

SESSION

A TOUR OF ADVANCED DATA MINING METHODOLOGIES

Chair(s)

Dr. Dan Steinberg

A Tour of Advanced Data Mining Methodologies: The CART Decision Tree

Dan Steinberg

Salford Systems, San Diego, California

Abstract - *The 1984 monograph, "CART: Classification and Regression Trees," co-authored by Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone (BFOS), represents a milestone in the evolution of Artificial Intelligence, Machine Learning, non-parametric statistics, and data mining. The work is important for the comprehensiveness of its study of decision trees, the technical innovations it introduces, its sophisticated examples of tree-structured data analysis, and its authoritative treatment of large sample theory for trees. CART citations can be found in almost any domain, including credit risk, targeted marketing, financial markets modeling, electrical engineering, quality control, biology, chemistry, and clinical medical research. This brief account is intended to introduce CART basics, touching on the major themes treated in the CART monograph, and to encourage readers to return to the rich original source for technical details, discussions revealing the thought process of the authors, and examples of their analytical style.*

Keywords: CART, Classification, Regression Trees, Breiman, Friedman

1 Overview

The CART decision tree is a binary recursive partitioning procedure capable of processing continuous and nominal attributes as targets and predictors. Data are handled in their raw form; no binning is required or recommended. Beginning in the root node, the data are split into two children, and the children are in turn each split into grandchildren. Trees are grown to a maximal size without the use of a stopping rule; essentially, the tree growing process stops when no further splits are possible due to lack of data. The maximal-sized tree is then pruned back to the root (essentially split by split) via the novel method of cost-complexity pruning. The next split to be pruned is the one

contributing least to the overall performance of the tree on training data (and more than one split may be removed at a time). The CART mechanism is intended to produce not one tree, but a sequence of nested pruned trees, each of which is a candidate to be the optimal tree. The "right-sized" or "honest" tree in the sequence is identified by evaluating, on independent test data, the predictive performance of every tree in the pruning sequence. Unlike C4.5, CART does not use an internal (training-data-based) performance measure for tree selection. Instead, tree performance is always measured on independent test data (or via cross validation) and tree selection proceeds only after test-data-based evaluation. If testing or cross validation has not been performed, CART remains agnostic regarding which tree in the sequence is best. This is in sharp contrast to methods such as C4.5 or classical statistics that generate preferred models on the basis of training data measures.

The CART mechanism includes (optional) automatic class balancing and automatic missing value handling, and allows for cost-sensitive learning, dynamic feature construction, and probability tree estimation. The final reports include a novel attribute importance ranking. The CART authors also broke new ground in showing how cross validation can be used to assess performance for every tree in the pruning sequence even though trees in different CV folds may not align on the number of terminal nodes. It is useful to keep in mind that BFOS addressed all these topics in the 1970's and that, in some cases, the BFOS treatment appears to remain the state of the art. The literature of the 1990s contains a number of articles that rediscover core insights first introduced in the 1984 CART monograph. Each of these major features is discussed separately below.

2 The Algorithm briefly stated

A complete statement of the CART algorithm that includes all relevant technical details is lengthy and complex; there are multiple splitting rules available for both classification and regression, separate handling of continuous and categorical splitters, special handling for categorical splitters with many levels, and provision for missing value handling. Following the tree-growing procedure, there is another complex procedure for pruning the tree, and finally, there is tree selection. Here we sketch out a simplified algorithm for tree growing. Formal statements of the algorithm are provided in the CART monograph. Here we offer an informal statement that is highly simplified.

2.1 Simplified tree pruning algorithm

Having grown the tree, CART next generates the nested sequence of pruned subtrees, and a simplified algorithm sketch for pruning follows that ignores priors and costs. This is different from the actual CART pruning algorithm and is included here for the sake of brevity and ease of reading. The procedure begins by taking the largest tree grown (T_{max}) and removing all splits, generating two terminal nodes that do not improve the accuracy of the tree on training data. This is the starting point for CART pruning. Pruning proceeds further by a natural notion of iteratively removing the weakest links in the tree, the splits that contribute the least to performance of the tree on test data. In the algorithm presented below we restrict the pruning action to parents of two terminal nodes. (This algorithm will be shown in the live presentation accompanying this paper; or, request by contacting abaldwin@salford-systems.com.)

The CART pruning algorithm differs from the above in employing a penalty on nodes mechanism that can remove an entire subtree in a single pruning action. The monograph offers a clear and extended statement of the procedure. We now discuss major aspects of CART in greater detail.

4 Splitting rules

CART splitting rules are always couched in the form:

An instance goes left if $CONDITION$, and goes right otherwise,

where the $CONDITION$ is expressed as "attribute $X_i \leq c$ " for continuous attributes. For categorical or nominal attributes the $CONDITION$ is expressed as membership in a list of values. For example, a split on a variable like CITY might be expressed as:

An instance goes left if CITY is in {Chicago, Detroit, Nashville} and goes right otherwise

The splitter and the split point are both found automatically by CART with the optimal split selected via one of the splitting rules defined below. Observe that because CART works with unbinned data, the optimal splits are always invariant with respect to order-preserving transforms of the attributes (such as log, square root, power transforms, etc). The CART authors argue that binary splits are to be preferred to multi-way splits because (1) they fragment the data more slowly than multi-way splits, and (2) repeated splits on the same attribute are allowed and, if selected, will eventually generate as many partitions for an attribute as required. Any loss of ease in reading the tree is expected to be offset by improved predictive performance.

The CART authors discuss examples using four splitting rules for classification trees (Gini, twoing, ordered twoing, symmetric gini), but the monograph focuses most of its discussion on the Gini, which is similar to the better known entropy (information-gain) criterion. For a binary (0/1) target the "Gini measure of impurity" of a node t is

$$G(t) = 1 - p(t)^2 - (1 - p(t))^2$$

where $p(t)$ is the (possibly weighted) relative frequency of class 1 in the node. Specifying

$$G(t) = -p(t)\ln p(t) - (1 - p(t))\ln(1 - p(t))$$

instead yields the entropy rule. The improvement (gain) generated by a split of the parent node P into left and right children L and R is

$$I(P) = G(P) - qG(L) - (1 - q)G(R)$$

Here, q is the (possibly weighted) fraction of instances going left. The CART authors favored the

Gini over entropy because it can be computed more rapidly, can be readily extended to include symmetrical costs (see below), is less likely to generate "end cut" splits, –splits with one very small (and relatively pure) child and another much larger child. (Later versions of CART have added entropy as an optional splitting rule.) The twoing rule is based on a direct comparison of the target attribute distribution in two child nodes:

$$I(\text{split}) = [25q(1-q)^u \sum_k |p_L(k) - p_R(k)|^2]$$

where k indexes the target classes, and p_L and p_R are

the probability distributions of the target in the left and right child nodes, respectively. (This splitter is a modified version of Messenger and Mandell, 1972.) The twoing "improvement" measures the difference between the left and right child probability vectors, and the leading $q(1-q)$ term, which has its

maximum value at $q=.5$, implicitly penalizes splits that generate unequal left and right node sizes. The power term u is user-controllable, allowing a continuum of increasingly heavy penalties on unequal splits; setting $u=10$, for example, is similar to enforcing all splits at the median value of the split attribute. In our practical experience the twoing criterion is a superior performer on multi-class targets as well as on inherently difficult-to-predict (e.g. noisy) binary targets. BFOS also introduce a variant of the twoing split criterion that treats the classes of the target as ordered. Called the "ordered twoing" splitting rule, it is a classification rule with characteristics of a regression rule because it attempts to separate low-ranked from high-ranked target classes at each split.

For regression (continuous targets), CART offers a choice of Least Squares (LS, sum of squared prediction errors) and Least Absolute Deviation (LAD, sum of absolute prediction errors) criteria as the basis for measuring the improvement of a split. As with classification trees the best split yields the largest improvement.

5 Prior probabilities and class balancing

Balancing classes in machine learning is a major issue for practitioners, because many data mining

methods do not perform well when the training data are highly unbalanced. For example, for most prime lenders default rates are generally below 5% of all accounts, in credit card transactions fraud is normally well below 1% and in internet advertising "click through" rates occur typically for far fewer than 1% of all ads displayed (impressions). Many practitioners routinely confine themselves to training data sets in which the target classes have been sampled to yield approximately equal sample sizes. Clearly, if the class of interest is quite small such sample balancing could leave the analyst with very small overall training samples. For example, in an insurance fraud study the company identified about 70 cases of documented claims of fraud. Confining the analysis to a balanced sample would limit the analyst to a total sample of just 140 instances (70 fraud, 70 not fraud).

It is interesting to note that the CART authors addressed this issue explicitly in 1984 and devised a way to free the modeler from any concerns regarding sample balance. Regardless of how extremely unbalanced the training data may be, CART will automatically adjust to the imbalance with no action, preparation, sampling, or weighting required by the modeler. The data can be modeled as they are found without any preprocessing.

To provide this flexibility CART makes use of a "priors" mechanism. Priors are akin to target class weights but are invisible in that they do not affect any counts reported by CART in the tree. Instead, priors are embedded in the calculations undertaken to determine the goodness of splits. In its default classification mode, CART always calculates class frequencies in any node relative to the class frequencies in the root. This is equivalent to automatically reweighting the data to balance the classes, and ensures that the tree selected as optimal minimizes balanced class error. The reweighting is implicit in the calculation of all probabilities and improvements and requires no user intervention; the reported sample counts in each node thus reflect the unweighted data. For a binary (0/1) target any node is classified as class 1 if, and only if,

$$\frac{N_1(\text{node})}{N_1(\text{root})} > \frac{N_0(\text{node})}{N_0(\text{root})}$$

Observe that this ensures that each class is assigned a working probability of $1/K$ in the root node when there are K target classes, regardless of the actual distribution of the classes in the data. This default mode is referred to as "priors equal" in the

monograph. It has allowed CART users to work readily with any unbalanced data, requiring no special data preparation to achieve class rebalancing or the introduction of manually constructed weights. To work effectively with unbalanced data it is sufficient to run CART using its default settings. Implicit reweighting can be turned off by selecting the "priors data" option. The modeler can also elect to specify an arbitrary set of priors to reflect costs, or potential differences between training data and future data target class distributions. Note: The priors settings are unlike weights in that they do not affect the reported counts in a node or the reported fractions of the sample in each target class. Priors do affect the class any node is assigned to as well as the selection of the splitters in the tree-growing process. (Being able to rely on priors does not mean that the analyst should ignore the topic of sampling at different rates from different target classes; rather, it gives the analyst a broad range of flexibility regarding when and how to sample.)

6 Missing value handling

Missing values appear frequently in real world, and especially business-related, databases, and the need to deal with them is a vexing challenge for all modelers. One of the major contributions of CART was to include a fully automated and effective mechanism for handling missing values. Decision trees require a missing value-handling mechanism at three levels: (a) during splitter evaluation, (b) when moving the training data through a node, and (c) when moving test data through a node for final class assignment. (See Quinlan, 1989 for a clear discussion of these points.) Regarding (a), the first version of CART evaluated each splitter strictly on its performance on the subset of data for which the splitter is not missing. Later versions of CART offer a family of penalties that reduce the improvement measure to reflect the degree of missingness. (For example, if a variable is missing in 20% of the records in a node, then its improvement score for that node might be reduced by 20%, or alternatively by half of 20%, etc). For (b) and (c), the CART mechanism discovers "surrogate" or substitute splitters for every node of the tree, whether missing values occur in the training data or not. The surrogates are thus available should a tree trained on complete data be applied to new data that includes missing values. This is in sharp contrast to machines that cannot tolerate missing values in the training data or that can only learn about missing value handling from training data that include missing values. Friedman (1975) suggests moving instances

with missing splitter attributes into both left and right child nodes and making a final class assignment by taking a weighted average of all nodes in which an instance appears. Quinlan (1989) opts for a variant of Friedman's approach in his study of alternative missing value-handling methods. In CART the missing value handling mechanism is fully automatic and locally adaptive at every node. At each node in the tree the chosen splitter induces a binary partition of the data (e.g., $X_1 \leq c_1$ and $X_1 > c_1$). A

surrogate splitter is a single attribute Z that can predict this partition where the surrogate itself is in the form of a binary splitter (e.g., $Z \leq d$ and $Z > d$). In other words, every

split becomes a new target that is to be predicted with a single split binary tree. Surrogates are ranked by an association score that measures the advantage of the surrogate over the default rule predicting that all cases go to the larger child node (after adjustments for priors). To qualify as a surrogate, the variable must outperform this default rule (and thus it may not always be possible to find surrogates). When a missing value is encountered in a CART tree the instance is moved to the left or the right according to the top-ranked surrogate. If this surrogate is also missing then the second ranked surrogate is used instead (and so on). If all surrogates are missing the default rule assigns the instance to the larger child node (after adjusting for priors). Ties are broken by assigning an instance to the left.

In the mobile phone example which will be discussed in the live presentation accompanying this paper (more details related to this case study example are available from Salford Systems; request by contacting abaldwin@salford-systems.com), consider the right child of the root node, which is split on TELEBILC, the land line telephone bill. If the telephone bill data are unavailable (for example, the household is a new one and has limited history with the company), CART searches for the attributes that can best predict whether the instance belongs on the left or the right side of the split. In this case we see that of all the attributes available the best predictor of whether the cost of the land line telephone is high (greater than 50) is marital status (never-married people spend less), followed by the travel time to work, age, and, finally, city of residence. Surrogates can also be seen as akin to synonyms in that they help to interpret a splitter. Here we see that those with lower telephone bills tend to be never married, live closer to the city center, younger, and concentrated in three of the five cities studied.

7 Stopping rules, pruning, tree sequences, and tree selection

The earliest work on decision trees did not allow for pruning. Instead, trees were grown until they encountered some stopping condition and the resulting tree was considered final. In the CART monograph the authors argued that no rule intended to stop tree growth can guarantee that it will not miss important data structure (e.g., consider the two-dimensional XOR (more details are available from Salford Systems: request by contacting abaldwin@salford-systems.com)). They therefore elected to grow trees without stopping. The resulting overly-large tree provides the raw material from which a final optimal model is extracted.

The pruning mechanism is based strictly on the training data and begins with a cost-complexity measure defined as

$$R_{\alpha}(T) = R(T) + \alpha|T|$$

where $R(T)$ is the training sample cost of the tree, $|T|$ is the number of terminal nodes in the tree, and α is a penalty imposed on each node. If $\alpha=0$ then the minimum cost-complexity tree is clearly the largest possible. If α is allowed to progressively increase, the minimum cost-complexity tree will become smaller because the splits at the bottom of the tree that reduce $R(T)$ the least will be cut away. The parameter α is progressively increased in small steps from 0 to a value sufficient to prune away all splits. BFOS prove that any tree of size Q extracted in this way will exhibit a cost $R(Q)$ that is minimum within the class of all trees with Q terminal nodes. This is practically important because it radically reduces the number of trees that must be tested in the search for the optimal tree. Suppose that a maximal tree has $|T|$ terminal nodes. Pruning involves removing a split generating two terminal nodes and absorbing the two children into their parent, thereby replacing the two terminal nodes with one. The number of possible subtrees extractable from the maximal tree by such pruning will depend on the specific topology of the tree in question. It can be shown that the number of unique subtrees that can be extracted from a perfectly balanced maximal tree can exceed

$(1.2 \exp(.2|T|))$ which for an 81 terminal-node

maximal tree is more than 10 million trees. However, given cost-complexity pruning for the

example which will be discussed in the live presentation accompanying this paper, we need to examine only 28 subtrees. (More details related to this example are available from Salford Systems: request by contacting abaldwin@salford-systems.com).

The optimal tree is defined as that tree in the pruned sequence that achieves minimum cost on test data. Because test misclassification cost measurement is subject to sampling error, uncertainty always remains regarding which tree in the pruning sequence is optimal. Indeed, an interesting characteristic of the error curve (misclassification error rate as a function of tree size) is that it is often flat around its minimum for large training data sets. BFOS recommend selecting the "*I SE*" tree that is the smallest tree with an estimated cost within 1 standard error of the minimum cost (or "*O SE*") tree. Their argument for the *I SE* rule is that in simulation studies it yields a stable tree size across repeated simulations whereas the size of the *O SE* tree size can vary substantially across simulations. The pruning sequence continues all the way back to the root as we must allow for the possibility that our tree will demonstrate no predictive power on test data.

8 Software availability

CART software is available from Salford Systems, at <http://www.salford-systems.com>; no-cost evaluation versions may be downloaded on request.

9 References

- [1] Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J. 1984. *Classification and Regression Trees*. Belmont:Wadsworth. Republished by CRC press.
- [2] Friedman, J. H. 1977. A recursive partitioning decision rule for nonparametric classification. *IEEE Trans. Computers* 26:404-408. Also available as Stanford Linear Accelerator Center Rep. SLAC-PUB-1373 Rev. 1975.
- [3] Olshen, R. 2001. A Conversation with Leo Breiman. *Statistical Science* 16:184-198.

SESSION

ASSOCIATION RULE MINING

Chair(s)

Dr. Philippe Lenca

Pruning for Extracting Class Association Rules without Candidate Generation

Emna Bahri and Stephane Lallich

Abstract—In this paper, we focus on a supervised learning task using association rules' algorithms (association based classification). These algorithms, developed for unsupervised learning, extract all the rules whose support and confidence exceed some prefixed threshold. The extraction of class association rules, using these algorithms, has several problems because of a-posteriori rules filtering. In the first stage, useless frequent itemsets are extracted, which are not included in the class item. Whereas the second stage can be simplified, since an itemset containing the class item produces only one class rule. In order to be able to work with a low threshold support, we propose FCP-Growth and an alternative FCP-Growth-P, which is an adaptation of FP-Growth having three main advantages. Firstly, it eliminates the frequent itemsets that do not contain a class item. Secondly, we give priority to the minority class during the construction of the class itemsets by adapting the threshold support. By doing this the same threshold support inside each class attributes is used. Thirdly, for better pruning, FCP-Growth-P uses the pruning techniques as used by Li on rules specialization. Any specialization of a rule, that does not reduce the number of counter-examples, is less interesting than the basic rule, both for confidence and various interestingness measures.

I. INTRODUCTION

We focus on a supervised learning task using association rules algorithms (association-based classification). In fact, class association rule mining algorithms provide rules which allow to discover useful information, hidden in dense databases. Association based classification methods usually consist of two main steps: The first is the discovery of frequent itemsets from association rule mining algorithms such as Apriori [1] or FP-Growth [3], whose principle will be presented in section 2. In the second stage, these algorithms subdivide the frequent itemsets in two disjointed parts in order to build all the possible rules and keeping only the rules whose confidence exceeds the confidence threshold, denoted minconf .

In the first stage, these algorithms extract all the frequent itemsets, whose support exceeds the support threshold, denoted as minsupp . It is the most expensive stage in terms of execution time, because the number of these frequent items depends exponentially on the number of items handled (for p items, there are potentially 2^p itemsets). Sometimes, in order to achieve this stage, we must increase the support threshold, which gives a central role to the support condition and makes the extraction of nuggets (namely the rules

having a very strong confidence and a very weak support) more difficult.

The use of unsupervised algorithms in order to determine the class rules, present various disadvantages. On the one hand, in the first stage, they extract many useless frequent itemsets, those which do not contain a class item. The extraction of useless frequent itemsets makes it impossible to reduce the support threshold. In addition, the second stage can be simplified, since an itemset containing a class item (class itemset) produces only one class rule. In addition, the use of a support threshold is usually applied to the totality of the minority class which is generally the most interesting and the most difficult to correctly predict. Finally, these various algorithms are based only on the support and confidence to extract the interesting rules, whereas various work ([9]) showed that we can improve the quality of the rules extracted by using other interesting measurements, more relevant than confidence and the support.

To avoid these problems, we present FCP-Growth (an adaptation of FP-Growth) which in the stage of frequent itemsets construction makes it possible to generate only the frequent class itemsets. Moreover, the same support threshold is used inside each class in order to take into account the possible imbalances of the classes. Lastly, we introduce one condition of pruning in FCP-Growth to avoid extracting specializations from rules which do not have any interest and thus, define the FCP-Growth-P algorithm.

This paper initially will present the state of the art of class association rules methods, while being interested more particularly in the various association rule mining algorithms (search of frequent itemsets) and the methods used for pruning (Section 2). Then, we explain the improvements made with FP-Growth (Section 3) and we compare the results of FCP-Growth-P and FCP-Growth with those of FP-Growth, after having applied them to seven different datasets (Section 4). We conclude by presenting the perspectives of this work (Section 5).

II. STATE OF THE ART

In this section, we present the principal algorithms of associative classification. We recall the various methods of extracting association rules in unsupervised learning and the pruning strategies.

Emna Bahri and Stephane Lallich are with university of Lyon, ERIC Laboratory, 5 avenue Pierre Mendès-France, France (phone: 0033 478 673 046; fax: 0033 478 772 375; email: emna.bahri — stephane.lallich@univ-lyon2.fr).

A. Associative classification methods

Since the first efforts of [5], various works show the good performances of association based classification in terms of error rate reduction. Association based classification deals with the prediction of the class from association rules, known as class association rules or predictive association rules. A class association rule is a rule whose consequent must be the indicating variable of one of the class attributes. Such a rule is thus $A \rightarrow c_i$, where A is a conjunction of boolean descriptors and c_i is the indicating variable of the i_e class attribute. The interest of class rule is to allow focusing on groups of individuals, possibly very small, homogeneous (same descriptors in antecedent) and presenting the same class. The association based classification methods proceed in two phases, a phase of construction class association rules "class association rule mining" followed by a phase of prediction from the class association rules obtained. The first phase consists of two steps; the first step is to extract frequent itemset, taking into account the support threshold chosen, the second step is to construct class association rules satisfying the threshold of fixed confidence. Among these methods, one will quote initially CBA (Classification Based on Association) [5], the first algorithm proposed. This algorithm is held in two steps:

- CBA-RG uses APRIORI to generate all the association rules which satisfy support and confidence threshold chosen at the beginning
- CBA-CB is a heuristic algorithm which ensures the prediction: each example of training is covered by the rule having the most raised confidence and this one is stored. The stored rules are classified by decreasing order of confidence and one applies the rule with a new example which covers it with stronger confidence.

In [8], the author proposed an improved version of CBA called CPAR (Classification based on Predictive Association Rules). This method uses a rule generation algorithm for classification called FOIL (First Order Inductive Learner) [6] to avoid the redundant rules by building rules to distinguish positive from negative tuples. For classification, CPAR does not select only the best rule (that having the highest confidence) but it selects K better rules for each class, compares the variance of each one and chooses the one that has more predictive accuracy.

Another improvement suggested is CMAR (Classification based on Multiple Association Rules) [4]. CMAR selects the rules based on a high confidence and the correlation between the analyzed rules. The used measurement is the Weighted X^2 which expresses the power of a rule according to the support and class' distribution. To perform effectiveness, CMAR employs a new structuring data, FP-tree, to store and select the rules; CMAR uses FP-Growth instead of algorithm Apriori.

For the data with great dimensions, we find the CAEP method (Classification by Aggregating Emerging Patterns) [2]. This method uses a new type of knowledge "emerging pattern (EPs)". The EPs are itemsets whose sup-

port increases significantly from one class to another and can even differentiate classes for certain example. These methods build classifiers based on rules selection.

To deal with the multi-label problem, the authors [7] propose MMAC (Multi-class and Multi-label Association Classification). In fact, the algorithm considers a multi-label as a list of rows of class label associated to each example. In addition, the algorithm uses an intermediary phase of recursive learning between the extraction of the rules and the construction of the classification rules. These methods of associative classification have good performances. However, they are based on a first phase of association rules extraction which is not in relation with the second phase of classification.

B. Extraction of association rules

During the first phase of the rules extraction, the usual methods of associative classification are based on the algorithms of association rules extraction used in unsupervised training. Among those, the most used are Apriori and FP-Growth.

The strategy used by Apriori [1] to generate the frequent itemsets is simple. It extracts those in an ascending order of their respective length, based on the antimonotonicity property of support's condition. According to this property, all the super-itemsets of not frequent itemset are not frequent and all sub-itemset of frequent itemset are frequent. On the level k of the itemsets lattice, Apriori generates all the itemsets candidates with length k by grouping all $k - 1$ itemsets and retains only itemsets whose frequency is higher or equal to minsupp. Then, using each frequent itemset X , Apriori examines all the rules of the type $X/Y \rightarrow Y$, $X \cap Y = \emptyset$ and retains only itemsets whose confidence is at least equal to minconf, the selected confidence threshold. These rules are then added to ER, the group of associations rules which satisfy prefixed support and confidence thresholds.

Contrary to Apriori which generates itemsets candidates and tests them to preserve only frequent itemsets, FP-Growth [2] builds the frequent itemsets without generation of candidates. In fact, it uses the strategy of divide-and-conquer. First, it compresses the frequent items found in the data base in frequent-pattern tree (FP tree) which contains the association of the itemsets. Then, it divides each association which will be presented by item frequent or pattern fragment. The FP-Growth transforms the problem of finding the frequent itemsets by searching the smallest and the concatenation of the suffix (last frequent itemset). This makes it possible to reduce the search cost.

Example of FP-Growth : To explain the principle of FP-Growth, one chooses the simplified database described in the table 1 which contains descriptive items (I) and class items (C).

- Firstly, FP-Growth scans the database and derives the set of frequent items and their support counts and compares items support count with minimum support count. For

TABLE I: Database example

TID	List of item-TID
T1	I1,I2,I5,C1
T2	I2,I4,C2
T3	I2,I1,I3,C1
T3	I1,I2,I4,C2

TABLE II: items support count

ITEMS	Support
I2	4
I1	3
I4	2
C1	2
C2	2
I3	1
I5	1

this example, we choose the minimum support count as 1. The set of frequent items is stored in order of descending support count in a *list*. Thus we have L : $I2 : 4, I1 : 3, I4 : 2, C1 : 2, C2 : 2, I3 : 1, I5 : 1$.

- Secondly, an FP-Tree is constructed by creating the root of the tree labeled null and scanning the database a second time but in this phase each transaction contains items in L order. For example, the first transaction "T1 : I1, I2, I5, C1" which contains items (I2, I1, C1, I5) in L order, leads to construct the first branch of tree with tree nodes (I2 : 1), (I1 : 1), (C1 : 1) and (I5 : 1). The second transaction T2 contains the items (I2, I4, C1) in L order which would result in a branch where I2 is linked to the root and I4 is linked to I2 and C1 is linked to I4. However, this branch would share a common **prefix**, I2, with the existing path for T1. Instead, we thus, increment the count of I2 by 1 and create a new node (I4 : 1), which linked as children of (I2 : 2).

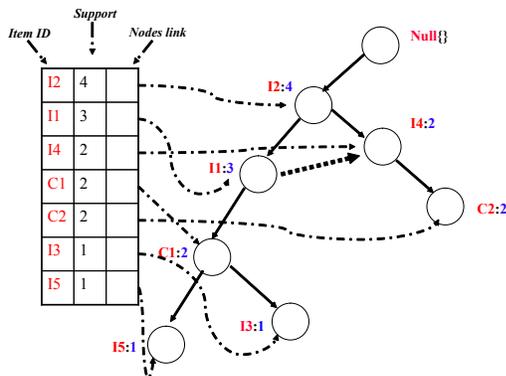


Fig. 1: Construction of FP-Tree

- Thirdly, the FP-Tree is mined by creating conditional (sub-)pattern base. The FP-Tree is mined as follows:

TABLE III: Mining FP-Tree

Item	Cond P	Cond FP-Tree	FP
I5	I1,I2,C1,	(I2:I1:I5:C1:1)	I2,I1,C1,I5:1
I4	I2:I1,I2,I1:1	(I2:2)	I2,I4:2

Start from each frequent length -1 pattern (**suffix pattern**), construct its **conditional pattern base** (set of prefix paths in FP-Tree), then construct its conditional FP-Tree. The pattern growth is achieved by concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-Tree.

In addition, it should be noted that the class association rules obtained by filtering of the rules extracted from the unsupervised algorithms are confronted with the imbalanced classes problem. In fact, in the stage of the frequent itemsets generation, these algorithms select the itemsets with a support threshold which does not take into account the weight of different classes to be predicted, which is disadvantage of the class rules associated with the least frequent modality. In our study, we chose to retain FP-Growth, because of its structuring (FP-Tree), which is regarded as more effective than Apriori. We have proposed FCP-Growth, an alternative of FP-Growth which builds only the itemsets of class, whose principle is exposed in ([13]). The aim of this paper is to improve the quality of FCP-Growth algorithm by introducing a pruning condition.

C. Pruning

During the frequent itemsets extraction, the various algorithms are based on only one principle of pruning, that is of the support condition. In the literature, there exists several ways of pruning which gives good performances. We find in particular :

- 1) The general rule (CSA): Let's have two rules R1 and R2, R1 is more general than R2 if $conf(R1) > conf(R2)$, or $conf(R1) = conf(R2)$ et $sup(R1) > sup(R2)$, or finally $conf(R1) = conf(R2)$, $sup(R1) = sup(R2)$ and R1 have less items. Pruning method consists of removing the less general rule. This method is used on CMAR
- 2) The most powerful rule: One can quote FOIL criterion ([6]), used by CPAR ([8]), which is based on the measure "FOIL - Prune(R) = (pos - neg) ÷ (pos + neg)" where *pos* and *neg* are the positive and negative numbers of t-tuples covered by the rule R. Another possible criterion is the correlation of the rule antecedent with the class measured by χ^2 . This criterion is used by CMAR
- 3) The rule behavior with the counter-examples: the works of Li ([11], [10] and [12]) show that if the specialization of a rule does not decrease the number of counter-examples, then neither this specialization, nor over-specialization is more interesting within the confidence measurement or some other interest measures. We chose to introduce into the FCP-Growth algorithm the latter criterion of pruning, used by Li with Apriori, to obtain FCP-Growth-P.

III. NEW APPROACH : FCP-GROWTH

In order to reduce the cost of the class association rules in terms of execution time and storage, algorithms must generate only frequent class itemsets, i.e. the frequent itemsets

which contain one of the class attributes. For this purpose, we have proposed FCP-Growth (Frequent Class Pattern), which is an alternative of FP-Growth that stores in FCP-Tree (Frequent Class Pattern Tree) only the frequent itemset c_iA containing a class item c_i . Such an itemset generates only one rule of class, $A \rightarrow c_i$. This allows us having profited, both in execution time and storage space.

At the same time, to solve the problem of the imbalanced classes, we propose to fix an adaptive support threshold, which is inversely proportional to the number of examples of each class, so that the minority class be in advantage during the construction of the class itemsets. If n_i is the number of examples which validate the class C_i , the i^e class attributes, the global support threshold σ , the support threshold adjustable to the class C_i is σ_i , with $n \times \sigma_i = n_i \times \sigma$. The adaptive support threshold is thus proportional to the class size, so that this threshold is in proportion to the class size and not of the database size, this adaptive support has the same value for each class attribute. This way, we prevent that the minority classes are systematically handicapped during the construction of the class itemsets and association rules.

Algorithm. Our adaptation has as a principle to add modules to the initial version of FP-Growth.

- 1) Pretreatment of the database: During this stage, we let the user choose the class modality from the various attributes of the database. Thus, the class to be predicted is determined. We reorganize the database into a table whose first column contains the various values assigned to this class. This method, enables us to build frequent FCP-tree of the items which contains the class to be predicted. Let's re-examine the transactional database using the FCP-Growth approach: During the pretreatment phase, we obtain the following database (table 4)

TABLE IV: Database after pretreatment

TID	List of item-TID
T1	$C1, I1, I2, I5$
T2	$C2, I2, I4$
T3	$C1, I2, I1, I3$
T4	$C2, I1, I2, I4$

TABLE V: Items according to class support

ITEMS	Supp For C1	Supp for C2
$I2$	4	4
$I1$	3	3
$I4$	2	2
$I3$	-	1
$I5$	-	1

- 2) Choice of the adjustable supports: This stage enables us not to fix a global support (as initial version), but a support for each class attributes depending on the number of transactions n_i which validate each class i . We choose σ and we define the support threshold

relative to c_i as $\sigma_i = \sigma \times \frac{n_i}{n}$. These support thresholds are adapted to each class attribute. This method supports the classes having an insufficient number of examples. We then construct the list L which contains firstly the class items ($C1$ or $C2$) and the items stored according to descending support count of $C1$ or $C2$. For example, let's define support count of $C1$ as 200 and support count of $C2$ as 100. If we define σ as 1% so the minimum support count of $C1$ is 2 and the minimum support count of $C2$ is 1. For example, we have two lists, $L1$ for the class $C1$ and $L2$ for the class $C2$:

- $L1 = (C1 : P1), (I2 : 4), (I1 : 3), (I4 : 2)$
- $L2 = (C2 : P2), (I2 : 4), (I1 : 3), (I4 : 2), (I3 : 1), (I5 : 1)$

Using these lists of items, the first transaction is stored in $L : (C1, I2, I1)$

- 3) Construction of FCP-tree and the class frequent patterns: This stage proceeds with the same way of FP-Growth. However, only the transaction which starts with a class to be predicted will be treated. Thus there will be the class attributes as child linked to the root of the FCP-Tree. For example, we create first the root of the tree labeled with null, scan the database for the second time and construct m child of the root if we have m class attributes. In the example below, we construct 2 children to the root $C1$ and $C2$ because we have 2 class attributes $C1, C2$. The first branch of the tree is construct with the first transaction. For the same example, the first branch of tree is linked to the child $C1$ of FP-Tree by three nodes $(I2 : 1), (I1 : 1)$. Like that, we are sure that, for each item, we have a conditional pattern that contains a class item.

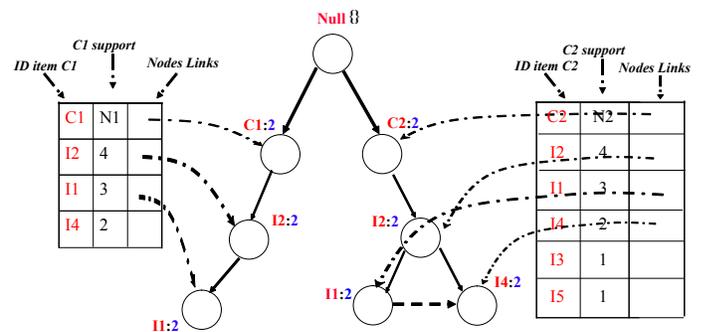


Fig. 2: Construction of FCP-tree

- 4) Mining the FP-Tree and generation of frequent itemsets: The FCP-tree is mined in the same way as FP-Tree. For each item, all conditional patterns are found and so that all frequent patterns (i.e., patterns having a higher support or equal to the selected support) are generated. This phase is called "mining frequent

patterns". Frequent itemsets are obtained by the association of each item with frequent pattern. All frequent itemsets found contain a class item. For example using FCP-Growth, we don't find the itemset $(I2, I4)$ of the example used in FP-Growth.

- 5) Pruning: Such as FP-Growth, The FCP-Growth algorithm uses the CSA pruning condition. In order to obtain an efficient pruning, we replace the CSA pruning condition by the pruning method of Li [12]. This method of pruning rests on fact that any specialization of a rule which does not decrease the number of counter-examples is less interesting than the starting rule. This method was proposed within the framework of Apriori, an algorithm of extraction of frequent itemsets with generation of candidates. To introduce it into FCP-Growth which is an algorithm of extraction of frequent itemsets without generation of candidates, we had to carry out some adaptations. For each transaction and a node construction (item I), we update and we determine the transaction numbers which validate the itemsets association from the beginning of the branch until item node $Supp(Ic)$ and the transaction numbers which contradicts this association $Supp(I\bar{c})$.

Let's have a class itemset Ac , The associated class rule is $A \rightarrow c$. The counter-examples of this rule correspond to the itemset $A\bar{c}$. Let's have the item x which associate the specialization $Ax \rightarrow c$. The pruning condition of Li is applied to optimal measures. An optimal measure is a measure m where $m(Ax \rightarrow c) \leq m(A \rightarrow c)$, if $supp(Ax\bar{c}) = Supp(A\bar{c})$. To integrate this constraint in an algorithm, it is necessary to be able to update the number of counter-examples of the class rule associated with the current class itemset. With FCP-Growth, children of the root correspond to a class attributes. For example, if we have the class itemset ca_1a_2 , and $ca_1a_2a_3$, the pruning condition compares the counter-examples supports of the two class rules $a_1a_2 \rightarrow c$ and $a_1a_2a_3 \rightarrow c$, so $p_{\bar{c}a_1a_2}$, and $p_{\bar{c}a_1a_2a_3}$.

FCP-Growth updates the supports $p_{ca_1a_2}$, and $p_{ca_1a_2a_3}$. To calculate $p_{\bar{c}a_1a_2} = p_{a_1a_2} - p_{ca_1a_2}$ and $p_{\bar{c}a_1a_2a_3} = p_{a_1a_2a_3} - p_{ca_1a_2a_3}$, we have to know $p_{a_1a_2}$ and $p_{a_1a_2a_3}$. Generally, for each node cA , we have to calculate p_{cA} and p_A , to predict $p_{\bar{c}A} = p_A - p_{cA}$. For the example chosen, if $p_{\bar{c}a_1a_2a_3} = p_{\bar{c}a_1a_2}$, we prune this node, because neither the specialized rule $a_1a_2a_3 \rightarrow c$ nor each over-specializations are more interesting than the rule $a_1a_2 \rightarrow c$.

IV. MAIN RESULTS

FCP-Growth and FCP-Growth-P, the algorithms which we propose to build directly the class frequent itemsets and the rules which result from this itemsets, rest on several principles:

- FCP-Growth is an adaptation of FP-Growth, known as faster than Apriori.

- During the construction of FCP-Tree, FCP-Growth eliminates the non relevant itemsets, those which do not contain class items, whereas the usual methods build all the rules and eliminate then the rules which are not rules of class. This characteristic should improve the execution time and make it possible to decrease the support threshold, which is particularly important in associative classification, where one is very interested by the nuggets.
- FCP-Growth uses an adaptive threshold support where the support threshold used in each class is not the same, in order to not favorite the small classes, this threshold is selected proportional to the size of the class, so we fixed the threshold support as the same percentage of each class.
- FCP-Growth-P is an alternative of FCP-Growth which use a pruning procedure based on the counter-examples. The pruning strategy applies the Li condition of counter-examples to the support, which makes it possible to eliminate uninteresting rules specializations. To evaluate the effectiveness of FCP-Growth and FCP-Growth-P, we tested them on 7 real datasets (table 6) and we calculated (tables 7 and 8), with a support threshold of 1%, the total number of itemsets (# Total Isets), the number of class itemsets C_i (# Isets C_i) and the number of non relevant itemsets (# No R Isets).

TABLE VI: Description of datasets

Datasets	#Exp	#Attr	#Classes	Classes freq
D100T20I6	100000	20	2	58%-42%
D100T20I2	100000	20	2	55%-45%
Retail data	63000	10	2	69%-31%
Adult	48842	14	2	75,22%-24,78%
Connect-4	67557	42	3	65,83%-24,62%-9,55%
Abalone	4177	8	3	53%-25%-22%
Census	48842	14	2	75.22%-24.78%

The comparison results of FCP-Growth and its alternative FCP-Growth-P, with FP-Growth results (tables 7 and 8) is carried out on the basis of 4 criterion:

- reduction of the total number of itemsets built
- increase of the number of class itemsets
- the good representation of the minority class
- the improvement of the cover rate (i.e. the examples proportion covered by a class itemset)

The results show, firstly, the importance of the non-relevant itemsets weight (the itemsets not containing the class item). On all seven treated datasets, it appears that between one the third and half of the itemsets generated by FP-Growth are not relevant, which affects the procedure unnecessarily. Moreover, Var1 measurement (obtained while dividing # Itemsets FCP-Growth by # Itemsets FP-Growth) shows that FCP-Growth as its FCP-Growth-P alternative make it possible to extract more rules of classes that FP-Growth, while preventing that the minority class is too underprivileged. This results proves that the performance of the algorithms is improved, thanks the use of adaptive threshold and the

TABLE VII: Results with 7 datasets

Datasets	Var1	FP-Gr	FCP-Gr	FCP-P	Var2
<i>Global support</i>	1,00%	1,00%	1,00%	1,00%	1,00%
D100T2016					
# Isets	-24,77%	1090	820	730	-10,97%
# Isets C1	12,22%	452	508	438	-13,72%
# Isets C2	28,46%	243	312	292	-6,48%
# No R Isets	-	395	0	0	-
# T Class Isets	17,98%	695	820	730	-,10,97%
D100T2012					
# Isets	-24,62%	1415	1075	967	-10,04%
# Isets C1	23,78%	565	699	610	-12,73%
# Isets C2	23,06%	306	376	357	-6,48%
# No R Isets	-	545	0	0	-
# T Class Isets	23,56%	870	1075	967	-10,04%
Retail data					
# Isets	-25,77%	330	245	224	-8,57%
# Isets C1	5,13%	121	127	117	-8,51%
# Isets C2	92,16%	40	118	107	-8,59%
# No R Isets	-	169	0	0	-
# T Class Isets	52,17%	161	245	224	-8,57%
Abalone					
# Isets	-14,19%	665	562	518	-7,89%
# Isets C1	7,25%	209	224	206	-12,43%
# Isets C2	36,50%	144	196	180	-5,19%
# Isets C3	12,89%	126	142	132	-4,14%
# No R Isets	-	176	0	0	-
# T Class Isets	17,57%	478	562	518	-7,89%
Adult					
# Isets	-14,60%	890	760	675	-11,18%
# Isets C1	2,8%	392	403	398	-14,53%
# Isets C2	43,33%	249	357	277	-7,82%
# No R Isets	-	249	0	0	-
# T Class Isets	18,56%	641	760	675	-11,18%
Connect-4					
# Isets	-11,04%	6543	5820	5184	-11,37%
# Isets C1	11,12%	2094	2328	2160	-15,80%
# Isets C2	14,15%	1439	2037	1657	-8,84%
# Isets C3	17,03%	1243	1455	1341	-7,82%
# No R Isets	-	1767	0	0	-
# T Class Isets	21,85%	4776	5820	5184	-11,37%
Census					
# Isets	-15,51%	780	659	615	-6,67%
# Isets C1	10,63%	328	362	335	-11,67%
# Isets C2	52,07%	195	297	280	-4,55%
# No R Isets	-	257	0	0	-
# T Class Isets	26%	523	659	615	-6,67%

elimination of the itemsets which are not class items itemsets. On the seven datasets, this increase varies between 17% and 23%.

To evaluate the FCP-Growth and FCP-Growth-P performances, we retained the cover rate which indicates the proportion of examples which are covered by at least a class itemset. The results of table 8 show clearly the superiority of FCP-Growth. Indeed, for each datasets, cover rate is increased to arrive approximately at more than 96% of cover. In addition, for a better quality and thanks to its pruning procedure, the FCP-Growth-P alternative, inheriting already the advantages of FCP-Growth in terms of increase in class itemsets, makes it possible to reduce the quantity of extracted class rules while the quality is the same. Indeed, var2 measurement of the table 7 (obtained by dividing # Itemsets FCP-Growth-P by # Itemsets FCP-Growth) shows that FCP-

Growth-P decreases the number of class rules extracted from 6,67% to 11,37% according to the datasets, while ensuring same cover rate.

Finally, it is noticed that the reduction in the class rules number, brought by the pruning condition of Li, results in a light increase the FCP-Growth-P execution time compared to FCP-Growth (Table 9). However the execution time of FCP-Growth-P remains definitely lower than that of FP-Growth.

TABLE VIII: Covers rate

Datasets	FP-Gr	FCP-Gr and FCP-P
D100T2016	63%	91%
D100T2012	75%	93%
Retail Data	67%	89%
Adult	69%	92%
Connect-4	73%	90%
Abalone	79%	96%
census	76%	92%

TABLE IX: Execution Time

Datasets	FP-Growth	FCP-Growth	FCP-Growth-P
D100T2016	23	12	14
D100T2012	20	9	11
Adult	18	10	12
Retaildata	6	2	4
Census	10	4	6
Connect-4	14	6	9
Abalone	12	6	8

V. CONCLUSION

We proposed FCP-Growth, an adaptation of FP-Growth to extract class itemsets for predictive association rules and FCP-Growth-P, its alternative which integrates pruning condition of Li. Both FCP-Growth and FCP-Growth-P build only class frequent itemsets. In addition, the support threshold used in each class is proportional to the size of class to avoid the imbalanced classes problem. The results show that FCP-Growth and FCP-Growth-P allow us to have profited both in execution time and storage space, thanks to the generation of only class frequent itemsets. Moreover, the cover rates shows that we obtain a quality of rules higher than the quality of the rules generated by FP-Growth. Based on the obtained results, we will use FCP-Growth-P like an algorithm of class rules extraction for the first phase of a powerful associative classification.

REFERENCES

- [1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.
- [2] Guozhu Dong, Xiuzhen Zhang, Limsoon Wong, and Jinyan Li. Caep: Classification by aggregating emerging patterns. In *Discovery Science*, pages 30–42, 1999.
- [3] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In Weidong Chen, Jeffrey Naughton, and Philip A. Bernstein, editors, *2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12. ACM Press, 05 2000.
- [4] Wenmin Li, Jiawei Han, and Jian Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *ICDM*, pages 369–376, 2001.

- [5] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.
- [6] J. R. Quinlan and R. M. Cameron-Jones. Induction of logic programs: FOIL and related systems. *New Generation Computing*, 13:287–312, 1995.
- [7] F. A. Thabtah, P. Cowling, and Yonghong Peng. MMAC: A new multi-class, multi-label associative classification approach. In *ICDM '04. Fourth IEEE International Conference on Data Mining*, pages 217–224 2004.
- [8] Xiaoxin Yin and Jiawei Han. CPAR : Classification based on predictive association rules. In *3rd SIAM International Conference on Data Mining (SDM'03)*, pages 331–335, San Francisco, CA.
- [9] Guillet.F. and Hamilton.H. Quality measures in data mining. *Springer-Verlag*, 2007.
- [10] Li, J. On optimal rule discovery. *IEEE Transformation on Knowledge and Data Engeneering*. 18(4), pages 460–471, 2006.
- [11] Li, J., H. Shen, and R. Topor. Mining the smallest association rule set for predictions. *IEEE International Conference on Data Mining*, pages 361, 2001.
- [12] Li, J. and Y. Zhang. Direct interesting rule generation. *IEEE International Conference on Data Mining*, 155, 2003.
- [13] Bahri, E. and S. Lallich. FCP-Growth: Class Itemsets for Class Association rules, *The 22nd International FLAIRS Conference* ,Sanibel Island, Florida, 2009.

Mining Frequent Itemsets by Transaction Decomposition with Itemset Clustering

I-En Liao, Member, IEEE, Ke-Chung Lin, and Hong-Bin Chen

Abstract - To speed up the task of frequent itemset mining, we proposed a technique, called Top-Down Mining algorithm (TDM), based on transaction decomposition in our previous work. Compared to the traditional algorithms, TDM has a unique feature that examines database itemsets in a top-down manner using decomposition. Besides, it reduces the number of database scans to at most two. However, the decomposition method may generate too many itemsets while decomposing transactions from the database. In this paper, we propose a new algorithm called TDC (Transaction Decomposition with Clustering) to alleviate the space problem in the decomposition method. The major feature of TDC is that it divides the original lattice into smaller pieces such that each portion can be solved by decomposition method independently. Without storing huge information in memory, the TDC method is able to discover frequent itemsets efficiently. We present experimental results on comparing the proposed method with existing algorithms. The results show that TDC outperforms other approaches for many cases. Even in the cases where the data set is huge or the user-specified minimum support is low, the TDC method still has the best performance. The advantage that TDC enjoys in terms of performance makes it a useful technique for discovering frequent itemsets in a very large database.

Keywords: Decomposition Method, Frequent itemset mining, Top-Down Mining

I. INTRODUCTION

In recent years, data mining has attracted lots of attention in the database community. One important reason is the huge database that is far from the ability of human to analyze. In general, the data mining technique can be divided into several categories: association rules, classification, clustering, etc. Association rule mining techniques discover the ways in which objects are associated in a database. The classification techniques learn a model from the training data set and predict the class label of unseen data based on the model. The clustering techniques partition a data set into clusters whose elements share common traits.

This research was partially supported by National Science Council, Taiwan, under contract no. NSC97-2221-E-005-083.

I-En Liao is a professor in the Department of Computer Science and Engineering at National Chung-Hsing University, Taichung, 402 Taiwan. (corresponding author to provide phone: +886-4-2284-0497 ext 705; fax: +886-4-22853869; e-mail:ieliao@nchu.edu.tw)

K. C. Lin is currently a Ph.D. candidate of the Department of Computer Science and Engineering at National Chung-Hsing University. (e-mail: phd9212@cs.nchu.edu.tw)

H. B. Chen is a master student of the Department of Computer Science and Engineering at National Chung-Hsing University. (e-mail: s9656008@cs.nchu.edu.tw).

Association rules mining is one the most important mining techniques in the data mining field. It is useful for discovering relationships among the different items. For example, given a database of sales transactions, it would be important to derive all associations among items such that the occurrence of one item of a transaction will imply the occurrence of another item in the same transaction. Thus, the information of customer's buying behavior can be analyzed, and the quality of marketing strategy can be also improved.

The problem of association rules mining can be formally stated as follows. Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of items and D be a multiset of transactions, where each transaction T contains a set of items in I . We call a subset $X \subseteq I$ an itemset and call X a k -itemset if X contains k items. The support of itemset X , denoted as $sup(X)$, is the fraction of transaction in D that contains all items in X . If $sup(X)$ is greater than or equal to the $minsup$ (user-specified minimum support), we call X a frequent itemset. An association rule is a conditional implication among itemsets, $X \Rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$, $X \neq \phi$, $Y \neq \phi$, and $X \cap Y = \phi$. The confidence of the association rule, given as $sup(X \cup Y) / sup(X)$, is the conditional probability among X and Y such that the appearance of X in T will imply the appearance of Y in T . Discovering all frequent itemsets from a database and forming the desired association rules among the frequent itemsets in a straightforward manner are called the problem of association rules mining.

Various approaches have been proposed to discover association rules. However, these approaches may incur problems related to either performance or space. Recently, we proposed a new mining technique, called Top-Down Mining (TDM) [8], for mining frequent itemsets. The major feature of TDM is that it examines database in a top-down manner and employs a transaction decomposition method to verify itemsets' supports. Simulation results showed that TDM has better performance in mining large database. Nevertheless, TDM incurs a space problem when it generates too many sub-patterns during the decomposition process. The major motivation of our research is to overcome this drawback and to develop an effective frequent itemset mining method.

In this paper, we proposed a new method, called TDC (Transaction Decomposition with Clustering), that combines the transaction decomposition method and the clustering technique to discover frequent itemsets. TDC divides the original lattice into smaller pieces such that each portion can be solved by pattern decomposition method independently. Experimental results show that this new algorithm has very short execution time and its performance is relatively independent of the minimum support threshold.

This paper is organized as follows. Section 2 presents the related work, and Section 3 defines the problem to be solved. Section 4 describes the details of the TDC method and Section 5 shows our experimental results. Finally, Section 6 concludes this paper.

II. RELATED WORK

Since Agrawal [1-2] introduced the frequent itemset mining problem, several algorithms for solving the problem have been proposed [1-7, 9-15]. The Apriori algorithm [1-2] utilizes a bottom-up strategy to discover frequent itemsets from a database. It uses a candidate generation method such that the frequent k -itemsets at one level can be used to construct candidate $(k+1)$ -itemsets at the next level. However, it has to scan the database many times for verifying frequent itemsets. DHP [11] improves the performance of Apriori. It uses a hash table to reduce the number of candidate 2-itemsets and employs a database trimming technique to lower the costs of database scanning. Pincer-Search [9] combines both the top-down and bottom-up searches. However, the number of database scans of Pincer-Search may be the same as that of Apriori in the worst case.

FP-Growth [6-7] is a different approach from the Apriori algorithm. Without candidate generation, the FP-Growth method uses a pattern growth approach to find the frequent itemsets. It first scans database twice and generates a FP-tree for storing the frequency information of the transaction database. By employing a divide-and-conquer method on the FP-tree, the support of each itemset is determined. Nevertheless, the recursive mining may decrease the whole mining performance. Thus, several approaches are proposed for improving the performance of FP-Growth [5, 12].

Combining clustering techniques with association-rule methods has been widely discussed recently. CDAR [14] partitions a database into several clusters by transaction length and employs a top-down manner to mine frequent itemsets. If an itemset X is frequent, both X and all its subsets will be output and then removed from the clusters because all of its subsets must be also frequent. Thus, CDAR improves the performance by decreasing the members in clusters. MaxClique [15] separates itemsets into various clusters alphabetically, and a sub-lattice is constructed for each cluster.

By traversing these sub-lattices individually, MaxClique is very efficient to discover frequent itemsets. Nevertheless, skewedness is a potential problem with the clustering step in MaxClique.

Recently, we proposed a hierarchical mining technique called TDM [8] to discover frequent patterns. The major feature of this technique is to examine database transactions in a top-down manner, resulting in lowering the I/O cost in counting patterns from a large database. However, the decomposition method may incur a space problem when it generates too many sub-patterns during the decomposition process.

To overcome the drawback of transaction decomposition method, this paper proposes a new algorithm, namely TDC (Transaction Decomposition with Clustering), which combines decomposition method with clustering to improve the throughput of frequent itemset mining.

III. PROBLEM DESCRIPTION

TDM can count the support of itemsets efficiently based on its transaction decomposition method. All of the subsets of a transaction T form a full lattice. The full lattice is partitioned into layers, and each itemset at the *layer* k has the length of k . In the decomposition method, each parent propagates its support to all of its children. The descendants in lower layers may have larger support than their real one due to multiple inheritances from the transactions. Thus, equation (1) is adopted in decomposition method to get the correct support of a descendant.

Assume that $T = \{t_1, t_2, \dots, t_n\}$ is a set of transactions in which each transaction has m items. Let x_k represent an k -itemset that is contained in each transaction in T . Each itemset x_k in the layer k will have $(m-k)$ parents in one transaction. Let P_{ij} represent the j th parent of x_k in the transaction t_i ($1 \leq k \leq i-1$).

$$sup(x_k) = \begin{cases} |T| & , if (m-k) = 1 \\ (\sum_{i=1}^n \sum_{j=1}^{m-k} sup(P_{ij})) / (m-k) & , if (m-k) > 1 \end{cases} \quad (1)$$

Figure 1 displays the procedure of counting an itemset in decomposition method. For an itemset I , the *Support_list* records its accumulated supports inherited from I 's ancestors in various layers.

Let us consider the database in TABLE I, which comprises four transactions: $\{(b, c, d, g, e), (a, b, c, g), (a, c, g)$ and $(a, b, c)\}$. Suppose that the support of each transaction is 1.

TABLE I
AN EXAMPLE OF TRANSACTION DATABASE

TID	Transaction	Layer
100	(b, c, d, g, e)	5
200	(a, b, c, g)	4
300	(a, c, g)	3
400	(a, b, c)	3

By using equation (1), $sup(ag)$ is derived as follows. We count $sup(ag)$ in two steps as it includes two portions, propagated from transactions in layer 4 and layer 3 (i.e., $\{abcg, acg\}$). First, since $\{abcg\}$ is in layer 4 and $\{ag\}$ is in layer 2, we start the calculation with $i=4$ and $k=2$. According to equation (1), the portion of $sup(ag)$ from $\{abcg\}$

$$sup(ag) = \left(\sum_{i=1}^1 \sum_{j=1}^{4-2} sup(P_{ij}) \right) / (4-2)$$

$$= (sup(acg) + sup(abg)) / (4-2),$$

$$sup(acg) = |T| = sup(abcg) = 1, \text{ and}$$

$$sup(abg) = |T| = sup(abcg) = 1,$$

where $\{acg\}$ and $\{abg\}$ are parents of $\{ag\}$. Thus, we obtain $sup(ag)$ propagated from $\{abcg\}$ as follows.

$$sup(ag) = (1+1)/(4-2) = 1.$$

Function *Itemset-count*

Input: I : an itemset to be counted; k : the layer of I ; PS : the set of all itemsets in layer $(k+1)$;

Output: $sup(I)$: the support of I ;

Method:

1. create an empty *Support_list* for I ;
2. **for** all parents p of $I \in PS$ **do**
3. **for** all entries $(x: y)$ in p 's *Support_list* **do**
4. **if** there exists an entry $(x: y')$ in I 's *Support_list* **then**
5. add y to y' in $(x: y')$
6. **else** insert the entry $(x: y)$ into I 's *Support_list*;
7. **end**
8. **end**
9. $sup(I) := 0$;
10. **for** all entries $(x: y)$ in I 's *Support_list* **do**
11. $sup(I) := sup(I) + (y/(x-k))$;
12. **end**
13. **if** I is also a transaction **then**
14. insert an entry $(k : \text{the support of the transaction } I)$ into the *Support_list*;
15. $sup(I) := sup(I) + (\text{the support of the transaction } I)$;
16. **return** $sup(I)$;

Figure 1. The Itemset-count procedure

Then, we count the portion of $sup(ag)$ propagated from $\{acg\}$. We set i to 3 and k to 2 in equation (1), since $\{acg\}$ is in layer 3 and $\{ag\}$ is in layer 2. By equation (1), we obtain this portion of $sup(ag)$ as follows.

$$sup(ag) = sup(T) = sup(acg) = 1$$

Finally, by summing up these supports from $\{abcg\}$ and $\{acg\}$, $sup(ag)$ is calculated (i.e., $1+1=2$).

However, the method may incur a space problem in pattern decomposition. It may generate too many patterns while decomposing the transactions from the database. For example, suppose that the length of a transaction T is n . The maximum number of decomposing T into subsets will be $(C(n, n/2))$. If decomposition problem can be improved, the transaction decomposition method will be a good method for handling complicated mining task.

In this paper, we combine a clustering method with decomposition method, and propose an efficient algorithm, called TDC, to discover frequent itemsets.

IV. TDC METHOD (TRANSACTION DECOMPOSITION WITH CLUSTERING)

TDC adopts two steps, itemset clustering and itemset decomposition, for performing the mining process. The itemset clustering technique, shown in Section 4.1, partitions a database into several small projected databases, which can be solved in memory. The TDC method, shown in Section 4.2, discovers frequent itemsets by combining the previous step and transaction decomposition method.

A. Itemset Clustering Technique

The TDM algorithm uses transaction decomposition method to calculate the support of an itemset. However, decomposing all itemsets from a transaction is not effective. For example, if there are n items in a transaction T , then the number of itemset decomposition from T will be $2^n - 1$ in the worst case. In fact, decomposing each itemset from T is not necessary because we only have to decompose subsets of candidate frequent itemsets from T . Thus, a useful clustering technique for finding candidate frequent itemsets in order to reduce the cost of decomposition method becomes important in TDC.

We modify Zaki's algorithm for generating maximal cliques and show the modified method in Figure 2. $[i].CoveringSet$ denotes a set of 1-item in frequent 2-itemsets with item i . $[i].CliqList$ denotes a list of cliques with item i . Each item's identifier in $[i].CoveringSet$ and in cliques of $[i].CliqList$ is not lower than that of i . Line 4 considers the items in the covering set when generating maximal cliques.

Line 5 generates the maximal cliques for items in the covering set for each class. Before inserting a new clique, all duplicates or subsets are eliminated (Line 7 - Line 8). Line 9 - Line 13 guarantees that each clique in *MaxCliqList* is a maximal clique.

<p>Procedure: <i>Max_Clique_Generation</i> Parameter: N_{max}: the maximum item's identifier; N_{min}: the minimum item's identifier; Input: F_2: the set of frequent 2-itemsets; Output: <i>MaxCliqList</i>: a list of maximal cliques; Method: 1. $MaxCliqList = \phi$; 2. for (i's identifier = N_{max}; i's identifier $\geq N_{min}$; $i--$) do 3. $[i].CliqList = \phi$; 4. for each $x \in [i].CoveringSet$ do 5. for each $cliq \in [x].CliqList$ do 6. $M = cliq \cap [i]$; 7. if ($M \neq \phi$) and ($\{\{i\} \cup M\} \neq$ any subset of cliques in $[i].CliqList$) then 8. add $\{\{i\} \cup M\}$ into $[i].CliqList$ 9. for each $cliq' \in [i].CliqList$ do 10. for each $cliq'' \in MaxCliqList$ do 11. if $cliq'' \subseteq cliq'$ then 12. remove $cliq''$ from <i>MaxCliqList</i>; 13. add $cliq'$ to <i>MaxCliqList</i>;</p>
--

Figure 2. Maximal clique generation procedure

We use an example to show that how we generate maximal cliques. Let F_2 be {13, 14, 15, 16, 23, 25, 34, 35, 45}. Based on F_2 , four covering sets {[1]:{3, 4, 5, 6}, [2]:{3, 5}, [3]:{4, 5}, [4]:{5}} are generated. For covering set [3], the intersection of covering set [3] and [4].*CliqList* (i.e., {3, 4, 5} \cap {4, 5}) is {4, 5}. Next, a *cliq*[3, 4, 5] (i.e., {3} \cup {4, 5}) is generated and inserted into [3].*CliqList* and *MaxCliqList*. Performing the same process on [2] and [1], *cliq*[2, 3, 5] and *cliq*[1, 3, 4, 5] are added into *MaxCliqList*. In addition, *cliq*[3, 4, 5] is removed from *MaxCliqList* since it is a subset of *cliq*[1, 3, 4, 5]. Therefore, two maximal cliques (*cliq*[1, 3, 4, 5] and *cliq*[2, 3, 5]) are available in this example.

Algorithm: TDC

Input: *DB*: a transactional database; min_sup : the minimum support.

Output: F : the set of frequent itemsets being mined

Parameters: L_k : the set of itemsets in layer k ;

Method:

1. Scan DB twice and generate frequent 2-itemsets F_2 ; 2. $F := F_2$; 3. if $F_2 \neq \phi$ then 4. $MaxCliqList = Max_Clique_Generation(F_2)$; 5. for each $MaxCliq \in MaxCliqList$ 6. project <i>DB</i> to <i>DB'</i> by items in <i>MaxCliq</i> and partition transactions in <i>DB'</i> into various layers according to the number of items in each transaction; 7. let t be the largest number of items in transactions; 8. $k := t$; 9. $L_k := \{c \in L_k \mid c \notin \text{a descendant of any frequent itemsets in } F\}$; 10. for each itemset $I \in L_k$ do begin 11. if <i>Itemset-count</i> (I, k, L_{k+1}) $\geq min_sup$ then 12. $F := F \cup \{I\}$; 13. $L_k := L_k - \{I\}$; 14. else 15. for each child I' of I do begin 16. $L_{k-1} := L_{k-1} \cup \{I'\}$; 17. end 18. end 19. $k := (k-1)$; 20. if $k > 2$ then go to line 8;

Figure 3. The TDC algorithm

B. The TDC Algorithm

Figure 3 shows the whole TDC algorithm. Line 1 scans database twice in order to get the set of frequent 2-itemsets F_2 . Line 4 generates all maximal cliques from F_2 . Based on maximal cliques, TDC partitions the database into a set of projected databases, and then for each projected database, perform its corresponding decomposition method (Line 5 - Line 20). Line 9 excludes those descendants of any frequent itemsets in F . Line 11 calls the *Itemset-count* method to count itemset's support. Line 16 adds the children of I to the set of itemsets in layer ($k-1$) for further mining in the next iteration.

As an example, we use TDC to mine frequent itemsets from *DB* in Table 1 and let the min_sup be 2. By scanning database twice, five frequent 2-itemsets $F_2 = \{ab, ac, bc, ag, cg\}$ are generated. The procedure *Max_Clique_Generation* is used for generating two maximal cliques (i.e., *cliq*[*abc*] and *cliq*[*acg*]). Based on the two cliques, TDC builds projected databases correspondingly. The two projected databases, D_1 and D_2 , are shown in Figure 4. TDC then employs the

decomposition method to discover frequent itemsets for each of projected databases.

By using equation (1), $sup(ac)$ of D_1 is derived as follows. Starting with calculation with $i=3$ and $k=2$, the portion of $sup(ac)$ from $\{abc\}$

$$sup(ac) = |T| = sup(abc) = 2,$$

where $\{abc\}$ is a parent of $\{ac\}$.

In addition, by accumulating the support of transaction (a, c) (i.e., 1) and the inherited support from $\{abc\}$, $sup(ac)$ is derived [i.e., $1+2=3$].

TID	Transaction	Layer	TID	Transaction	Layer
100	(b, c)	2	100	(c, g)	2
200	(a, b, c)	3	200	(a, c, g)	3
300	(a, c)	2	300	(a, c, g)	3
400	(a, b, c)	3	400	(a, c)	2

(a) The projected database D_1 for $cliq[abc]$ (b) The projected database D_2 for $cliq[acg]$

Figure 4. Two new databases projected from the sample database in Table 1

V. EXPERIMENTAL RESULTS

To understand the performance of TDC, we implemented four algorithms, Apriori [1-2], Pincer-Search [9], TDM[8] and TDC, and use them to mine frequent itemsets from various databases. The experiments were performed on an AMD Althlon(tm) 64 X2 Dual Core Processor 4600+ 2.41GHz PC with 1.5GB memory running the Microsoft Windows-XP. We used the C++ language to code the four algorithms and generated the test databases using the data generator program of IBM Data Mining Research Group [16]. Each database is characterized by four or five parameters specified in its name. TABLE II summarizes the parameters used in our experiments.

In particular, we only present the performance comparison among TDM and the other algorithms in the first tested data set. For the other three data sets, TDM incurs a space problem while decomposing itemsets from transactions. Thus, we compare the throughput among Apriori, Pincer-Search and TDC on those three data set group.

TABLE II

THE PARAMETERS OF TEST DATABASES

D	Number of transactions per database
T	Average number of items per transaction
I	Average length of maximum frequent patterns
N	Number of items per database
L	Number of patterns

First, we tested the scalability with various numbers of items. The $minsup$ was set to 20% in the database T10I4D100KL50 with various numbers of items. In most of cases, the decomposition method employed in various projected database makes TDC the second best algorithm. When the number of items is 500, TDM incurs the space problem because it generates too many sub-itemsets. Compared with Apriori and Pincer-Search, TDC achieves better performance by eliminating repeated database scans. Figure 5 shows the execution results.

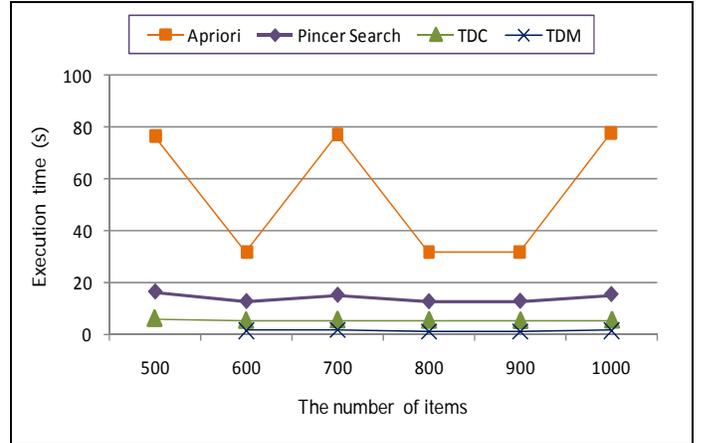


Figure 5. Performance under various numbers of items.

The running time of the three algorithms on T10I6N1000D100K with respect to various minimum supports is shown in Figure 6. The performance of TDC depends on the number of maximal cliques and the maximum number of items of these cliques. As the minimum support was set to 0.25%, 172 maximal cliques were generated by TDC and the maximum number of the maximum cliques was 4. The efficiency of TDC is superior to other algorithms in this experiment. To know whether a candidate pattern exists in a transaction, Apriori and Pincer-Search need to verify all transactions in each database scan. When the minimum support decreases, the cost of repeatedly scanning transactions makes the performance of Apriori and Pincer-Search become low.

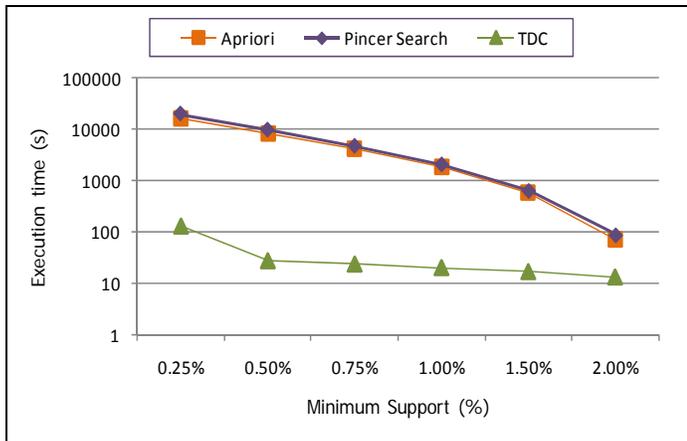


Figure 6. Performance under various minimum support on T10I6D100KN1000

Figure 7 shows the running time of the three algorithms on T10I6D100KN1000L50 with respect to various minimum supports. TDC outperforms Apriori and Pincer-Search in the data set. The length of maximal frequent itemsets in the data set is 9 and TDC generates 22 maximal cliques in the data set. As the minimum support was set to 10%, the number of database scans of Apriori is equal to that of Pincer Search, resulting in almost the same performance.

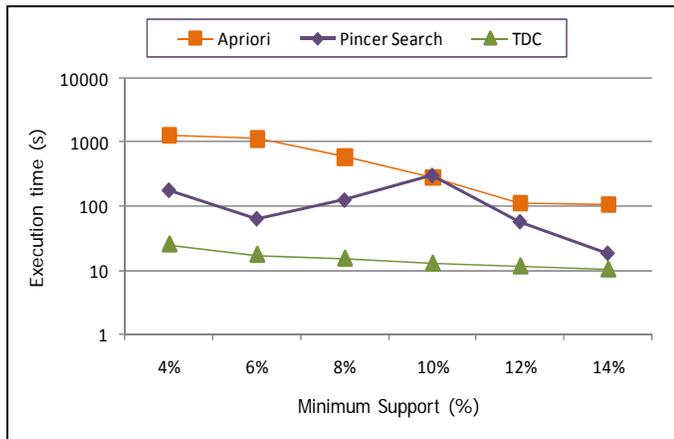


Figure 7. Performance under various minimum support on T10I6D100KN1000L50

For the fourth database group, the scalability with the number of transactions was tested. The *minsup* was set to 1.5% in the databases T10I4N1000 with various numbers of transactions, from 10K to 1000K. We illustrate the execution results in Figure 8. When the size of database increases, the cost of scanning database makes the performance of Apriori and that of Pincer-Search get worse. In contrast, even if the database size becomes large, TDC can mine frequent itemsets efficiently.

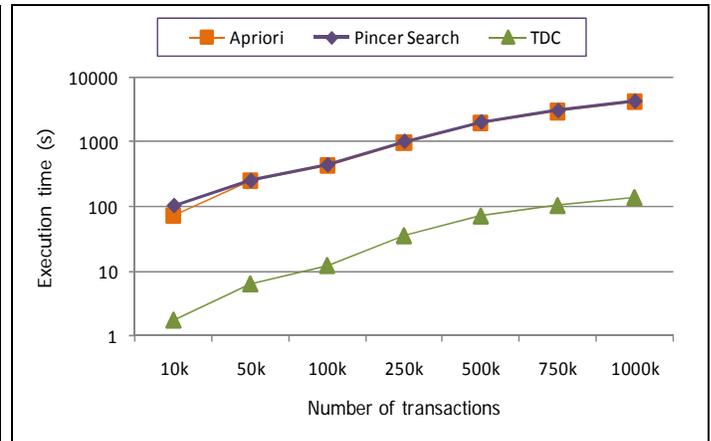


Figure 8. Performance under various numbers of transactions

From the experimental results, TDC is an efficient algorithm for mining frequent itemsets in the database. Even if the size of the database becomes large, TDC is also very efficient. Therefore, TDC is suitable for different types of high performance applications.

VI. CONCLUSION

By contrast with the traditional approaches of mining frequent itemsets, we proposed an efficient method called TDC algorithm. TDC employs a clustering technique to alleviate the space problem in the transaction decomposition method. It partitions the original database into smaller projected databases such that each portion can be solved by decomposition method independently. Experimental results show that the proposed TDC algorithm outperforms its counterpart algorithms in execution time.

REFERENCES

- [1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," *Proc. of the ACM SIGMOD Conf. on Management of Data*, pp. 207-216, 1993.
- [2] R. Agrawal and R. Srikant, "Fast Algorithm for Mining Association Rules in Large Databases," *Proc. of 20th VLDB Conf.*, pp. 487-499, 1994.
- [3] D. Burdick, M. Calimlim, J. Flannick, J. Gehrke, and T. Yiu, "MAFIA: A Maximal Frequent Itemset Algorithm," *IEEE Transactions on Knowledge and Data Engineering*, Vol.17, No.11, pp.1490-1504, 2005.
- [4] A. Ghoting, G. Buehrer, S. Parthasarathy, D. Kim, A. Nguyen, Y. Chen, and P. Dubey, "Cache-Conscious Frequent Pattern Mining on Modern and Emerging Processors," *The International Journal on Very Large Data Bases*, Vol. 16, Issue 1, pp. 77-96, 2007.

- [5] Gösta Grahne and Jianfei Zhu, "Fast Algorithms for Frequent Itemset Mining Using FP-trees," *IEEE Transactions on Knowledge and Data Engineering*, Vol.17, No.10, pp.1347-1362, 2005.
- [6] Jiawei Han, Jian Pei, and Yiwen Yin, "Mining Frequent Patterns without Candidate Generation," *Proc. of the ACM-SIGMOD Conf. Management of Data*, pp. 1-12, 2000.
- [7] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," *Data Mining and Knowledge Discovery*, Vol. 8, Issue 1, pp. 53-87, 2004.
- [8] Kuen-Fang Jea, Ke-Chung Lin, I-En Liao, "Mining Hybrid Sequential Patterns by Hierarchical Mining Technique," *International Journal of Innovative Computing Information and Control*. (Accepted)
- [9] D. Lin and Z. M. Kedem, "Pincer-Search: A New Algorithm for Discovering the Maximum Frequent Set," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 14, No. 3, 2002, pp.553-566.
- [10] S. Orlando, C. Lucchese, P. Palmerini, R. Perego, and F. Silvestri, "kDCI: a Multi-Strategy Algorithm for Mining Frequent Sets," *Proc. of IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, 2003.
- [11] J.S. Park, M. Chen, and P.S. Yu, "Using a Hash-based Method with Transaction Trimming for Mining Association Rules," *IEEE Transactions on Knowledge and Data Engineering*, Vol.9, No.5, pp.813-825, 1997.
- [12] B. Racz, "nonordfp: An FP-growth variation without rebuilding the FP-tree," *Proc. of IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, 2004.
- [13] Mingjun Song and Sanguthevar Rajasekaran, "Finding Frequent Itemsets by Transaction Mapping," *Proceedings of the 2005 ACM symposium on Applied computing*, 2005, pp. 498-492.
- [14] Y.J. Tsay and Y.W. Chang-Chien, "An efficient cluster and decomposition algorithm for mining association rules," *Information Sciences*, 160, 2004, pp. 161-171.
- [15] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New Algorithms for Fast Discovery of Association Rules," *Proc. of 3rd Knowledge Discovery & Data Mining conf.*, pp. 283-286, 1997.
- [16] <http://researchweb.watson.ibm.com>

Linked list based high utility itemsets mining using pattern growth method

Fan Wu, Ya-Han Hu, and Kai-Chung Pai

Department of Management Information System, National Chung Cheng University, Chiayi, Taiwan, ROC

Abstract – *In the past, several researches in frequent pattern mining focused on relative frequent patterns and did not consider the utility of merchandise. Indeed, the property of profit or revenue is also meaningful to business. Methods on mining high utility itemsets, considering profit, do not have anti-monotone property. Considering this problem, Liu et al. proposed the TwoPhase algorithm with anti-monotone property; but, it is based on generation-and-test approach. On the other hand, Erwin et al. proposed CTU-Tree data structure and CTU-Mine algorithm to mine high utility itemsets; however, it is inefficient in traversal the CTU-Tree. In this paper, we develop the double layer index and linked lists method using pattern growth approach to enhance the performance of CTU-Mine. Moreover, the proposed method with the property of anti-monotone and lexicon order in each linked list can quickly find the specific pattern and utilize multi-processing approach to enhance the efficiency both for providing a real insight into business strategy.*

Keywords: Utility mining, frequent pattern., data mining

1 Introduction

Data mining has become more and more important in today's society. Many applications such as basket analysis for customer discrimination and cross sale assists decision maker to gain more competition strategies. The traditional association rule mining algorithm discovers association rule in the procedure of Apriori [1] which finds large itemsets through scanning database in a series of iterations. In each iteration, the algorithm computes the support from a large database for the sake of pruning the infrequent itemsets, and then discovering the large ones. Two main drawbacks exist in Apriori algorithm: (i) multiple scans of the database to compute the frequency of itemsets. (ii) a huge number of association rules being generated. In order to solve the two problems on Apriori, Han et al. proposed the FP-growth algorithm [2] which discovers the frequent patterns by generating the FP-tree which represents the frequency of each item in a transaction database as a binary value. In this way, the FP-growth algorithm only scan entire database twice, and generally outperform the Apriori algorithm.

However, the merchandises with high profit are often purchased rarely and Apriori and FP-growth algorithms only consider the quantity or frequency of items without considering the profit. Indeed, profit of the product is also significant for decision maker to identify real world problems. For instance, the frequency of selling computers may occur less than merchandises for livelihood, but the former will be more profitable than the later. Obviously, high utility itemsets, considering both quantity and profit, are more useful in usual application, and they do not have anti-monotone property of large itemsets which is mined from traditional Apriori method. Liu et al. [3] proposed an algorithm called *TwoPhase* based on candidate generation-and-test approach like Apriori, and they used a measure called *transaction weighted utility* (twu) which has anti-monotone property to prune the items without high twu. But twu may over-estimate real utility of the itemsets; further high twu itemsets pruning is required. Another research also extend Apriori algorithm for mining high utility itemset [4]. In order to solve the problem which is based on candidate generation-and-test approach, Erwin et al. proposed the efficient high utility algorithm [5] based on transaction weight utility method to prune the item without high twu, and then construct Compressed Transaction Utility tree (CTU-tree) to store all patterns with high twu measure. The way to construct CTU-tree is based on the FP-growth approach. However, it is inefficient in traversal CTU-Tree for the sake of discovering high utility patterns because CTU-Mine algorithm uses recursive function and it can not multi-process candidate pattern in figuring the high utility pattern in the period of traversing the tree. Moreover, the nodes in CTU-tree will leave some empty patterns and its corresponding empty arrays, and waste disk storage on storing the tree.

In order to solve the problems of not capable of multi-processing candidate pattern and empty arrays, in this paper we propose a high utility itemset mining method based on FP-growth. Unlike the CTU-Tree, we store the high twu pattern in form of linked lists which performs linear search and don't have empty arrays in our data structure. Using our method will mine high twu pattern in specific linked list and not entire ones. In each linked list, we use double layer indexes to efficient find the pattern. One is the right-most sorted item, or postfix item, of each pattern; the other is the length of pattern (k-itemset). The first index is called item index, and second index is called pattern index. At the time inserting the node in specific linked list, we will check the value of pattern to

sorting the pattern in lexicon order. Hence, we won't check complete linked list when we want to find the pattern in a linked list, and save the traversal time in finding high utility patterns. Moreover, when traversing high utility pattern in our linked list, we just need to traversal the linked lists which item index is the right-most sorted item and we can apply multi-processing in traversal specific linked list. Through these mechanisms, we can have better performance at the time when we traversal high utility pattern compared with CTU-Mine algorithm.

2 Related Work

In this section, we firstly introduce the notation and basic definition of the transaction database. In a transaction database, let $D = \{T_1, T_2, T_3, \dots, T_n\}$ be a transaction database. Each transaction T_i is composed by $\{i_1, i_2, i_3, \dots, i_m\}$ and contains a set of m items from $I = (i_1, i_2, i_3, \dots, i_r)$ where $m \leq r$; i.e. $T_i \subseteq I$. An itemset X with k items from I is called a k -itemset. A transaction T_i contains an itemset X if and only if $X \subseteq T_i$.

2.1 High utility itemset Mining

In each transaction (T_i), the purchased quantity of item i_p in transaction T_q is denoted as $o(i_p, T_q)$. External utility, denoted as $s(i_p)$, is the number (i.e. the profit of merchandise) in the utility table (see Fig. 1(b)). In each transaction, the utility of each item (i_p) is defined as $o(i_p, T_q) \times s(i_p)$. The itemset X is the subset of I , and the utility $u(X, T_i)$ of X in each transaction (T_q) is defined as:

$$u(X, T_q) = \sum_{i_p \in X} u(i_p, T_q) \quad (1)$$

In transaction database, the utility of all itemset, denoted as $u(X)$, is defined as:

$$u(X) = \sum_{T_i \in D \wedge X \subseteq T_i} u(X, T_i) \quad (2)$$

The destination of high utility itemset mining is to discover the itemset which is higher than user-defined min_utility . Liu et al. [3] proposed the concept of *Transaction Utility (tu)* and *Transaction Weighted Utility (twu)* to prune the itemset without high twu and measuring items with twu has downward closure property.

The transaction utility of a transaction called $tu(T_i)$ is sum of utility of all item in a transaction:

$$tu(T_i) = \sum_{i_p \in T_i} u(i_p, T_i) \quad (3)$$

The transaction Weight Utility of itemset X is called $twu(X)$ which is sum of the transaction utility (tu) of transactions containing X :

$$twu(X) = \sum_{T_i \in D \wedge X \subseteq T_i} tu(T_i) \quad (4)$$

For example, Fig. 1 (a) is a transaction database of a 3C market; Fig. 1 (b) is the utility table of transaction database.

As shown in Fig. 1 (a), column 2 to 7 are item counts of each item; the last column is the transaction utility of each transaction, denoted as $tu(X)$ which can be calculated by fomula 3. Sum of each $tu(X)$ in this example is 1118. If decision maker would like to find 10% of total transaction utility to find the high utility itemset, the min_utility will equal to $1118 \times 10\% = 111.8$

Item	A	B	C	D	E	F	tu(x)
T1	0	0	1	1	0	6	62
T2	0	2	2	0	0	0	240
T3	0	1	1	1	0	4	158
T4	1	0	0	1	1	0	45
T5	0	1	0	1	0	2	134
T6	0	0	1	1	1	2	59
T7	1	1	0	0	0	10	130
T8	0	1	1	0	0	0	120
T9	1	1	0	0	0	10	130
T10	0	0	2	0	0	0	40
total	3	7	8	5	2	34	1118

Item	Profit	
Desk	A	10
PC	B	100
Desk Lamp	C	20
Printer	D	30
Ink	E	5
Paper	F	2

(b) Utility table

(a) a transaction database

Fig. 1. The example of transaction database and its utility table

Besides, based on the formula 4, we can calculate the twu value of each item. For example, item A shows in T4, T7 and T9, and the values of tu are 45, 130 and 130 respectively. Hence, $twu(A) = 45 + 130 + 130 = 305$. Fig. 2 is the twu value of each item.

item	A	B	C	D	E	F
twu(X)	305	912	679	458	104	673

Fig. 2. The twu value of each item

Since we have known the min_utility is 111.8, we can know that $twu(E) = 104 < 111.8$. So item E belongs to low twu itemset, we will prune item E at this time.

2.2 Compressed Transaction Utility tree (CTU-tree)

Erwin et al. [5] proposed the *Compressed Transaction Utility tree* (CTU-tree) method which stores all high twu patterns and quantity of each pattern. Continuing the example in Fig. 1, CTU-tree shows in Fig. 3.

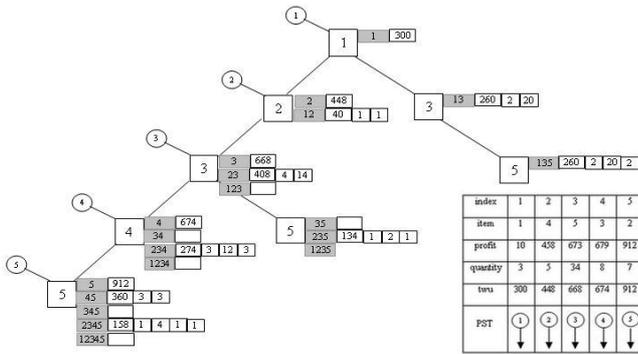


Fig.3. CTU-Tree and Item table

As shown in Fig. 3, the item table stores the high twu items sorted in ascending order of their twu values. Each item has its own index, profit, quantity, twu and pointer to the sub-tree. The shadowed part of each node is the pattern and the right of the shadowed part is the twu of each pattern and its corresponding quantity of each item. Each node stores the pattern with their corresponding prefix nodes, and each pattern has the linked to the array of twu value which is initialized to zero and the corresponding quantity of each item in a pattern. Notice that CTU-tree may produce the pattern with null twu and its corresponding array. When the item is numerous and not generation many pattern, a lot of null twu and its corresponding array will be generated.

After constructing the CTU-tree, the next step is to mine all high utility patterns by traversal all patterns and calculating the real utility of the patterns whose TWU is higher than user-defined *min_utility*. However, CTU-Mine is a recursive algorithm and the procedure to mine high utility patterns is hard to apply multi-processing in traversal time because each pattern need to wait for the former sub-tree scan over. It is not efficient in mining progress.

3 Our Method

In this section, we based on pattern growth method [2] to store high transaction weighted utility pattern in linked list without producing null patterns and its corresponding array like CTU-tree. Storing item in form of linked list will perform linear search and spread the pattern in specific linked list instead of recursive search only in a tree structure. The following three sections will introduce three phases of our method.

3.1 Phase I : Data preprocessing - prune low twu item

We discover the high transaction weighted utility pattern based on the method of Liu, et al. [3] with the following steps:

Step 1 Scan transaction database to get all 1-item counts.

Step 2 Based on the profit of each item in utility table, we calculate the utility of each transaction (transaction utility, tu) and the transaction weighted utility (twu) value of each item.

Step 3 Based on the user-defined threshold (i.e. 10%), we calculate the minimum utility, donated as *min_utility*, to be the threshold of pruning low twu item. The way to calculate the *min_utility* is the sum value of each transaction utility (tu) multiplied by the user-defined ratio. The item which is below the *min_utility* will be pruned.

Step 4 Each item will be sorted in ascending order by the twu value.

Continuing the example in section 2.1, we have done the steps 1 to 3. Hence, next we will sort the twu value in ascending order, and then construct the item table. The item table shows in Fig. 4 and consists four information: (a) item (b) profit (c) twu (d) reordered item index. In Fig. 4, we can get the following order: A, D, F, C, B, and the next phase we will use this order to encode our high utility linked list.

item	A	D	F	C	B
profit	10	30	2	20	100
twu(X)	305	458	673	679	912
reordered item index	1	2	3	4	5

Fig. 4. The item table

3.2 Phase II : Construction of high twu linked list

In phase I, we have scanned database one time and pruned the item without high twu. The outcome presents in Fig. 5 and we will use these two tables: Fig. 5(a) is the sorted transaction database which is sorted by twu value in ascending order and Fig. 5 (b) is the item table. In phase II, we will scan database one more time to construct the high utility linked list, and use double indexes to find the pattern efficiently: one is the right-most sorted item with high twu; the other is the length of pattern (k-itemset). The first index is called item index, and the other index is called pattern index. The steps are described in the following:

Step 1 Construct the nodes which length is one and no information in each node.

Step 2 Scan transaction database to update our linked lists.

Step 3 Update the linked lists until the last transaction.

Item \ T _i	A	D	F	C	B	tu(x)
T ₁	0	1	6	1	0	62
T ₂	0	0	0	2	2	240
T ₃	0	1	4	1	1	158
T ₄	1	1	0	0	0	40
T ₅	0	1	2	0	1	134
T ₆	0	1	2	1	0	54
T ₇	1	0	10	0	1	130
T ₈	0	0	0	1	1	120
T ₉	1	0	10	0	1	130
T ₁₀	0	0	0	2	0	40
total	3	5	34	8	7	

(a) sorted transaction database

item	A	D	F	C	B
profit	10	30	2	20	100
twu(X)	305	458	673	679	912
reordered item index	1	2	3	4	5

(b) the item table

Fig. 5. the data for constructing the high twu linked list

In step 1, α is number of rest item after the phase I. Firstly we construct α linked list, and each linked list has its own index from 1-1 to α -1. The first index is the last item of each pattern, and the second index is the length of pattern (k-itemset). Each pattern will store in a node of linked list with the corresponding index. For example, if the linked list index is 2-1, it represents that the linked list stores the patterns which reordered item index is 2 and pattern length is 1 (1-itemset).

In step 2, we will transform the information of each transaction into specific node. When each pattern insert to the corresponding linked list, some other information also fill into each node. One is the value of tu, another is the quantity of each item in the pattern. Notice that we will use the reordered item index to replace the item of each pattern. Since the items are sorted by twu in phase I, the patterns which are ready to insert in each transaction are described in the following: (i) store the 1-item and its information (tu, quantity) in the corresponding linked list. (ii) scan the pattern from left to right item then combine the k-pattern ($k \geq 2$), and also store the information of each pattern. For example, In Fig. 5 (a), the item and quantity in T1 are D:1, C:1, F:6. Then we can get 5 patterns in T1: D, C, F, DC, and DCF. The corresponding index is 1, 2, 3, 12 and 123, and then judge the number of each pattern in order to insert the pattern and its corresponding tu and the quantity into specific linked list. Hence the 5 patterns in T1 are inserted into the linked list which their index is 1-1, 2-1, 3-1, 2-2, and 3-3. When the index of linked list does not exists, create new linked list with corresponding index. It is worth noting that we will accumulate tu and quantity value when the pattern has already

existed. On the other hand, when the pattern does not exist in the linked list, we will add new pattern with tu and the quantity of each item in the linked list. Notice that when inserting a node into a specific linked list, we will check the reordered item index of each pattern and insert the pattern in the right position. So the node of each linked list will be sorted in lexicon order. Lexicon order means that we will check the pattern value and the former pattern will always smaller than latter one. Repeat the procedure in step 2 until the last transaction in the database.

Fig. 6 represents the procedure in constructing the high utility linked list before scanning the T3. At the beginning we construct the 1-item linked list from item index 1 to 5 because of remaining 5 item in the item table (See Fig. 6 (a)). The next we will scan the transaction one by one. T1 have 5 patterns to

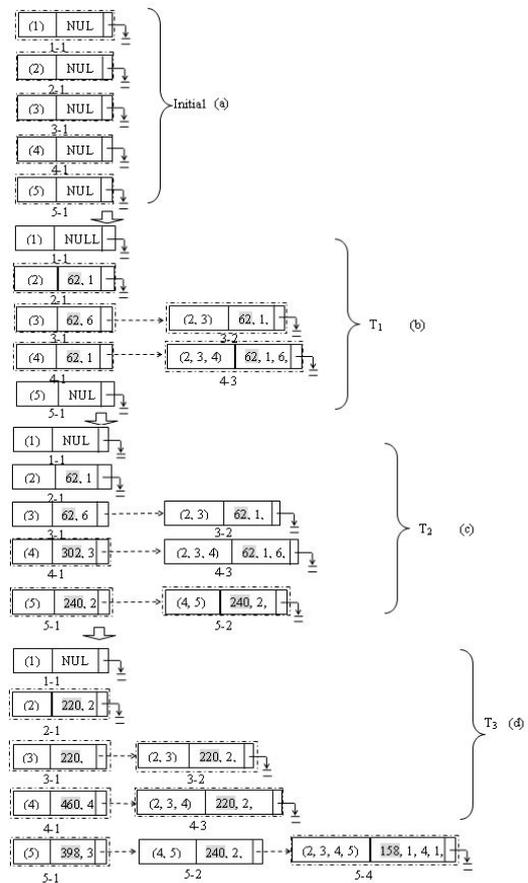


Fig. 6. Process of generating high utility linked list before transaction

insert into linked lists: 2, 3, 4, 23 and 234. We will take the right-most item as item index and the length of pattern to as pattern index, and insert the node with its pattern and corresponding information into the linked list (See Fig. 6 (b)). When inserting a new node in a specific linked list, we also check the value of index to sort each node in lexicon order. We will also accumulate the node with its twu and quantity when the value of pattern is the same. The same procedure is

also performed in Fig. 6 (c) and Fig. 6 (d) which represent transaction 2 and 3, respectively.

As shown in Fig. 6, the frames with dotted line are the place which is updated in each transaction. The dotted arrows mean the linked list points out the different linked list with the same right-most item but different pattern length, and the solid ones mean that the connection of nodes in a linked list. Notice that in Fig. 6 it doesn't exist any solid arrow because there is only one node in each linked list. Scanning from T1 to T3 can get the following pattern: 1, 2, 3, 23, 4, 234, 5, 45 and 2345. The corresponding accumulated twu are 220, 220, 220, 460, 220, 398, 240 and 158, separately. Every pattern will also store the quantity of each item.

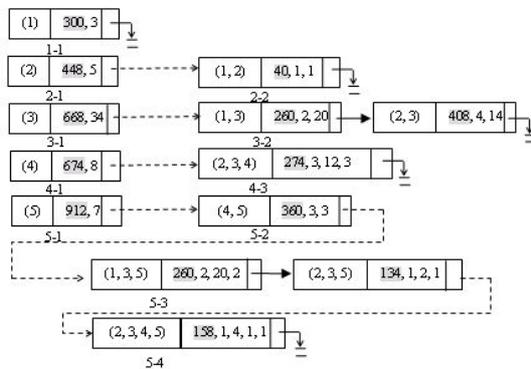


Fig. 7. The result of high utility linked lists

Fig. 7 represents the high utility linked lists after scanning all transaction in the transaction database. The solid arrows are shown in linked list index 3-2 and 5-3, and represent that there is more than one node in the linked list.

3.3 Phase III : Discover high utility patterns

The high utility linked list which is shown in Fig. 7 stores all high utility patterns measured by transaction weighted utility of each item, and every pattern represents that the item will purchase together. However, using twu to measure the high utility pattern may over-estimate the pattern because the twu itself is an over-estimation of real utility itemset value. So further pruning of high twu patterns are necessary for mining high utility ones. Hence, we will calculate the real utility of the pattern which is larger than $min_utility$. The steps are listed in the following:

Step 1 set a queue to store the patterns which is larger than $min_utility$. Then insert the pattern which length is one and generating the combination of 2-pattern because twu of each item will be larger than $min_utility$. Finally, delete all 1-pattern from the queue.

Step 2 take the fist pattern in the queue and check the pattern in the linked list. If its twu value is larger than

$min_utility$, first calculate real utility and insert new candidate patterns into the queue. If the real utility is larger than $min_utility$, output the pattern to the high utility patterns. Second, if the pattern length is k , insert the patterns which length is $(k+1)$ to the queue. For instance, if pattern (1,2) is larger than $min_utility$, we will insert pattern (1,2,3), (1,2,4), ..., (1,2, α) into the queue. On the other hand, if its twu value is not larger than $min_utility$, we don't do anything. At the end of this step, delete the first pattern at the beginning of the queue.

Step 3 continue the procedure in step 2 until the queue is empty.

Continuing the example, in step 1 we set a queue to store the patterns. Insert 1-pattern into the queue from 1 to 5, and then insert the combination of the patterns which length is 2. We will insert the following patterns: (1,2), (1,3), (1,4), (1,5), (2,3), (2,4), (2,5), (3,4), (3,5), (4,5). Then calculate real utility of each 1-pattern, for example, pattern 1 represents item A and $twu(A) = 300 > min_utility$. Since pattern 1 is larger than $min_utility$, calculate the real utility of pattern A which equals $profit(A) \times quantity(A) = 10 \times 3 = 30 < min_utility$. Hence pattern A is not high utility item. Finally, delete 1-pattern in the queue.

In step 2, we take the first pattern in the queue. The first pattern will be (1, 2). We only find the pattern (1, 2) in linked list index 2-2 and its twu value is $40 < min_utility$ 111.8, so next we won't calculate real utility of pattern (1, 2) and delete (1, 2) from the queue. The next pattern in the queue is pattern (1, 3), and only one pattern in index 3-2 contains (1, 3). And its twu is $260 > min_utility$. So we will calculate the real utility of pattern (1, 3) = $60 (10 \times 2 + 2 \times 20) < min_utility$. Hence pattern (1, 3) is not high utility pattern. Since the twu of pattern (1, 3) is larger than $min_utility$, we will insert pattern (1, 3, 4), (1, 3, 5) into the queue. This procedure will be continuing until the queue is empty. The final result of the high utility itemset is (2), (4), (5), (1, 5), (2, 3), (2, 4), (2, 5), (3, 5), (4, 5), (1, 3, 5), (2, 3, 4), (2, 3, 5), (2, 4, 5), (3, 4, 5), (2, 3, 4, 5).

Notice that we just need to traversal the linked lists which item index is the right-most item of each pattern and we can apply multi-processing in traversal specific linked list. For example, we can check pattern (1, 2) and (1, 3) at the same time.

Through our method, we will get all high utility patterns. The result can be used by the decision makers, and let them know what patterns are the most profitable. This will provide the decision makers to make their selling strategy.

3.4 Our proposed algorithm

The inputs data and output data in our algorithm are shown in Table 1, and the symbol list is shown in Table 2.

TABLE I
INPUTS AND OUTPUTS DATA IN OUR METHOD

Inputs	transaction database (D), profit of each item ($profit_{item}$), a minimum utility of w% of total transaction utility.
Outputs	High utility itemsets

TABLE II
SYMBOL LISTS

Symbol	Explanation
<i>Transaction database (D)</i>	The database including all transactions.
<i>profit_{item}</i>	profit of each item
<i>w</i>	a minimum utility of w% of total transaction utility
<i>T_q</i>	the transactions q in T _{DB}
<i>i_p</i>	the item i _p
<i>min_utility</i>	Minimum utility which is calculated by total transaction utility \times w%
<i>tu</i>	Transaction utility
<i>twu</i>	Total weighted utility

Our proposed algorithm (D, $profit_{item}$, w)

```
// First scan
// Step I : data pre-processing - prune low utility item
1. for each transaction  $T_q \in D$ 
  1.1 for each item  $i_p \in T_q$ 
    1.1.1 if  $i \in$  utility table
      calculate (the quantity of each  $i_p$ )  $\times$  (the
      profit of each  $i_p$ )
    end if
  end for
  1.2 transaction utility = sum of utility of each  $T_q$ 
  end for
2. sum of each  $tu(T_q)$  for total transaction utility
3.  $min\_utility = total\ transaction\ utility \times w\%$ 
4. calculate the total weighted utility( $twu$ ) for each  $i_p$ 
5. sort transaction database in  $twu$  by ascending order
6. if the  $twu$  of each  $i_p < min\_utility$ 
  prune the item
end if
// Step II : Encode the high utility linked list
// second scan
7. construct the item table with remaining items and their
profit and  $twu(X)$ 
8. initialize the linked list of the remaining item
9. scan each transaction and the corresponding pattern and
their pattern quantity
10. insert the pattern into the corresponding linked list in
lexicon order
11. if the pattern exists in the linked list
  accumulate the exist pattern and its  $tu$  and quantity
else
  add new pattern and its  $tu$  and quantity to the linked
  list
end if
// Step III: Discover high utility pattern
```

```
12. set a queue
13. insert pattern which length is one and generating the
combination of 2-pattern
14. delete all 1-pattern from the queue
15. while (queue != null)
  15.1 take the first pattern in the queue
  15.2 traversal the pattern in the same item index
  15.3 if the accumulated  $twu$  of this pattern  $> min\_utility$ 
    calculate the real utility and insert extend patterns
    which length is (k+1) to the queue
    15.3.1 if real utility  $> min\_utility$ 
      output the pattern and its utility to high utility
      pattern
    end if
  else
    break
  end if
15.4 delete the first pattern from of the queue
end while
```

4 Discussion

In this section, we list the advantages of our method.

1. Anti-monotone property: We use the measure called *transaction weighted utility* (twu) which has anti-monotone property. Utilize this property can reduce the search space by pruning item which is low twu .
2. Lexicon order in each linked list: While inserting the pattern into linked list, check the pattern to insert the right position. Utilizing the property of lexicon order won't check the complete linked list while discovering the high utility itemsets.
3. Utilize multi-processing approach: Our method perform linear search instead of recursive search in CTU-Mine. We use queue to place the pattern which is ready to discover in linked lists, and this feature can perform multi-processing to enhance the efficiency in discovering high utility patterns.

The preliminary experiment result with no lexicon order and multi-processing features is put in appendix. We use IBM Synthetic Data Generator [6] to generate Synthetic dataset to simulate our method.

5 Conclusions

In this paper, we develop the double layer index and linked lists method using pattern growth approach to enhance the performance of CTU-Mine [5]. The proposed method with the property of anti-monotone and lexicon order in each linked list can quickly find the specific pattern and utilize multi-processing approach to enhance the efficiency in discovering high utility patterns. Compared to CTU-Tree, our

linked list is more condensable because of no generating empty array.

Through the mechanisms mentioned above, we can improve the performance of CTU-Mine algorithm.

6 References

- [1] Agrawal, R. and R. Srikant, "Fast Algorithms for Mining Association Rules," in Proceedings of the 20th International Conference on Very Large Data Bases. 1994. p. 487-499.
- [2] Han, J., J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," in Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. 2000. p. 1-12.
- [3] Liu, Y., W.-K. Liao, and A. Choudhary, "A Fast High Utility Itemsets Mining Algorithm," in International Conference on Knowledge Discovery and Data Mining. 2005, ACM: Chicago, Illinois. p. 90 - 99.
- [4] Tao, F., F. Murtagh, and M. Farid, "Weighted Association Rule Mining using Weighted Support and Significance Framework," in Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. 2003: Washington, D.C. p. 661 – 666
- [5] Erwin, A., R.P. Gopalan, and N.R. Achuthan, "CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach," in Seventh International Conference on Computer and Information Technology. 2007. p. 71-76.
- [6] IBM. IBM Synthetic Data Generator. [cited; Available from: <http://www.almaden.ibm.com/software/quest/resources/index.shtml>].

7 Appendix

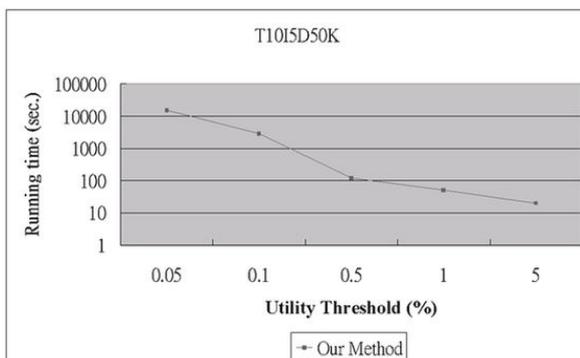


Fig. 8. The preliminary experiment result with no lexicon order and multi-processing features

SESSION

BUSINESS INTELLIGENCE

Chair(s)

Dr. Robert Stahlbock

OLAP For Multicriteria Maintenance Scheduling

Walter Cai[†], David C. Anastasiu[‡], Mingji Xia[§], and Byron J. Gao[‡]

[†]Memorial High School, Madison, WI, USA

[‡]Department of Computer Science, Texas State University - San Marcos, San Marcos, TX, USA

[§]Department of Computer Science, University of Wisconsin - Madison, Madison, WI, USA

Abstract—Widely used in decision support systems, OLAP (Online Analytical Processing) technology facilitates interactive analysis of multi-dimensional data of varied granularities. In this paper, we demonstrate an interesting application of OLAP in solving multicriteria maintenance scheduling problems. Maintenance scheduling has many important applications, such as maintenance of inverted indexes for search engines and maintenance of extracted structures for unstructured data management systems. We introduce the design and implementation of *Scube*, an OLAP-based web service that allows users to navigate in a multi-dimensional simulation cube and comprehensively evaluate maintenance schedules. Based on the evaluation, the best trade-off schedule can be selected.¹

Keywords: OLAP, multicriteria maintenance scheduling, Pareto set, decision support

1. Introduction

Since its first introduction in the early 90's, OLAP (Online Analytical Processing) technology has evolved to be a must-have marketing weapon for business executives to systematically organize, understand, and use enterprise-wide data to make strategic decisions [14]. OLAP facilitates interactive analysis of multi-dimensional data of varied granularities. Most OLAP applications are business-oriented, including sales, financial and management reporting, marketing, business process management, budgeting and forecasting.

In this paper, we investigate an interesting application of OLAP in solving multicriteria maintenance scheduling problems. Maintenance scheduling is a non-typical category of scheduling problems, where jobs need to be periodically maintained over a long time span. It has many important applications, such as maintenance of inverted indexes for search engines, maintenance of extracted structures for unstructured data management systems, and maintenance of materialized views in data warehouses.

Due to the explosive growth of the web, crawling web pages has become increasingly challenging. As of ten years ago, the typical time for a major search engine to crawl one

billion pages was more than one week, and it took up to six months for a new page to be indexed by popular search engines [5]. To substantially improve up-to-dateness of inverted indexes and save on network bandwidth, incremental crawling [5][4] was introduced, where crawlers estimate how often pages change, and then *schedule* pages for revisit based on their estimated change frequency and importance.

The challenge of managing unstructured data represents the largest opportunity since managing relational data [9]. An essential step in unstructured data management is to extract structures embedded in unstructured data, enabling structured queries like "how many citations did Jim Gray receive in 2009?" Web sources are highly dynamic, maintaining extracted structures is a labor intensive undertaking, and developing *scheduling* techniques to improve up-to-dateness and reduce maintenance cost is critical [10].

Such maintenance scheduling tasks are generally subject to communication and computation capacity constraints. While we want to keep the jobs maintained as in-time as possible, we also want to consume as few resources (e.g., workload) as possible. Taking account of several criteria enables us to propose more realistic solutions to the decision maker [21]. Therefore, maintenance scheduling problems are best modeled as multicriteria optimization problems, where multiple incommensurable objectives need to be optimized simultaneously.

At the abstract level, in a typical multicriteria maintenance scheduling problem, we have many jobs with different maintenance periods and maintenance costs. During each period, a job needs to be maintained at least once. We also have a daily workload capacity. The task is then to find a feasible schedule minimizing the long-term tardiness as well as workload.

Unlike single objective optimization problems, multi-objective or multicriteria problems do not have a single optimal solution. Instead, all non-dominated solutions form an optimal *Pareto set*, or *Pareto front*. There are two conceptual steps in solving multicriteria problems: search and decision making. Search refers to the optimization process in which the feasible set is sampled for Pareto solutions. Decision making refers to selecting a suitable compromise solution from the Pareto set. A human decision maker is required to make the often difficult trade-offs among conflicting objectives [13].

¹This work originated from a student summer project Walter Cai did under the direction of Dr. Byron J. Gao at the University of Wisconsin - Madison.

Maintenance schedules are long-term schedules having a huge number of measure points, e.g., one tardiness value for each job with respect to each due day, and one workload value for each day. The performance of maintenance schedules cannot be well captured by a single measure (e.g., largest tardiness or workload) or at a single point (e.g., tardiness or workload for a particular day). For example, schedule s_1 with a bigger maximum tardiness may be much more favorable over schedule s_2 if s_1 performs better than s_2 in terms of tardiness for most due days of most jobs. Thus, maintenance schedules need to be comprehensively evaluated on their overall performance, which usually requires human intervention.

OLAP technology, facilitating interactive analysis of multi-dimensional data of varied granularities, is ideal in assisting decision makers to compare and evaluate the overall performance of alternative maintenance schedules by navigating the data cube. In this paper, we introduce the design and implementation of *Scube* (scheduling cube), an OLAP-based web service that allows users to comprehensively evaluate maintenance schedules.

Outlines. In Section 2, we discuss related work. In Section 3, we model multicriteria maintenance scheduling and introduce OLAP preliminaries. In Section 4, we discuss in detail the architecture and design of *Scube*. In Section 5, we discuss the implementation of the system. In Section 6, we present initial empirical evaluations of *Scube* and a case study. Section 7 concludes the paper.

2. Related Work

[5] studies how to refresh a local copy of an autonomous data source to maintain the copy fresh in the context of managing web data, in particular, inverted index maintenance. It defines freshness, which informally represents the fraction of up-to-date pages in the local collection.

Based on the same motivation of keeping inverted indexes up-to-date, [4][11][17] discuss designs of incremental crawlers with different topical foci, e.g., how web pages evolve over time, what distribution better models changes of web pages, and how to achieve scalability.

[6] theoretically studies a crawler scheduling problem minimizing the fraction of time that pages spend out of date, assuming Poisson page change processes and a general distribution for page access time.

Maintenance scheduling applications abound in unstructured data management systems [9][10]. In such systems, unstructured data are fetched periodically and structured to enable SQL-like queries. It is one of the central issues to schedule the workflow wisely and keep the structured data as fresh as possible while minimizing communication and computation costs.

Scheduling theory first appeared in the mid 1950's. Scheduling concerns the allocation of limited resources to

tasks over time and is normally formulated as optimization problems [18][7]. Maintenance scheduling is a non-typical scheduling problem. Instead of completion time, it concerns periodic in-time maintenance of jobs over a long time span. [1][3][19][2] theoretically study a maintenance scheduling formulation called "windows scheduling problem".

Multicriteria optimization has been studied for decades [12][8][20]. An excellent overview for multicriteria scheduling can be found in [21]. The majority of scheduling problems are single objective. Taking account of several criteria enables us to propose to the decision maker more realistic solutions [21]. Multicriteria problems do not have a single optimal solution and a decision maker is usually involved in the problem solving procedure [13].

OLAP technology has been widely used in enterprise decision support systems [14]. To our knowledge, we are the first to apply OLAP to multicriteria optimization. For a thorough coverage on OLAP, interested readers can refer to the many books dedicated to data warehousing and applications, e.g., [15][16].

3. Preliminaries

In this section, we briefly introduce OLAP preliminaries and a generic problem formulation for multicriteria maintenance scheduling.

OLAP. OLAP systems are built based on the multi-dimensional model, where the central concept is data cube. A *data cube* consists of a large set of numeric facts called measures that are categorized by dimensions. Hierarchical in nature, dimensions are the entities or perspectives with respect to which an organization wants to keep records. Data cubes are typically created from star schemas or snowflake schemas, with star schemas being more popular.

Typical OLAP operations include roll-up, drill-down, drill-across, drill-through, slice, dice, and pivot. There are also other statistical operations available such as ranking and computing moving averages and growth rates. These operations allow users to navigate a data cube along dimension hierarchies and view the cube from different perspectives.

In particular, roll-up summarizes data by climbing up a concept hierarchy of a dimension (or by dimension reduction) to have a "higher view", e.g., from a city view to a country view. Drill-down is the reverse of roll-up going from a higher level summary to a lower level summary. Slice performs a selection on one dimension of the cube, resulting in a subcube. Dice defines a subcube by performing a selection on two or more dimensions. Pivot changes the dimensional orientation and rotates the data axes in order to provide an alternative presentation of the data. Drill-across executes queries involving more than one fact table. Drill-through uses relational SQL facilities to drill through the bottom level of a data cube down to its back-end relational tables [14].

Modeling of Multicriteria Maintenance Scheduling.

In a typical multicriteria maintenance scheduling problem, there are n jobs to be maintained. Each job labeled i needs to be maintained at least once every w_i days and each maintenance costs c_i workload. There is a daily workload capacity of k . The task is to find a feasible schedule minimizing the long-term tardiness as well as workload.

The measure *tardiness* is either 0 (maintained in time) or a positive number indicating the number of days after the due day of a maintenance. A *feasible* schedule is one satisfying all the given constraints. The notion of *day* here is symbolic, it can be time slot of any length.

We use a tuple $I = (k, (c_1, w_1), \dots, (c_n, w_n))$ to denote an instance of a multicriteria maintenance scheduling problem. In I , each job i has a *maintenance window* of size w_i and maintenance cost of c_i .

Note that, real maintenance scheduling problems are almost always *dynamic*, where jobs come and go, and maintenance costs and window sizes change randomly. For example, web pages are created and deleted every day. Blogs would get updated more frequently during holiday seasons. Authors may receive higher citation rates after winning some major awards. Therefore in a dynamic maintenance scheduling instance I , the parameters k , c_i , and w_i are not constants, but variables.

4. Architecture of Scube

In this section, we discuss in detail the design, architecture and workflow of Scube.

The architecture of Scube is shown in Figure 1. In the system, a dynamic multicriteria scheduling instance I can either be automatically generated by the data generation module or provided by users. Some built-in schedulers in the scheduling module will then be applied to I to generate a set of alternative schedules that approximate the Pareto set. Users can also apply external schedulers to generate schedules for I and upload them. Then, from I and the set of alternative schedules, the cube computation module computes a simulation cube. The cube navigation module allows users to navigate the cube and evaluate the schedules in three modes: SQL, CQL, and Graphic. Based on interactive and comprehensive evaluations, users can decide the best trade-off schedule for I .

Data Generation Module. Scube provides a synthetic data generator that generates dynamic maintenance scheduling instances. The instances can vary in size n , length, daily workload capacity k , distribution (Gaussian or Poisson) of maintenance cost c_i and window size w_i . Since Scube performs a simulation calculating measures for each day, a valid instance must end at some finite length.

Dynamic factors (creation and deletion of jobs, change of parameters) are added randomly under a uniform or Poisson distribution.

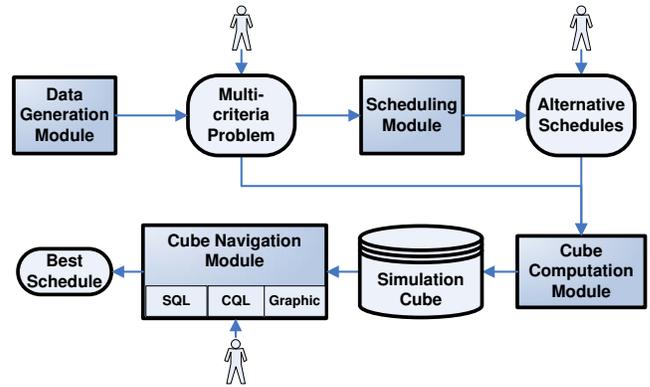


Fig. 1: Scube architecture.

Scube was designed to provide web services. Thus, it also allows users to upload their own scheduling instances conforming to a simple XML style format as described in the following.

Multicriteria Problem. In instance I , maintenance cost c_i and window size w_i can be updated at any specified day. A new job is created if it appears in I with a new id. Job i is deleted if c_i becomes 0. The daily workload capacity k can be modified as well. I ends on the day when all jobs are deleted.

An example instance I is given in Figure 2 (left hand side), demonstrating the required simple format. On day 1, two jobs 1 and 2 are inserted, where $c_1 = c_2 = 1$ and $w_1 = w_2 = 4$. The daily capacity $k = 2$. On day 3, k is updated to 3, c_1 is updated to 0.5, and w_1 is updated to 8. On day 5, all jobs are deleted and I is ended.

Scheduling Module. The scheduling module contains a repository of schedulers for users to choose. Currently it includes two built-in schedulers, EDD (Earliest Due Day first) and MMEDD (Multicriteria Maintenance version of EDD).

The simple greedy algorithm EDD provides optimal solutions for many non-maintenance scheduling problems minimizing the maximum tardiness [7]. In EDD, jobs are ordered by non-decreasing order of due dates (breaking ties arbitrarily) and scheduled in that order.

EDD minimizes tardiness but cares little about workload. If used in maintenance scheduling, EDD would consume too much workload unnecessarily. Non-maintenance scheduling concerns early completion of jobs, whereas in maintenance scheduling, the concern is long-term in-time maintenance. Jobs will not be completed. Each maintenance of a job simply generates a new due day for the same job. Thus EDD would end up generating too many due days unnecessarily. MMEDD minimizes tardiness while using workload wisely. We omit the algorithmic details of MMEDD as they are

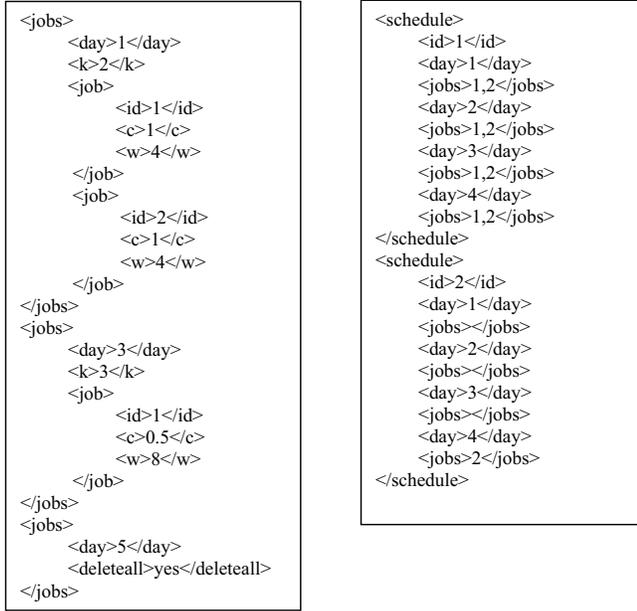


Fig. 2: Example problem and schedules.

beyond the scope of this paper.

A chosen scheduler will generate a set of alternative schedules for input instance I . The scheduling step can be executed multiple times using different schedulers, in other words, the generated alternative schedules to be evaluated are not necessarily from the same scheduler.

Instead of using the `Scube` built-in schedulers, users can also apply external scheduling mechanisms on I and upload the resulting schedules for evaluation, as long as those schedules conform to a simple XML style format as described in the following.

Alternative Schedules. An example set of generated alternative schedules is given in Figure 2 (right hand side), demonstrating the required simple format. There are two schedules in the set with id 1 and 2 for the instance I given in the left hand side. Schedule 1 is an “exhausting” schedule, according to which both jobs need to be maintained every day from day 1 to day 4. Schedule 2 is an optimal schedule, according to which no maintenance is needed from day 1 to day 3, and only job 2 needs to be maintained on day 4. Note that in I , the due day of job 1 is updated on day 3 and it is not due on day 4 anymore, but on day 8.

Cube Computation Module. The cube computation module takes as input the problem instance I and the set of alternative schedules to compute all the tardiness and workload measures. The computed values are then inserted into the fact table of the simulation cube. Algorithm 1 presents the pseudo code for this computation.

In line 6 of Algorithm 1, by comparing the current day

Algorithm 1 Cube Computation

Input: I : scheduling instance; S : set of schedules

Output: F : fact table of the simulation cube

- 1: process I and store each update of c_i and w_i for job i into C_i and W_i respectively, where C_i and W_i are arrays of (day, value) pairs;
 - 2: **for each** $s \in S$
 - 3: **for each** day $d \leq$ the end day of s
 - 4: **for each** job i
 - 5: **if** (job i is scheduled) **then**
 - 6: compare d with $D[i]$ to obtain the tardiness for job i on day d w.r.t. schedule s and insert into F ;
 - 7: consult C_i to obtain the current cost of job i and insert into F as the maintenance workload spent on job i on day d w.r.t. schedule s ;
 - 8: consult W_i and update $D[i]$;
 - 9: **end if**
 - 10: **end for**
 - 11: **end for**
 - 12: **end for**
-

d with the current due day of job i that is scheduled (and maintained) on day d , we get the tardiness for job i on day d w.r.t. schedule s . We insert this tardiness value into the fact table properly.

In line 7, by consulting C_i we can get the current maintenance cost for the scheduled job i . This cost is inserted into the fact table as the workload spent on job i on day d w.r.t. schedule s .

Note that, the series of due days for job i are not prefixed. They depend on the actual schedule as well as the dynamics of job i itself in the input instance I . In line 8, after a maintenance, the next due day for job i is updated. For this update, we need to know the current window size for job i as it may change at any given day.

The algorithm is linear in the number of jobs, the number of days, and the number of schedules.

Simulation Cube. The data cube in `Scube` is called simulation cube because the facts collected and stored in the cube are based on a simulation of some maintenance scenario. In this study, we assume jobs are always maintained as scheduled.

The design of the simulation cube in `Scube` adopts a star schema, which involves a large central fact table containing the bulk of data with no redundancy, and a set of smaller attendant dimension tables one for each dimension [14].

In `Scube`, there are three dimension tables, *time*, *jobs*, and *schedules*. The “time” dimension has a concept hierarchy of “day”, “week”, “month”, and “year”. For simplicity, they are totally ordered, i.e., a year has 12 months, a month has 4 weeks, and a week has 7 days. The “day” attribute is

Table 1: Comparison of Navigation Modes

Modes	Functionality	Usability	Implementation
SQL	Very Good	Very Bad	Very Easy
CQL	Very Good	Good	Easy
Graphic	Very Good	Very Good	Hard

the primary key in the time dimension table with positive integers as domain. The "time" dimension allows users to evaluate schedules based on their performance on different periods of time of varied granularity.

The "jobs" dimension table has job "id" as primary key. The two other attributes are "cost" and "window", with domains of {high, normal, low} and {long, normal, short} respectively. Jobs are put into different "cost" and "window" categories based on their calculated average cost and average window size. The "jobs" dimension allows users to evaluate schedules based on their performance on different types of jobs. Note that the "jobs" dimension has a partial order concept hierarchy.

The "schedules" dimension table has schedule "id" as primary key. It also has a "type" attribute with domain of {busy, normal, lazy}, allowing comparing and evaluating schedules by groups. *Busy* schedules are those generated (e.g., by MMEDD) for $(k', (c_1, w_1), \dots, (c_n, w_n))$ with larger $k' \leq k$ values. Such schedules tend to squander workload for the minimization of tardiness. *Lazy* schedules, on the contrary, favor economic use of workload. *Normal* schedules are in between the two.

The fact table has the primary key of every dimension table as one of its attributes. These attributes are foreign keys to the corresponding dimension tables, and they together form the primary key for the fact table. The two measure attributes are "tardiness" and "workload". There is a tardiness value, 0 or a positive number, for each scheduled maintenance and NULL otherwise. Similarly, there is a positive workload value for each maintenance, which is equal to the cost of the job being maintained. We use 0 as the default value for workload, which would not cause any confusion as in the tardiness case.

Cube Navigation Module. The cube navigation module allows users to interactively evaluate the alternative schedules by navigating the simulation cube. There are three navigation modes, SQL, CQL, and Graphic.

In the SQL (Structured Query Language) mode, a query window is provided taking any standard SQL query over the fact table. In the CQL (Cube Query Language) mode, only several types of predefined intuitive cube navigation queries are allowed in the query window. In the Graphic mode, the CQL operations can be performed on a visible cube without typing textual queries.

A comparison summary for the three navigation modes is shown in Table 1. In terms of functionality, the SQL mode obviously provides every possible structured way of

exploring the fact table. With less querying power in general, the CQL and Graphic modes however provide sufficiently good querying capability, as they are tailored to facilitate cube navigation.

In terms of usability, the SQL mode is very bad because very few regular users are trained to write SQL queries. The CQL mode, however, is good as its syntax is simple, intuitive, and easy to learn. The Graphic mode obviously has the best usability.

In terms of implementation, the SQL mode is very easy only requiring implementation of a simple interface. The CQL mode is also fairly easy, as CQL can be considered as a selective subset of SQL queries plus some syntactic sugar to improve usability. The Graphic mode is hard to implement, but it can be essential if we want to make *Scube* truly accessible to lay users.

To enable any mode of navigation, users need to know the schema of the fact and dimension tables. *Scube* allows users to browse such information easily.

Currently, CQL defines the most basic OLAP operations, *up* (roll-up), *down* (drill-down), and *slice*. A CQL query takes the following format:

Operation Dimension [Value]

For example, "up time" means to perform a one level roll-up on the "time" dimension. "down time 2" means to perform a two level drill-down on the "time" dimension. The very top level of any dimension is the absence of the dimension. In addition, "reset" resets the cube to a default state, which corresponds to the apex cuboid in the lattice of cuboids of the simulation cube.

Scube allows daisy-chaining multiple commands by separating them with a semicolon, allowing users to jump from one cuboid to another directly, as demonstrated by the following example:

reset; down time 2; down jobs.cost 1

In the above example, conceptually the cube is first reset to the default state. Then it is drilled-down by 2 levels along the "time" dimension, and then it is drilled-down by 1 level along the "cost" attribute of the "jobs" dimension. Recall that the "jobs" dimension has a partial order concept hierarchy. It is inefficient if all these conceptual steps are actually computed one after the other, as the user is only interested in the final state of the cube. *Scube* analyzes the multiple commands in a daisy-chained command and translate them into a single aggregation group-by SQL query, greatly reducing processing time.

Best Schedule. Alternative schedules are typically non-dominated and "goodness" of schedules is generally subject to user preferences. After interactively querying the simulation cube, users will obtain comprehensive knowledge about the overall performance of those alternative schedules, from which they are responsible to decide the "best" schedule.

5. Implementation

Scube was implemented in PHP using an OOP architecture, utilizing standard web application development strategies such as the Front Controller Pattern, Model View Controller, and the Template Pattern. This makes the application easily maintainable and extendable. On the server side, Scube made use of several industry standard frameworks, such as the Zend PHP Framework² and Smarty PHP Template Engine³.

On the client side, Scube used Ajax, implemented using the jQuery JavaScript Framework⁴, to enable a rich user experience without unnecessary delays caused by repeated browser page refreshes.

In Scube, users follow three steps to complete a scheduling task: present the problem, schedule the problem, and navigate the cube. All initiated actions within the three steps are handled in the background by the server via Ajax requests, which update the browser window with appropriate responses. Scheduling a problem generally takes much longer time than submitting a problem. Thus all submitted problems are queued, and the server checks the queue each minute for new items.

Scube allows users to delete schedules for a given problem. When a schedule is deleted, the data is removed from the simulation cube.

To allow concurrent access to the Scube service, the system generates a random session id for each new application session. Users are allowed to change this id. During subsequent sessions, users can continue to work on a problem by providing the same session id.

6. Evaluation

Scube⁵ is open for public access and evaluation. In this section, we present our initial usability and scalability studies on Scube. We also introduce a citation information monitoring system⁶ as a maintenance scheduling case study.

User Study. Usability is a fundamental concern for decision making tools like Scube. Decision makers are business-oriented. They prefer systems that are intuitive, friendly, and easy to learn and to use.

For an initial test on the usability of Scube, we prepared 6 questions regarding learning time, response time and user-friendliness. The questions were distributed to 10 users, most are IT professionals located in Austin (TX, USA) and Wilmington (NC, USA) in a clinical research organization. The questions and the received average scores on a scale of 1 to 10 are given in the following.

²<http://framework.zend.com/>

³<http://www.smarty.net/>

⁴<http://jquery.com/>

⁵<http://dmlab.cs.txstate.edu/scube/>

⁶<http://dmlab.cs.txstate.edu/citation/>

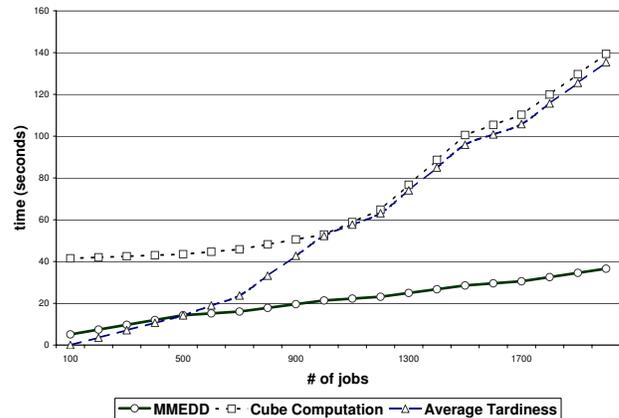


Fig. 3: Scalability.

- 1) Is the system easy to learn? (9.6)
- 2) Is the interface user-friendly? (8.8)
- 3) Does the system respond quickly? (8.8)
- 4) Are the directions clear? (9.6)
- 5) Would you consider using such a system if you had to evaluate alternative long schedules? (9.3)
- 6) How do you rate the system overall? (9.1)

The scores are higher than expected possibly because most participants are IT professionals and better than average at adapting to new technical tools and web services. The system architecture (Q1) received a high score, which is evident for the intuitive and logical design of the Scube architecture as shown in Figure 1. The simple xml style formats for the instance and schedules are also easy to digest.

The interface (Q2) received a lower score possibly because we have not fully implemented the Graphic mode for cube navigation. If users are not familiar with SQL, the SQL navigation mode can be very unfriendly. Although the CQL navigation mode is much more friendly, it may still require some basic understanding on OLAP tools.

The system response performance (Q3) received a lower score possibly because the users did not understand that long-term many-job maintenance tasks could easily take a couple of minutes to schedule. It takes even longer time to calculate the measures for numerous measure points and load them into the simulation cube. Thus, the longer response time may have appeared unexpected.

Overall, the initial deployment of Scube is satisfactory. The participants felt it could well be a convenient tool for comprehensively evaluating long schedules.

Scalability Study. In Scube, the most time-consuming modules are scheduling (MMEDD) and cube computation. We experimentally evaluated their scalability performance.

In this series of experiments, we used our data generator to generate instances of different sizes, where daily capacity

k was set to 10 and length of schedules was set to 1000 days. The maintenance cost was set to 1 for every job and the window size values were random numbers in the range of 28 to 100 under the normal distribution.

For each instance size, 5 experiments were performed and the average time was taken. The results are presented in Figure 3, which shows the scalability of MMEDD and cube generation, and thus, *Scube*.

A similar series of experiments were performed for instances generated under the Poisson distribution, and the results exhibit a similar trend as in Figure 3.

Figure 3 also shows the average tardiness in these experiments, for the purpose of showing an insight that is typical in maintenance scheduling, i.e., the tardiness increases with the increase of number of jobs.

Citation Information Monitoring. This research was initially motivated by a citation monitoring service, where we monitor the dynamics of citations for authors (currently 8988 of them) in the database community, and provide online services taking temporal queries for citation. A pilot system, still under continuous development and evaluation, has been deployed and up running for about 10 months. This application is a typical case of dynamic maintenance scheduling problem.

Tardiness directly impacts the functionality of the system. For example, if we crawl Jim Gray once every 30 days, we will not be able to properly answer queries like “what citations did Jim Gray receive in the last 10 days?”

Workload is just as critical. Actually we have received complaints from some web sources for intensive crawling. We definitely need to reduce the total workload.

Thus, in this project we have paid great attention to scheduling. The application is also exposed to a typical dynamic environment, where new authors enter the community at random times and citation rates change dynamically.

The window size w_i for each author i needs to be estimated each day so as to quickly capture dynamic changes of citation rates. The basic idea for this estimation is to calculate a weighted average for the number of citations received by author i in the past certain number (e.g., 300) of days. Each author also needs to be assigned a different maintenance cost, which can be estimated based on the number of publications of the author. More publications lead to bigger maintenance cost.

7. Conclusion

In this paper, we demonstrated an interesting application of OLAP in solving multicriteria maintenance scheduling problems. Such problems can properly model many important applications in information monitoring, such as maintenance of inverted indexes for search engines, maintenance of extracted structures for unstructured data management, and maintenance of materialized views in data warehouses, to

name a few. We introduced the design and implementation of *Scube*, an OLAP-based web service that helps users to comprehensively evaluate alternative maintenance schedules and make decisions on the best trade-off schedule to select.

Future work includes continuous development and refinement of *Scube*, e.g., on the Graphic navigation mode. The built-in repository of schedulers in *Scube* can be expanded as well. Last but not least, although we provided initial empirical evaluation on *Scube*, for a more convincing justification, it is important if we can apply *Scube* to real applications and see the actual long-term benefits it generates.

References

- [1] A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. *Math. Oper. Res.*, 27(3):518–544, 2002.
- [2] A. Bar-Noy and R. E. Ladner. Windows scheduling problems for broadcast systems. *SIAM J. Comput.*, 32(4):1091–1113, 2003.
- [3] A. Bar-Noy, R. E. Ladner, and T. Tamir. Windows scheduling as a restricted version of bin packing. *ACM Transactions on Algorithms*, 3(3), 2007.
- [4] J. Cho and H. Garcia-Molina. The evolution of the web and implications for an incremental crawler. In *VLDB*, 2000.
- [5] J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In *SIGMOD*, 2000.
- [6] E. Coffman, Z. Liu, and P. Weber. Optimal robot scheduling for web search engines. *Journal of Scheduling*, (1):15 – 29, 1998.
- [7] C. S. David R. Karger and J. Wein. *Scheduling Algorithms*. Algorithms and Theory of Computation Handbook, 1998.
- [8] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.
- [9] A. Doan, J. F. Naughton, A. Baid, X. Chai, F. Chen, T. Chen, E. Chu, P. DeRose, B. J. Gao, C. Gokhale, J. Huang, W. Shen, and B.-Q. Vuong. The case for a structured approach to managing unstructured data. *CIDR*, 2008.
- [10] A. Doan, R. Ramakrishnan, F. Chen, P. DeRose, Y. Lee, R. McCann, M. Sayyadian, and W. Shen. Community information management. *IEEE Data Eng. Bull.*, 29(1):64 – 72, 2006.
- [11] J. Edwards, K. McCurley, and J. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In *WWW*, 2001.
- [12] M. Ehrgott. *Multicriteria optimization*. Springer, 2005.
- [13] G. W. Evans. An overview of techniques for solving multiobjective mathematical programs. 30(11):1268 – 1282, 1984.
- [14] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2006.
- [15] C. Imhoff and N. Galemno. *Mastering Data Warehouse Design: Relational and Dimensional Techniques*. John Wiley & Sons, 2003.
- [16] R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, 2002.
- [17] L. Lim, M. Wang, and S. Padmanabhan. Dynamic maintenance of web indexes using landmarks. In *WWW*, 2003.
- [18] M. Pinedo. *Scheduling - Theory, Algorithms, and Systems*. Prentice Hall, 1995.
- [19] R. H. Shoshana Anily, Celia A. Glass. The scheduling of maintenance service. *Discrete Applied Mathematics*, 82(1–3):27–42, 1998.
- [20] R. E. Steuer. *Multiple Criteria Optimization: Theory, Computations, and Application*. John Wiley & Sons, 1984.
- [21] V. T'kindt and J.-C. Billaut. *Multicriteria Scheduling*. Springer, 2006.

Using Social Ties to Predict Missing Customer Information

Stamatis Stefanakos

D1 Solutions AG, Zypressenstrasse 71, 8040 Zurich, Switzerland.

Email: stamatis.stefanakos@d1solutions.ch

Abstract—“*Show me who your friends are and I will tell you who you are.*” This popular saying captures the essence of our approach on dealing with missing customer information in databases. We argue that, provided relationship information among customers is available, one can aggregate information along social ties in order to predict missing data or better understand customer behavior.

We focus our analysis in the case of a mobile telecommunications operator. We use the operator’s network log to define a call graph on the active customer base. From the call graph, based on the contact frequency and the types of calls, we extract an approximation of the social ties for each customer. We then use the defined social ties to aggregate information from the contacts of each customer in order to predict missing data. We analyze the case of customer age and indicate potential uses of our approach.

Keywords: Customer behavior; Social networks; Telecommunications; Missing data.

1. Introduction

Dealing with missing data attributes is an important task in any data mining or Customer Relationship Management (CRM) application. In the case of data mining, missing information can lead to underperforming predictive models or wrong analyses. In the case of CRM processes, missing information may lead to underperforming campaigns. For example, when a CRM manager designs a campaign specifically targeted to particular age segments, the customers whose age is missing from the database will inevitably have to be excluded from the campaign. Otherwise one would risk addressing customers with the wrong campaign.

Deteriorating data quality is a major problem in databases which might be hard to detect and might lead to wrong analyses, underperforming predictive models, or badly targeted CRM campaigns. In order to deal with these issues, measures have to be taken already at the operational system level. An incoming call in a call center can be an opportunity to collect missing data—as long as the agents have been trained to do so and can be timely informed. Similarly, an incoming customer call can be used to correct incorrect data—as long as the errors have been identified and the identification is fed into the operational system. Statistical methods and data mining can be helpful for detecting anomalies in the data and can thus serve as a first step in collecting correct data and improving data quality.

In the case of telecommunication companies, where data-warehouses (DWH) need to store and historize information about millions of customers, these issues are usually magnified. Missing information and deteriorating data quality can be common in data models spanning several dimensions such as revenues, network usage, socio-demographics, product relationships, payment information, etc.

Mobile operators usually offer both prepaid and postpaid rate-plans to their customers. Prepaid rate-plans require the subscriber to pay in advance. This is usually done by refilling credits through the internet or by buying the extra credit from kiosks. Postpaid rate-plans are invoice-based and usually come bundled with attractive handsets but also with contracts binding the customer for one or two years. The existence of a contract is already a guarantee of better data quality. As will also see later in this paper, information of prepaid customers is characterized from bad data quality as a result of the rate-plan’s nature.

A very important source of information for both prepaid and postpaid customers is the network log. The network log, contains a record for every event (such as a voice call, a short message (SMS), or a data transfer) that occurs in the network. These records contain information about the parties involved, the time and the date and the duration of the event, geographical information, volume information, as well as other technical data. The data records stored in the network log are known as call-detail records (CDRs).

Information from the network log is very frequently used in data mining and CRM applications and might even be summarized in several aggregates in the DWH. Information such as the number of calls a customer receives or makes within a certain period of time, the number of short messages, the data usage, the incoming–outgoing ratios are stored and aggregated in the DWH and often used in predictive models or in the design of CRM campaigns. For example, a common approach of mobile operators is to design campaigns offering new multimedia services to users who have exhibited heavy usage of similar services in the past; this information is obtained from the CDRs.

What is less often used in CRM and data mining applications is the *relationship* information contained in the network log. Relationship information can reveal the social ties between the customers. A social network is in essence integrated in every such network log as for each customer we not only know how many calls she makes in a given time period but also *whom* she contacts and how often.

1.1 Our Contribution

We argue that relationship information contained in the network logs (or relationship information obtained from any kind of social network) can be used in order to

- predict missing data attributes for customers with missing information,
- define customer segments without having to exclude from the segmentation customers with missing information, and
- generally obtain a better understanding of the customer's needs and behavior.

Our approach can be generalized to apply to any business that can obtain relationship information for its customers. Here, however, we focus in the case of telecommunication providers in which such information is readily available in the network logs. We show how social ties can be obtained by analyzing the network logs. The social ties can then be used to aggregate information available for a customer's connections in order to get a better understanding of the customer behavior or even predict information that is missing from the customer's profile.

Our approach, viewed in the context of customer development focuses on the customer's network value as opposed to the traditionally used customer value. As defined in [1] the network value of a customer is the expected profit from sales to others that results from marketing to that customer. Similarly, one can define customer segments by segmenting the behavior of the social ties of the customer and not directly the behavior of the customer. We believe that in certain cases CRM campaigns are better designed having in mind "with whom my customer communicates" rather than "who my customer is."

We study our theory for one single data dimension: Customer age. We first identify, in the data-warehouse we had access to, typical problems in data quality. We establish that this information is missing for around 10% of the customer base. We then define the social ties using four weeks of call-detail records. We show how the age dimension can be aggregated through the social ties and compute the resulting error in the prediction when using only a simple average aggregation.

In our study, we distinguish between the postpaid and the prepaid customer segments. While one could argue that the two are really two different kinds of rate-plans, we observe great differences in data quality as well as in the accuracy of our predictions. Telecommunication companies very often do not have much information about their prepaid customers (see, e.g., [2]). For this type of customer, our approach is particularly interesting because even if a company for some reason does not gather data for any of its prepaid customers, some insights about these customers can be propagated via their social contacts to postpaid customers (for which data should exist).

1.2 Previous Work

The problem of dealing with missing data attributes has been extensively studied in the field of statistics. See, e.g., the book of Little and Rubin [3]. A study on how to deal with missing values in data sets that are specifically used in data mining can be found in [4]. Also, approaches to predict missing data values using data mining have been proposed in [5] and in [6]. We are not aware of any work that uses social networking information in order to predict missing attributes from databases.

A vast amount of literature exists on the topic of social networks. Early research on social networks was in the social sciences. The focus of this research was concerning the connectivity of social networks arising from, e.g., friendship networks, the size of the average connecting paths, and their degree distribution. A famous such study is Milgram's analysis [7] of a contact network in the United States (later popularized as the "small world phenomenon").

Research in social networks has recently boomed along with the emergence of popular internet social-networking applications that allow for the collection and the study of vast data-sets. Barabasi et al. [8], e.g., study the evolution of scientific collaborations. Kossinets et al. [9] study the social network obtained from email communication of faculty and staff of a large university.

Several researchers have also studied networks arising from the call-detail records of telephone operators. Nanavati et al. [10] study call graphs created from mobile CDRs. They study several structural parameters of the call graph and introduce a topological model to capture the shape of such graphs. Pandit et al. [11] study the problem of finding dense communities in a social network defined by a telecom call graph. They apply to this problem an approximation algorithm for the densest subgraph problem by [12]. Dasgupta et al. [13] build a churn prediction model for the prepaid segment of a large mobile communication provider based on social tie calculation.

Several studies also exist regarding data mining in telecommunications and the analysis of customer behavior. See for example [14], [15], [16] and the references therein.

2. Defining The Call Graph

Our study is based on anonymized data from the network log of a Swiss mobile operator. The network log that we can access captures all events within the particular operator's network. Thus, we do not have information about connections made outside of that network, across the networks of other operators, or to the wireline network.

The operator's network log stores around 15 million records per day. Each record contains information on the calling parties, the type of call, the duration or the data volume of the connection, the devices used, the cell on which the connection originated, etc. Figure 1 shows the

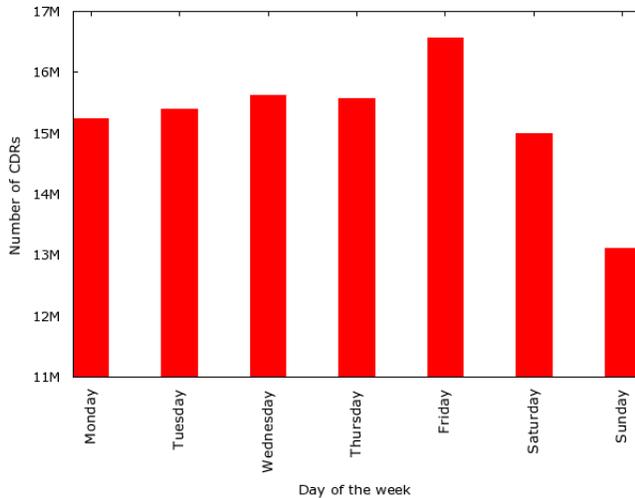


Fig. 1: The distribution of CDRs per day of the week.

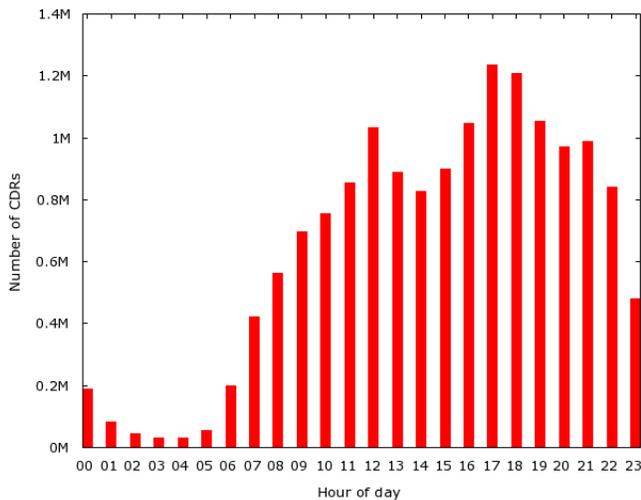


Fig. 2: The distribution of CDRs per hour for a single day.

distribution of the call-detail records per day within a week. Figure 2 shows the distribution of the call-detail records per hour for a single day.

For our study we have collected call-detail records spanning four weeks. We have excluded all records in which one of the calling parties lies outside of the operator's network. We omit, for example, calls to or from international numbers and calls to or from wireline numbers. Furthermore, we only maintain records of voice calls or SMS. From the available information we only maintain the indication on whether the connection is a voice call or an SMS, the anonymized identifiers of the two parties, the direction of the connection, and in the case of voice calls the duration of the call.

The aggregated data is then used to construct the call graph. For every subscriber, we obtain a list of other sub-

scribers with which a connection was established in the four week period. Not all of these contacts constitute social-ties however. It is clear that a single call within four weeks can be an occasional communication and that no real tie may exist between the two parties. Identifying the real ties within the call graph (and perhaps even defining what a "real tie" should be) is a challenging task. We have decided to resolve this issue in an ad-hoc manner: For a tie to exist, we require bi-directional communication in at least one of the two communication channels (voice and SMS) and a minimum of four connections in any direction within the time-span that we study. Thus, if subscriber A sends five SMS to subscriber B but does not receive any then this link is dropped. On the other hand if A sends three SMS to B and receives one from B then we maintain a link in the call graph between A and B . The resulting graph consists of one million nodes (postpaid and prepaid subscribers) and four millions unidirectional edges.

3. Case Study: Customer Age

The customer's age is a very important information for telecommunication companies. CRM and marketing activities are often designed for customer clusters emerging from micro-segmentations along few dimensions. Age is a dimension that decisively determines these segments and in many cases might be the only dimension used to determine the segments. Often, special offers are targeted to people of a certain age.

The age of the customer is collected during the sign-up process: Once a contract is made or a prepaid card is registered, the date of birth of the customer is entered into the operational system and transferred to the DWH. There are several reasons why such an important piece of information might be missing for certain customers. Customers that started using the operator's services several years ago, for example, when the registration of prepaid cards was not obligatory might have not registered their cards and thus have entered no date of birth into the system. Also, during registration the agents might not enter the information correctly into the system in order to speed up the process and serve more customers. Another reason why a prepaid customer's age information is card-swapping: Prepaid telephone cards, since they are not bound to any contract, might be registered by one person and then passed on to another one. For example, a father might register prepaid cards that will be later used exclusively by his children. Hence, the information stored in the database about this subscriber will not reflect the real user of the telephone.

Figures 3 and 4 illustrate the age distribution of the operator's postpaid and prepaid customer base. The distribution plots instantly reveal mistakes in the data: In the case of postpaid customers, 2% of all birth-dates lie outside the "normal" range between 0 and 100. In the case of prepaid customers, this percentage rises to around 8%. Some of

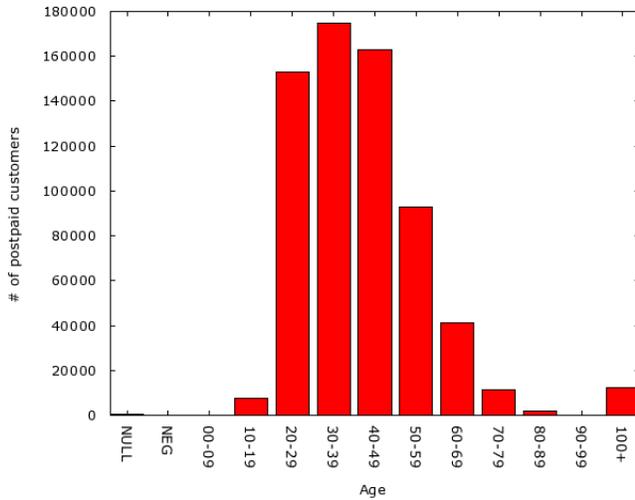


Fig. 3: Age distribution of the postpaid customer base.

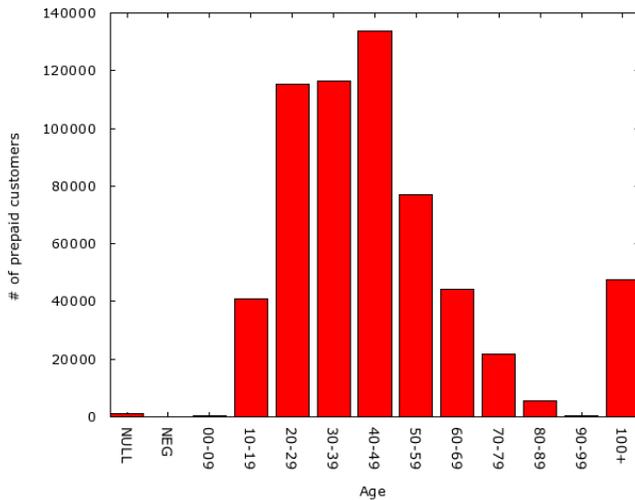


Fig. 4: Age distribution of the prepaid customer base.

this is due to missing birth-dates. Most of the birth-dates, however, are not missing but are obviously wrong: Some lie in the future (resulting in negative age), many have the same value around 256 years ago. This is most likely the default value displayed in the operational system when entering the information for a customer. The agents, when in hurry, simply skip entering the correct date and the default date propagates into the system.

These observations thus indicate that we do not have correct age information for about 10% of the customer base. This percentage refers to these customers whose date of birth is clearly wrong in our database. From this data alone, we can not estimate up to what extent legitimate looking birthdates are correct or wrong. We see two consequences: First, the customers whose wrong data can be easily identified will

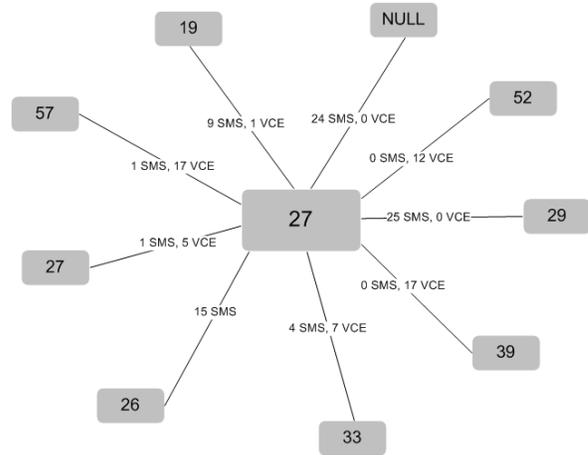


Fig. 5: Example of the call graph induced by a single 27-year old customer.

have to be excluded from any campaigns that are designed primarily according to age segments. Second, customers who appear to have a correct date of birth will enter the campaigns but depending on the accuracy of the information might receive a campaign designed for a different age segment.

The aggregation of relationship information from the network log, however, allows us to predict the missing information and in some cases even identify wrong but legitimate looking data. For each customer we have the following information at our disposal: The social ties, as defined in Section 2, together with the age (if it is available) for each of his or her ties. For each social tie we also know whether the tie originates from voice calls or SMS (or both) and the corresponding volumes. If we focus on a customer who is say, 27 years old, the call graph augmented with age information might look as shown in Figure 5. Taking a real example, we observe the distribution of SMS and voice calls per social tie shown in Figure 6.

Figure 6 indicates that indeed it might be possible to predict up to some degree the age of a customer from the ages of the social ties. For this particular example, just by taking a simple average of the ages of all social ties results in a pretty accurate prediction of the customer's age. One of course can improve on this and implement techniques to enhance the prediction accuracy. Such measures could include removing first a certain min and max percentile of the ages before averaging, or using the corresponding connection volumes to weight the importance of a social tie. While we can not hope for accurate prediction of the ages of all customers, an approximation should be useful for the design of CRM campaigns. Furthermore, if one is interested in understanding the customer's behavior the aggregated age is no longer a prediction but an indicator of behavior.

In Figure 7, we see the result of a prediction of all

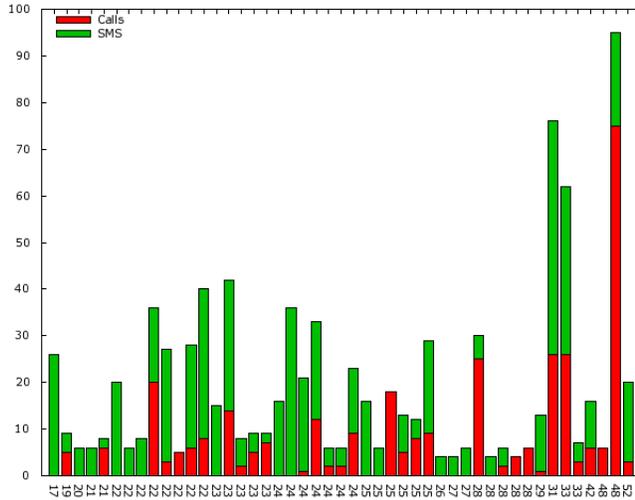


Fig. 6: SMS and voice volumes for all social ties of a 27-year old customer shown across the ages of the contacts.

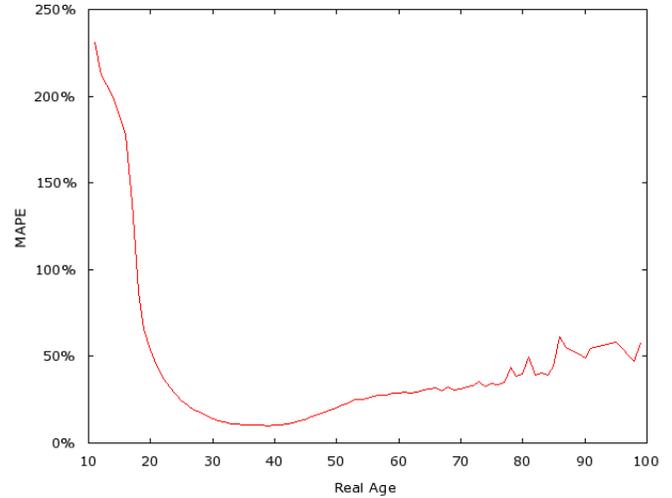


Fig. 8: Mean absolute percentage error per age for postpaid customers.

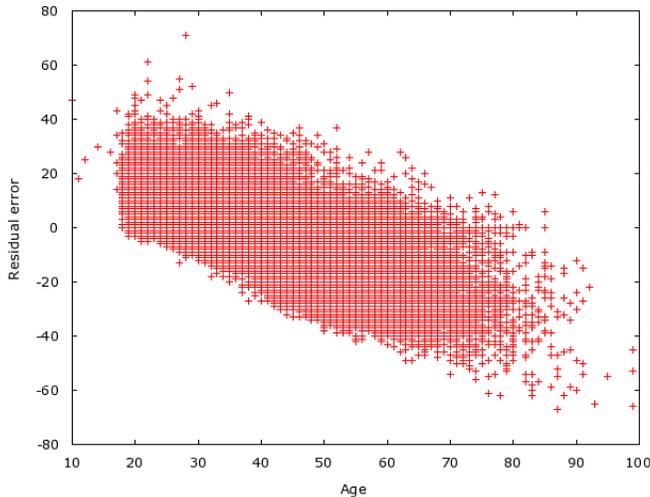


Fig. 7: Residual error of the age prediction through social link aggregation for postpaid customers.

postpaid customers' ages by using a simple average of the ages of their social ties. The volumes of the calls and the SMS are not taken into account. All clearly wrong ages (i.e., all missing ages, negative ages, or ages around the value of 256 that result from skipping to enter a customer's age in the operational system) have also been removed from the calculations. The chart shows the residual error of the predicted age against the known age. As expected, the residual error is more evenly distributed around zero for ages around 40. We tend to over-predict ages on the left of that range and under-predict ages on the right of that range. This is logical since young people will tend to communicate with older people and vice versa.

Figure 8 shows the mean absolute percentage error per

age for postpaid customers. The error is minimized between ages 30 and 40 and is at its highest for ages close to ten years old. Overall, for both postpaid and prepaid customers the mean absolute percentage error is 24.6%. For postpaid only the error drops to 20.5% while for prepaid the error rises to 30.6%. The larger error for the prepaid customer base can partly be due to the different age distribution of that segment. The prepaid customer base has a higher percentage of customers between 10 and 20 years old where the error is higher. The big difference in prediction accuracy for the prepaid customer segment, however, is also due to the lower data quality available as well as due to the effects of card swapping.

4. Conclusion

We have outlined an approach for dealing with missing customer information in databases. The core idea relies on the existence of relationship information between customers. In the case of telecommunications, this is available in the network log. The relationship information can be used to extract the social ties of each customer. Then, information available for other customers can be aggregated through the social ties in order to fill-in missing or incorrect information. Our first analyses indicate that this is a promising approach which can help predict missing information or identify wrong information in the database.

This approach can also be employed to better understand customer behavior. The aggregation of customer information through social ties reveals insights about the social circle of the customers and their behavior and communication patterns. Such insights can be much more valuable than a single attribute. In order to calculate expected customer value, for example, we can perform a similar aggregation of revenues

to identify customers with high-spending social circles. In the case of prepaid subscribers, customer information might be missing completely or to a very large extent. Social ties can be used in order to propagate information available for the postpaid subscribers to the prepaid segment.

It is important to note that this is only a first step in our study: We create social ties using an ad-hoc rule and the prediction mechanism we employ is very simple. Thus, our results should only be understood as a proof of concept. Further study is needed to understand how one can robustly define social ties from the customers' relationship information. Also, more elaborate techniques can be used to aggregate information from the social ties. Finally, more advanced data mining techniques should be tested for the task of predicting information for a single customer using the information available for the social ties.

References

- [1] P. Domingos, "Mining social networks for viral marketing," *IEEE Intelligent Systems*, vol. 20, no. 1, pp. 80–82, 2005.
- [2] A. Levisse, N. Manuel, and M. Sjolund, "Getting more from prepaid mobile services," *The McKinsey Quarterly*, February 2008.
- [3] R. Little and D. Rubin, *Statistical analysis with missing data*. Wiley New York, 1987.
- [4] J. W. Grzymala-Busse and M. Hu., "A comparison of several approaches to missing attribute values in data mining." in *Rough Sets and Current Trends in Computing*, 2000, pp. 378–385.
- [5] J. Li, N. Cercone, and R. Cohen, "Addressing missing attributes during data mining using frequent itemsets and rough set based predictions," in *IEEE International Conference on Granular Computing 2007*, 2007, pp. 294–294.
- [6] N. Setiawan, P. Venkatachalam, and A. Hani, "Missing attribute value prediction based on artificial neural network and rough set theory," in *IEEE International Conference on BioMedical Engineering and Informatics 2008*, 2008, pp. 306–310.
- [7] S. Milgram, "The small world problem," *Psychology Today*, vol. 2, pp. 60–67, 1967.
- [8] A. Barabasi, H. Jeong, E. Ravasz, Z. Neda, A. Schubert, and T. Vischek, "Evolution of the social network of scientific collaborations," *Physica*, vol. 311, pp. 590–614, 2002.
- [9] G. Kossinets, J. Kleinberg, and D. Watts, "The structure of information pathways in a social communication network," in *Proc. 14th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 2008, pp. 435–443.
- [10] A. A. Nanavati, S. Gurumurthy, G. Das, D. Chakraborty, K. Dasgupta, S. Mukherjea, and A. Joshi, "On the structural properties of massive telecom call graphs: findings and implications," in *Proceedings of the 15th ACM international conference on Information and knowledge management*, 2006, pp. 435–444.
- [11] V. Pandit, N. Modani, S. Mukherjea, A. Nanavati, S. Roy, and A. Agarwal, "Extracting dense communities from telecom call graphs," in *Communication Systems Software and Middleware and Workshops (COMSWARE 2008)*, 2008, pp. 82–89.
- [12] M. Charikar, "Greedy approximation algorithms for finding dense components in a graph," in *Proceedings of the 5th workshop on Approximation Algorithms (APPROX)*, 2000, pp. 84–95.
- [13] K. Dasgupta, R. Singh, B. Viswanathan, D. Chakraborty, S. Mukherjea, A. A. Nanavati, and A. Joshi, "Social ties and their relevance to churn in mobile telecom networks," in *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, 2008, pp. 668–677.
- [14] F. Malabocchia, L. Buriano, M. Mollo, M. Richeldi, and M. Rossotto, "Mining telecommunications data bases: an approach to support the business management," in *IEEE Network Operations and Management Symposium*, vol. 1, no. 15–20, 1998, pp. 196–204.
- [15] L. Yan, R. H. Wolniewicz, and R. Dodier, "Predicting customer behavior in telecommunications," *IEEE Intelligent Systems*, vol. 19, no. 2, pp. 50–58, 2004.
- [16] G. M. Weiss, "Data mining in the telecommunications industry," in *Encyclopedia of Data Warehousing and Mining*, 2008.

Advanced Implementation Techniques for Scientific Data Warehouses

Gangadhar Payyavula¹, Lakshmi Tulasi Ravulapalli²

¹B.Tech Student, Computer Science & Engineering, QIS College of Engineering & Technology, Vengamukkapalem, Ongole, Andhra Pradesh, India

²Head of the Department, Computer Science & Engineering, QIS College of Engineering & Technology, Vengamukkapalem, Ongole, Andhra Pradesh, India

Abstract - R & D Organisations handling many Research and Development projects produce a very large amount of Scientific and Technical data. The analysis and interpretation of these data is crucial for the proper understanding of Scientific / Technical phenomena and discovery of new concepts. Data warehousing using multidimensional view and on-line analytical processing (OLAP) have become very popular in both business and science in recent years and are essential elements of decision support, analysis and interpretation of data. Data warehouses for scientific purposes pose several great challenges to existing data warehouse technology. This paper provides an overview of scientific data warehousing and OLAP technologies, with an emphasis on their data warehousing requirements. The methods that we used include the efficient computation of data cubes by integration of MOLAP and ROLAP techniques, the integration of data cube methods with dimension relevance analysis and data dispersion analysis for concept description and data cube based multi-level association, classification, prediction and clustering techniques.

Keywords: Scientific Data Warehouses, On-line analytical processing (OLAP), Data Mining, On-Line Analytical Mining (OLAM), DBMiner, Data Cubes.

1 Introduction

R & D Organisations handling many Research and Development projects produce a very large amount of Scientific and Technical data. The analysis and interpretation of these data is crucial for the proper understanding of Scientific / Technical phenomena and discovery of new concepts. Data warehousing and on-line analytical processing (OLAP) are essential elements of decision support, which has increasingly become a focus of the database industry. Many commercial products and services are now available, and all of the principal database management system vendors now have offerings in these areas. Decision support places some rather different requirements on database technology compared to traditional on-line transaction processing

applications. Data Warehousing (DW) and On-Line Analytical Processing (OLAP) systems based on a dimensional view of data are being used increasingly in traditional business applications as well as in applications such as health care and bio-chemistry for the purpose of analyzing very large amounts of data.

The use of DW and OLAP systems for scientific purposes raises several new challenges to the traditional technology. Efficient implementation and fast response is the major challenge in the realization of On-line analytical mining in large databases and scientific data warehouses. Therefore, the study has been focused on the efficient implementation of the On-line analytical mining mechanism. The methods that we used include the efficient computation of data cubes by integration of MOLAP and ROLAP techniques, the integration of data cube methods with dimension relevance analysis and data dispersion analysis for concept description and data cube based multi-level association, classification, prediction and clustering techniques. We describe back end tools for extracting, cleaning and loading data into a scientific data warehouse; multidimensional data models typical of OLAP; front end client tools for querying and data analysis; server extensions for efficient query processing; and tools for metadata management and for managing the warehouse. These methods will be discussed in detail.

2 OLAP + Data Mining → On-Line Analytical Processing

On-line analytical processing (OLAP) is a powerful data analysis method for multi-dimensional analysis of data warehouses. Motivated by the popularity of OLAP technology, we use an On-Line Analytical Mining (OLAM) mechanism for multi-dimensional data mining in large databases and scientific data warehouses. We believe this is a promising direction to pursue for the scientific data warehouses, based on the following observations.

1. Most data mining tools need to work on integrated, consistent, and cleaned data, which requires costly data cleaning, data transformation and data integration as pre-processing steps. A data warehouse constructed by such pre-processing serves as a valuable source of cleaned and integrated data for OLAP as well as for data mining.

2. Effective data mining needs exploratory data analysis. A user often likes to traverse flexibly through a database, select any portions of relevant data, analyze data at different granularities, and present knowledge/results in different forms. On-line analytical mining provides facilities for data mining on different subsets of data and at different levels of abstraction, by drilling, pivoting, filtering, dicing and slicing on a data cube and on some intermediate data mining results. This, together with data/knowledge visualization tools, will greatly enhance the power and flexibility of exploratory data mining.

3. It is often difficult for a user to predict what kinds of knowledge to be mined beforehand, by integration of OLAP with multiple data mining functions. On-line analytical mining provides flexibility for users to select desired data mining functions and swap data mining tasks dynamically.

However, data mining functions usually cost more than simple OLAP operations. Efficient implementation and fast response is the major challenge in the realization of On-line analytical mining in large databases and scientific data warehouses. Therefore, our study has been focused on the efficient implementation of the On-line analytical mining mechanism. The methods that we used include the efficient computation of data cubes by integration of MOLAP and ROLAP techniques, the integration of data cube methods with dimension relevance analysis and data dispersion analysis for concept description and data cube based multi-level association, classification, prediction and clustering techniques. These methods will be discussed in detail in the following subsections.

2.1 Architecture for On-Line Analytical Mining

An OLAM engine performs analytical mining in data cubes in a similar manner as an OLAP engine performs on-line analytical processing. Therefore, it is suggested to have an integrated OLAM and OLAP architecture as shown in below Figure.1., where the OLAM and OLAP engines both accept users on-line queries (instructions) and work with the data cube in the analysis. Furthermore, an OLAM engine may perform multiple data mining tasks, such as concept description, association, classification, prediction, clustering, time-series analysis, etc. Therefore, an OLAM engine is more sophisticated than an OLAP engine since it usually consists of multiple mining modules which may interact with each other for effective mining in a scientific data warehouse.

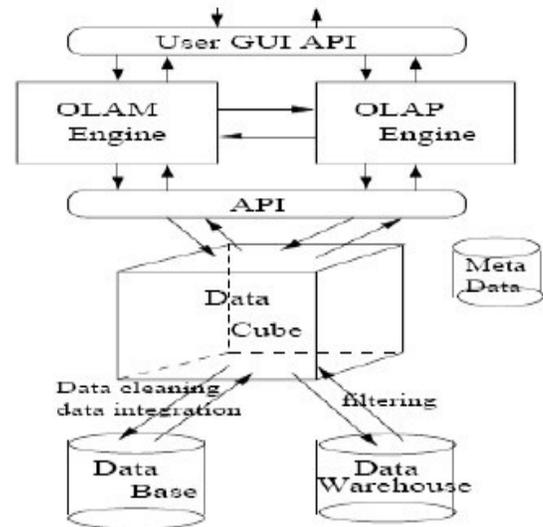


Fig.1. An integrated OLAM and OLAP architecture

Since some requirements in OLAM, such as the construction of numerical dimensions, may not be readily available in the commercial OLAP products, we have chosen to construct our own data cube and build the mining modules on such data cubes. With many OLAP products available on the market, it is important to develop on-line analytical mining mechanisms directly on top of the constructed data cubes and OLAP engines. Based on our analysis, there is no fundamental difference between the data cube required for OLAP and that for OLAM, although OLAM analysis may often involve the analysis of a larger number of dimensions with finer granularities, and thus require more powerful data cube construction and accessing tools than OLAP analyses. Since OLAM engines are constructed either on customized data cubes which often work with relational database systems, or on top of the data cubes provided by the OLAP products, it is suggested to build on-line analytical mining systems on top of the existing OLAP and relational database systems rather than from the ground up.

2.2 Data cube construction

Data cube technology is essential for efficient on-line analytical mining. There have been many studies on efficient computation and access of multidimensional databases. These lead us to use data cubes for scientific data warehouses.

The attribute-oriented induction method adopts two generalization techniques (1) attribute removal, which removes attributes which represent low-level data in a hierarchy, and (2) attribute generalization which generalizes attribute values to their corresponding high level ones. Such generalization leads to a new, compressed generalized relation with count and/or other aggregate values accumulated. This is similar to the relational OLAP (ROLAP) implementation of the roll-up operation.

For fast response in OLAP and data mining, the later implementation has adopted data cube technology as follows, when data cube contains a small number of dimensions, or when it is generalized to a high level, the cube is structured as compressed sparse array but is still stored in a relational database (to reduce the cost of construction and indexing of different data structures). The cube is pre-computed using a chunk-based multi-way array aggregation technique. However, when the cube has a large number of dimensions, it becomes very sparse with a huge number of chunks. In this case, a relational structure is adopted to store and compute the data cube, similar to the ROLAP implementation. We believe such a dual data structure technique represents a balance between multidimensional OLAP (MOLAP) and relational OLAP (ROLAP) implementations. It ensures fast response time when handling medium-sized cubes/cuboids and high scalability when handling large databases with high dimensionality.

Notice that even adopting the ROLAP technique, it is still unrealistic to materialize all the possible cuboids for large databases with high dimensionality due to the huge number of cuboids it is wise to materialize more of the generalized, low dimensionality cuboids besides considering other factors, such as accessing patterns and the sharing among different cuboids. A 3-D data cube/cuboid can be selected from a high-dimensional data cube and be browsed conveniently using the DBMiner 3-D cube browser as shown in Figure.2. Where the size of a cell (displayed as a tiny cube) represents the entry count in the corresponding cell, and the brightness of the cell represents another measure of the cell. Pivoting, drilling, and slicing/dicing operations can be performed on the data cube browser with mouse clicking.

2.3 Concept description

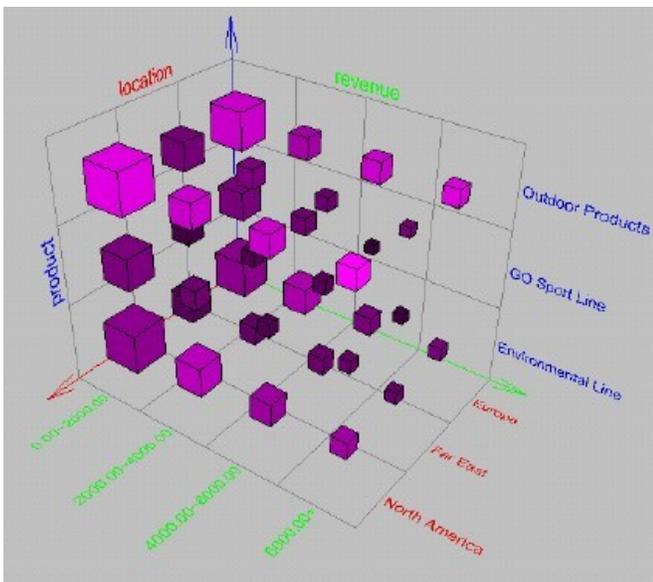


Fig.2. Browsing of a 3-dimensional data cube in DBMiner

Concept/class description plays an important role in descriptive data mining. It consists of two major functions, data characterization and data discrimination (or comparison).

Data characterization summarizes and characterizes a set of task-relevant data by data generalization. Data characterization and its associated OLAP operations, such as drill-down and roll-up (also called drill-up).

2.4 Database System Architecture

The Database System used for the scientific data warehouses can use a centralized architecture, containing ETL, Data Warehousing, OLAP and Data Mining in a single platform. The overall system architecture is seen in below Figure.3.

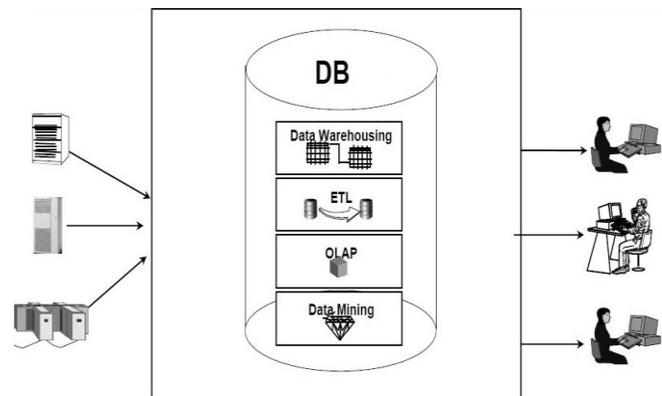


Fig.3. Database System Architecture

This type of architecture can reduce the administration costs because of its single platform, and also reduces the implementation costs. This Architecture supports faster deployment and improved scalability and reliability.

The OLAP in this type architecture can empower end user to do own scientific analysis, can give ease of use. This also provides easy Drill Down facility to the users. This architecture can provide virtually no knowledge of tables required for the users. This architecture can also improve exception analysis and variance analysis.

This architecture gives user the multidimensional view of data and can provide easy Drill Down, rotate and ad-hoc analysis of data. It can also support iterative discovery process. It can provide unique descriptions across all levels of data.

The DB modifies & summarizes (store aggregates) of the scientific data and adds historical information to the DB.

3 OLAP++ System Architecture

The overall architecture of the OLAP++ system is seen in Figure.4. The object part of the system is based on the OPM tools that implements the Object Data Management Group (ODMG) object data model and the Object Query Language (OQL) on top of a relational DBMS, in this case the ORACLE RDBMS. The OLAP part of the system is based on Microsoft's SQL Server OLAP Services using the Multi-Dimensional eXpressions (MDX) query language.

When a SumQL++ query is received by the Federation Coordinator (FC), it is first parsed to identify the measures, categories, links, classes and attributes referenced in the query. Based on this, the FC then queries the metadata to get information about which databases the object data and the OLAP data reside in and which categories are linked to which classes. Based on the object parts of the query, the FC then sends OQL queries to the object databases to retrieve the data for which the particular conditions holds true. This data is then put into a "pure" SumQL statement (i.e. without object references) as a list of category values.

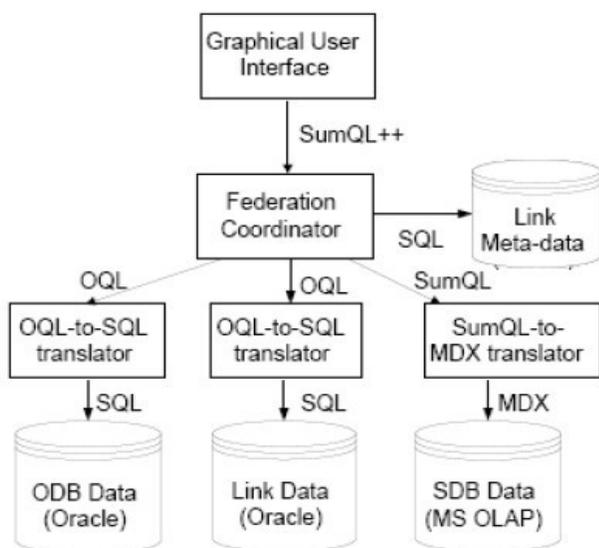


Fig.4. OLAP++ Architecture

This SumQL statement is then sent to the OLAP database layer to retrieve the desired measures, grouped by the requested categories. The SumQL statement is translated into MDX by a separate layer, the "SumQL-to-MDX translator", and the data returned from OLAP Services is returned to the FC. The reason for using the intermediate SumQL statements is to isolate the implementation of the OLAP data from the FC. As another alternative, we have also implemented a translator into SQL statements against a "star schema" relational database design. The system is able to support a good query performance even for large databases while making it possible to integrate existing OLAP data with

external data in object databases in a flexible way that can adapt quickly to changing query needs.

3.1 Back End Tools and Utilities

Data warehousing systems use a variety of data extraction and cleaning tools, and load and refresh utilities for populating warehouses. Data extraction from "foreign" sources is usually implemented via gateways and standard interfaces (such as Information Builders EDA/SQL, ODBC, Oracle Open Connect, Sybase Enterprise Connect, Informix Enterprise Gateway).

Data Cleaning

Since a data warehouse is used for decision making, it is important that the data in the warehouse be correct. However, since large volumes of data from multiple sources are involved, there is a high probability of errors and anomalies in the data. Therefore, tools that help to detect data anomalies and correct them can have a high payoff. Some examples where data cleaning becomes necessary are: inconsistent field lengths, inconsistent descriptions, inconsistent value assignments, missing entries and violation of integrity constraints. Not surprisingly, optional fields in data entry forms are significant sources of inconsistent data. There are three related, but somewhat different, classes of data cleaning tools. *Data migration* tools allow simple transformation rules to be specified; e.g., "replace the string *gender* by *sex*". Warehouse Manager from Prism is an example of a popular tool of this kind. *Data scrubbing* tools use domain-specific knowledge (e.g., postal addresses) to do the scrubbing of data. They often exploit parsing and fuzzy matching techniques to accomplish cleaning from multiple sources. Some tools make it possible to specify the "relative cleanliness" of sources. Tools such as Integrity and Trillum fall in this category. *Data auditing* tools make it possible to discover rules and relationships (or to signal violation of stated rules) by scanning data. Thus, such tools may be considered variants of data mining tools. For example, such a tool may discover a suspicious pattern (based on statistical analysis) that a certain car dealer has never received any complaints.

Load

After extracting, cleaning and transforming, data must be loaded into the warehouse. Additional pre-processing may still be required: checking integrity constraints; sorting; summarization, aggregation and other computation to build the derived tables stored in the warehouse; building indices and other access paths; and partitioning to multiple target storage areas.

The load utilities for data warehouses have to deal with much larger data volumes than for operational databases. There is only a small time window (usually at night) when the warehouse can be taken offline to refresh it. Sequential loads

can take a very long time, e.g., loading a terabyte of data can take weeks and months! Hence, pipelined and partitioned parallelisms are typically exploited. Doing a full load has the advantage that it can be treated as a long batch transaction that builds up a new database. While it is in progress, the current database can still support queries; when the load transaction commits, the current database is replaced with the new one. Using periodic checkpoints ensures that if a failure occurs during the load, the process can restart from the last checkpoint. However, even using parallelism, a full load may still take too long. Most commercial utilities (e.g., RedBrick Table Management Utility) use incremental loading during refresh to reduce the volume of data that has to be incorporated into the warehouse. Only the updated tuples are inserted. However, the load process now is harder to manage. The incremental load conflicts with ongoing queries, so it is treated as a sequence of shorter transactions (which commit periodically, e.g., after every 1000 records or every few seconds), but now this sequence of transactions has to be coordinated to ensure consistency of derived data and indices with the base data.

Refresh

Refreshing a warehouse consists in propagating updates on source data to correspondingly update the base data and derived data stored in the warehouse. There are two sets of issues to consider: *when* to refresh, and *how* to refresh. Usually, the warehouse is refreshed periodically (e.g., daily or weekly). Only if some OLAP queries need current data (e.g., up to the minute stock quotes), is it necessary to propagate every update. The refresh policy is set by the warehouse administrator, depending on user needs and traffic, and may be different for different sources. Refresh techniques may also depend on the characteristics of the source and the capabilities of the database servers. Extracting an entire source file or database is usually too expensive, but may be the only choice for legacy data sources. Most contemporary database systems provide replication servers that support incremental techniques for propagating updates from a primary database to one or more replicas. Such replication servers can be used to incrementally refresh a warehouse when the sources change. There are two basic replication techniques: data shipping and transaction shipping. In data shipping, a table in the warehouse is treated as a remote snapshot of a table in the source database.

After_row

triggers are used to update a snapshot log table whenever the source table changes; and an automatic refresh schedule (or a manual refresh procedure) is then set up to propagate the updated data to the remote snapshot. In transaction shipping (e.g., used in the Sybase Replication Server and Microsoft SQL Server), the regular transaction log is used, instead of triggers and a special snapshot log table. At the source site, the transaction log is sniffed to detect updates on replicated

tables, and those log records are transferred to a replication server, which packages up the corresponding transactions to update the replicas. Transaction shipping has the advantage that it does not require triggers, which can increase the workload on the operational source databases. However, it cannot always be used easily across DBMSs from different vendors, because there are no standard APIs for accessing the transaction log. Such replication servers have been used for refreshing data warehouses. However, the refresh cycles have to be properly chosen so that the volume of data does not overwhelm the incremental load utility. In addition to propagating changes to the base data in the warehouse, the derived data also has to be updated correspondingly.

3.2 Conceptual Model and Front End Tools

A popular conceptual model that influences the front-end tools, database design, and the query engines for OLAP is the *multidimensional* view of data in the warehouse. In a multidimensional data model, there is a set of *numeric measures* that are the objects of analysis. Examples of such measures are sales, budget, revenue, inventory, ROI (return on investment). Each of the numeric measures depends on a set of *dimensions*, which provide the context for the measure.

For example, the dimensions associated with a sale amount can be the city, product name, and the date when the sale was made. The dimensions together are assumed to *uniquely* determine the measure. Thus, the multidimensional data views a measure as a value in the multidimensional space of dimensions. Each dimension is described by a set of attributes. For example, the Product dimension may consist of four attributes: the category and the industry of the product, year of its introduction, and the average profit margin. For example, the soda Surge belongs to the category beverage and the food industry, was introduced in 1996, and may have an average profit margin of 80%. The attributes of a dimension may be related via a hierarchy of relationships. In the above example, the product name is related to its category and the industry attribute through such a hierarchical relationship.

Another distinctive feature of the conceptual model for OLAP is its stress on *aggregation* of measures by one or more dimensions as one of the key operations; e.g., computing and ranking the *total* sales by each county (or by each year). Other popular operations include *comparing* two measures (e.g., sales and budget) aggregated by the same dimensions. Time is a dimension that is of particular significance to decision support (e.g., trend analysis). Often, it is desirable to have built-in knowledge of calendars and other aspects of the time dimension.

3.3 Front End Tools

The multidimensional data model grew out of the view of business data popularized by PC spreadsheet programs that were extensively used by business analysts. The spreadsheet is still the most compelling front-end application for OLAP. The challenge in supporting a query environment for OLAP can be

crudely summarized as that of supporting spreadsheet operations efficiently over large multi-gigabyte databases. Indeed, the Essbase product of Arbor Corporation uses Microsoft Excel as the front-end tool for its multidimensional engine. We shall briefly discuss some of the popular operations that are supported by the multidimensional spreadsheet applications. One such operation is *pivoting*. Consider the multidimensional schema of Figure.5. represented in a spreadsheet where each row corresponds to a sale. Let there be one column for each dimension and an extra column that represents the amount of sale. The simplest view of pivoting is that it selects two dimensions that are used to aggregate a measure, e.g., sales in the above example. The aggregated values are often displayed in a grid where each value in the (x, y) coordinate corresponds to the aggregated value of the measure when the first dimension has the value x and the second dimension has the value y. Thus, in our example, if the selected dimensions are city and year, then the x-axis may represent all values of city and the y-axis may represent the years. The point (x, y) will represent the aggregated sales for city x in the year y. Thus, what were values in the original spreadsheets have now become row and column headers in the pivoted spreadsheet. Other operators related to pivoting are *rollup* or *drill-down*. Rollup corresponds to taking the current data object and doing a further group-by on one of the dimensions. Thus, it is possible to roll-up the sales data, perhaps already aggregated on city, additionally by product. The drill-down operation is the converse of rollup. *Slice_and_dice* corresponds to reducing the dimensionality of the data, i.e., taking a projection of the data on a subset of dimensions for selected values of the other dimensions. For example, we can slice_and_dice sales data for a specific product to create a table that consists of the dimensions city and the day of sale.

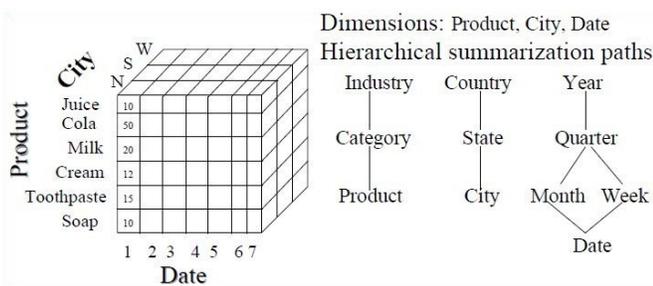


Fig.5. Multidimensional data

4 Advantages

On-line analytical processing (OLAP) is a powerful data analysis method for multi-dimensional analysis of data warehouses. OLAM engine may perform multiple data mining tasks, such as concept description, association, classification, prediction, clustering, time-series analysis, etc. Therefore, an OLAM engine is more sophisticated than an OLAP engine since it usually consists of multiple mining modules which may interact with each other for effective mining. Based on

our analysis, there is no fundamental difference between the data cube required for OLAP and that for OLAM, although OLAM analysis may often involve the analysis of a larger number of dimensions with finer granularities, and thus require more powerful data cube construction and accessing tools than OLAP analyses.

The attribute-oriented induction method adopts two generalization techniques (1) attribute removal, which removes attributes which represent low-level data in a hierarchy, and (2) attribute generalization which generalizes attribute values to their corresponding high level ones. Such generalization leads to a new, compressed generalized relation with count and/or other aggregate values accumulated.

Data warehousing systems use a variety of data extraction and cleaning tools, and load and refresh utilities for populating warehouses. Data extraction from “foreign” sources is usually implemented via gateways and standard interfaces. *Data Cleaning, Load, Refresh and After_row* operations can be performed more efficiently. Data cleaning is a problem that is reminiscent of heterogeneous data integration, a problem that has been studied for many years. But here the emphasis is on *data* inconsistencies instead of schema inconsistencies. Data cleaning, as we indicated, is also closely related to data mining, with the objective of suggesting possible inconsistencies.

This architecture gives user the multidimensional view of data and can provide easy Drill Down, rotate and ad-hoc analysis of data. It can also support iterative discovery process. It can provide unique descriptions across all levels of data. The OLAP in this type architecture can empower end user to do own scientific analysis, can give ease of use. This also provides easy Drill Down facility to the users. This architecture can provide virtually no knowledge of tables required for the users. This architecture can also improve exception analysis and variance analysis. Provides high query performance and keeps local processing at sources unaffected and can operate when sources unavailable. Can query data not stored in a DBMS through Extra information at warehouse

The use of DW and OLAP systems for scientific purposes raises several new challenges to the traditional technology. Efficient implementation and fast response is the major challenge in the realization of On-line analytical mining in large databases and scientific data warehouses. Therefore, the study has been focused on the efficient implementation of the On-line analytical mining mechanism. The methods that we used include the efficient computation of data cubes by integration of MOLAP and ROLAP techniques, the integration of data cube methods with dimension relevance analysis and data dispersion analysis for concept description and data cube based multi-level association, classification, prediction and clustering techniques. We describe back end tools for extracting, cleaning and loading data into a scientific data warehouse; multidimensional data models typical of OLAP;

front end client tools for querying and data analysis and tools for metadata management and for managing the warehouse.

5 Conclusion

Data warehousing using multidimensional view and on-line analytical processing (OLAP) have become very popular in both business and science in recent years and are essential elements of decision support, analysis and interpretation of data. Data warehouses for scientific purposes pose several great challenges to existing data warehouse technology. This paper provides an overview of scientific data warehousing and OLAP technologies, with an emphasis on their data warehousing requirements. The methods that we used include the efficient computation of data cubes by integration of MOLAP and ROLAP techniques, the integration of data cube methods with dimension relevance analysis and data dispersion analysis for concept description and data cube based multi-level association, classification, prediction and clustering techniques. We describe back end tools for extracting, cleaning and loading data into a scientific data warehouse; multidimensional data models typical of OLAP; front end client tools for querying and data analysis; server extensions for efficient query processing; and tools for metadata management and for managing the warehouse.

6 References

- [1] Microsoft Corporation. OLE DB for OLAP Version 1.0 Specification. Microsoft Technical Document, 1998.
- [2] The OLAP Report. Database Explosion. <www.olapreport.com/DatabaseExplosion.htm>. February 18, 2000.
- [3] T. B. Pedersen and C. S. Jensen. Research Issues in Clinical Data Warehousing. In Proceedings of the Tenth International Conference on Statistical and Scientific Database Management, pp. 43–52, 1998.
- [4] T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. Supporting Imprecision in Multidimensional Databases Using Granularities. In Proceedings of the Eleventh International Conference on Statistical and Scientific Database Management, pp. 90–101, 1999.
- [5] T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. Extending PractiPre-Aggregation in On-Line Analytical Processing. In Proceedings of the Twentyfifth International Conference on Very Large Data Bases, pp. 663–674, 1999.
- [6] T. B. Pedersen and C. S. Jensen. Multidimensional Data Modeling for Complex Data. In Proceedings of the Fifteenth International Conference on Data Engineering, 1999. Extended version available as TimeCenter Technical Report TR-37, <www.cs.auc.dk/TimeCenter>, 1998.
- [7] Inmon, W.H., *Building the Data Warehouse*. John Wiley, 1992.
- [8] <http://www.olapcouncil.org>
- [9] Codd, E.F., S.B. Codd, C.T. Salley, “Providing OLAP (On-Line Analytical Processing) to User Analyst: An IT Mandate.” Available from Arbor Software’s web site <http://www.arborsoft.com/OLAP.html>.
- [10] Kimball, R. *The Data Warehouse Toolkit*. John Wiley, 1996.
- [11] Barclay, T., R. Barnes, J. Gray, P. Sundaresan, “Loading Databases using Dataflow Parallelism.” SIGMOD Record, Vol. 23, No. 4, Dec. 1994.
- [12] O’Neil P., Quass D. “Improved Query Performance with Variant Indices”, To appear in Proc. of SIGMOD Conf., 1997.
- [13] Harinarayan V., Rajaraman A., Ullman J.D. “Implementing Data Cubes Efficiently” Proc. of SIGMOD Conf., 1996.
- [14] Chaudhuri S., Krishnamurthy R., Potamianos S., Shim K. “Optimizing Queries with Materialized Views” Intl. Conference on Data Engineering, 1995.
- [15] Widom, J. “Research Problems in Data Warehousing.” Proc. 4th Intl. CIKM Conf., 1995.
- [16] R. G. G. Cattell et al. (Eds). *The Object Database Standard: ODMG 2.0*. Morgan Kaufmann, 1997.
- [17] E. Thomsen. *OLAP Solutions*. Wiley, 1997.
- [18] R. Winter. Databases: Back in the OLAP game. *Intelligent Enterprise Magazine*, 1(4):60–64, 1998.
- [19] Wu, M-C., A.P. Buchmann. “Research Issues in Data Warehousing.” Submitted for publication.
- [20] Levy A., Mendelzon A., Sagiv Y. “Answering Queries Using Views” Proc. of PODS, 1995.
- [21] Seshadri P., Pirahesh H., Leung T. “Complex Query Decorrelation” Intl. Conference on Data Engineering, 1996.
- [22] Widom, J. “Research Problems in Data Warehousing.” Proc. 4th Intl. CIKM Conf., 1995. Gupta A., Harinarayan V., Quass D. “Aggregate-Query Processing in Data Warehouse Environments”, Proc. of VLDB, 1995.

Analyzing Student Retention with Data Mining

Kevin Daimi and Ruth Miller
 Department of Mathematics and Computer Science
 University of Detroit Mercy, Detroit, MI, 48221-3038
 {daimikj, millerru}@udmercy.edu

ABSTRACT

With the growing competition in higher education, student retention is one of the most critical issues for higher education institutions. Institutions continuously monitor student retention progress using various approaches. Many institutions have hired an external consultant to help with the process of retention. Others have created committees to look into the retention problem. In this paper, student retention will be analyzed using data mining. In particular, a classification model will be employed to investigate the profile of students most likely leaving a college for a competitor one. Hypothetical data will be used for the analysis purposes.

Keywords

Student Retention, Data Mining, Classification, Clustering, WEKA

I. INTRODUCTION

Student retention is considered as a costly and problematic process for higher education institutions. Enrollment has, in general, been increasing. However, keeping students in these institutions is a very challenging problem. Institutions are working very hard to deal with this complex issue. Administrators and faculty have teamed together to solve this crisis or at least minimize its consequences. To this end, many studies that analyze this problem from various perspectives exist. Below, some of these studies will be mentioned.

Reason [13] reviewed extensive literature on retention. He concluded that the fast conversion of demographics of the undergraduate student population is an indication to modify our understanding of variables that forecast undergraduate student retention. Based on the reviewed literature, he further concluded that the variables; high school GPA, college entrance exam scores, first-year college GPA, socioeconomic

status, race-ethnicity, and gender must be included as predictor variables in all retention studies. The approach followed in our paper has included those variables as part of the attributes set.

Community colleges enroll nearly half of the undergraduates in the U.S. They serve as a major vehicle for supplying a considerable number of transfer students for universities. Roman [14], a common measure of accountability in these colleges is student retention. She concentrated on the relationship between admission officers and student retention, and stressed that "Admission officers play a crucial role in the broader student life cycle that extends from prospect to alumnus." Transforming this relationship into attributes that will serve data mining tasks is by no means easy. In fact, no Student Database contains such attribute. However, a survey could help in isolating some variables.

Gansemer-Topf et al [4] examined the impact of institutional expenditures and grants (student financial aid) on retention. They analyzed this impact over a 10-year period and examined the variations between institutions with low- and high-admissions discriminatory standards. They concluded that spending on institutional grants positively impacted retention and graduation rate at low-admissions selectivity standards. Due to the importance of financial aid, it is adopted by our paper as one of the attributes for mining the hypothetical retention dataset.

Mezick [6] conducted a research using libraries data collected by various organizations and from various sources. She concluded that the strongest relationships existed between student retention and total library expenditures, total library materials costs, and serial costs for institutions categorized as baccalaureate colleges within the Carnegie Classification System.

There are a number of definitions of Data Mining in the literature. However, they all have in common the following; "extraction", "knowledge", and "large data." Therefore one definition will suffice.

Data Mining is the process of performing data analysis and may uncover important data patterns, contributing greatly to business strategies, knowledge bases, and scientific and medical research. Data mining normally has a number of models including classification, prediction, clustering and association. A classification model is used to classify database records into a number of predefined classes based on certain criteria. For example, a university may classify student records for retention purposes as a “leaving,” or “staying” [2] [5] [9] [18] [19].

To our knowledge peer-reviewed journals and conferences research papers dealing with applying data mining to student retention are not available. The captured literature involved either short posters or commercial (companies) reports. This is due to the fact that educational researchers are not trained in data mining and therefore they are not aware that data mining can contribute. On the other hand, data miners avoid such research areas since real data is private and cannot be easily disclosed. However, a number of studies exist for customer retention in various industries including insurance, and telecommunications.

According to Rygielski et al [16] “Through data mining –the extraction of hidden predictive information from large databases– organizations can identify valuable customers, predict future behaviors, and enable firms to make proactive, knowledge-driven decisions.” Data mining tools made it possible to obtain answers to critical business questions in timely manner. Without data mining tools, the process of reaching the answers used to be very time-consuming. In fact, it is these answers that made customer relationship management promising.

Chu et al [1] indicated that inhibiting of customer churn through customer retention is a central concern of Customer Relationship Management (CRM). If a company could minimize customer churn, its profit is maximized. They suggested a hybridized architecture to treat with customer retention issues. This was achieved through predicting churn probability and proposing retention policies. They concluded that their policy model paradigm implies an exciting and essential technique in exploring the characteristics of churning groups.

To continue to be competitive, hotel industry needs to create a vital customer retention strategy [7]. A key to the successful establishment of such a strategy relies upon customer relationship

management. Accordingly, hotels should recognize the most profitable approaches to develop and maintain a steadfast customer relationship. In an effort to assist hotels figure out their customers’ preferences and the ways to interact with customers, they proposed using data mining techniques.

Min and Emam [8] stressed that chronic driver turnover can negatively impact a trucking firm’s effectiveness through disrupting delivery services, delays due to equipment repairs, and undue recruiting expenses. For a trucking firm to survive and to recruit and retain qualified drivers, who are less likely to cause turnover, data mining techniques were suggested. The goal was to discover which drivers were at higher risk of switching to another firm or job.

Rosset et al [15] discussed the notable business problem of estimating the outcome of retention endeavors on the lifetime value of a customer in the telecommunication industry. They indicated they were presenting a novel segment-based technique modeled after the segment-level view marketing analysts typically deploy. Later, they described how this technique was formulated to estimate the effects of retention on lifetime value.

The insurance industry presents many interesting problems that could be tackled by the research community. Smith et al [17] presented a case study involving two problems from the insurance domain; the understanding of customer retention patterns (whether customers will renew or terminate their insurance policy), and identifying types of policy holders who are at more risk. They solved them using various data mining techniques.

Despite the lack of refereed papers on mining student retention datasets using classification, an interesting application of data mining to a student database was introduced by Erdogan et al [3]. They studied the relationship between students’ university entrance examination (SAT and ACT) results and their success with the aid of cluster analysis and k-mean techniques. As illustrated below, our hypothetical dataset contains attributes covering SAT and ACT results.

This paper tries to analyze student retention using the data mining task, classification. A hypothetical dataset will be used for classification model. The classification task is performed using the Waikato Environment for Knowledge Analysis, Weka-3-5-7, [19].

II. STUDENT RETENTION ARCHITECTURE

The student retention system architecture is exemplified in Fig. 1. It is implemented in WEKA. Real data is supposed to be gathered from student databases, public data, surveys, and forms. However, as real data is not available currently for this research, the collected hypothetical data is cleaned and transformed by the Data Preparation Engine to obtain the hypothetical dataset (Student Retention Dataset). The Student Retention Dataset contains records (instances) for 1000 students. These records are composed of a number of fields (attributes) as explained below. The dataset is transformed into a standard format by the Transformation Engine. The goal of this transformation is to create an ARFF file format required by WEKA. The Classification Engine will run Decision Trees on the ARFF format file to obtain the results. The results are displayed via the User Interface unit. In addition to displaying the results, the results could be visualized graphically via the User Interface. Visualization plays an important role in making the discovered knowledge understandable and interpretable by human. The User Interface component also allows selection of the file to be mined and the classification algorithm.

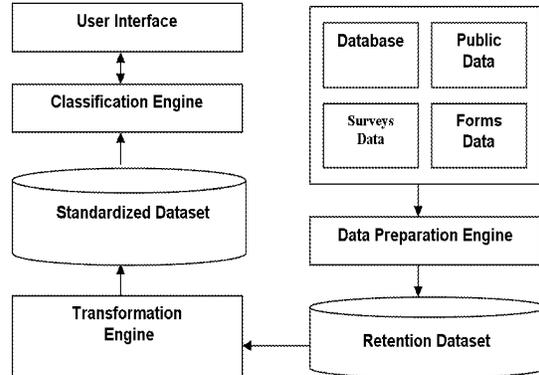


Figure 1: Student Retention System Architecture

III. ATTRIBUTES SELECTION

For the purpose of student retention analysis, the following self-explanatory attributes, written in WEKA's ARFF format, are regarded necessary:

```

@ATTRIBUTE Major {Undecided, Chemistry,
                  Biology ...}
@ATTRIBUTE State {AK, AL, AR, AZ, CA, CO,
                 CT, DC, DE ...}
@ATTRIBUTE Sex {F, M}
@ATTRIBUTE SATMath NUMERIC
  
```

```

@ATTRIBUTE SATComposite NUMERIC
@ATTRIBUTE ACTMath NUMERIC
@ATTRIBUTE ACTComposite NUMERIC
@ATTRIBUTE NoOfCreditsSoFar NUMERIC
@ATTRIBUTE CummulativeGPA NUMERIC
@ATTRIBUTE LastTermGPA NUMERIC
@ATTRIBUTE HighSchoolGPA NUMERIC
@ATTRIBUTE Probation {Yes, No}
@ATTRIBUTE MaritalStatus {Single, Married,
                          Separated, Divorced}
@ATTRIBUTE FamilyIncome NUMERIC
@ATTRIBUTE NumberOfKids NUMERIC
@ATTRIBUTE Employeed {Yes, No}
@ATTRIBUTE DistanceFromHome NUMERIC
@ATTRIBUTE FinancialAid {Yes, No}
@ATTRIBUTE TransferStudent {Yes, No}
@ATTRIBUTE Adult {Yes, No}
@ATTRIBUTE Honors {Yes, No}
@ATTRIBUTE ClassesTaughtByAdjuncts
    NUMERIC
@ATTRIBUTE StudentsOrganizationMembership
    {Yes, No}
@ATTRIBUTE FirstMathPerformance {A, A-, B+,
    B, B-, C+, C, C-, D+, D, D-, F}
@ATTRIBUTE Leaving {Yes, NO}
  
```

Note that “*Undecided*” is included as a value in the “*Major*” attribute to save having another attribute for Decided/Undecided. Furthermore, a distance of “0” in the “*DistanceFromHome*” attribute will indicate a student is living on campus. This will save an additional attribute for living on-Campus or off-campus. When SAT test is not required, 0.0 is used for both “*SATMath*” and “*SATComposite*.” The attribute “*Leaving*” is the class.

IV. CLASSIFICATION MODEL

For the classification model, the *C4.5* decision tree algorithm, which was developed by Quinlan [10, 11], is adopted. In fact *C4.5* is a modification of the original *ID3* developed by Quinlan [12] too. *C4.5* included a number of improvements including dealing with numeric data, missing data, and noisy data. In addition generating rules from trees was introduced. *WEKA* implements a later and slightly improved version, *C4.5 revision 8*. This is referred to as *J4.8*. The results of implementing the retention model are obtained using *J4.8*.

The *J4.8* algorithm was first trained using 1000 instance dataset contained in the “*train.arff*” file. Table I provides few sample instances of the training dataset represented as columns instead rows.

It was then tested using 400 instance dataset provided by “*test.arff*” file. The time taken to build model was 0.08 seconds. The time taken to test the model on training data was 0.03 seconds. The resulting number of leaves was 57, and the size of tree was 88. The classification model was saved as “*retention.model*.” Table II illustrates the training and testing statistics. The trained and tested model was obtained using the command line below.

```
java weka.classifiers.trees.J48 -C 0.25 -M 2 -t
C:\train.arff -d C:\retention.model -T C:\
test.arff
```

The command line option *M* specifies the minimum number of instances per leaf. It is set to 2 in this command line. The confidence factor of 0.25 is provided by the command line option *C*.

TABLE I
Sample Instances from Training Dataset

ATTRIBUTE SET	ROW-1	ROW-2	ROW-3	ROW-4	ROW-5
Major	Biochemistry	Undecided	Chemistry	ElectricalEng	Chemistry
State	MI	MI	NY	MI	AZ
Sex	F	M	M	M	M
SATMath	0	0	610	0	530
SATComposite	0	0	1680	0	1010
ACTMath	29	29	27	20	31
ACTComposite	22	22	29	28	34
NoOfCreditsSoFar	43	45	41	40	41
CumulativeGPA	3.8	3.7	2.3	2.1	2.9
LastTermGPA	3.1	3.1	2.4	3.3	3.1
HighSchoolGPA	3.5	3.4	3.2	2.4	3.4
Probation	NO	NO	NO	NO	NO
MaritalStatus	Single	Single	Married	Married	Single
FamilyIncome	61000	44000	49000	200000	89000
NumberOfKids	0	0	1	1	0
Employed	NO	NO	NO	NO	NO
DistanceFromHome	63	60	46	99	0
FinancialAid	YES	NO	YES	YES	YES
TransferStudent	NO	NO	NO	YES	YES
Adult	NO	NO	YES	YES	NO
Honors	NO	NO	NO	NO	NO
ClassesTaughtByAdjuncts	1	2	2	1	0
StudentsOrganization	NO	YES	YES	YES	YES
FirstMathPerformance	F	B+	B	C-	A-
Leaving	NO	NO	YES	YES	NO

TABLE II
Training and Testing Statistics Summary

SUMMARY VARIABLE	TRAINING VALUE	TESTING VALUE
Correctly Classified Instances	887	353
Incorrectly Classified Instances	113	47
Kappa statistic	0.6948	0.5995
Mean absolute error	0.1133	0.1121
Root mean squared error	0.238	0.238
Relative absolute error	52.5457 %	50.7898 %
Root relative squared error	72.6037 %	70.9178 %
Total Number of Instances	1000	400

Despite the fact that hypothetical data was used, the percentage for correctly classified instances for both training and testing is reasonably high. The percentage of correctly classified instances for the training and testing phases are 88.7% and 88.25%

respectively. Further training and testing will be required to improve these percentages.

In addition to getting the decision tree with *J4.8*, another technique, *PART* was applied to catch classification rules. *PART* obtains rules from partial decision trees. It builds the tree using *C4.5*'s heuristics with the same user-defined parameters as *J4.8* [19]. Below are sample rules produced by *PART*. Note that the class name (Leaving), “IF,” and “THEN” were added to clarify the rules. The first value in parentheses represents the number of instances reaching that rule, and second value indicates the number of instances that were incorrectly classified.

1. IF state = MI AND MaritalStatus = Single AND Adult = YES AND Sex = M AND FamilyIncome > 66000 THEN Leaving = NO (5.0/1.0)
2. IF state = MI AND Adult = NO AND Honors = NO AND SATcomposite <= 1930 AND Major = Biology AND Transfer = NO THEN Leaving = NO (9.0/1.0)
3. IF state = MI AND Honors = YES AND ClassesTaughtByAdjuncts <= 3 THEN Leaving = YES (14.0/4.0)
4. IF state = MI AND Major = Computer-Science AND StudentsOrganizationMembership = NO THEN Leaving = YES (6.0/1.0)
5. IF NoOfCreditsSoFar <= 54 AND state = WI AND Adult = NO AND FamilyIncome > 61000 THEN Leaving = YES (10.0/1.0)

V. IMPLEMENTING THE MODEL

The classification model was applied to an 11 instance dataset file, “*run.arff*,” with unknown class. In this file, the class column contains “?” for all the instances. Table III exhibits few instances of the “*run.arff*” file (instances 1 - 4, 7). Note that “Instance #” is not an actual attribute. To implement the model, “*retention.model*,” the command below was used. The parameter *p* is used to indicate the range of attributes to display. Here we are asking for all (1-25) attributes as we have 25 attributes.

```
java weka.classifiers.trees.J48 -p 1-25 -l C:\
retention.model -T C:\run.arff
```

The predicted values for the unknown classes are displayed in Table IV. Note that the attribute “*PredictedLeaving*” is added by WEKA. Table V shows the predictions for the eleven instances of the “*run.arff*” file together with their prediction

accuracy (confidence). The prediction accuracy allows us to focus on predictions that are more convincing. Consequently, we could filter out any instance with predicted value below certain percentage. For example, we could ignore instances with an accuracy < 85%.

TABLE III
Sample Instances with Unknown Classes

ATTRIBUTE SET	ROW-1	ROW-2	ROW-3	ROW-4	ROW-7
Instance #	1	2	3	4	7
Major	undecided	Biochemistry	Chemistry	Biochemistry	undecided
State	MI	AZ	OH	MI	IL
Sex	M	M	F	F	M
SATMath	630	520	670	0	500
SATComposite	1800	1920	730	0	1990
ACTMath	21	22	34	31	30
ACTComposite	26	31	26	29	29
NoOfCreditsSoFar	43	42	40	18	75
CumulativeGPA	2	2.6	3.7	1.5	3.9
LastTermGPA	1.8	2.4	2.1	2.3	4
HighSchoolGPA	3.6	3.5	3.8	3.6	3.6
Probation	NO	NO	NO	NO	NO
MaritalStatus	Single	Single	Single	Single	Married
FamilyIncome	44000	164000	37000	95000	61000
NumberOfKids	3	2	2	2	0
Employeed	NO	NO	NO	NO	NO
DistanceFromHome	23	60	0	0	64
FinancialAid	NO	YES	YES	YES	YES
TransferStudent	YES	NO	NO	YES	NO
Adult	NO	NO	NO	NO	YES
Honors	NO	NO	NO	NO	NO
ClassesTaughtByAdjuncts	1	2	1	2	2
StudentsOrganization	NO	YES	NO	YES	NO
FirstMathPerformance	A	D	D+	C	A
Leaving	?	?	?	?	?

Although the data used is hypothetical, a number of interesting points could be observed:

- Undecided students will not necessarily leave
- Number of children is generally not a factor
- Students out of state are not leaving
- Students with low grades (D, D+) in their first math class will not necessarily leave
- Number of classes taught by adjuncts is not necessarily an indication of leaving
- Low number of total credits (mixed with other factors) could encourage leaving
- It is not necessarily true transfer students are most likely to leave
- Low GPA in the last term does not necessarily justify leaving. The student that is predicted to leave (instance 4) has GPA = 2.3, which higher than what the student in instance 1 has (1.8)

TABLE IV
Sample Instances with Predicted Classes

ATTRIBUTE SET	ROW-1	ROW-2	ROW-3	ROW-4	ROW-7
Instance #	1	2	3	4	7
Major	undecided	Biochemistry	Chemistry	Biochemistry	undecided
State	MI	AZ	OH	MI	IL
Sex	M	M	F	F	M
SATMath	630	520	670	0	500
SATComposite	1800	1920	730	0	1990
ACTMath	21	22	34	31	30
ACTComposite	26	31	26	29	29
NoOfCreditsSoFar	43	42	40	18	75
CumulativeGPA	2	2.6	3.7	1.5	3.9
LastTermGPA	1.8	2.4	2.1	2.3	4
HighSchoolGPA	3.6	3.5	3.8	3.6	3.6
Probation	NO	NO	NO	NO	NO
MaritalStatus	Single	Single	Single	Single	Married
FamilyIncome	44000	164000	37000	95000	61000
NumberOfKids	3	2	2	2	0
Employeed	NO	NO	NO	NO	NO
DistanceFromHome	23	60	0	0	64
FinancialAid	NO	YES	YES	YES	YES
TransferStudent	YES	NO	NO	YES	NO
Adult	NO	NO	NO	NO	YES
Honors	NO	NO	NO	NO	NO
ClassesTaughtByAdjuncts	1	2	1	2	2
StudentsOrganization	NO	YES	NO	YES	NO
FirstMathPerformance	A	D	D+	C	A
PredictedLeaving	NO	NO	NO	YES	NO
Leaving	?	?	?	?	?

TABLE V
Predictions with Prediction Accuracy

Instance #	Prediction	Confidence
1	NO	0.875
2	NO	0.946
3	NO	0.946
4	YES	0.750
5	NO	0.946
6	NO	0.946
7	NO	0.946
8	NO	0.946
9	NO	0.946
10	NO	0.946
11	NO	0.946

VI. CONCLUSIONS

Student retention is one of the most critical problems in higher education institutions. To help in planning student retention policies, Data Mining has been applied to build a classification model to inform institutions how likely a particular student is going to continue with the institution in the next term. By using classification trees and rules to predict departing students, the model will help institutions in understanding students' retention risks, obtaining a list of most likely leaving students, and enlightening student retention policies. The results obtained above could have been well obtained using the GUI version of WEKA. However, the command line has been used to supply the classification accuracy.

Normally, repeated 10-fold cross-validations are utilized when the dataset is finite to avoid over-fitting. However, since the goal of this paper is to demonstrate how data mining could be used to analyze retention models, a separate testing dataset was adopted. Furthermore, the work has been carried out on hypothetical data. For real world, retention data is large enough to demand a separate training and testing datasets.

The confidence factor of 0.25 that is used in the first command line above works well in many situations. However, if the rate of the correctly classified instances tends to be unappealing, this factor should be set to a lower value to force more extensive pruning.

Another value that could be modified is the minimum number of instances per leaf. In the first command line above, this is set to 2 (two instances per leaf). If the real world dataset contains many inferential data, this value should be bigger.

To gain further insight into the retention problem, it is recommended that another data mining task, *clustering*, be carried out. Both the original dataset and the dataset obtained after the implementation stage need to be clustered to envision relationships among leaving students, and between them and staying students.

REFERENCES

- [1] B. Chu, M. Tsai, and C. Ho, Toward a Hybrid data Mining Model for Customer Retention, *Knowledge-Based Systems*, 20:703-718, 2006.
- [2] K. Cios, W. Pedrycz, R. Swiniarski, and L. Kurgan, *Data Mining: A Knowledge Discovery Approach*, Springer, 2007.
- [3] S. Z. Erdogan, and M. Timor, A Data Mining Application in a Student Database, *Journal of Aeronautics and Space Technologies*, 2(2): 53-57, 2005.
- [4] A. M. Gansemer-Topf, and J. H. Schuh, Institutional Grants: Investing in Student Retention and Graduation, *NASFAA Journal of Student Financial Aid*, 35(3):5-20, 2005.
- [5] J. Han, and M. Kamber, Morgan Kaufmann, 2006.
- [6] E. M. Mezick, Return on Investment: Libraries and Student Retention, *The Journal of Academic Librarianship*, 33(5) 561-566, 2007.
- [7] H. Min, H. Min, and A. Emam, A Data Mining Approach to Developing the Profiles of Hotel Customers, *International Journal of Contemporary Hospitality Management*, 14(6):274-285, 2002.
- [8] H. Min, and A. Emam, Developing the Profiles of Truck Drivers for Their Successful Recruitment and Retention, *International Journal of Physical Distribution & Logistics Management*, 33(2):149-162, 2003.
- [9] S Parthasarathy, Data Mining at the Crossroads: Successes, Failures and Learning From Them, In Proceedings of The 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 1053-1055, San Jose, CA, 2007.
- [10] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
- [11] J. R. Quinlan, Improved use of continuous attributes in c4.5, *Journal of Artificial Intelligence Research*, 4:77-90, 1996.
- [12] J. R. Quinlan, Induction of decision Trees, *Machine Learning*, 1:81-106, 1986.
- [13] R. Reason, Student Variables that Predict Retention: Recent Research and New Developments, *NASPA Journal*, 40(4):172-191, 2003.
- [14] M. A. Roman, Community College Admission and Student Retention, *Journal of College Admission*, pp. 18-23, Winter, 2007
- [15] S. Rosset, E. Neumann, U. Eick, N. Vatnik, and Y. Idan, Customer Lifetime Value Modeling and Its Use for Customer Retention Planning, In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 332-340, Edmonton, Canada, 2002.
- [16] C. Rygielski, J. Wang, and D. C. Yen, Data Mining Techniques for Customer Relationship Management, *Technology in Society*, 24:483-502, 2002.
- [17] K. A. Smith, R. J. Willis, and M Brooks, An analysis of Customer Retention and Insurance claim Patterns Using Data Mining: A Case Study, *Journal of the Operational Research Society*, 51:532-541, 2000.
- [18] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Addison-Wesley, 2006.
- [19] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 2005.

Automobile Insurance Knowledge Mining

Sulaiman A. Al-Hudhaif, Associate Professor

Abstract—Business information received from advanced data analysis and data mining is a critical success factor for companies wishing to maximize competitive advantages. In many cases, extraction of knowledge for automobile insurance customer relationships is considered the most reliable methodology for making right decisions for many customer relationship situations. The use of traditional tools and techniques to obtain knowledge is inefficient and does not give the right information at the right time. Data mining should provide tactical insights to support strategic directions.

This paper introduces a methodology that can help insurance experts to quickly identify candidate elements containing one or more features of interest from a database which is so massive that no human individual could investigate without assistance of knowledge extraction techniques. Insurance data, as well as other customer data, such as age, nationality and job, are classified according to several factors, e.g. automobile type, premium limits, number of installments, ... etc. These factors are used to categorize customers in different levels ranging from excellent cases to fraud cases. From the generated association rules, some interesting rules can be used in the early prediction of insurance decisions.

1. Introduction

During the last decade, huge amounts of data were generated and became available, and hence the label “The Information Revolution Era”. Every automation task involves storing data, e.g. bank transactions, school registries, medical data, e-commerce, etc. As a result, the database of society's capabilities of both generating and collecting data has been increased rapidly. Millions of databases have been used in business, management, government administration, scientific and engineering data management, and many other applications. The number of such databases has been growing rapidly because of the availability of powerful and affordable tools and database systems [6].

This paper is financially supported by the Research Center of the College of Business Administration (CBA), King Saud University (KSU).

Dr. Sulaiman A. Al-Hudhaif is an Associate Professor at CBA, KSU. He is currently working as the CBA Associate Dean for Quality and Development.

His postal address is: P. O. Box 86924, Riyadh 11632, Saudi Arabia.

Office phone: 00966-1-4674040.

Fax: 00966-1-4674113.

Email: shudhaif@ksu.edu.sa

In many cases, data collected through automobile owners' records need to be analyzed. Analyzing such data is considered the most reliable method for early detection of fraud cases, such as, the size of the claim, and the short time period between buying the policy and the first claim.

Customer data are considered among the most difficult to read due to differences in their types and levels. Due to the huge volume of data to be read by insurance experts, the accuracy rate tends to decrease, and, therefore, automatic reading of data becomes highly desirable. Data stored in terabytes of storage need a serious effort by individual humans at manually examining and characterizing the data objects. Also, in many cases, it has been proven that double reading of such data by two experts increases the accuracy, but at high costs. That is why the computer aided systems are necessary to assist the automobile insurance staff to achieve a high level of efficiency and effectiveness.

In this paper, three problems are emphasized:

- Feature Selection and Collection Process
- Data Cleansing Techniques, and
- Data Mining Techniques.

A suitable data mining technique was used to discover information that many traditional business analysis and statistical techniques fail to deliver [1, 4, 8, 16]. Additionally, the application of data mining techniques exploits the value of data warehouse by converting expensive volumes of data into valuable assets for future tactical and strategic business development. Management information systems should provide advanced capabilities that enable the user to ask more sophisticated and pertinent questions. It empowers the right people by providing the specific information they need.

The rest of the paper is organized as follows: Section 2, illustrates the major processes of knowledge discovery along with the major tasks in the data mining process. Section 3, explains the data mining technique selection. Section 4 describes feature selection and collection processes. Section 5 and Section 6, focus on the different techniques used in data cleansing, and data transformation. In Section 7, the system implementation and experimental results are given. Finally, the paper is concluded in Section 8.

2 Data Mining Technique Selection

The term “Knowledge discovery in databases” is defined as the process of identifying useful and novel structure (model) in data [13, 14, 16]. It could be viewed as a multi-stage process. The stages are summarized as follows:

- Data gathering, e.g., databases, data warehouses, web crawling.
- Data cleansing; eliminating errors, e.g., GPA = 7.3.
- Feature extraction; obtaining only the interesting attributes of data.
- Data mining; discovering and extracting meaningful patterns.
- Visualization of data.
- Verification and evaluation of results; drawing conclusions.

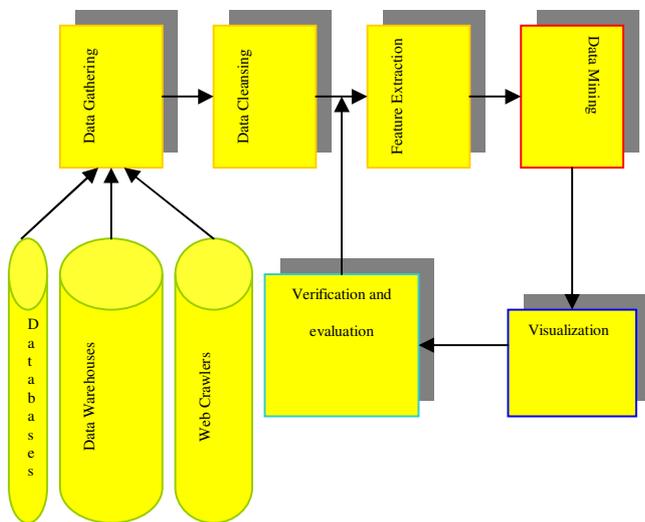


Figure 2.1 Process of Identifying Structure

Data mining is considered the main step in the knowledge discovery process that is concerned with the algorithms used to extract potentially valuable patterns, associations, trends, sequences and dependencies in data [8, 14, 67, 77, 98, 99, 100, 104, 126]. Key business examples include web site access analysis for improvements in e-commerce advertising, fraud detection, screening and investigation, retail site or product analysis, and customer segmentation.

Data mining techniques could be categorized either by tasks or by methods. Data mining tasks:

- Association Rule Discovery.
- Classification.
- Clustering.
- Sequential Pattern Discovery.
- Regression.
- Deviation Detection

Most researchers refer to the first three data mining tasks as the main data mining tasks. The other tasks are either related more to some of other fields, such as regression, or considered as a sub-field in one of the main tasks, such as sequential pattern discovery and deviation detection.

Data Mining Methods:

- Algorithms for mining spatial, textual, and other complex data
- Incremental discovery methods and re-use of discovered knowledge
- Integration of discovery methods
- Data structures and query evaluation methods for data mining
- Parallel and distributed data mining techniques
- Issues and challenges for dealing with massive or small data sets
- Fundamental issues from statistics, databases, optimization, and information processing in general as they relate to problems of extracting patterns and models from data.

Because of the limited space, the paper focuses on discussing the main techniques in data mining tasks.

2.1 Association Rule Discovery

Association rule discovery or association mining that discovers dependencies among values of an attribute was introduced by Agrawal *et al* [13], and has emerged as a prominent research area. The association mining problem, also referred to as the market basket problem, can be formally defined as follows. Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of items as $S = \{s_1, s_2, \dots, s_m\}$ be a set of transactions, where each transaction $s_i \in S$ is a set of items that is $s_i \subseteq I$. An association rule denoted by $X \Rightarrow Y$, where $X, Y \subset I$ and $X \cap Y = \Phi$, describes the existence of a relationship between the two item sets X and Y .

Example 2.1

In the following Table given a set of records each of which contains some set of items from a given collection of items. Produce dependency rules that predict occurrences of one item based on the occurrences of some other items.

Transaction ID	Items
1	Milk, Bread, Diaper
2	Milk, Bread
3	Coke, Milk, Bread, Diaper
4	Coke, Coffee, Bread
5	Bread, Coke
6	Milk, Coke, Coffee

Several measures have been introduced to define the strength of the relationship between item sets X and Y, such as support, confidence, and interest. The definitions of these measures, from a probabilistic model are given below.

- Support $(X \rightarrow Y) = P(X, Y)$; the percentage of transactions in the database that contain both X and Y.
- Confidence $(X \rightarrow Y) = P(X, Y) / P(X)$; the percentage of those transactions containing Y in transactions containing X.
- Interest $(X \rightarrow Y) = P(X, Y) / P(X) \cdot P(Y)$, represents a test of statistical independence.

Based on example 2.1, some of the Discovered Rules are:

Rule	Support	Confidence	Interest
Bread \rightarrow Milk	3/6	3/5	9/10
Milk, Bread \rightarrow Diaper	2/6	2/3	2

2.2. Classification

Classification is the process of discovering a model for the class in terms of the remaining attributes. It is an important problem in the rapidly emerging field of data mining [31, 62, 71, 89]. The problem can be stated as follows. We are given a training dataset consisting of records. Each record is identified by a unique record ID and consists of fields corresponding to the attributes. An attribute with a continuous domain is called a continuous attribute. An attribute with finite domain of discrete values is called a categorical attribute. One of the categorical attributes is the classifying attribute or class, and the values in its domain are called class labels.

2.3 Clustering

Clustering is grouping some points in some space into a small number of clusters, each cluster consisting of points that are "near" in some sense. To consider whether a set of points is close enough to be considered a cluster, we need a distance measure $D(x, y)$ to tell us how far points x and y are. The usual axioms for a distance measure D are:

- $D(x, x) = 0$, a point is distance 0 from itself.
- $D(x, y) = D(y, x)$, D is symmetric.
- $D(x; y) \leq D(x; z) + D(z; y)$, the triangle inequality.

Based on the writer's assessment that involves problem definition and used data volume and type, the association mining methodology has been chosen as the most suitable for data mining.

3. Association Mining Techniques

The Apriori algorithm [13] is considered the most famous algorithms in the area of association mining. Apriori starts with a seed set of item sets found to be large in previous pass, and uses it to generate new potentially large item sets (called candidate item sets). The actual support for these candidates is counted during the pass over the data, and non-large candidates are thrown out. The main outlines of the Apriori algorithm are described in [13].

The AprioriTid algorithm [12] is a variation of the Apriori algorithm. The AprioriTid algorithm also uses the "apriori-gen" function to determine the candidate itemsets before the pass begins. The main difference from the Apriori algorithm is that the AprioriTid algorithm does not use the database for counting support after the first pass. Instead, the set $\langle TID, \{X_k \rangle$ is used for counting. (Each X_k is a potentially large k-itemset in the transaction with identifier TID.) The benefit of using this scheme for counting support is that at each pass other than the first pass, the scanning of the entire database is avoided. But the downside of this is that the set $\langle TID, \{X_k \rangle$ that would have been generated at each pass may be huge. Another algorithm is called AprioriHybrid, where the basic idea of the AprioriHybrid algorithm is to run the Apriori algorithm initially, and then switch to the AprioriTid algorithm when the generated database (i.e. $\langle TID, \{X_k \rangle$) would fit in the memory.

Among all other different techniques used for association mining, the AprioriTid algorithm has been chosen as the basic association mining technique.

4. Feature selection and collection

As web and database technologies advance, data accumulates in a huge quantity that is beyond a human's capacity of data processing. Data mining, as a multidisciplinary joint effort between databases, statistics, and machine learning, is used in turning huge amounts of data into knowledge. Researchers have realized that the effective use of data mining tools requires data preprocessing to achieve successful data mining. Feature selection is one of the important techniques in data preprocessing for data mining [7, 9]. The number of features is reduced by removing irrelevant, redundant, or noisy data; this leads to speeding up the data mining operation, and improving the performance in predictive accuracy.

Feature selection Process is used in selecting a subset of original features [11, 15]. The optimality of a feature subset is measured by an evaluation criterion. Finding an optimal feature subset has been shown to be NP-hard problem. The four basic steps in feature selection process are:

- Subset generation
- Subset evaluation

- Stopping criterion
- Result validation

In subset generation, the generation procedure searches and produces feature subsets for evaluation based on a defined search strategy. For each candidate feature subset, the evaluation criterion is applied, evaluated, and compared with the previous best one. The best subset is decided by comparing the current best subset and the new subset. The better one is chosen as the new best subset. The two processes of subset generation and evaluation are repeated until a given stopping criterion is satisfied. The selected best subsets are validated by using either prior knowledge or applying different tests using synthetic or real data sets. Feature selection can be found in many areas of data mining such as classification, clustering, association rules, and regression.

For automobile insurance data selection, customers' features have been studied. The needed features have been collected from a large set of features. The selection is based on two conditions. These conditions are

- Feature applicability, where the features that are not available or applicable in the region are removed.
- Feature intensity, where the features that have less than 50% intensity are removed.

Based on the above two conditions, the number of selected features is 26.

In table 4.1, a list of all features recommended for this research is given.

The data set contains 147 records. In Table 4.2, a sample of the collected data set is presented. The sample has 16 features and 10 records. The description of data values is given in Table 4.3.

5. Data Cleansing

Data Cleansing is a preprocess step for data mining. The process of data cleansing is normally computationally expensive, hence it was not possible to do with old technologies. Nowadays, faster computers allow data cleansing to be performed in an acceptable amount of time on a large amount of data. There are many issues in the data cleansing area that interest researchers. These issues consist of dealing with missing data, determining erroneous data, etc. Different issues require different approaches.

Unfortunately, there is no commonly accepted definition for data cleansing. Different definitions depend on the particular area in which it is applied. The major areas that involve data cleansing are data warehousing, knowledge discovery in databases (KDD), and total data quality management

(TDQM). This section focuses on data cleansing for data warehousing and mining.

In this paper, and based on the writer's assessment, for data cleansing, a similarity technique has been chosen; it is based on similarity measures to remove out-of-order data for data cleansing.

No.	Feature Name
1	Age
2	Job
3	Nationality
4	City
5	Driving Experience
6	Marital Status
7	Car Manufacturer
8	Car Model
9	Model Year
10	No of Cylinders
11	Use of the Car
12	Car Type
13	Relationship
14	Insured
15	Insurance Type
16	Insurance Period
17	Insurance Installments
18	Accidents in Last 2 Years
19	Type of Losses
20	No of Accidents in the Last Two Years
21	Extra Expenses Paid for Each Accident
22	In Each Accident Total Losses
23	In Each Accident Paid Losses
24	Got Discount
25	Percentage of discount
26	Reason for having discount

Table 4.1 Recommended Features for Research

Record	Feature							
	1	2	3	4	5	6	7	8
1	26	1	1	1	9	2	Hyundai	Sonata
2	23	2	1	1	4	1	Toyota	Hilux
3	24	1	1	1	5	1	Nissan	Sunny
4	28	1	1	2	3	1	Nissan	Maxima
5	18	2	1	2	3	1	Range Rover	--
6	22	2	1	1	5	1	Toyota	Camry
7	18	2	1	1	3	1	Toyota	Camry
8	18	2	1	1	3	1	Nissan	Altima
9	23	1	1	1	5	1	Hyundai	Sonata
10	32	1	1	1	17	2	Ford	Victoria

Table 4.2 (a) Sample of the Collected Data Set

Record	Features							
	9	10	11	12	13	14	15	16
1	2008	6	1	1	1	1	2	4
2	2004	4	1	1	1	1	1	1
3	2001	4	1	1	1	1	1	1
4	2007	6	1	1	1	1	2	4
5	2007	8	1	1	1	1	2	1
6	2007	4	1	1	1	1	2	3
7	2006	6	1	1	1	1	1	2
8	2005	6	1	1	1	1	2	2
9	2003	4	1	1	1	1	2	1
10	1999	8	1	1	1	1	1	1

Table 4.2 (b) Sample of the Collected Data Set

	Feature Name	Description and Ranges
1	Age	Years
2	Job	Employee, Others
3	Nationality	Saudi, none Saudi
4	City	Riyadh, others
5	Driving Experience	Number
6	Marital Status	Single, Married, Widower, Divorced
7	Automobile Manufacturer	15 types
8	Automobile Model	Maximum of 12 Models per Car Manufacture
9	Model Year	Year
10	No of Cylinders	2.5, 4, 6, 8, 12
11	Use of the Automobile	Private, Commercial
12	Automobile Type	Private, Taxi, Bus, Truck, Trill, Gas Trill
13	Relationship	Owner, Driver, Rent, Others
14	Insured	Yes / No
15	Insurance Type	Limited, Full Coverage
16	Insurance Period	Number

Table 4.3 Description of Data Values

6. Data Transformation

In data transformation, the data are transformed into forms appropriate for mining. Data transformation can involve one or more of the following:

- Smoothing where noise values are removed from the data. Such techniques include binning, and clustering.
- Aggregation, where summary or aggregation operations are applied to the data.
- Generalization of the data, where low-level or primitive (raw) data are replaced by higher-level concepts through the use of concept hierarchies.
- Normalization, where the attribute values are scaled to fall within a small specified range, such as -1.0 to 1.0, or 0.0 to 1.0.
- Attribute construction (or feature construction), where new attributes are constructed and added from the given set of attributes in order to help improve the accuracy and understanding of structure in high-dimensional data.

Data discretization techniques are used to reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals. Interval labels can then be used to replace actual data values. Replacing numerous values of a continuous attribute by a small number of interval labels thereby reduces and simplifies the original data. This leads to a concise, easy-to-use knowledge-level representation of mining results.

After review, all the fundamental preprocessing steps, only the steps needed to prepare car insurance data will be chosen. So the following steps will be used as data preprocessing steps:

- Data cleansing for missing and noisy data.
- Data reduction to reduce the dataset size by removing the irrelevant and useless attributes from the dataset.
- Data transformation to suitable formats for mining by using Data discretization with histogram technique.

The following are two examples of data discretization.

Example 6.1 For the first 10 records in the data set, the discretization process will replace the “job” feature, which has either 1 or 2 value, by the “Employee” and “Others” features. Each feature has value 1 if the person agrees with the feature and 0 if the person disagrees with the feature. Figure 6.1 shows the discretization process.

Record	Job	→	Record	Employee	Others
1	1	→	1	1	0
2	2	→	2	0	1
3	1	→	3	1	0
4	1	→	4	1	0
5	2	→	5	0	1
6	2	→	6	0	1
7	2	→	7	0	1
8	2	→	8	0	1
9	1	→	9	1	0
10	1	→	10	1	0

Figure 6.1 Job Discretization Mapping

Example 5.2 For the first 10 records in the data set, the discretization process will replace the “Age” feature, which has a continuous domain, by five features; (from 17-21, 22-28, 29-35, 36-50, 51-..). Each feature has value 1 if the person agrees with the feature, and 0 if the person disagrees with the feature. Figure 6.2 shows the discretization process.

Record	Age	→	Record	Age				
				17-21	22-28	29-35	36-50	>50
1	26	→	1	0	1	0	0	0
2	23	→	2	0	1	0	0	0
3	24	→	3	0	1	0	0	0
4	28	→	4	0	1	0	0	0
5	18	→	5	1	0	0	0	0
6	22	→	6	0	1	0	0	0
7	18	→	7	1	0	0	0	0
8	18	→	8	0	0	0	0	0
9	23	→	9	0	1	1	0	0
10	32	→	10	0	0	0	0	0

Figure 6.2 Age Discretization Mapping

7. System Results

Section 3 presents the AprioriTid algorithm, which is a variation of the Apriori algorithm. It has been shown that the AprioriTid algorithm does not use the database for counting support after the first pass. Instead, a set containing those transaction id's with their potentially large item sets is used in the next passes. At each pass other than the first pass, the scanning of the entire database is avoided.

The AprioriTid algorithm has been implemented by using C++, and on a 2.4 GHz machine, with 4 GB of RAM and running windows Vista. The data set used was obtained from one of the major insurance companies in Saudi Arabia.

Figure 7.1 gives the system interface, where four inputs are required:

- Input file; the data should be in a binary format
- Output file; either frequent item sets or frequent association rules
- Minimum Support; as a percentage
- Minimum Confidence; as a percentage

There are two output modes. The first is frequent item sets mode, and the other is the frequent association rule mode.

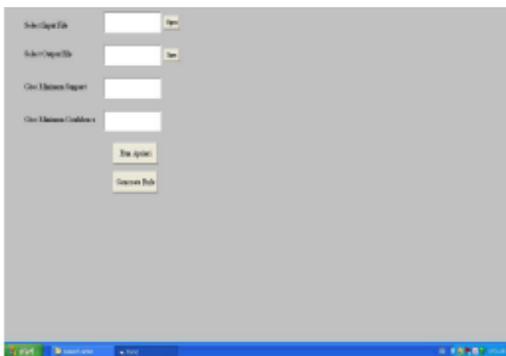


Figure 7.1 System Interface

The initial results show that there is a strong dependency between some of the data features, where the *Dependency Mining* technique does not only discover the trivial dependencies between the automobile insurance data features, but also discovers the non-trivial dependencies.

8. Conclusion

This paper has discussed the problem of processing large volume of car insurance data to be read by insurance experts, the accuracy rate that tends to decrease, and automatic reading of data that is becoming highly desirable. Data stored in terabytes of storage need a serious effort by individuals to manually examine and characterize data objects. In many cases, forming contract deals and fraud detection in car insurance business represent a very important factor. That is why the computer-aided systems are necessary to assist car insurance experts to achieve high level of efficiency and effectiveness. Different methods have been used to classify customer types and/or detect fraud cases. The majority of such methods use features that are extracted using processing techniques. Other methods have been introduced that are based on different techniques such as the fuzzy set theory, Markov models and neural networks. Most of the computer-aided methods have proved to be powerful tools that can assist insurance experts and lead to better results in categorizing customers. Based on the writer's assessment, a similarity technique has been chosen which is based on similarity measures to remove out-of-order data for data cleansing. Following the data cleansing process, a suitable data mining technique has been chosen. In this process, the discovered knowledge should be non-trivial, where many of the known traditional business analysis and statistical techniques fail to deliver. This paper has presented most of the state of the art techniques used for data mining. Based on the problem definition and the data volume and types used, the association mining methodology has been chosen as the most suitable methodology for data mining.

The system has been implemented and used against the data of one of the major insurance companies in Saudi Arabia. The experiments have been run on a 2.4 GHz machine, with 4 GB of RAM and running windows Vista. From the initially generated association rules, some interesting rules have emerged that can be used in the decision making process in car insurance operations.

References

- [1] "Survey of DHS Data Mining Activities", Department of Homeland Security, 2006.
- [1] A.M. Hafez and V.V. Raghavan, "A Matrix Approach for Association Mining," the ISCA 10th International Conference on Intelligent Systems, June 13-15, 2001.
- [2] A. Williams, Mining Association Rules in Medical Image Data Sets. Master Thesis, Computer Science Department, University of Manitoba, 2003.
- [3] Leopold and Kindermann J. Text Categorization with support Vector machines, how to Represent texts in Input Space? Machine Learning, 46:423-444, 2002.

- [4] E.H. Han, G. Karypis and V. Kumar, "Scalable Parallel Data Mining for Association Rules," Proc. 1997 ACM-SIGMOD Int. Conf. On Management of Data, Tucson, Arizona, 1997.
- [5] Brassard and P. Bratley. Fundamentals of Algorithms. Prentice Hall, New Jersey, 1996.
- [6] Muslea, *et al*, "Selective Sampling with Redundant Views", In Proceedings of the National Conference on Artificial Intelligence, 2000.
- [7] J.Lin, and P.J.Haug, "Data Preparation Framework for Preprocessing Clinical Data in Data Mining", AMIA Symposium Proceeding, pp489-493, 2006.
- [8] C.Ordonez, N.Ezquerria, C.A.Santana, "Constraining and Summarizing Association Rules in Medical Data", Knowl Inf Syst , pp.259-283 , 2006.
- [9] Bollmann-Sdorra, A.M. Hafez and V.V. Raghavan, "A Theoretical Framework for Association Mining Based on the Boolean Retrieval Model," DaWaK 2001, September 2001.
- [10] Amit Mandvikar, and Huan Liu.Class, "Specific Ensembles for Active Learning Digital Imagery". The SIAM International Conference on Data Mining. Florida, 2004.
- [11] S.Kotsiantis, D.Kanellopoulos, "Association Rules Mining: A Recent Overview", GESTS International Trans on Computer Science and Engineering, Vol.32 (1), pp. 71-82, 2006.
- [12] R. Agrawal and R. Srikant, "Fast Algorithms for Mining association rules," Proc. Of 20th VLDB Conference, 1994.
- [13] R. Agrawal, T. Imielinski and A.Swami, "Mining Association Rules between Sets of Items in Large Databases," Proc. ACM-SIGMOD Int. Conf. On Management of Data, Washington, D.C. 1993.
- [14] R. Kohavi and G.H. John. Wrappers for Feature Subset Selection. Artificial Intelligence, 97(1-2):273-324, 1997.
- [15] Fayyad, G. Piattsky-Shapiro and P. Smyth, "From Data Mining to Knowledge Discovery: An Overview," Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, CA, pp.1-30.

SESSION

REAL-WORLD DATA MINING APPLICATIONS, CHALLENGES, AND PERSPECTIVES

Chair(s)

Dr. Mahmoud Abou-Nasr

Relevant Feature Selection and Generation in High Dimensional Haptic-based Biometric Data

¹Nizar Sakr, ²Fawaz A. Alsulaiman, ³Julio J. Valdés, ²Abdulmotaleb El Saddik, ^{1†}Nicolas D. Georganas

¹Distributed and Collaborative Virtual Environments Research Lab, University of Ottawa, Canada

²Multimedia Communications Research Lab, University of Ottawa, Canada

³National Research Council Canada, Institute for Information Technology

¹{nsakr, georganas}@discover.uottawa.ca, ²{fawaz, abed}@mclab.uottawa.ca, ³julio.valdes@nrc-cnrc.gc.ca

Abstract—In this paper, relevant feature selection and generation in high dimensional haptic-based biometric data is introduced. Feature selection and generation methods are investigated and derived to minimize the size of the high dimensional haptic feature vectors. Feature selection techniques used are based on concepts from rough set theory in order to select minimal information-preserving subsets of the original attributes. Feature generation methods are also presented that make use of non-linear transformations to map the high dimensional feature space into another space of smaller dimension while minimizing some error measure of information loss. In addition, Support Vector Machines (SVM), k-nearest neighbors (k-NN), Naïve Bayes and Random Forest classifiers are exploited to evaluate the accuracy of the selected and generated minimal feature vectors. The obtained results demonstrate that great haptic feature reduction can be achieved (over 98%) with minimal impact on classification accuracy.

I. INTRODUCTION

The emerging field of haptics, which enables the sensing and manipulation of virtual environments through touch, has been recognized as an important element in the area of human computer interaction. Its applications are wide, and span many areas, including medicine, rehabilitation, education and entertainment. Many of these current and future applications will involve the analysis and interpretation of the acquired haptic information in order to possibly reveal certain patterns in the data. Haptic data, which depict trajectory, cutaneous as well as kinesthetic information, essentially consist of position, velocity, orientation, torque and force information, that are directly acquired from a haptic interface upon a user's interaction with a predefined virtual environment. However, regardless of the intended haptic application, the number of resulting features are significantly large (in the thousands range) as the recorded information typically consists of multidimensional time-varying data. Consequently, this paper aims to derive and investigate different approaches to determine relevant attributes in high dimensional haptic datasets. The dataset considered in this study is generated using a haptic-enabled biometric application developed at DISCOVER laboratory of the University of Ottawa. Biometric systems provide a solution to ensure that

protected services are solely accessible by a legitimate user. This is achieved while relying on users' behavioral and/or physiological characteristics. In recent years, the possible use of haptic devices in biometric systems has been suggested to enable improved user identification/verification performance over more traditional techniques, such as those based on handwritten signatures.

Relevant feature selection and generation in high dimensional haptic data is nearly unexplored in the literature. In [1], [2], Orozco *et al.* make use of the same dataset exploited in this paper in order to demonstrate the feasibility of a haptic-based user authentication system. The authors however, distribute the high dimensional attributes of each signature across different instances, i.e. position, velocity, orientation, torque and force data acquired at time t_1 are assigned to instance $Inst_1$, data acquired at time t_2 are assigned to instance $Inst_2$, etc..., yielding a number of instances per user signature with only few attributes per instance. It is evident however that a more logical and adequate approach would be to assign all the generated haptic data attributes per signature to a single instance, i.e. each instance contains the entire (haptic-based) signature for a single user. This approach can lead to better analysis and interpretation of the haptic dataset, and improved discrimination between users. This however comes at the expense of having to deal with instances that possibly consist of thousands of attributes.

In this paper, feature selection and generation methods are investigated and derived to minimize the size of the high dimensional haptic feature vector. Feature selection methods used are based on concepts from rough set theory in order to select minimal information-preserving subsets of the original features. Conversely, the suggested feature generation techniques make use of non-linear transformations that map the high dimensional feature space into another space of smaller dimension while minimizing some error measure of information loss. Furthermore, Support Vector Machines (SVM), k-nearest neighbors (k-NN), Naïve Bayes and Random Forest classifiers are exploited in order to evaluate the accuracy of the selected and generated minimal feature vectors.

The rest of the paper is organized as follows. In Section II the haptic data acquisition procedure will be illustrated.

†N.D. Georganas holds a Cátedra de Excelencia at the Univ. Carlos III de Madrid and is visiting researcher at IMDEA Networks, on leave from the School of Information Technology and Engineering, University of Ottawa.

In Section III, dimensionality reduction based on feature selection is presented. In Section IV dimensionality reduction based on feature generation is described. In Section V the experimental results are provided. In Section VI a discussion of the experimental results is conducted. Finally, conclusive remarks are outlined in Section VII.

II. HAPTIC DATA ACQUISITION

In this section, the haptic-enabled virtual check application, as well as the acquired data will be described.

A. Haptic-enabled Virtual Environment

The experiments are performed using the Reachin Display [3], which integrates a haptic device with stereo graphics for an immersive and high quality 3D experience. The Reachin visuo-haptic interface enables users to see and touch virtual objects at the same location in space. This approach enables a superior integration of vision and touch than a conventional 2D screen-based display. The haptic stimulus is sensed using the SensAble PHANTOM Desktop force-feedback device, which is equipped with an encoder stylus that provides 6-degree-of-freedom single contact point interaction and positional sensing. In the case presented here, the visual stimuli consist of a virtual pen and a virtual check on which users can record their handwritten signature. The latter haptic-enabled virtual environment has been selected since handwritten signatures have been widely accepted as a mean to prove authenticity and authorship of a document. Conversely, the haptic stimuli are force and frictional feedback that attempt to mimic the tactile sensations felt when signing a traditional paper check. More specifically, the check is built on an elastic membrane surface with particular texture features, providing the users with a user-friendly and realistic feel of the virtual object. Moreover, similarly to conventional dynamic signature verification technologies, the virtual check application records a wide array of attributes that depict a user's physical and behavioral traits.

B. Haptic Datasets

The haptic-based handwritten signatures are diligently obtained from 13 different participants, where 10 signatures are collected per individual. A database is generated that is itself composed of a set of flat files (users' haptic-based signatures), that were collected on a workstation equipped with MS-OS 2000 and a XENON processor at the DISCOVER lab of the University of Ottawa. The data acquired depict various distinct haptic features as a function of time. A number of haptic data types are considered that characterize the instantaneous state of the haptic system, including, three-dimensional position, force (pressure exerted on the virtual check), torque, and angular orientation. Furthermore, the multi-feature and multidimensional haptic data are recorded at 100 Hz. As the data is time-varying, the resulting number of attributes per signature is in fact the number of haptic data types considered (position, force, torque, ...) times the number of samples recorded per data type during each

signature acquisition. This evidently leads to significantly large feature vectors that encompass thousands of haptic-based attributes.

III. DIMENSIONALITY REDUCTION BASED ON FEATURE SELECTION

In this section, a rough sets approach to feature reduction is described. Concepts from rough set theory are initially introduced, followed with a concise description of the Rosetta software and the various algorithms that were used for our analysis.

A. Rough Sets Concepts

The Rough Set Theory [4] bears on the assumption that in order to define a set, some knowledge about the elements of the dataset is needed. This is in contrast to the classical approach where a set is uniquely defined by its elements. In the Rough Set Theory, some elements may be indiscernible from the point of view of the available information and it turns out that vagueness and uncertainty are strongly related to indiscernibility. Within this theory, knowledge is understood to be the ability of characterizing all classes of the classification. More specifically, an information system is a pair $\mathbf{A} = (U, A)$ where U is a non-empty finite set called the universe and A is a non-empty finite set of attributes such that $a : U \rightarrow V_a$ for every $a \in A$. The set V_a is called the value set of a . For example, a decision table is any information system of the form $\mathbf{A} = (U, A \cup \{d\})$, where $d \in A$ is the decision attribute and the elements of A are the condition attributes.

a) *Implicants*: It has been described [5] that an m -variable function $f : \mathbf{B}^m \rightarrow \mathbf{B}$ is called a *Boolean function* if and only if it can be expressed by a Boolean formula. An *implicant* of a Boolean function f is a term p such that $p \leq f$, where \leq is a partial order called the inclusion relation. A *prime implicant* is an implicant of f that ceases to be so if any of its literals are removed. An implicant p of f is a prime implicant of f in case, for any term q , the implication of Eq-1 holds.

$$p \leq q \leq f \Rightarrow p = q \quad (1)$$

b) *General Boolean Reasoning Solution Scheme*: It has been described [5] that following the presentation of earlier work, the general scheme of applying Boolean reasoning to solve a problem P can be formulated as follows:

- 1) Encode problem P as a system of simultaneously-asserted Boolean equations as in Eq-2, where the g_i and h_i are Boolean functions on \mathbf{B} .

$$P = \begin{cases} g_1 = h_1 \\ \vdots \\ g_k = h_k \end{cases} \quad (2)$$

- 2) Reduce the system to a single Boolean equation (e.g. $f_p = 0$) as in Eq-3.

$$f_p = \sum_{i=1}^k (g'_i \cdot h_i + g_i \cdot h'_i) \quad (3)$$

- 3) Compute $BCF(f_p)$, the prime implicants of f_p .
- 4) Solutions to P are then obtained by interpreting the prime implicants of f_p .

c) *Discernibility Matrices*: An information system \mathcal{A} defines a matrix M_A called a discernibility matrix. Each entry $M_A(x, y) \subseteq A$ consists of the set of attributes that can be used to discern between objects $x, y \in U$ according to Eq-4.

$$M_A(x, y) = \{a \in A : \text{discerns}(a, x, y)\} \quad (4)$$

Where, $\text{discerns}(a, x, y)$ may be tailored to the application at hand.

d) *Indiscernibility Relations and Graphs*: A discernibility matrix M_A defines a binary relation $R_A \subseteq U^2$. The relation R_A is called an *indiscernibility relation* [5] (See Eq-5) with respect to A , and expresses which pairs of objects that we cannot discern between.

$$xR_Ay \Leftrightarrow M_A(x, y) = \emptyset \quad (5)$$

An alternative way to represent R_A is via an *indiscernibility graph* (IDG), which is a graph $G_A = (U, R_A)$ with vertex set U and edge set R_A . It has been stated [5] that G_A is normally only interesting to consider when R_A is a tolerance relation, in which case G_A may be used for the purpose of clustering or unsupervised learning.

e) *Discernibility Functions*: A *discernibility function* [5] is a function that expresses how an object or a set of objects can be discerned from a certain subset of the full universe of objects. It can be constructed relative to an object $x \in U$ from a discernibility matrix M_A according to Eq-6.

$$f_A(x) = \prod_{y \in U} \left\{ \sum a^* : a \in M_A(x, y) \text{ and } M_A(x, y) \neq \emptyset \right\} \quad (6)$$

The function $f_A(x)$ contains $|A|$ Boolean variables, where variable a^* corresponds to attribute a . Each conjunction of $f_A(x)$ stems from an object $y \in U$ from which x can be discerned and each term within that conjunction represents an attribute that discerns between those objects. The prime implicants of $f_A(x)$ reveal the minimal subsets of A that are needed to discern object x from the objects in U that are not members of $R_A(x)$.

In addition to defining discernibility relative to a particular object, discernibility can also be defined for the information system \mathcal{A} as a whole. The full discernibility function $g_A(U)$ (See Eq-7) expresses how all objects in U can be discerned from each other. The prime implicants of $g_A(U)$ reveal the minimal subsets of A we need to discern all distinct objects in U from each other.

$$g_A(U) = \prod_{x \in U} f_A(x) \quad (7)$$

f) *Reducts*: If an attribute subset $B \subseteq A$ preserves the indiscernibility relation R_A then the attributes $A \setminus B$ are said to be *dispensable*. An information system may have many such attribute subsets B . All such subsets that are minimal

(i.e. that do not contain any dispensable attributes) are called *reducts*. The set of all reducts of an information system \mathcal{A} is denoted $RED(\mathcal{A})$.

In particular, minimum reducts (those with a small number of attributes), are extremely important, as decision rules can be constructed from them [6]. However, the problem of reduct computation is NP-hard, and several heuristics have been proposed [7].

B. Rosetta

In this study, The ROSETTA software [5] is used to perform data analysis within the framework of rough set theory. The algorithms that were exploited are discussed in the following sections, and they can be classified into two distinct categories: discretization and reduct computation.

1) *Discretization*: Discretization is a preprocessing step that most symbolically based data mining methods require in order to provide adequate performance. It essentially corresponds to defining a coarser view of the world through making the attributes value sets smaller. The transformation is performed in such a manner that ranges or intervals are used instead of the exact data themselves. The observations can then be treated as qualitative rather than quantitative information.

- NaiveScaler

The naive algorithm is a basic heuristic that may result in a greater number of cuts (that determine the intervals) than are necessary. Its description makes the assumption that all condition attributes A are numerical. For every condition attribute a , its value set V_a can be sorted to obtain the following ordering:

$$v_a^1 < \dots < v_a^i < \dots < v_a^{|V_a|} \quad (8)$$

Then let C_a denote the set of all determined cuts for attribute a , defined as depicted in Eq-(9), Eq-(10), and Eq-(11).

$$X_a^i = \{x \in U \mid a(x) = v_a^i\} \quad (9)$$

$$\Delta_a^i = \{v \in V_d \mid \exists x \in X_a^i \text{ such that } d(x)=v\} \quad (10)$$

$$C_a = \left\{ \frac{v_a^i + v_a^{i+1}}{2} \mid |\Delta_a^i| > 1 \text{ or } |\Delta_a^{i+1}| \text{ or } \Delta_a^i \neq \Delta_a^{i+1} \right\} \quad (11)$$

The set C_a corresponds to all cuts midway between two observed values, with the exception of those that are not needed because the corresponding objects have the same decision value. In the case when no cuts are found for an attribute, this discretization technique leaves the attribute unprocessed. Furthermore, missing values are ignored during the search for cuts.

- BROrthogonalScaler

This is a Rosetta implementation of the algorithm introduced in [8]. It is based on the combination of the aforementioned NaiveScaler algorithm and a Boolean reasoning procedure for discarding all but a small subset of the generated cuts. The algorithm is initiated by first creating a Boolean function f (using Eq-12) from the set of candidate cuts computed using Eq-11.

$$f = \prod_{(x,y)} \sum_a \left\{ \sum c^* \mid c \in C_a \text{ and } a(x) < c < a(y) \text{ and } \partial_{A(x)} \neq \partial_{A(y)} \right\}. \quad (12)$$

The prime implicant of f is then computed using the greedy algorithm of Johnson [9], described in Section III-B.2. However, in certain cases, this discretization technique may result in no cuts being deemed necessary for some attributes. This occurs when such attributes are not needed to preserve discernibility. The Rosetta implementation of this algorithm leaves such attributes untouched.

- EqualFrequencyScaler

In order to discretize attributes that may have been left untouched when the BROrthogonalScaler algorithm is used, equal frequency binning can be performed, which is a simple unsupervised and univariate discretization technique. The procedure consists of fixing a number of intervals n and subsequently examining the histogram of each attribute. As a result, $n - 1$ cuts are determined so that approximately the same number of objects fall into each of the n intervals.

2) *Reduct Computation*: Two different algorithms to compute reducts or reduct approximations were used. Reduct approximations generate attribute subsets that in a sense “almost” preserve the indiscernibility relation. As reduct computation is NP-hard, reduct approximation can be found to be necessary especially when the number of attributes $|A|$ is very high.

- JohnsonReducer

Johnson’s algorithm as implemented in Rosetta, invokes a variation of a greedy algorithm to compute a single reduct [9]. The algorithm has a natural bias towards finding a single prime implicant of minimal length. A step by step illustration of the algorithm is depicted below, where $w(S)$ denotes a weight for set S in \mathcal{S} that is computed from the data.

- 1) Let $B = \emptyset$
- 2) Let a denote the attribute that maximizes $\sum w(S)$, where the sum is taken over all sets S in \mathcal{S} that contain a . The Rosetta implementation resolves ties arbitrarily.
- 3) Add a to B .
- 4) Remove all sets S from \mathcal{S} that contain a .
- 5) If $S = \emptyset$ return B . Otherwise, goto step 2.

TABLE I
DISTANCE FUNCTIONS

Metric Name	Metric Formulation
Absolute Value of the Cosine of the Angle Between Two Vectors	$d(x, w) = \left \frac{(\sum_i x_i y_i)}{ x y } \right $
Bray Curtis	$d(x, w) = \sum_i \frac{1}{\sum_j (x_j + w_j)} x_i - w_j $
Normalized Canberra	$d(x, w) = \frac{1}{N} \sum_i \frac{1}{(x_i + w_i)} x_i - w_i $
Normalized Euclidean	$d(x, w) = \sqrt{\frac{1}{N} \sum_i (x_i - w_i)^2}$
Chebyshev	$d(x, w) = \lim_{k \rightarrow \infty} \left(\sum_i x_i - w_i ^k \right)^{1/k}$
Gower Dissimilarity	$\delta_{ij} = \frac{1}{S_{ij}} - 1$ where $S_{ij} = \sum_{k=1}^p s_{ijk} / \sum_{k=1}^p w_{ijk}$
Normalized City Block	$d(x, w) = \frac{1}{N} \sum_i x_i - w_i $
Normalized Clark	$d(x, w) = \frac{1}{N} \sqrt{\sum_i \frac{(x_i - w_i)^2}{(x_i - w_i)^2}}$
Normalized Gower	$d(x, w) = \frac{1}{N} \sum_i \frac{ x_i - w_i }{Range_i}$

Computation of approximate reducts is performed by aborting the loop when enough sets have been removed from \mathcal{S} , instead of requiring that \mathcal{S} has to be entirely emptied.

- SAVGeneticReducer

This method is an implementation of a genetic algorithm for computing minimal hitting sets, as depicted in [10], [11]. The algorithm has support for both cost information and approximate solutions. The algorithm’s fitness function f is defined as follows:

$$f(B) = (1 - \alpha) \times \frac{cost(A) - cost(B)}{cost(A)} + \alpha \times \min \left\{ \varepsilon, \frac{||S \text{ in } \mathcal{S} \mid S \cap B \neq \emptyset||}{|\mathcal{S}|} \right\}, \quad (13)$$

where \mathcal{S} is the set of sets that correspond to the discernibility function. The parameter α defines a weighting between subset cost and hitting fraction, while ε controls approximate solutions as it represents a minimal value for the hitting fraction. Subsets B of A are found through the evolutionary search driven by the fitness function, where sets that have a hitting fraction of at least ε , are collected in a “keep list”.

IV. DIMENSIONALITY REDUCTION BASED ON FEATURE GENERATION

The construction of a smaller feature space can also be performed via a nonlinear transformation that maps the

original set of objects under study \mathcal{O} into another space of small dimension $\hat{\mathcal{O}}$, with an intuitive metric. A feature generation approach of this kind implies information loss as it usually involves a non-linear transformation ($\varphi : \mathcal{O} \rightarrow \hat{\mathcal{O}}$). There are essentially three kinds of spaces generally sought [12]: *i)* spaces preserving the structure of the objects as determined by the original set of attributes or other properties (unsupervised approach), *ii)* spaces preserving the distribution of an existing class or partition defined over the set of objects (supervised approach) and *iii)* hybrid spaces. In this work, unsupervised spaces are constructed because data structure is one of the most important elements to consider when the location and adjacency relationships between different objects in the new space could give an indication of the *similarity relationships* [13], [14] between the objects, as given by the set of original attributes [15]. The mapping φ is constructed while minimizing some error measure of information loss [16]. If δ_{ij} is a dissimilarity measure between any two objects $i, j \in \mathcal{O}$, and $\zeta_{\hat{i}\hat{j}}$ is another dissimilarity measure defined on objects $\hat{i}, \hat{j} \in \hat{\mathcal{O}}$ ($\hat{i} = \varphi(i), \hat{j} = \varphi(j)$), a frequently used error measure associated to the mapping φ is:

$$\text{Sammon error} = \frac{1}{\sum_{i < j} \delta_{ij}} \sum_{i < j} \frac{(\delta_{ij} - \zeta_{\hat{i}\hat{j}})^2}{\delta_{ij}} \quad (14)$$

Frequently used dissimilarity (and distance) measures δ_{ij} are presented in Table I. In order to optimize the Sammon error measure, the classical Fletcher-Reeves algorithm is exploited. The Fletcher-Reeves method is a well known technique used in deterministic optimization [17]. It assumes that the function f is roughly approximated as a quadratic form in the neighborhood of a N dimensional point \mathbf{P} . $f(\vec{x}) \approx c - \vec{b} \cdot \vec{x} + \frac{1}{2} \vec{x} \cdot \mathbf{A} \cdot \vec{x}$, where $c \equiv f(\mathbf{P})$, $\vec{b} \equiv -\nabla f|_{\mathbf{P}}$ and $[\mathbf{A}]_{ij} \equiv \frac{\partial^2 f}{\partial x_i \partial x_j}|_{\mathbf{P}}$.

The matrix \mathbf{A} whose components are the second partial derivatives of the function is called the Hessian matrix of the function at \mathbf{P} . Starting with an arbitrary initial vector \vec{g}_0 and letting $\vec{h}_0 = \vec{g}_0$, the conjugate gradient method constructs two sequences of vectors from the recurrence $\vec{g}_{i+1} = \vec{g}_i - \lambda_i \mathbf{A} \cdot \vec{h}_i$, $\vec{h}_{i+1} = \vec{g}_{i+1} - \gamma_i \mathbf{A} \cdot \vec{h}_i$, where $i = 0, 1, 2, \dots$

The vectors satisfy the orthogonality and conjugacy conditions $\vec{g}_i \cdot \vec{g}_j = 0$, $\vec{h}_i \cdot \mathbf{A} \cdot \vec{h}_j = 0$, $\vec{g}_i \cdot \vec{h}_j = 0$, $j < i$ and λ_i, γ_i are given by $\lambda_i = \frac{\vec{g}_i \cdot \vec{g}_i}{\vec{h}_i \cdot \mathbf{A} \cdot \vec{h}_i}$, $\gamma_i = \frac{\vec{g}_{i+1} \cdot \vec{g}_{i+1}}{\vec{g}_i \cdot \vec{g}_i}$.

It can be proven [17] that if \vec{h}_i is the direction from point \mathbf{P}_i to the minimum of f located at \mathbf{P}_{i+1} , then $\vec{g}_{i+1} = -\nabla f(\mathbf{P}_{i+1})$, therefore, not requiring the Hessian matrix.

The φ mappings obtained using an approach of this kind are implicit, as the images of the transformed objects are computed directly and the algorithm does not provide a function representation. Hence, the accuracy of the mapping depends on the final error obtained in the optimization process. Explicit mappings can however be obtained from these solutions using neural networks, genetic programming, and other techniques.

V. RESULTS

The example high dimensional haptic dataset selected is that of [1], [2], and that was briefly described in Section II. They essentially consist of haptic-based handwritten signatures recorded from 13 different participants, where 10 signatures were collected per individual. In order to ensure accurate discrimination between the signatures, the obtained feature vectors were normalized to a common length of 10000. Essentially, the acquired haptic data types are re-sampled (upsampled/downsampled) when necessary to ensure a common feature vector length across all instances. The latter feature vector length was selected in such a manner to minimize the information loss that is most apparent when downsampling is performed. Consequently, the computed preprocessed dataset contains 130 instances, where each consists of 10000 features. Feature selection is then performed using the discretization/reduct computation strategy discussed in Section III, where the preprocessed dataset is initially discretized using the *BROrthogonalScaler* and the *EqualFrequencyScaler* algorithms. As mentioned in Section III-B, *EqualFrequencyScaler* is only used to discretize attributes that may have been left untouched by the *BROrthogonalScaler* algorithm. Reduct computation is then performed using two different algorithms: *JohnsonReducer* and *SAVGeneticReducer*.

A selection of the reducts generated using the latter algorithms are shown in Table II, where it can be observed that when *JohnsonReducer* is used, a reduct with only 3 attributes was obtained. Moreover, in order to evaluate the accuracy of the computed reducts, Support Vector Machines (SVM), k-nearest neighbors (k-NN), Naïve Bayes and Random Forest classifiers were exploited. For comparison purposes, the original preprocessed dataset (10000 attributes) is initially equally divided into training and test sets. Classification of the test set using the aforementioned classifiers is then performed, where the corresponding results are shown in Table II. Five-fold cross-validation (CV) results obtained using the entire preprocessed dataset are also presented. Similarly, the rough sets-based (reduced) datasets were also equally divided into training and test sets. The obtained classification (training/testing) and cross-validation results are illustrated in Table II.

In Table III, the results obtained with the reduced datasets computed using the non-linear feature generation technique presented in IV, are shown. It can be observed that the non-linear mapping is performed from the original 10000 attributes to 3 and 10 attributes, while relying on the different distance measures previously depicted in Table I. Consequently, classification and cross-validation results with 18 different generated datasets are presented.

VI. DISCUSSION OF RESULTS

Many observations can be made from the results presented in Table II and Table III. First, if we consider

TABLE II
CLASSIFICATION RESULTS USING ORIGINAL ATTRIBUTES.

Datasets	Number of Attributes	SVM		k-NN		Naïve Bayes		Rand. Forest	
		5-fold CV	Train/ Test	5-fold CV	Train/ Test	5-fold CV	Train/ Test	5-fold CV	Train/ Test
All	10000	92.3%	95.4%	80.0%	83.0%	85.3%	70.7%	88.4%	98.4%
<i>JohnsonReducer</i> Reduct	3	75.4%	75.4%	75.4%	70.8%	70.0%	75.4%	73.8%	78.5%
<i>SAVGeneticReducer</i> Reduct 1	96	88.5%	87.7%	70.8%	70.8%	85.4%	80.0%	88.5%	93.8%
<i>SAVGeneticReducer</i> Reduct 2	185	91.5%	93.8%	74.6%	76.9%	82.3%	81.5%	86.9%	95.4%

TABLE III
CLASSIFICATION RESULTS USING GENERATED ATTRIBUTES.

Datasets	Number of Attributes	SVM		k-NN		Naïve Bayes		Rand. Forest	
		5-fold CV	Train/ Test	5-fold CV	Train/ Test	5-fold CV	Train/ Test	5-fold CV	Train/ Test
AVCA	3	68.5%	70.8%	74.6%	75.3%	59.2%	53.8%	70.0%	67.6%
	10	75.4%	75.4%	73.8%	76.9%	73.8%	73.0%	81.0%	80.0%
Bray Curtis	3	77.7%	78.5%	77.6%	83.0%	75.3%	73.8%	70.7%	72.3%
	10	91.5%	92.3%	87.6%	90.7%	83.0%	80.0%	87.6%	83.0%
Normalized Canberra	3	73.8%	75.4%	73.0%	78.4%	69.2%	70.7%	69.2%	67.6%
	10	89.2%	90.8%	80.0%	86.1%	84.6%	78.4%	81.5%	78.4%
Chebyshev	3	40.0%	38.5%	33.8%	30.7%	32.3%	41.5%	30.0%	29.2%
	10	34.6%	38.5%	29.2%	30.7%	35.3%	40.0%	25.3%	35.3%
Gower Similarity	3	82.3%	81.5%	77.6%	73.8%	81.5%	72.3%	72.3%	69.2%
	10	87.7%	83.1%	73.0%	75.3%	86.1%	81.5%	80.7%	76.6%
Normalized City Block	3	79.2%	70.8%	81.5%	78.4%	80.7%	76.9%	66.9%	60.0%
	10	88.5%	92.3%	76.1%	83.0%	84.6%	81.5%	78.4%	80.0%
Normalized Clark	3	36.9%	44.6%	37.6%	32.3%	36.9%	33.8%	36.9%	32.3%
	10	46.9%	43.1%	43.0%	41.5%	38.4%	30.7%	38.4%	35.3%
Normalized Euclidean	3	79.2%	81.5%	76.9%	75.3%	70.0%	70.7%	60.0%	55.3%
	10	87.7%	84.6%	76.1%	72.3%	83.0%	64.6%	74.6%	75.3%
Normalized Gower	3	85.4%	80%	77.6%	78.4%	80.7%	72.3%	68.4%	66.1%
	10	89.2%	93.8%	73.0%	80.0%	84.6%	75.3%	78.4%	78.4%

the classification and cross-validation results presented in Table II, it can be seen that results obtained when using only 3 of the original 10000 attributes, selected using Johnson's reduct computation algorithm, are in fact quite reasonable. For example, a 5-fold cross-validation on the original dataset where each instance consists of 10000 attributes, using SVM, yielded a classification result of 92.3%. Conversely, a 5-fold cross-validation on *JohnsonReducer*'s 3-attributes-per-instance dataset (i.e. a 99.97% reduction of the original number of attributes), using SVM, still resulted in a moderate 75.4% classification accuracy. Moreover, when the 185 attributes selected using the *SAVGeneticReducer* algorithm are exploited, the classification and cross-validation results are both very comparable to the results obtained when all the original 10000 attributes are used. Also, as it can be presumed, reducts with a greater number of attributes resulted in better classification accuracy. However, a quick comparison of the results obtained using the *SAVGeneticReducer* algorithm-

generated reducts, reveals that only a slight classification improvement is achieved when the 185-attributes reduct is exploited as opposed to the more reduced 96-attributes reduct. Nevertheless, this small improvement remains crucial, particularly in security-related applications, such as in biometric systems as investigated in this work.

The results presented in Table III, which served to demonstrate the possible reduction in the number of attributes when performing a non-linear mapping of the original feature space to another space of a smaller dimension, are for the most part quite impressive. It can be observed that the best results were however achieved when the non-linear mapping is performed while relying on the Bray Curtis and Normalized Gower distance measures. Conversely, the worst classification results were obtained when the Chebyshev and the Normalized Clark distance measures were exploited. Also, if a comparison is performed between the non-linear mappings of the original attributes to 3 and 10 attributes, it can be seen that

the 10-attributes datasets, for almost all exploited classifiers, and regardless of the distance measure used (excluding the Chebyshev and the Normalized Clark distance measures), generally achieved better classification results. This being said, it would be desirable to further investigate this problem to determine the minimum number of generated-attributes necessary that would result in the best possible discrimination between different users. In other words, it would be of interest to determine the minimum number of generated-attributes required such that the achieved classification/cross-validation accuracy is at least as high as that obtained when the original dataset (10000 attributes) is considered.

VII. CONCLUSIONS

Feature selection and generation using concepts from rough set theory and non-linear transformations respectively, proved to be a promising way to reduce the very high dimensionality of haptic feature spaces. Classification and cross-validation experiments using datasets of haptic-based handwritten signatures demonstrated the accuracy of the suggested methods. It is also worth mentioning that the capability of the proposed feature generation technique to map a high dimensional haptic feature space to a smaller three-dimensional space with very little information loss is very encouraging as it opens up new avenues for studying various haptic characteristics using 3D visualizations, e.g. through 3D virtual reality spaces. As for the future work, it would be of interest to combine the suggested feature reduction and generation techniques to further reduce the dimensionality of the feature space while minimizing the possibly inevitable information loss.

REFERENCES

- [1] A. El Saddik, M. Orozco, Y. Asfaw, S. Shirmohammadi and A. Adler, "A Novel Biometric System for Identification and Verification of Haptic Users," *IEEE Transactions on Instrumentation and Measurement*, vol. 56, no. 3, pp. 895–906, 2007.
- [2] M. Orozco, M. Graydon, S. Shirmohammadi, and A. El Saddik, "Experiments in Haptic-Based Authentication of Humans," *Journal of Multimedia Tools and Applications*, vol. 37, no. 1, pp. 73–92, 2007.
- [3] *Reachin Technologies ab. Reachin Display*. <http://www.reachin.se/products/>.
- [4] Z. Pawlak, *Rough sets: Theoretical aspects of reasoning about data*. Kluwer Academic Publishers, Netherlands, 1991.
- [5] A. Ohm, *Discernibility and rough sets in medicine: tools and applications*. Ph.D. Thesis. Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway, 1999.
- [6] J. Bazan, A. Skowron, and P. Synak, "Dynamic reducts as a tool for extracting laws from decision tables," In *Proceedings of the 8th International Symposium on Methodologies for Intelligent Systems*, Charlotte, NC, USA, Oct. 16-19, 1994. *Lecture Notes in Artificial Intelligence* 869, Springer-Verlag, pp. 346–355.
- [7] J. Wrblewski, "Ensembles of classifiers based on approximate reducts," *Fundamenta Informaticae* 47 IOS Press, pp. 351–360, 2001.
- [8] H. S. Nguyen and A. Skowron, "Quantization of real-valued attributes," In *Proceedings of the Second International Joint Conference on Information Sciences*, pp. 34-37, Wrightsville Beach, NC, Sept. 1995.
- [9] D. S. Johnson, "Approximation algorithms for combinatorial problems," In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, pp. 38-49, 1974.
- [10] S. Vinterbo and A. hrn, "Minimal approximate hitting sets and rule templates," In *Predictive Models in Medicine: Some Methods for Construction and Adaptation*, Department of Computer and Information Science, Dec. 1999. NTNU report 1999:130. [<http://www.idi.ntnu.no/staalv/dev/thesis.ps.gz>].
- [11] S. Vinterbo and A. hrn, "Minimal approximate hitting sets and rule templates," *International Journal of Approximate Reasoning*, vol. 25, no. 2, pp. 123-143, 2000.
- [12] J. Valdés and A. Barton, "Virtual reality spaces for visual data mining with multiobjective evolutionary optimization: Implicit and explicit function representations mixing unsupervised and supervised properties," In *Proceedings of the IEEE Congress of Evolutionary Computation*, Vancouver, Canada, pp. 5592-5598, 2006.
- [13] J. L. Chandon and S. Pinson, *Analyse typologique. Théorie et applications*. Masson, Paris, 1981.
- [14] I. Borg and J. Lingoes, *Multidimensional similarity structure analysis*. Springer-Verlag, 1987.
- [15] J. Valdés, "Virtual reality representation of information systems and decision rules: an exploratory technique for understanding data knowledge structure," in *Lecture Notes in Artificial Intelligence*, ser. LNAI, vol. 2639. Springer-Verlag, pp. 615-618, 2003.
- [16] J. W. Sammon, "A non-linear mapping for data structure analysis," *IEEE Transactions on Computers*, vol. C18, pp. 401-408, 1969.
- [17] W. Pres, B. Flannery, S. Teukolsky, and W. Vetterling, *Numeric Recipes in C*. Cambridge University Press, 1992.

A Prime Number-Based Method for Interactive Frequent Pattern Mining

Mohammad Nadimi-Shahraki¹, Norwati Mustapha², Md Nasir Sulaiman², Ali Mamat²

¹ Computer Engineering Dept., Islamic Azad University, Najafabad branch, Iran
Ph.D. Candidate of Computer Science, University of Putra Malaysia

² Faculty of Computer Science and Information Technology,
University of Putra Malaysia (UPM), Selangor, Malaysia.

Abstract - Data mining is a key step in knowledge discovery process. Frequent patterns play an important role in data mining tasks such as clustering, classification, and association analysis. In some real applications such as process control and web usage mining, finding new correlations between patterns by changing minimum support threshold called interactive mining is very useful. So far a few works have been introduced for interactive mining. Almost all current algorithms can not keep all information in memory and they need to update and reconstruct frequent pattern model for interactive mining. This research proposes a new method for interactive mining of frequent patterns based on prime number characteristics. The proposed method introduces a data transformation technique, a novel tree structure called *PC_Tree* and an efficient miner algorithm. The transformation technique transforms each transaction into only a positive integer to reduce the size of database. The novel tree structure has been designed to capture all positive integers to avoid of database rescanning and updating of frequent pattern model those can enhance the performance of interactive mining. The experiment results verify the compactness and efficiency of the proposed method.

Keywords: Data transformation, Data mining, Frequent pattern mining, Interactive mining, Prime number.

1 Introduction

Knowledge discovery or Extracting knowledge from large amount of data is a desirable task in competitive businesses. Data mining is an essential step in knowledge discovery process. Since the introduction of the Apriori algorithms [1], frequent patterns mining plays an important role in data mining tasks such as clustering, classification, prediction and association analysis. Many efficient algorithms have been introduced to solve the problem of frequent pattern mining more efficiently. They are almost based on three fundamental frequent patterns mining methodologies: Apriori, FP-growth and Eclat [5].

In some real applications such as process control and web usage mining, finding new correlations between items by changing minimum support threshold called interactive mining is very useful. When users change the minimum support threshold, finding frequent patterns with respect to new minimum support threshold in an acceptable response time is expected. Avoiding database rescanning and updating of frequent pattern model is the basic idea to implement an efficient algorithm. So far, a few efficient interactive mining algorithms have been introduced with acceptable response time. They have mostly improved FP-tree and usually suffer from FP-tree problems inherently [9]. Unlike the previous methods, this research proposes a new method for interactive mining of frequent patterns by using prime number characteristics. This method is an improvement of previous method that we presented for maximal frequent mining [10]. However the tree structure, search space pruning technique, and traversing technique have mostly been changed in mining algorithm.

The data transformation technique transforms each transaction into only a positive integer that presents all properties of the transaction. The experiments shows that by applying this transformation technique, the size of real transaction database can be reduced more than half. The novel tree structure called Priime-based and Compressed Tree or *PC_Tree* is to capture all information about transactions by keeping the positive integers. The tree construction is based on simple definitions presented in section 3. This method also introduces an efficient miner algorithm called *PC_Miner*. The *PC_Miner* makes use prime numbers characteristics and some nice properties of *PC_Tree* written in section 3 to prune the search space and enhance efficiency of tree traversing. There is no need to database rescanning and updating of frequent pattern model by using the information kept in the tree. The experiment results verify the compactness and efficiency of the method.

The rest of the paper is organized as follows. Section II introduces the problem and reviews some efficient related works. The proposed method is described in section III. The experimental results show in section IV and section V contains some conclusions and future works.

2 Problem and Related Work

2.1 Problem Description

Frequent patterns are itemsets or substructures that exist in a dataset with frequency no less than a user specified threshold. Let $L = \{i_1, i_2 \dots i_n\}$ be a set of items. Let D be a set of database transactions where each transaction T is a set of items and $|D|$ be the number of transactions in D . Given $P = \{i_j \dots i_k\}$ be a subset of L ($j \leq k$ and $1 \leq j, k \leq n$) is called a pattern. The support of a pattern P or $S(P)$ in D is the number of transactions in D that contains P . Pattern P will be called frequent if its support is no less than a user specified support threshold $\min_sup \sigma$ ($0 \leq \sigma \leq |D|$). The problem of frequent pattern mining is finding all frequent patterns (FP) in D with respect to σ showed by FP (σ).

Let σ' as new minimum support threshold then the problem of interactive frequent pattern mining is to find all frequent patterns in D with respect to new minimum support σ' or FP (σ') in an acceptable response time (without database rescanning and with minimum updating of current frequent patterns model).

2.2 Related Work

When users change minimum support threshold, one possible solution is to the update problem and rerun the frequent patterns mining algorithm again with respect to new minimum support threshold. This approach, though simple, has an unacceptable response time. Because all computation done for finding out current frequent patterns has been missed and all frequent patterns have to be found again from scratch. Avoiding database rescanning, tree reconstructing and updating of frequent pattern model is the basic idea to enhance the performance of interactive mining algorithms. A few interactive mining algorithms have been introduced so far based on this idea. Current efficient interactive mining algorithms have mostly improved FP-tree to capture all information about transactions. They usually suffer from FP-tree problems inherently and need to update frequent pattern model and in worst case database rescanning [9].

Cheung and Zaiane [3] proposed the FELINE algorithm and introduced Compressed and Arranged Transaction Sequences Tree or CATS Tree. CATS Tree was a novel data structure and an extension of FP-Tree to improve storage compression. It mines frequent patterns without the generation of candidate itemsets. CATS Tree is constructed by one database scan. Their experiment results showed that CATS Tree and the FELINE algorithm were well-suited for interactive mining. But, it still needs lots of swapping, merging, and splitting of tree nodes, because items in the trees are arranged base on a global frequency-dependent ordering.

Koh and Shieh [6] proposed AFPIM algorithm for

interactive mining by adjusting the FP-tree structure. Similarly, it keeps only frequent items however an item is frequent if its frequency is no less than preMinSup that is defined lower than the min_sup . Obviously determining a proper preMinSup is a problem in itself. Also modifications of transactions may affect the frequency of items because items are arranged by a descending global frequency ordering. In worst case the tree requires to reconstruction.

Leung et al. [7] proposed a canonical-order tree structure or CanTree which is unaffected by frequency changes. CanTree captures all information of transaction database to use for both incremental and interactive mining. Their experiments showed that CanTree solves the weaknesses of the FELINE and AFPIM algorithms. However CanTree compaction is less than FP-tree. It appears that CanTree may take a large amount of memory.

Zhu and Lin [12] proposed an incremental updating frequent pattern tree (IUFPTree) that was an extension of FP-Tree and IUFPTree algorithm for interactive mining. They improved FP-Tree by using Trailer Table and adding a new field called status in Header table. The Trailer Table was used as an optimization technique for reducing the size of database during the update process. The status field can help to reuse discovered frequent patterns. They presented that IUFPTree can mine FP faster than FP-Growth and also it is suitable for interactive mining. But it suffers from FP-tree weaknesses inherently. Also when minimum support threshold changed to smaller value, IUFPTree needs to reconstruction.

3 The Proposed Method

In this research, we proposed a new method for interactive frequent pattern mining. The proposed method introduces an efficient data transformation technique, a novel tree structure called PC_Tree and PC_Miner algorithm.

3.1 Data Transformation Technique

The transformation and presentation of database is an essential consideration in almost all algorithms. The most commonly database layout is the horizontal and vertical layout [11]. In both layouts, the size of database is very large. The data transformation is a useful technique which can reduce the size of database. Obviously, reducing of the size of database can enhance the performance of mining algorithms. Our method uses a prime-based data transformation technique to reduce the size of transaction database. It transforms each transaction into a positive integer called Transaction Value (TV) during of the PC_Tree construction as follows: Given transaction $T = (tid, X)$ where tid is the transaction-id and $X = \{i_j \dots i_k\}$ is the transaction-items or pattern X . While the PC_Tree algorithm

reads transaction T, the transformation procedure considers a prime number p_r for each item i_r in pattern X, and then TV_{tid} is computed by Eq. (1). Therefore, the transaction database can be represented in a compacted layout with smaller size.

$$TV_{tid} = \prod_j^k p_r \quad (1)$$

The data transformation technique utilizes Eq. (1) based on simple following definitions:

“A positive integer N can be expressed by unique product $N = p_1^{m_1} p_2^{m_2} \dots p_r^{m_r}$ where p_i is prime number, $p_1 < p_2 < \dots < p_r$, and m_i is a positive integer, called the multiplicity of p_i ” [4].

For example, $N = 1800 = 2^3 * 3^2 * 5^2$. Fundamentally, there is no duplicated item in transaction T. Hence we restrict the multiplicity only to $m_i = 1$ without losing any significant information. Therefore N can be produced by $p_1 p_2 \dots p_r$.

To facilitate the transformation process used in our method, let's examine it through an example. Let item set $L = \{A, B, C, D, E, F\}$ and the transaction database, DB, be the first two columns of Table 1 with eight transactions. The fourth column of Table 1 shows TV_{tid} transformed for all transactions. The average items length is smaller than real items length in benchmark datasets.

TABLE 1
THE TRANSACTION DATABASE DB AND ITS TRANSACTION VALUES

TID	Items	Encoded	TV
1	A, B, C, D, E	2, 3, 5, 7, 11	2310
2	A, B, C, D, F	2, 3, 5, 7, 13	2730
3	A, B, E	2, 3, 11	66
4	A, C, D, E	2, 5, 7, 11	770
5	C, D, F	5, 7, 13	455
6	A, C, D, F	2, 5, 7, 13	910
7	A, C, D	2, 5, 7	70
8	C, D, F	5, 7, 13	455

For example third transaction $T = (3, \{A, B, E\})$ in Table 1 can presents good identification numbers bought by a customer; $T_0 = (3, \{55123450, 55123452, 55123458\})$ from a market. Although the length of items in T_0 is bigger than T but both T and T_0 can be transformed into the same TV 66 by using our data transformation technique. Hence it is an item-length independent transformation technique. The experiments showed that by applying this data transformation technique, the size of real transaction database can be reduced more than half.

3.2 PC_Tree Construction

Using tree structure in mining algorithms makes two possibilities to enhance the performance of mining. Firstly, data compressing by a well-organized tree structures like FP-tree. Secondly, reducing search space by using pruning techniques. Thus the tree structures have been considered as a basic structure in previous data mining research [5, 8, 9]. This research introduces a novel tree structure called PC_Tree (Prime-based encoded and Compressed Tree). Unlike the previous methods, the PC_Tree is based on prime number characteristics. It makes use of both possibilities data compressing and pruning techniques to enhance efficiency.

A PC_Tree includes of a root and some nodes that formed sub trees as children of the root or descendants. The node structure consisted mainly of several different fields: value, local-count, global-count, status and link. The value field stores TV to records which transaction represented by this node. The local-count field set by 1 during inserting current TV and it is increased by 1 if its TV and current TV are equal. The global-count field registers support of pattern P which presented by its TV.

In fact during of insertion procedure the support of all frequent and infrequent patterns is registered in the global-count field. It can be used for interactive mining when min_sup is changed by user frequently. The status field is to keep tracking of traversing. When a node visited in the traversing procedure the status field is changed from 0 to 1. The link field is to form sub trees or descendants of the root.

Fig. 1 shows step by step construction of PC_Tree for transactions shown in Table 1 which summarized in Figure 1(a).

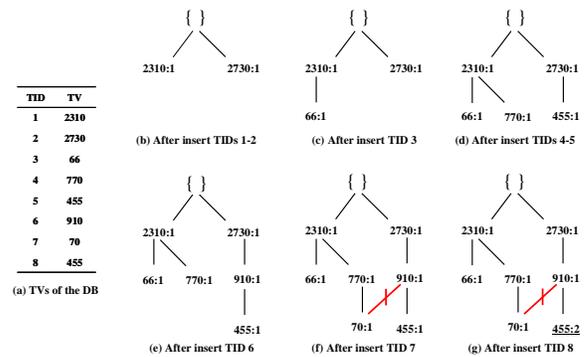


Fig. 1 Step by step PC_Tree construction

The construction operation mainly consists of insertion procedure that inserts TV(s) into PC_Tree based on definitions below:

Definition 1: If TV of node n_r and n_s is equal then $r = s$. Insertion procedure increases local - count field of node n_r

by 1 if the current TV is equal with TV of n_r .

Definition 2: $R = (n_i, n_{i-1}, \dots, n_j, root)$ is a descendant iff

TV of node $n_r \in R$ ($i \leq r \leq j$) can divide all TVs kept in nodes descendant $R_r = (n_{r+1}, n_{r+2}, \dots, n_j, root)$.

Definition 3: TV of the root is assumed null and can be divided by all TVs.

Definition 4: Node n_r can be belonged to more than one descendant.

Fig. 1(b) shows insertion of the first and second transactions. The second TV with value 2730 can not be divided by the first TV with value 2310 and it creates a new descendant using definition 2 and 3. Transactions 3-6 are inserted into their descendant based on definition 2 shown in Figure 1(c) - (e). Insertion of the seventh transaction applies definition 4 when TV 70 is belonged to two descendants (second descendant shown in the dashed line) shown in Figure 1(f). The TV of eighth transaction with value 455 is equal with the fourth TV, then the local-count field of fourth TV is increased by 1 using definition 1 shown in Figure 1(g) (shown in underline).

Each TV in PC_Tree represents a pattern P and the support of pattern P or S (P) is registered in the global-count field. Given pattern P and Q have been presented by TVP and TVQ respectively, the PC_Tree has some nice below properties:

Property 1: The S (P) is computed by traversing only descendants of TVP.

Property 2: P and Q belong to descendant R and S (P) < S (Q) iff TVP can be divided by TVQ.

Property 3: All nodes are arranged according to TV order, which is a fixed global ordering. In fact the PC_Tree is an independent-frequency tree structure.

Property 4: Important procedures are almost done only by two simple mathematic operations product and division. Obviously using mathematic operation enhances the performance instead of string operation.

3.3 PC_Miner Algorithm

The PC_Miner algorithm traverses the completed PC_Tree to discover Frequent Patterns in top-down direction. Fig.1 (g) shows a completed PC_Tree for transactions presented in Table 1. However, it didn't show some information like global-count stored in the tree. There is no need to database scan again and also the miner algorithm makes use of a combined pruning strategy including both superset infrequency and subset frequency pruning. As a result the search space is reduced, which dramatically reduces the computation and memory requirement and enhances the efficiency. The superset infrequency pruning is assisted by the frequency of items computed during the PC-Tree construction. Table 2 shows

the item frequency and supposed prime number for transaction database shown in table 1.

When the database remains unchanged, the PC_Miner can use the PC_Tree built in memory to mine frequent patterns with respect to new minimum support threshold changed by users. Hence, our method constructs the PC_Tree once and the PC_Miner uses it many for different minimum support threshold without tree reconstruction or database rescanning.

TABLE 2
FREQUENCY OF ITEMS AND SUPPOSED PRIME NUMBERS

Item	Prime number	Item Frequency
A	2	6
B	3	3
C	5	7
D	7	7
E	11	3
F	13	4

On the other hand, as explained in previous section during of insertion each TV in the PC_Tree, the following procedures are done.

- Item-frequency counting.
- Local-counting.
- Global-counting.
- Descendant constructing.

The global-counting procedure set the global-count field for each TV kept in PC_Tree. Therefore during of tree traversing, to find the frequent patterns, PC_Miner only compare the amount of each global-count with min_sup. It means there is no need to count the frequency of itemsets again or updating of frequent pattern model.

4 Experimental Results

In this section, we evaluate the performance of our method. All experiments were performed in a time-sharing environment in a 2.4 GHz PC. All the algorithms are implemented using Microsoft Visual C++ 6.0. We performed comprehensive experimental analysis on the performance of PC_Tree on several synthetic and real datasets. However, as explained in section 2 the problem is interactive mining for real application especially with dense database and high correlation between consecutive patterns. Therefore according to the space limitation and the problem specifications, we only show the result of compactness evaluation on synthetic sparse datasets and the effect of minimum support threshold changing on runtime in dense datasets.

In first experiment we use synthetic sparse datasets T10I4D100k generated by the program developed at IBM Almaden Research Center [1]. The number of transactions, the average transaction length and the average frequent

pattern length of T10I4D100k are set to 100k, 10 and 4 respectively. We consider $p\%$ of this dataset where p will be increased from 10 to 100 to evaluate how the data transform technique can compact the size of dataset. Fig.2 shows comparison of the size of original dataset with the size of transformed dataset using our data transform technique.

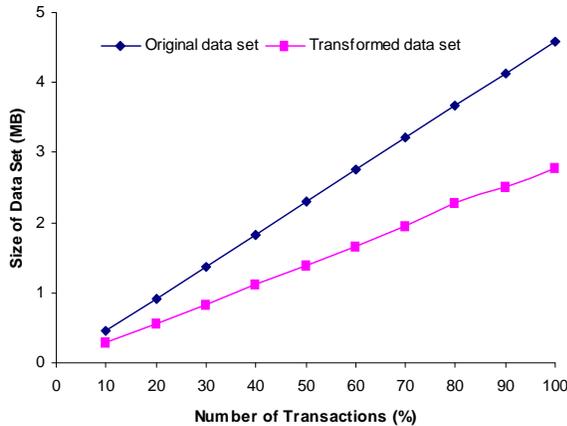


Fig.2. Compactness of data transformation technique

In second experiment, the effect of minimum support threshold changing on runtime is evaluated. This experiment is conducted by using real dense datasets mushroom from UC Irvine Machine Learning Depository [2]. The mushroom dataset records consists of the characteristics of various mushroom species, and the number of records, the number of items and the average record length are set to 8124, 119 and 23 respectively.

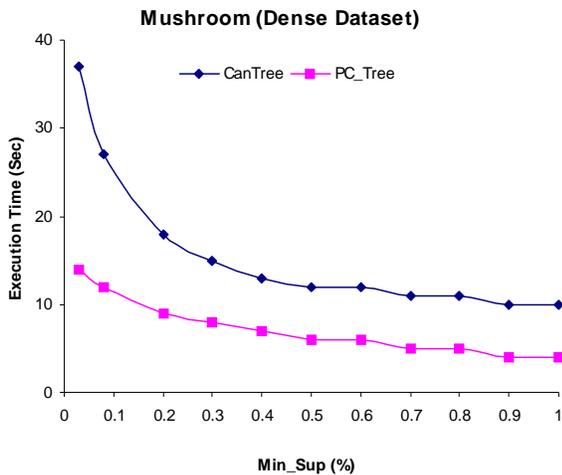


Fig. 3 Runtime comparison: PC_Tree vs. CanTree

In [7] CanTree showed that it outperforms other efficient interactive mining algorithms say AFPIM, CATS tree. Hence, we only state the performance comparison of PC_Tree with CanTree. The results are based on the average of multiple runs. The runtime includes tree construction and mining time. As shown in Fig. 3, when the minimum support threshold decreases from 5 to 1, the PC_Tree were superior to the CanTree in the execution time.

5 Conclusions and Future Works

We proposed a new method for interactive mining of frequent patterns especially in real applications with dense database. Unlike previous interactive algorithms, this method is based on simple prime number characteristics. Moreover important procedures are almost done only by two simple mathematic operations product and division.

Most contribution is introducing a data transformation technique to reduce the size of transaction database, a novel tree structure PC_Tree (Prime-based and Compressed Tree) and PC_Miner algorithm. The tree captures the content of the transactions by using the TV(s) transformed during in tree construction. Therefore the PC_Miner can avoid of database rescanning and tree reconstructing. Also the PC_Tree has some useful properties that the PC_Miner uses them to avoid of updating of frequent pattern model. The experimental results verified the performance of the proposed method.

Many directions can be considered as future works. The proposed method can be improved for incremental mining of frequent patterns where database transactions are inserted, deleted, and/or modified. Moreover the data transformation technique during of tree construction has a strong potential to hide the data and association rules. Hence, the proposed method can be extended for privacy preserving data mining.

References

- [1] Agrawal R. and R. Srikant, FAST Algorithms for Mining Association Rules, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, vol. 1215, pp. 487499, 1994.
- [2] Blake C. and C. Merz, Uci Repository of Machine Learning Databases, University of California – Irvine, Irvine, Ca, 1998.
- [3] Cheung W. and O. R. Zaiane, Incremental Mining of Frequent Patterns without Candidate Generation or Support Constraint, presented at IDEAS 2003. IEEE Computer Society Press, 2003, pp. 111-116.
- [4] Cormen T. T., C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*: MIT Press Cambridge, MA, USA, 1990.
- [5] Han J., H. Cheng, D. Xin, and X. Yan, Frequent Pattern Mining: Current Status and Future Directions, *Data Mining*

and *Knowledge Discovery*, vol. 15, pp. 55-86, 2007.

- [6] Koh J. L. and S. F. Shieh, An Efficient Approach for Maintaining Association Rules Based on Adjusting Fp-Tree Structures, *LECTURE NOTES IN COMPUTER SCIENCE*, pp. 417-424, 2004.
- [7] Leung C. K. S., Q. I. Khan, and T. Hoque, Cantree: A Tree Structure for Efficient Incremental Mining of Frequent Patterns, *Proc. ICDM 2005*, pp. 274-281, 2005.
- [8] Mustapha N., M. N. Sulaiman, M. Othman, and M. H. Selamat, Fast Discovery of Long Patterns for Association Rules, *International Journal of Computer Mathematics*, vol. 80, pp. 967-976, 2003.
- [9] Nadimi-Shahraki M., N. Mustapha, M. N. Sulaiman, and A. Mamat, Incremental Updating of Frequent Pattern: Basic Algorithms, *Proceedings of the second International Conference on Information Systems Technology and Management (ICISTM 08)*, pp. 145-148, 2008.
- [10] Nadimi-Shahraki M., N. Mustapha, M. N. B. Sulaiman, and A. B. Mamat, Maximal frequent Itemset Mining Using Database Encoding, *Proceeding of the ICDMKE'08*, pp. 299-302.
- [11] Zaki M. J., Scalable Algorithms for Association Mining, *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, pp. 372-390, 2000.
- [12] Zhu Q. and X. Lin, Mining Frequent Patterns with Incremental Updating Frequent Pattern Tree, *proceeding of the WCICA 2006*.

MRC: Multi Relational Clustering approach

Majid Rastegar-Mojarad¹, Behrouz Minaei-Bidgoli²

¹Faculty of Engineering, Persian Gulf University, Bushehr 75169, Iran

²Assistant Professor of Computer Engineering, Iran University of Science and Technology
Narmak, Tehran, Iran

²CRCIS (NOOR): Data Mining Research group, Computer Research Center of Islamic Sciences, P.O.
Box 37185-3857, Qom, Iran

Abstract— *Clustering is a process of partitioning data objects into groups based on the similarity measures. Most of the existing methods perform clustering within a single table, but most of the real-world databases, however, store information in multiple tables. We propose a new method which is called Multi Relational Clustering (MRC) for clustering a relational database. The MRC approach uses existing clustering algorithms for clustering every table of database. Tables in a database are related to each other based on foreign keys. The MRC approach divides the tables into two categories: dependent and independent tables. A dependent table is a table that includes entities attributes, as well as fields related to the other entities which belong to the other tables. In fact a dependent table includes one or more foreign keys. The MRC approach firstly, clusters independent tables then utilizes these results for clustering dependent tables. The MRC clusters each table by existing clustering algorithm with respect to its fields. An important feature of the MRC approach is ability of clustering several tables in parallel. The proposed approach is very simple and is developed under SQL very efficiently. We offer a version of implementation of k-Means in SQL and use it for clustering a database by MRC approach. Our experiments show that the MRC is efficient for clustering a huge database in a relational environment.*

Keyword: Multi Relational Data Mining, clustering, SQL, Relational Database

1. INTRODUCTION

Data mining algorithms look for patterns in data. While most existing data mining approaches look for patterns in a single data table, multi-relational data mining (MRDM) approaches look for patterns that involve multiple tables (relations) from a relational database. Recently, the most common types of patterns and approaches in data mining have been extended to the multi-relational cases [1].

Data clustering refers to the problem of partitioning a data set into homogeneous groups (called clusters) in such a way that patterns within a cluster are more similar to each

other than patterns belong to different clusters. Most clustering algorithms proposed in the literature [2]-[7] exploit a similarity (or dissimilarity) measure.

Clustering in multi-relational environments has been studied in [8]-[12]. In [8]-[10] the similarity between two objects is defined based on joinable tuples. These approaches, although provide similarity definitions in multi-relational environments, it is usually very expensive to compute such similarities because an object is often joinable with hundreds or thousands of tuples. Also the cross-relational clustering which is called CROSSCLUS [11], [12] needs very complex computation to determine distance between two records.

Authors of [13], [14] introduce a SQL implementation of the popular K-means clustering algorithm in order to integrate it with a relational DBMS. They compare K-means implementations in SQL and C++ with respect to speed and scalability. They also study the time of export data sets outside of the DBMS. Experiments show that SQL overhead is significant for small data sets, but relatively low for large data sets, whereas export times become a bottleneck for C++.

We propose a new approach clustering in multi relational environment. Our approach is a Multi Relational Clustering which is called MRC, efficiently works for large databases with many tables. The MRC divides tables in two categories. One category is independent tables and another category is dependent. Independent tables include features of only one entity. In fact they do not include any foreign keys. These tables do not need results of clustering of other tables. But dependent tables need results of clustering of other tables because these tables have at least a foreign key and we must replace foreign key with appropriate field for clustering. We first cluster independent tables and then transmit the results to tables that need them, and after that cluster dependent tables. After clustering all tables, a new table is generated which is called the Final table and includes the results of clustering tables. This table is clustered at the final step. Clustering Final table is similar to dependent tables. Figure 1 shows this approach. The MRC approach can cluster independent tables in parallel.

Also we can cluster some dependent tables in parallel which are not dependent to each other. This feature of proposed approach leads to a high improvement in clustering efficiency especially to cluster large databases which include many tables.

We use of K-means algorithm for clustering each table of our database. We present a new version of implementation K-means into SQL. Implementation into SQL has many benefits when have large database. These benefits described in follow.

The MRC has a reasonable accuracy. The MRC can use each existing clustering algorithm for clustering each table. We can select an algorithm for a table with respect to the type of its fields. So, this feature improves database clustering accuracy rather than when we use an algorithm for clustering all of them.

The paper is organized as follow: Section 2 introduces definitions and an overview of multi relational data mining. Also we explain an implementation of data mining algorithms into SQL in this section. Section 3 proposes MRC approach to perform clustering in a database with multi tables. We present implementation of k-means into SQL in section 4. Section 5 Contains experiments to evaluate performance with real database. Section 6 discusses related work. Section 7 provides general conclusions.

2. Multi Relational Data Mining

Multi Relational Data Mining is inspired from the relational model [15]-[17]. This model presents a number of techniques to store, manipulate and retrieve complex and structured data in a database consisting of a collection of tables. It has been the dominant paradigm for industrial database applications during the last decades, and it is at the core of all major commercial database systems, commonly known as relational database management systems (RDBMS). A relational database consists of a collection of named tables, often referred to as relations that individually behave as the single table that is the subject of Propositional Data Mining. Data structures which are more complex than a single record are implemented by relating pairs of tables through so-called foreign key relations. Such a relation specifies how certain columns in one table can be used to look up information in corresponding columns in another table, thus relating sets of records in the two tables [18].

2.1. Clustering in Multi Relational

Clustering in multi-relational environments has been studied in [8]-[10]. For computing similarity between two tuples, they consider not only the attributes of the two tuples, but also those of the tuples related to them (joinable

via a few joins) [8], [9]. However, these approaches suffer from poor efficiency and scalability, because in a large database an object is often joinable with thousands of tuples via a few joins, which makes it very expensive to compute similarity between each pair of objects. Moreover, they use all features indiscriminately and may not generate good clustering results that fit the user's goal [11].

In [9] based on recent advances in the area of multi-relational distance metrics and present RDBC1, a bottom-up agglomerative clustering algorithm that relies on distance information only.

CrossClus [11], [12] is a new algorithm which performs cross-relational clustering with user's guidance. CrossClus is carried out as follows: A user specifies a clustering task and selects one or a small set of features pertinent to the task. This method takes care of both quality in the feature extraction and efficiency in clustering. The CrossClus depends on the user and calculates many complex measures.

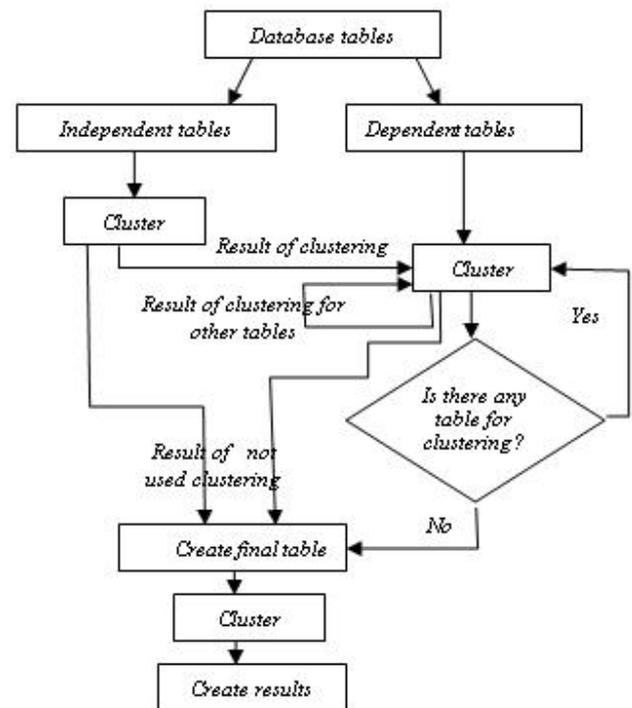


Fig. 1. A schema of the MRC approach

But we propose a new approach for clustering a database which use of existing algorithm that perform clustering in a table. We can cluster each table with an appropriate clustering algorithm. Our approach improves clustering's speed and clusters a database with a reasonable accuracy. Our approach has appropriate efficiency for databases that have many tables.

¹ Relational Distance-Based Clustering

2.2. Motivation to implement algorithm in SQL

There are several reasons for which implementing algorithm in SQL turned out to be an interesting problem. One of the main points is that this task looked trivial at first sight, but it turned out to be challenging when we faced the problem of handling large data sets with high dimensionality. In [19] the reasons that motivated us to implement our algorithm in SQL are mentioned.

3. The MRC approach

We propose a new approach clustering in multi-relational environment. Our approach is called MRC. The MRC divides tables in two categories. One category is independent tables and another category is dependent. Independent tables include features of only one entity. In fact they don't include foreign key. These tables do not need results of clustering of other tables. But dependent tables need results of clustering of other tables because these tables have at least one foreign key and we must replace foreign key with appropriate field for clustering. We first cluster independent tables and then transmit the results to tables that need them, and after that cluster dependent tables. After clustering all tables, a new table is generated which is called the Final table and includes the results of clustering tables. This table is clustered at the final step. Clustering Final table is similar to dependent tables. Figure 1 represents this approach.

3.1. Independent table

Independent tables do not need clustering results of any other tables. In relational databases, these tables generate for entities where participate in relationship. These tables include only attributes of one entity and do not have any foreign key or fields belong to other entities. These tables can be clustered with existing methods which perform clustering within a single table. An independent table can be determined with the table's schema. If there is no any field related to other entities, then this table is an independent table.

For clustering these tables, we first select an appropriate algorithm for each table considering the type of fields, then perform the clustering, and after that applying the generated results for clustering dependent tables with making the Final table. The MRC produces a label of generated cluster and use of this label for clustering dependent tables. This label must represent distance between points. We use center of clusters as the clusters labels. We replace any foreign key in dependent tables with proper label. Foreign key in the dependent table is the primary key in the independent table. We replace foreign keys with cluster's label. These Cluster's label generate

after clustering tables which include these foreign key as primary key.

After performing clustering an independent table, the MRC creates a table that represents distance between each other centers of clusters. We called this table distance table. We use of distance table for clustering dependent table which includes primary key of that independent table. Some of independent tables don't use to cluster other tables. These tables belong to entities which have a relationship with target entity. These tables have one field in Final table. Distance between two centers of two clusters can be computed with distance metric that was used by the clustering algorithm.

3.2. Dependent table

We call those tables which need the clustering results of the other tables, dependent tables. In relational database, these tables generate for multi-value attribute, and for fields which represent relationship between entities. These tables include foreign key. These foreign keys should be replaced with appropriate fields which are proper for clustering dependent tables. The MRC replaces foreign keys with other corresponding fields. These corresponding fields are produced after clustering those tables in which the foreign keys have been used as primary keys. Suppose table A has two fields A_F1 and A_F2. First field is foreign key that in table B is primary key. This primary key in B is B_F1. The MRC uses clustering result of table B in clustering table A. To determine distance between two records of table A, we should determine the distance between their fields A_F1. The MRC determines this distance using result of clustering table B.

Existence a value for field A_F1 means that this record has at least related record in table B. Instead of calculating distance between two values of field A_F1, the MRC calculates distance between their related records in table B. The MRC uses results of clustering table B to reduce total time of clustering. We do not calculate again distance between two records of table B. The centers of clusters are used to estimate the distance. Here we utilize distance table that we generated after clustering table B. In relationship with m-n multiplicity between table A and B, one record in table A has relation with several records in table B. In this case, the MRC considers which cluster of B includes most of related records. The MRC uses center of this cluster to estimate distance. Using this method, the accuracy is reduced but clustering's efficiency is increased. The MRC is a proper method for any databases that include many tables with a high dependency between them.

3.3. Final table

The MRC clusters independent tables firstly, whereas clustering independent tables do not need result of other

tables, so we can cluster all of the independent tables concurrently. Secondly, the MRC clusters dependent tables. It is possible some of them depend on other tables. So we may cluster them in several steps.

Finally, we generate a table which is called the Final table. The Final table fields are results of some independent and dependent tables. Clustering the Final table is the last step of clustering of the database. In fact the MRC cluster each table of the database, and then cluster the clustering results.

4. Implementation

We implement the MRC approach for clustering Iranian Families' Income-Expense of 2001-2005 national statistics. To cluster each table, we upgrade the proposed algorithm in [13], [14]. There exists an implementation of K-means by SQL. We upgrade that to generate appropriate results after clustering each independent table, for uses to cluster dependent table. After clustering tables, our implementation creates and clusters the Final table that includes clustering results of other tables.

4.1. A new implementation of K-means in SQL

We present a new implementation of K-means into SQL which able to cluster a relational database. We utilize algorithm in [13], [14] and apply it for our purpose. We explain some parts of algorithm which we change or increase it.

Independent tables are clustered by the proposed algorithm in [13], [14]. After clustering a table, we generate a new table. This table is called distance table and includes three fields. This table represents distance between centers of generated clusters. Required query to generate this table shows here. This query calculates distance between two centers and inserts it together identifier of those in Distance table. We generate a distance table for each table. The Distance tables are a small table, because they have three fields and few records. We can cluster all of independent table concurrently.

```

Insert Into DistanceTable
Select C1.j, C2.j, SQRT (Sum ((C1.val - C2.val) * (C1.val
- C2.val)))
From C as C1, C as C2
Where C1.l = C2.l
Group by C1.j, C2.j

```

The table C includes three fields. J shows cluster center identifier, L shows dimension and Val represents value of center j in dimension l. When we want clustering dependent tables, we utilize appropriate distance table

instead of calculate distance between foreign keys. Each record of dependent table may relate to more one record of independent table. Each record of dependent table must relate at least to one cluster center of independent table clusters center. So we generate following query for this purpose. It selects a cluster center which includes most related records. If two clusters have same number of related record, this query selects that cluster with lower identifier. This query uses two tables. One of them is Relationship table which represents relation between records of independent table and dependent table. Another table includes record's identifier of independent table and its cluster center. We join these tables and extract required result. The Relationship table has two fields, i and j. This table represents a relation between record i of dependent table with record j of independent table. Another table is R84output. That has two fields, k and clus. In this table, k is record identifier of independent table and clus represents its cluster.

```

Select t1.i, min(t1.clus) as label
From
  (Select i, Clus, Count (*) as y3
  From R84output, Relationship
  Where Relationship.j = R84Output.k
  Group by i, Clus
  ) as t1
,
  (Select t3.i, max (t3.y2) as e
  From
    (Select i, Clus, count(*) as y2
    From R84output, Relationship
    Where Relationship.j = R84output.k
    Group by i, clus) as t3
  Group by i
  ) as t2
Where t1.i = t2.i and t1.y3 = t2.e
Group by t1.i

```

Now we can calculate distance between records in dependent tables. To calculate Euclidean distance, at first we calculate distance between fields except foreign keys. After that we add distance between centers of related clusters to calculated distance. Below query performs this task. This query uses three tables, YD, R84output and R84distance. The YD has three fields, i, j and Distance. J and i represent record identifier and the Distance shows distance between two records. The R84Distance table has three fields, i, j and Distance. The R84Distance table shows distance between two centers of independent table's clusters. The YD table represents distance between two records of dependent table. The R84output table described before.

```
Update YD
```

Set distance= sqrt((Distance* Distance)+(dis.Distance
 * dis.Distance))
 From YD,
 (Select R1.k as k1, R2.k as k2,
 R84distance.Distance
 From R84output as R1, R84output as R2,
 R84distance
 Where R1.clus = R84distance.i and R2.clus =
 R84distance.j
) as dis
 Where dis.k1= YD.i and dis.k2 = YD.j

In our implementation, we must select one of records as cluster center. So we determine cluster center by proposed algorithm in [13], [14], then we find the nearest record to that as cluster's center in our implementation.

After clustering independent and dependent tables, we generate the Final table. The Final table includes result of tables which don't use in clustering other tables. Clustering this table performs like dependent table.

In this section, we present the important parts of our implementation and we don't mention details and other parts which exist in [13], [14].

4.2. Code Generator and used database

We generate a Code Generator which produces the required SQL code to cluster a database. The SQL code generator is programmed in the C# language, which connected to the SQL Server 2000. It allows user to select tables, fields, dependency between tables and target entity. We can cluster a database without the user, but these guidance of user help us to generate more meaningful results. The SQL code for clustering is created by the Code Generator automatically.

Our approach is tested by a real database. This database is Iranian Families' Income-Expense of 2001-2005 national statistics. It has 42 tables that include families' income and expense for foodstuffs, wearing stuff, training, traveling and curing.

5. Results

We implement the MRC approach in SQL. All tests were done on a 2GHz Pentium-4 PC with 1GB memory, running Windows XP. We test the algorithm with different size of database and different number of tables. We report result in follow.

In the first experiment, we consider clustering's time with respect to size of database. So we select 10 relations of database and in each run we change size of database. The time for each clustering shows in table 1.

TABLE 1
 CONSUMING TIME TO CLUSTER DATABASES WITH DIFFERENT SIZE

Database size(MB)	Time (Second)
5	10
10	23
50	45
100	102
200	194
500	321
1000	504

We test clustering's time with respect to number of relation in database. We chose 100 MB for size of database and change number of relation. Table 2 represents the obtained result.

TABLE 2
 CONSUMING TIME TO CLUSTER DATABASES WITH DIFFERENT NUMBER OF TABLES

Number of tables	Time (Second)
5	75
10	102
20	142
30	185
40	218
45	251

We change our implementation with optimizing in K-means algorithm [14]. These changes have efficient effect on our results, but we don't mention those results.

6. related work

The Clustering has been extensively studied for decades in different disciplines including statistics, pattern recognition, database, and data mining, using probability-based approaches [20], distance based approaches [3], [21], subspace approaches [22], and many other types of approaches.

Multi Relational Data Mining studied in many articles. But clustering in multi relational environment studied in a few articles. In [8]-[13] proposed some algorithms. But none of them proposed an approach. The RDBC [9] is a bottom-up agglomerative clustering algorithm that relies on distance information only. CrossClus [11], [12] is a clustering algorithm depends on user, and it is very complicated, so implementation it in SQL is very difficult.

Research on implementing data mining algorithms in SQL or extending SQL for data mining purposes includes the following: Association rules mining is explored in [23]. SQL extensions to define complex aggregate functions having tables as parameters are proposed in [24]; these extensions are used to tackle the problem of mining

association rules. The MSQL language [25] provides extensions to query a set of discovered association rules.

Several clustering algorithms implement in SQL. Programming the EM clustering algorithm in SQL is explored in [19]. K-means clustering in SQL appeared in [13], [14], where Standard K-means and some optimizations were introduced.

Implementation data mining algorithm in multi relational environment into SQL is a new topic for research.

7. Conclusion

In this paper, we propose an approach to cluster a relational database. This approach is called MRC. The MRC is very simple and efficient for clustering a huge relational database. Most of existing clustering algorithms work only on single table. We use them for clustering several tables that related to each other. In the MRC can employs different algorithms to cluster tables. Because these algorithms select with respect to type of table's fields so this feature improves accuracy.

The MRC can cluster several tables in parallel. We implement the MRC into SQL. Implementation in SQL is very important for clustering large databases. Whereas the MRC uses common clustering algorithm to cluster a relational database, so if we can implement these algorithms in SQL, we can cluster a relational database by SQL code. Guidance of users can help the MRC to cluster with high accuracy and generate result with respect to users' goal.

8. Acknowledgments

We are thankful for any support of Persian Gulf University, Bushehr, Iran, as well as Data Mining Research group at Computer Research Center of Islamic Sciences (CRCIS), NOOR co. P.O. Box 37185-3857, Qom, Iran

References

- [1] S. Dzeroski, "Multi-relational data mining: An introduction," SIGKDD Explorations. pp. 1-16, 2003.
- [2] A. Jain, R.C. Dubes, Algorithms for clustering. Prentice-Hall, Englewood Cliffs 1988.
- [3] L. Kaufman, P. J. Rousseeuw, Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley and Sons 1990.
- [4] B.S. Everitt, S. Landau, M. Leese, Cluster analysis. Arnold, London 2001.
- [5] R.J. Hathaway, J.C. Bezdek, Y. Hu, "Generalized fuzzy C-Means clustering strategies using Lp norm distances," IEEE Trans Fuzzy Systems, Vol. 8, No. 5, pp. 576-582, 2000.
- [6] R.N. Dave, R. Krishnapuram, "Robust clustering methods: a unified view," IEEE Trans Fuzzy Systems, pp. 270- 293, 1997.
- [7] N.B. Karayiannis, M.M. Randolph-Gips, "Soft learning vector quantization and clustering algorithms based on non- Euclidean norms: multinorm algorithms," IEEE Trans Neural Netw, pp. 89-102, 2003.
- [8] T. Gärtner, J. W. Lloyd, P. A. Flach, "Kernels and Distances for Structured Data," Machine Learning, pp. 205- 232, 2004.
- [9] M. Kirsten, S. Wrobel, "Relational Distance-Based Clustering," ILP, pp. 261-270, 1998.
- [10] M. Kirsten, S. Wrobel, "Extending K-Means Clustering to First-order Representations," ILP, pp. 112-129, 2000.
- [11] X. Yin, J. Han, and P. S. Yu, "CrossClus: User-Guided Multi-Relational Clustering," In the journal of Data Mining and Knowledge Discovery (DAMI), Springer, pp. 321-348, 2007.
- [12] X. Yin, J. Han, and P. S. Yu, "Cross-Relational Clustering with User's Guidance," In 11th Int'l. Conf. on Knowledge Discovery and Data Mining (KDD'05), Chicago, IL, pp. 344-353, 2005.
- [13] C. Ordonez, "Integrating K-Means Clustering with a Relational DBMS Using SQL," IEEE Trans. Knowl. Data Eng. pp. 188-201, 2006.
- [14] C. Ordonez, "Programming the K-Means Clustering Algorithm in SQL," Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining, pp. 823-828, 2004.
- [15] C. Date, An Introduction to Database Systems. Volume I, The Systems Programming Series, Addison-Wesley, 1986.
- [16] I. D. Ullman, Principles of Databases and Knowledge-Based Systems. Volume I, Computer Science Press, 1988.
- [17] J. Ullman, J. Widom, A First Course in Database Systems. Prentice Hall, 2001.
- [18] A. J. Knobbe, Multi-Relational Data Mining. (Frontiers in Artificial Intelligence and Applications). IOS Press, 2006.
- [19] C. Ordonez, P. Cereghini, "SQLEM: Fast Clustering in SQL Using the EM Algorithm," Proc. ACM SIGMOD Conf., pp. 559-570, 2000.
- [20] P. Cheeseman, "AutoClass: A Bayesian Classification System," In Proc. Int'l. Conf. Machine Learning, Alberta, Ann Arbor, MI, pp. 54-64, 1988.
- [21] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," In Proc. Berkeley Symp. Mathematics, Statistics and Probability, Berkeley, CA, pp. 281- 298, 1967.
- [22] C. C. Aggarwal, P. S. Yu, "Finding Generalized Projected Clusters in High Dimensional Spaces," In Proc. ACM SIGMOD Int'l. Conf. Management of Data, Dallas, TX, pp. 70-81, 2000.
- [23] H. Jamil, "Ad Hoc Association Rule Mining as SQL3 Queries," Proc. IEEE Int'l Conf. Data Mining, pp. 609-612, 2001.
- [24] H. Wang, C. Zaniolo, and C. R. Luo, "ATLAS: A Small but Complete SQL Extension for Data Mining and Data Streams," Proc. Very Large Databases Conf., pp. 1113-1116, 2003.
- [25] T. Imielinski, A. Virmani, "MSQL: A Query Language for Database Mining," Data Mining and Knowledge Discovery, vol. 3, no. 4, pp. 373-408, 1999.

Concurrent Agent-enabled Extraction of Computational Fluid Dynamics (CFD) Features in Simulation

Clifton Mortensen, Robert S. Woodley, *Member, IEEE*, and Steve Gorrell

Abstract—High fidelity, large scale simulations of complex systems pose a very difficult challenge to the scientist trying to understand the physical domain and characteristics of their system. Often it is impossible to manually search through all the data that can come out of just one of these simulations which may range from Gigabytes to Terabytes. Furthermore, it may be impossible to visualize the multi-dimensional interactions that are occurring in the data. What is needed is a tool that will concurrently mine the vast data sets that are produced and alert the scientist of important events (either planned or unplanned). 21st Century Systems, Inc. and Brigham Young University introduce our *Concurrent Agent-enabled Feature Extraction (CAFÉ)* concept to answer this challenge. CAFÉ identifies key features that are needed by CFD researchers. CAFÉ assists the researcher by automatically performing the algorithmic tests for the features and recommends the features that are most interesting. This is based on a search criterion performed concurrently within a simulation run. The initial results show that our concept is able to identify key features early in a simulation run. This allows the researcher to see how the simulation is progressing.

Keywords—Massive data set data mining, Concurrent simulation data mining, Computational Fluid Dynamics simulation, Agent-based data mining

I. INTRODUCTION

FIGURE 1 shows a conceptual view of the CAFÉ tool. Software running a physics-based simulation produces enormous amounts of data. The data is mined with various algorithms contained in agents. The feature extraction transforms the multi-variate data into reduced order information and is exchanged amongst the agents and with the operator. The transformed data is much easier to share, allowing the system to tune itself and guide the data-mining efforts. This pseudo genetic algorithm approach relies on swarm technology to guide the tuning. It is well known that swarming algorithms [1], [2] reach near optimal solutions given adequate coverage of the domain. CAFÉ's agents will cover the space of the domain.

This work was supported by the Air Force Office of Scientific Research under the STTR award FA9550-08-C-0058.

C. Mortensen is with the Department of Mechanical Engineering, Brigham Young University, Provo, UT, USA (phone: 801-422-2320; e-mail: cmort22@gmail.com).

R. Woodley is with 21st Century Systems, Inc., Ft. Leonard Wood, MO, 65473, USA (phone: 573-329-8526, fax: 573-329-8509 e-mail: robert.woodley@21csi.com).

S. Gorrell is with the Department of Mechanical Engineering, Brigham Young University, Provo, UT, USA (phone: 801-422-2759; fax: 801-422-0516; e-mail: sgorrell@byu.edu).

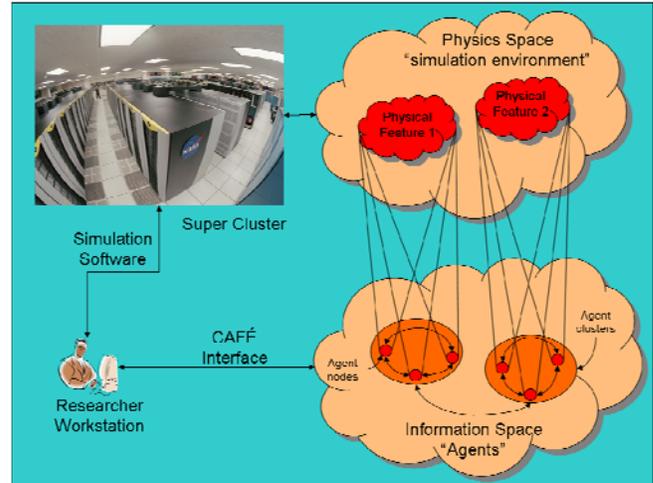


Figure 1: CAFÉ conceptual picture showing how the physics domain maps to nodes in the information space. These nodes communicate among each other, direct the data-mining activity, and interact with the operator.

In Section II.B we explain what is involved in a CFD simulation. The key point to realize is the amount of time it takes to run a CFD simulation. For a CFD simulation of a complex system, it may take hundreds of hours on expensive computing clusters to run a complete simulation. Suppose that after running the simulation for a week and then analyzing the results the researcher discovers that an unanticipated vortex developed that dramatically cuts the efficiency of the turbomachine. He now has to rerun that simulation, along with the expense that goes with it, and perform the run again fixing the problem.

What if there was a tool that could extract critical features concurrently while the simulation runs showing where key features are occurring? Further suppose that this tool would be able to provide this information far earlier in the simulation cycle so to not waste the researcher's valuable computing time. The CAFÉ tool is being designed to perform this task. We present here some initial findings and feasibility of the idea of concurrent data mining of massively large data sets.

II. METHODS

A. An agent-based solution

The mathematics of performing feature extraction is well known as we will show in the following section. The

innovation applied to the problem is the use of an agent-based structure that was designed for decision support software applications.

Due to the size of the data sets being generated, what often happens is that much of the data is discarded until the simulation has converged. Some simulations may be so massive that even the converged simulation may have too much data (in the terabytes) to store for analysis. What commonly occurs is that either features or 2D planes are extracted from the raw data and the remainder of the data is discarded. The parallel agent structure that we use on the CAFÉ project is able to simultaneously extract features from the data as it is generated thereby allowing the system to hold onto information from earlier in the simulation run.

This parallel extraction that occurs with the agent structure can then be used to show the researcher the progress of the simulation, or alert the user of the development of strong feature signatures such as a strong vortex. The researcher may look at the data concurrently with the simulation and decide to continue with the simulation or not.

The agents are armed with some advanced tools, described in Section II.C.2, to automatically perform the feature extraction. Unlike the final post-processing and analysis, CAFÉ must be able to make decisions on the feature extractions without the assistance of user input. By providing rules for extraction and a mathematically rigorous method for evaluating the uncertainty in the feature extraction we are able to determine when a feature is true or simply an artifact of an un-converged simulation. The agent structure, based on decision support software, allows for these plus provides the alert system when human analysis is warranted of a feature.

B. Computational fluid dynamics (CFD) simulations

CFD simulations numerically solve the governing equations of fluid motion. A common formulation is the Navier-Stokes equations formed from the application of Newton's second law to fluid motion combined with the conservation of mass. The result is nonlinear partial differential equations with analytical solutions available in only the simplest fluid flow cases. These equations are further complicated by the chaotic behavior of turbulent flow. Fluid flows generally belong to one of two regimes: laminar or turbulent. Laminar flows are simpler flows characterized by smooth and parallel streamlines, while turbulent flows are characterized by seemingly random, chaotic fluid particle motions. Large scale CFD simulations commonly contain turbulent flows.

CFD simulations involve a few basic steps: creation of a computer-aided design (CAD) model, generation of a computational grid, numerical solution, and data post-processing. Obtaining the numerical solution is an iterative process. An initial velocity field is guessed and then at each iteration a new velocity field is attained for the entire computational domain. General CFD generated data are scalar and vector variables which are stored at the node

locations of the computational grid. General scalar variables include: x, y and z node locations, density, and temperature. General vector variables include: velocity and momentum. Velocity components are the basis for the majority of feature extraction algorithms.

There are multiple techniques to determine when a simulation is complete. In unsteady simulations variables such as velocity and pressure can be monitored. When these variables become periodic it is a good indicator that a solution has converged. Also, residuals can indicate numerical convergence. A residual is defined as an imbalance of a quantity within a computational cell [3]. Residuals will vary dramatically early in a simulation and will dampen out as a solution reaches numerical convergence.

Compressors in gas turbine engines have seen some of the most rigorous application of CFD to understand the complex fluid flows involved. Gorrell et al. [4], [5] have run simulations of transonic fan stages with 166 million grid points and simulations of entire fans with over 300 million grid points. These simulations typically run on 900 to 1200 processors. The amount of data generated from this type of simulation is in terabytes. Present approaches to post-processing require the researcher to slowly sift through data and find useful information based on intuition and previous experience. Fortunately fluid flows are comprised of basic features that serve as building blocks for specific flow structures. Extraction of these fluid flow features can help researchers visualize, analyze and interpret the data.

C. The CAFÉ concept

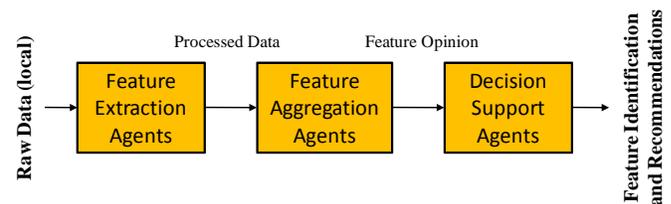


Figure 2: Information flow diagram for CAFÉ.

Figure 2 shows the information flow for CAFÉ. Agents running feature extraction algorithms at the local level identify possible features. These features are aggregated and evaluated at the feature aggregation level. The aggregation agents are running evidential reasoning algorithms. Finally, the feature opinions from the aggregation level are collected and ranked for the user by the decision support agents. The decision support agents are interactive with the user using the agent-based framework called the Agent Enhanced Decision Guide Environment (AEDGE[®]) framework.

1) Feature extraction

CAFÉ has concentrated on three basic fluid structures: vortices, shock waves, and separation and attachment lines. A vortex is one of the most common and most useful flow features. The physical structure of a vortex is made up of two

dependent parts: the vortex core line and the swirling fluid motion around the core line. Vortex extraction algorithms specifically suited for complex data sets focus on extraction of core lines rather than extraction of vortex regions.

Two vortex extraction algorithms were selected for CAFÉ: the eigenvector method of Sujudi-Haimes [6] and the higher-order method of Roth-Peikert [7]. These algorithms have their respective strengths and weaknesses. The Sujudi-Haimes algorithm is perfect for linear flows and works well with strongly rotating vortices. It does have problems with curved and weakly rotating vortices. The Roth-Peikert algorithm correctly identifies curved and weakly rotating vortices. Both fail to capture accelerating, curved vortices.

Shock waves occur in fluid flow only when the velocity of the fluid exceeds the speed of sound. Shock waves are characterized by sudden discontinuities in pressure, density and velocity. Detection of a shock wave in CFD data is comparable to edge detection in image processing applications.

Two shock wave extraction algorithms were selected for CAFÉ: the Lovely-Haimes algorithm [8] and the Ma-Vermeer algorithm [9]. These two shock algorithms have outputs that are slightly different. The output of the Lovely-Haimes algorithm is a volume that encompasses a shock, while the output of the Ma-Vermeer algorithm is a surface designed to locate the shock exactly. Both of these algorithms will be enhanced in CAFÉ through the use of multiple scalar values to compute derivatives, i.e. using density and pressure instead of one or the other.

Separation and attachment lines are lines on the surface of physical bodies where the fluid flow abruptly moves away from, or returns to, the surface of a body.

Two algorithms developed by Kenwright [10], [11] were selected: the phase plane algorithm and the parallel vector algorithm. Both algorithms were designed to detect straight lines and fail in detecting curved lines. The phase plane algorithm is better suited for unstructured grids, and it extracts disjointed line segments. The parallel vector algorithm works well with curvilinear grids and extracts continuous line segments.

All of the algorithms implemented thus far in CAFÉ have the same basic composition: perform computations on subsets of the flow domain, apply some filtering mechanism, and aggregate the remaining selected regions into features. This structure allows the feature aggregation agents to make decisions about the believability of features and their components before the selected regions have been aggregated.

This structure has two immediate consequences. First, since the agents operate on the feature before it has been aggregated, the parts of a feature with a high believability can easily be selected while disregarding other parts with a low believability. Second, it allows a combination of features that have been extracted by multiple algorithms. In other words,

if nodes are extracted close together by two separate extraction algorithms, they can be operated on by a feature aggregation agent and then combined into one distinct feature rather than two disjoint features.

2) Feature aggregation

One of the key functionalities of CAFÉ is the ability to identify features and combinations of features. As mentioned above, certain feature extraction algorithms work better than others for a given situation. Additionally, since CAFÉ is trying to perform concurrent extraction some extracted features may actually be incorrect depending on what iteration the simulation is performing. As a result CAFÉ needed a way to select features according to the believability of the feature based on when in the simulation the feature occurs, what conditions the feature is extracted under, and if previous iterations contain the same feature.

The tool we employ to quantify the believability of a feature is called the Evidential Reasoning Network[®] (ERN[®]) tool. The ERN tool uses a three valued logic that measures the belief, disbelief, and uncertainty in an opinion and intrinsically handles these in an algebraic space. The three valued logic uses subjective logic developed by Jøsang [12], [13]. For our purposes we set the belief as the strength of the feature (i.e., how strong a vortex signature is detected), the disbelief is set by the environmental conditions (i.e., for a curved flow the Sujudi-Haimes method for vortex core detection is less than ideal), and the relative residual is the uncertainty. As will be shown in the results, as the simulation progresses the CFD is balancing the Navier-Stokes and continuity equations to come up with a stable solution. Plots of the residuals are a good indication on the convergence of the simulation. Early in the simulation little can be determined and the residual plots show quite large residuals with high volatility. From these, we form an opinion on the feature for further processing at successive iterations culminating in the final decision support to the researcher.

Early in the simulation, residuals are high and there typically are large variations in the solution. The ERN tool keeps track of previous solutions and compares the current solution with the new solution. If the solution contains large variations, ERN discounts these solutions as being unlikely. However, as the simulation progresses, the real solutions begin to appear. ERN aggregates the solutions when they repeatedly appear. This produces a trust value that we use to tell the operator that a possible solution (i.e., a vortex core) is developing. We can show that core line to the researcher far earlier than traditional methods (often in half the iterations traditionally required).

The agent structure explained above facilitates the ERN by providing a large input space. Each feature agent is a node in the network. So, as a feature such as a vortex core is identified at one agent node, nearby agents will act to either bolster that assertion or discount the assertion. In this way, small anomalies can be automatically filtered out.

3) Decision Support

Ultimately, the goal of CAFÉ is to assist the researcher in performing CFD simulations. CAFÉ assists by providing information on the feature identified, what filters to use to better visualize the feature, provide visualization cues, compare features (size, strength, etc.), guide the selection of initial conditions, and do so concurrently with the simulation.

a) Feature information

As the features are identified CAFÉ performs two actions: evaluate the viability of the feature and provide information about how the feature was found. We discussed evaluating the viability of a feature by using the ERN tool to form a judgment opinion in Section II.C.2. The additional information includes the location, size, and strength of the feature as well as indicating which extraction algorithm found it. This last information is critical in the researcher's trust of the feature. For example, if the algorithm extracts a vortex core in a curved flow using the Roth-Peikert method, the researcher will have greater confidence than if the extraction algorithm was the Sujudi-Haimes, which works well in linear flow.

b) Filter selection

Often filters are applied to weed out extraneous information. For example, small eddies may occur around the main vortex core. These eddies are often a nuisance since they obscure the main core. A filter may be able to remove the extra eddies. CAFÉ will be able to provide information on filter parameter settings to remove certain extraneous data.

c) Visualization cues

In conjunction with the filters, CAFÉ will provide the researcher with where to look in the data to see the features. One of the advantages of using the agent architecture is that it is easy to develop decision trees for what to look for in the data. The researcher will be able to select the criterion of the types and characteristics of features. The agent structure then communicates back to the user which agents contain the information in a rank order.

d) Feature comparison

Feature comparison is performed by the ERN tool. The bigger, stronger, more certain the feature is, the higher its trust value will be. CAFÉ uses the trust value to perform the feature ranking.

e) Initial condition selection

Finally, as the CAFÉ system matures, we plan to adapt machine learning such that it may learn various solutions to conditions and provide feedback to the user on making initial condition guesses. The goal is not to attempt an exact solution, but a better educated guess at the initial flow solution. The better the initial guess, the faster the

convergence will be. It is also possible that the initial guess may be so far out of line that the system will not converge. By applying basic learning rules we can produce better initial guesses so that the system will have a better chance at convergence.

III. EXPERIMENTAL RESULTS

A. Test setup

Initial testing has been done to determine the feasibility of concurrent data mining and begin the formation of agent opinions. A CFD simulation was run on a blunt fin using the steady Reynolds Averaged Navier-Stokes (RANS) equations. The grid was generated as a structured curvilinear grid with 44,000 nodes. The Reynolds number based on the fin diameter was 630,000 and the one equation Spalart-Allmaras method was used to model turbulence. The flow field was initialized using a small velocity in the downstream direction. The solution reached full convergence at 900 iterations. This simulation is not of the magnitude where concurrent data mining is required, but it does yield useful information about concurrent data mining that can be used in the formation of agent opinions.

Concurrent data mining was replicated by exporting the entire flow field data set at specified iteration numbers throughout the flow solution. Vortex core lines were then extracted from each of the saved data sets using the Roth-Peikert and the Sujudi-Haimes methods. Both algorithms extracted features without any agent alterations.

B. Test results

Figure 3 displays the fully converged simulation residuals. Each of the residuals drops at least two orders of magnitude and then remains at a constant value. Also, the residuals contain a relatively low volatility and exhibit an overall decreasing pattern. This information can be used to set the uncertainty of agent opinions.

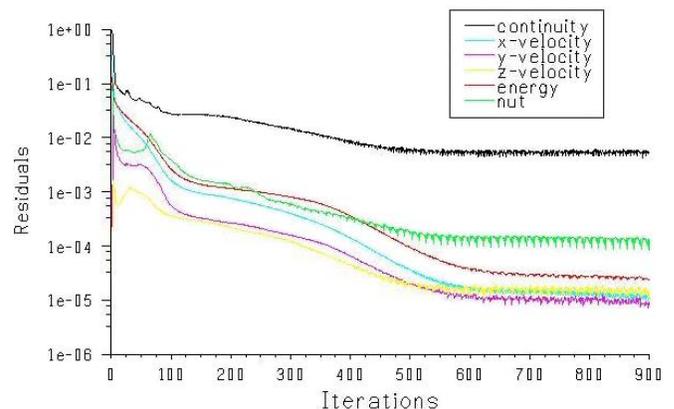


Figure 3: Residual plot of the fully converged CFD simulation.

Early in a simulation, agent opinions will contain a high amount of uncertainty. As the simulation progresses and the residuals decrease, the agent uncertainty will also decrease. When residuals remain constant the uncertainty will decrease

quickly to zero which coincides with a converged solution. Most solution residual behavior will not be so well behaved, but a converging solution will follow the same basic progression: large residuals early, relatively decreasing residuals following and then some period of residuals at a constant value. This progression is used as a template for the uncertainty of agent opinions.

The two algorithms differed slightly as to what iteration number the unconverged solution extracted cores, showing only slight variation from the extracted cores of the converged solution. The Roth-Peikert method reached this mark at 540 iterations into the solution or 60% of fully converged. This is to be expected since 540 correspond to the location in Figure 3 where the residuals begin to flatten out. The Sujudi-Haimes method unexpectedly reached this mark at 495 iterations into the solution or 55% of fully converged. This was unexpected because it was 45 iterations sooner than the Roth-Peikert method and at 495 iterations the solution residuals are still decreasing. This information is used in an agent opinion to more accurately calculate the time when an algorithms uncertainty is approaching zero.

Figure 4 is a comparison of the extracted core lines from the two different algorithms at full solution convergence. Black represents Sujudi-Haimes, green represents Roth-Peikert, and yellow represents the sections that were extracted by both algorithms. In all of the figures displaying the blunt fin, the flow is moving from the lower right to the upper left.

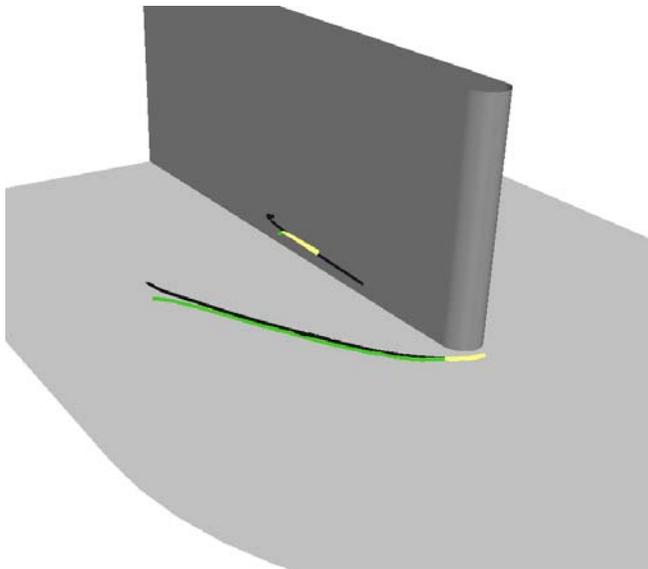


Figure 4: Comparison of the Roth-Peikert (green) and the Sujudi-Haimes (black) extracted vortex core lines at full solution convergence. Yellow represents the fact that both algorithms extracted the line. Both methods agree well on the horseshoe line, but differ on the length of the fin line.

The core line that runs along the fin will be called the ‘fin’ line. This fin line is extracted in the same area by both algorithms, but differs in length. The Sujudi-Haimes method extracts a fin line that extends beyond the Roth-Peikert fin

line in both the upstream and downstream directions. This represents a case where CAFÉ would verify the existence of a vortex core line at the locations where the two methods overlap and it would need to test the belief that the remaining parts of the vortex core line were correctly identified.

The core line that begins upstream of the fin and wraps around the fin will be called the horseshoe line. The horseshoe line has only small variations between the two methods with the largest variation downstream. Since CAFÉ operates on nodes before they are aggregated this feature can be combined into one feature with the highest belief value.

Both algorithms similarly identify some core lines and differently identify others. This reinforces the idea that multiple extraction methods are required to correctly identify all vortices within the flow domain.

Figures 5-7 display the extraction results obtained from the Roth-Peikert algorithm. The green lines represent extracted vortex core lines from the fully converged data set and the blue lines represent extracted core lines from the unconverged data sets. In order to give a visual comparison, the core lines extracted from the converged solution, i.e. the algorithm’s correct features, are displayed with the core lines extracted at intermediate steps. When a location in the domain contains a core line from the converged and unconverged data sets, the unconverged solution (blue) line is displayed on top.

Similarly, Figures 8-10 display the extraction results obtained from the Sujudi-Haimes algorithm. The black lines represent extracted vortex core lines from the fully converged data set and the red lines represent extracted core lines from the unconverged data sets.

A useful characteristic of the extracted core lines can be seen from the evolution of the horseshoe line. At 180 iterations (Figures 5 & 8), or 20% of full convergence, the horseshoe line has its basic form but is upstream of its final position. As the solution continues, this feature moves downstream and resolves from right to left in the flow direction until it almost completely resolves at 360 iterations (Figures 7 & 10), or 40% of full convergence. This resolving from upstream to downstream trend is also especially prevalent in the Sujudi-Haimes’ fin line. It resolves strongly from upstream to downstream. This information is used to add belief to agent opinions. A feature extracted upstream will have a higher belief than a feature extracted downstream.

It is interesting to note that as the solution progresses the two extraction algorithms extract different incorrect features. These features have been filtered out of Figures 5–10. This information is used to increase the disbelief in an agent opinion when feature structures are radically different.

The blunt fin simulation is similar to a simulation run by Hung [14] where the existence of a horseshoe vortex was validated. This verifies the accuracy of the vortex features extracted by CAFÉ.

IV. CONCLUSION AND FUTURE WORK

We have presented here the conceptual idea of using an agent based data mining system for extracting Computational Fluid Dynamics (CFD) features in extremely large data sets. The agent-based approach allows for the use of highly parallel extraction techniques, thereby maximizing the use of computing clusters. The CAFÉ concept utilizes an evidential reasoning engine to evaluate the viability of extracted features. This culminates in a tool that is able to analyze the data concurrently with the CFD simulation and provide the researcher with the decision support to maximize their research time.

The research is in its initial stages but already shows considerable promise. We have been able to identify how certain features tend to grow as the simulation converges. We are able to identify when a converged pattern is forming long before the end of a typical simulation. This fact alone may save hours of simulation time from being wasted.

Our next step on concurrent feature recognition will be to analyze a larger, more complex computational data set. Concurrent extraction of other existing features will be performed to see if they behave similar to vortex core lines.

Our future plans for the CAFÉ concept is to work at integrating the various components from the disjoint set of algorithms into a cohesive package. Quite a bit of general software engineering is still needed to make the system useful to the researcher. Additionally, there are several components that need to be implemented especially in the filter selection, visualization cues, and initial condition estimating. We plan to expand the concept beyond that of CFD simulations into a more general massive data set mining tool.

ACKNOWLEDGMENT

We thank the Air Force Office of Scientific Research for sponsoring this project. We further acknowledge Dr. Rhonda Vickery for the use of the feature extraction code.

REFERENCES

- [1] M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic," *New Ideas in Optimization*, McGraw-Hill, Maidenhead, UK, England, 1999.
- [2] R. Woodley. "Ants on the AEDGE: Personality in software agents," in *Proceedings of the 25th Army Science Conference*, November, 2006.
- [3] Fluent, *FLUENT 6.3 User's Guide*, Fluent Inc., 2006.
- [4] S. Gorrell, M. List, and G. Turner. "Investigation of Loss Generation in an Embedded Transonic Fan Stage at Several Gaps Using High Fidelity, Time-Accurate CFD," ASME paper GT2008-51220, 2008.
- [5] Yao, J., Gorrell, S. E., and Wadia, A. R., "High-Fidelity Numerical Analysis of Per-Rev-Type Inlet Distortion Transfer in Multistage Fans – Part II: Entire Component Simulation and Investigation." ASME Paper

- GT2008-50813, accepted for publication in *ASME Journal of Turbomachinery*, June 2008.
- [6] D. Sujudi, and R. Haimes. "Identification of Swirling Flow in 3D Vector Fields," AIAA paper 95-1715, Am. Inst. of Aeronautics and Astronautics, 1995.
- [7] M. Roth and R. Peikert, "A Higher-Order Method for Finding Vortex Core Lines," In *Proc. Visualization '98*, pp. 143–150, 1998.
- [8] D. Lovely and R. Haimes. "Shock Detection from Computational Fluid Dynamics Results," *14th AIAA Computational Fluid Dynamics Conference*, vol. 1, pp. 296–304, Norfolk, VA, July, 2000.
- [9] K.-L. Ma, J. Rosendale, and W. Vermeer. "3D Shock Wave Visualization on Unstructured Grids," In *Proc. Symposium on Volume Visualization*, pp. 87–94, 104, 1996.
- [10] D. N. Kenwright. "Automatic Detection of Open and Closed Separation and Attachment Lines," *IEEE Visualization '98*, pp. 151–158, 1998.
- [11] D. N. Kenwright, C. Henze and C. Levit. "Feature Extraction of Separation and Attachment Lines," *IEEE Trans. on Visualization and Computer Graphics*, vol. 5, pp. 135–144, 1999.
- [12] A. Jøsang. "A Logic for Uncertain Probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, pp. 279–311, 2001.
- [13] A. Jøsang. "Subjective Evidential Reasoning," *Proceedings of the International Conference on Information Processing and Management of Uncertainty (IPMU2002)*, 2002.
- [14] C. M. Hung and P. G. Buning. "Simulation of Blunt-fin-induced Shock-wave and Turbulent Boundary-layer Interaction," *Journal of Fluid Mechanics*, vol. 154, pp. 163–185, 1985.

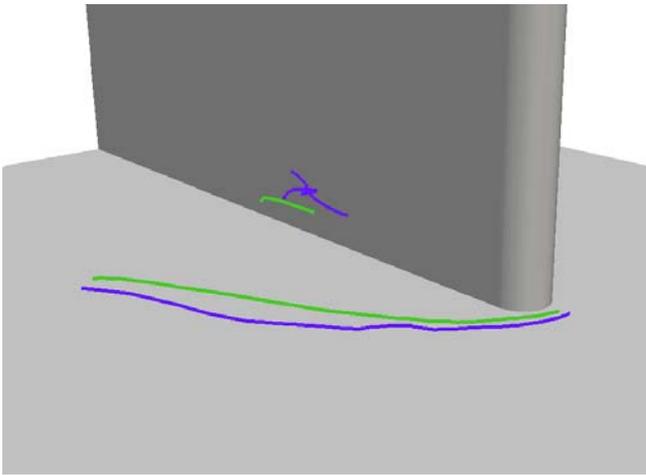


Figure 5: Comparison of Roth-Peikert extracted vortex core lines at 20% of converged solution - 180 iterations (blue) and 900 iterations (green). Horseshoe line begins to take shape upstream of final location. Fin line incorrect.

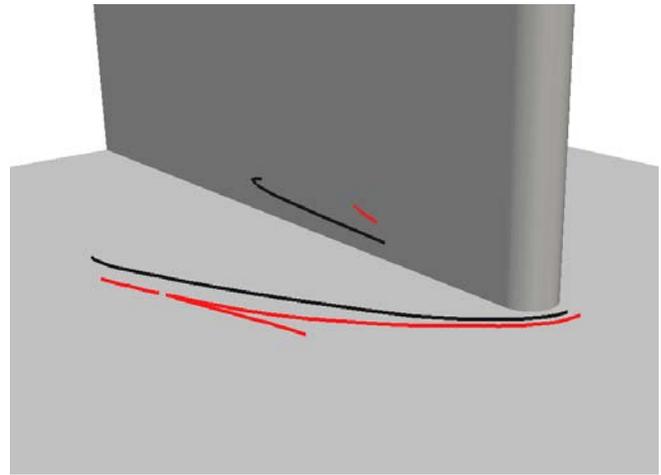


Figure 8: Comparison of Sujudi-Haimes extracted vortex core lines at 20% of converged solution - 180 iterations (red) and 900 iterations (black). Horseshoe line begins to take shape upstream of final location. Fin line incorrect.

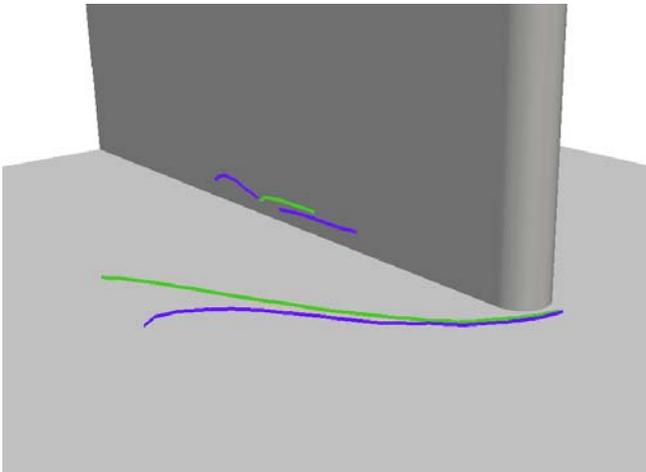


Figure 6: Comparison of Roth-Peikert extracted vortex core lines at 30% of converged solution - 270 iterations (blue) and 900 iterations (green). Horseshoe line begins to correctly resolve upstream. Fin line starts to resolve.

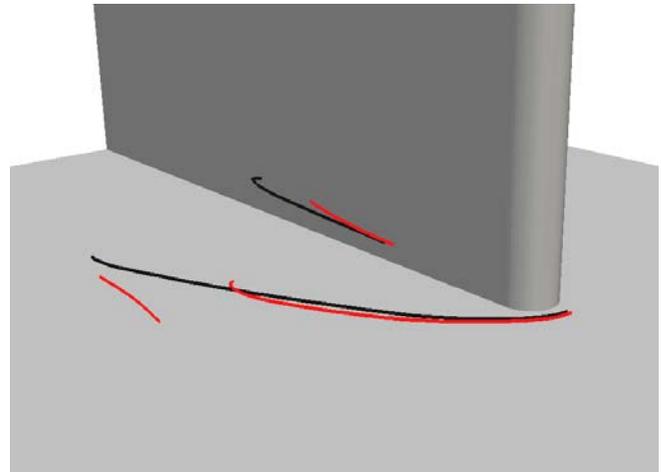


Figure 9: Comparison of Sujudi-Haimes extracted vortex core lines at 30% of converged solution - 270 iterations (red) and 900 iterations (black). Horseshoe line begins to correctly resolve upstream. Fin line also starts to resolve upstream.

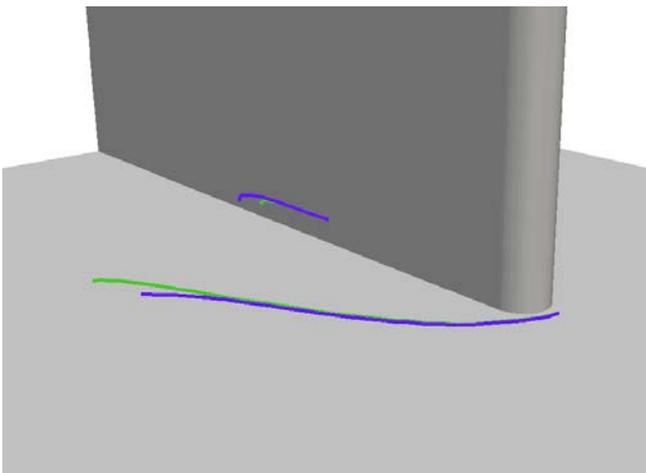


Figure 7: Comparison of Roth-Peikert extracted vortex core lines at 40% of converged solution - 360 iterations (blue) and 900 iterations (green). Horseshoe line and fin line almost correctly resolved.

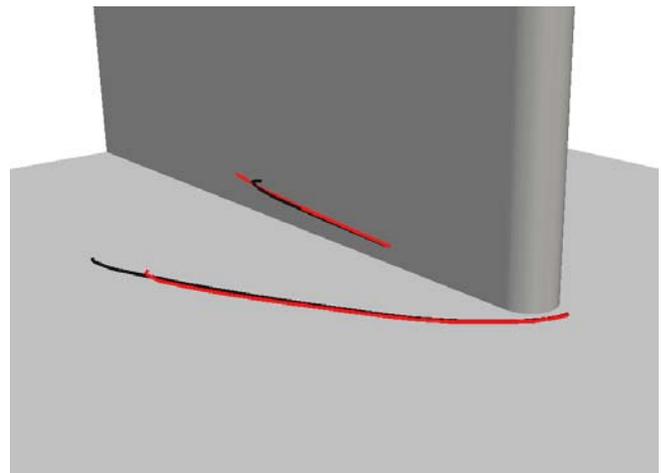


Figure 10: Blunt fin comparison of Sujudi-Haimes extracted vortex core lines at 40% of converged solution - 360 iterations (blue) and 900 iterations (white). Horseshoe line and fin line almost correctly resolved.

Fish or Shark – Data Mining Online Poker

Ulf Johansson and Cecilia Sönströd

Abstract— In this paper, data mining techniques are used to analyze data gathered from online poker. The study focuses on short-handed Texas Hold'em, and the data sets used contain thousands of human players, each having played more than 1000 hands. The study has two, complementary, goals. First, building predictive models capable of categorizing players into good and bad players, i.e., winners and losers. Second, producing clear and accurate descriptions of what constitutes the difference between winning and losing in poker. In the experimentation, neural network ensembles are shown to be very accurate when categorizing player profiles into winners and losers. Furthermore, decision trees and decision lists used to acquire concept descriptions are shown to be quite comprehensible, and still fairly accurate. Finally, an analysis of obtained concept descriptions discovered several rather unexpected rules, indicating that the suggested approach is potentially valuable for the poker domain.

Keywords: Concept description, Decision trees, Decision lists, Comprehensibility, Online poker, Texas Hold'Em

I. INTRODUCTION

In this application paper, we apply a bag of data mining tricks with the intention of studying datasets acquired from online poker. The overall goal is to analyze player profiles, trying to pin-point what separates successful, i.e. winning, players from players losing money. More specifically, the purpose is to combine explorative, descriptive and predictive techniques, in order to make it possible to not only predict if a certain player profile represents successful play, but also to obtain concept descriptions for winning and losing players. In addition, we compare strategies used at low-limit and middle-limit.

II. BACKGROUND

Texas Hold'em (often referred to as just "Hold'em") is the most popular poker game played online. The first subsection therefore presents the rules of the game. The second subsection discusses general basic strategy. The third subsection describes *short-handed play*, which is the focus of this study. Short-handed play is defined as six or fewer players at a table. The fourth subsection, finally, gives a brief description of a family of software products, used by online poker players for, among other things, profiling their opponents.

A. Texas Hold'em poker

When playing Hold'em, each player is dealt two private

cards face down. These cards are referred to as *hole cards*. Now the initial betting round takes place. After that, three public cards (the *flop*), are placed face up in the middle of the table. The second betting round follows. When the betting round has finished, another public card (the *turn*), is placed alongside the flop. Next is the third betting round. After that, the final, fifth, public card (the *river*) is turned up, followed by the final betting round. Each player still remaining in the pot now combines the public cards with her hole cards to obtain a five card poker hand. When doing so, a player may use one, both or none of her hole cards. Naturally, the player now (at the *showdown*) having the best poker hand wins the pot.

The betting structure varies between games, but the games studied in this paper are *fixed limit*; i.e., betting is restricted to bets and raises of predefined amounts. As an example, a fixed limit game at the level \$0.5-\$1, would have the following betting structure: Before the deal the player to the left of the dealer must place a forced bet of \$0.25 (the *small blind*) whereas the next player (the *big blind*) must make a similar bet of \$0.50. In the first betting round, the player to the left of the big blind is first to act. Her options are to either concede the hand (*fold*), pay an amount equal to the big blind to remain in the hand (*call*) or make a bet twice the size of the big blind (*raise*). After a raise, all players must, of course, match the size of the raised bet to remain in the hand, or raise the bet again to \$1.50 (*reraise*). Normally, only three raises and reraises are allowed, making the betting limit (called the *cap*) for the first round \$2.00. In the second betting round, the first remaining player to the left of the dealer acts first. Now she has the option to remain in the hand without making a bet (*check*), but as soon as one player makes a bet (\$0.50), all players must again either fold, call or raise. The third and fourth betting rounds are identical to the second, with the important difference that all bets are doubled; i.e. a bet or raise is now \$1.00, and the cap is \$4.00.

B. Basic strategy

As most beginners soon find out, Hold'em is a very sophisticated game, requiring mastery of many different skills. Some of these skills, most importantly the ability to read and interpret physical tells, are, however, of no value when playing online. On the other hand, the very short time available for each decision (compared to when playing "live") makes it extremely important to be able to quickly assess the situation and act accordingly. Furthermore, many players elect to play on multiple tables simultaneously, reducing the available time even more. From this, it is fair to say that most decisions online actually have to be made based on a very limited analysis of the specific situation. Or,

put in another way, players must rely on a *basic strategy* (almost an “auto-pilot”), which they normally will adhere to.

It must be noted that although the term basic strategy seems to suggest a simple mapping, from a situation to an action, this involves a number of subtleties. First of all, the situation is almost always both complex and only partially observable. In addition, in all betting rounds except the last, future, random events (new cards), will seriously change the situation. Finally, the actions taken by a player will of course also change the situation. Using terminology from agent theory (see e.g. [1]) the environment for poker is in fact *multi-agent, partially observable, strategic, sequential, static* and *discrete*. So, a basic strategy is actually a mapping from an *observed* situation to an action. Naturally, a perfect strategy would always maximize the expected profit, given the current situation, so a successful basic strategy must both be able to correctly identify the current situation, and to recommend the best action given this situation. Specifically, the strategy should, in contrast to game theory, not assume that all opponents play optimally. Based on this, some key abilities needed for estimating the expected values for the different actions, are listed below:

- The ability to guess an opponent’s hand, or more correctly, to narrow down an opponent’s range of hands.
- The ability to calculate current (pot) odds and to estimate future (implied) odds.
- The ability to calculate probabilities for making certain hands.
- The ability to predict the effects of a certain action.

For some players, their basic strategy is very deliberate, typically the result of seriously studying of poker literature, their own previous play and their opponents. For other players, their basic strategy is, at best, just a number of “rules of thumb”. Although there exist both intentional (e.g. when a player “mixes up” her play in order to be more deceptive) and unintentional (e.g. when a player is upset after suffering a *bad beat*) deviations from the basic strategy, the quality of the basic strategy will ultimately determine the player’s success.

Most poker literature, aimed at beginners and intermediate players, discuss basic strategy in terms of two dimensions; *loose-tight* and *passive-aggressive*; see e.g. [2][3]. The loose-tight dimension primarily captures how many hands a specific player decides to play, and how far she will take the hand. The passive-aggressive dimension describes how much a player, when playing a hand, bets and raises rather than checks and calls. The overall recommendation is to play tight and aggressive; i.e., to play very few hands, but when playing make a lot of bets and raises, as long as the expectation appears to be positive. The underlying assumption is that this way of playing will lead to a relatively high percentage of pots won when contending.

Based on this description, four player archetypes have emerged; the *calling station*, the *rock*, the *maniac* and the

solid player. A calling station is the weakest opponent and is loose-passive, i.e., plays a lot of (poor) hands and constantly takes them too far. In addition, she prefers to always call and rarely raises. A calling station will almost surely lose money quickly; as a matter of fact this player is most often referred to as a *fish*, i.e., easy prey for the *sharks*. A rock is a tight-passive player who plays mainly premium hands. The conservative hand selection makes it rather hard to win any significant money from a rock, but the fact that she is quite predictable and does not play aggressively enough, makes her only mildly successful, typically winning a few small pots to approximately break-even. The maniac is loose-aggressive, playing a lot of hands, just like the calling station, but the difference is that the maniac constantly bets, raises and reraises. A maniac will inevitably lose money in the long run, but if lucky during a session, may leave it as a big winner. In addition, the maniac forces other players to adapt to her style, thus introducing giant fluctuations in everybody’s bankroll. The solid player, finally, is tight-aggressive, mostly playing premium hands and a few speculative hands. The most basic trademark of a solid player is that she rarely calls, most often she either folds or raises.

Naturally, the level (i.e. how large the bets are) affects the quality of the poker played. Until recently, most small stakes Hold’em games were very loose-passive; the predominant player was often the calling station. One reason for this is that tight play requires a lot of discipline, it is much more fun to gamble by playing many more hands. Another reason is that most players playing at the small stakes tables were unaware of even the most basic principles for successful strategies. The last two years, however, online poker has changed fundamentally. Most importantly, the number of beginners has decreased significantly, while many players have improved their play, mainly from acquiring a lot of experience. This is especially true for fixed limit Hold’em, since nowadays most beginners will actually start playing No limit Hold’em right away. In addition, as the quality improves, poor players will start to lose consistently, and eventually either quit or try to somehow develop their play. Either way, the games, even at the lowest levels, now tend to be more competitive.

C. Short-handed play

In this paper, we study only short-handed play. Historically, Hold’em theory relates mainly to full tables; i.e., tables with nine or ten players. Even though the mechanics of the two games are identical, playing short-handed is profoundly different. First of all, since the blinds remain the same, a player now has to pay the same amount to play six hands instead of ten. Even more importantly, the probability that at least one opponent has a “good” hand is dramatically decreased. The position (i.e. if you act first or last during a betting round) is also even more important when playing short-handed. Specifically, a player will play a large proportion of all hands from the inherently hard blind positions. The overall (and extremely simplified) implication is that a player has to adapt to short-handed play, mainly by

being significantly more aggressive, but also by playing more hands.

The poker boom saw hundreds of new books, all trying to convince the reader that she could indeed be a consistent winner, just by following the advice given in the book. Since most beginners started by playing fixed limit, small stakes Hold'em at full ring tables, a large majority of these books focused on exactly that form of poker. Furthermore, the strategies suggested were clearly biased towards fairly safe play, i.e., tailored to make a small but certain profit at the loose-passive tables dominant at the time; see e.g. [4][5].

In books focusing on full ring play, the advice for short-handed play tends to be either very vague, such as "play more hands more aggressively", or outright incorrect like "you can regard short-handed play as a full table where the first four players have folded". Furthermore, the number of books focusing on short-handed play, especially addressing the improved level of play, is surprisingly small. One possible exception is [6] but here the presentation must be considered quite impeding. This is clearly a book written by poker professionals for an audience consisting of quite sophisticated players with high ambitions. With this in mind, the best source for learning the essential skills for short-handed play is instead regarded to be discussion forums on the Internet, where very good players share their thoughts on all forms of poker. Most famous are probably the very active forums at Two Plus Two Publishing [7] which include specific discussion groups for, for instance, short-handed limit play, on different levels.

D. Profiling software

Many players use software tools like Poker Office¹ Poker Tracker² or Hold'em manager³ to automatically collect information about both their own play and their opponents. These programs, which are legal to use at most but not all online poker sites, track a large number of attributes for each player and hand, including every action taken by the player, several attributes describing the situation and, of course, the outcome of the hand. Naturally, only information available to the player can be stored. Specifically, cards held by opponents remain unknown, unless they were involved in a showdown. With this in mind, some players use these tools mainly for analyzing their own play between sessions. Other players, however, use the tools primarily for assessing their opponents, both when choosing which tables to play, and during actual hands. Generally speaking, these programs produce profiles, consisting of several attributes, for each player. The exact attributes used, and how the information is aggregated and presented, differ slightly between the programs, but they are all able to describe a specific player using the two dimensions; *loose-tight* and *passive-aggressive*. Naturally, the exact overall profit achieved by a player is also stored, making it fairly straightforward to, based on this attribute alone, target weaker players while avoiding players who have been successful in the past.

¹ www.pokerooffice.com

² www.pokertracker.com

³ www.holdemmanager.net

III. METHOD

As mentioned in the introduction, the overall purpose of this study is to analyze data collected from online poker. More specifically, the first goal is to obtain accurate predictive models, making it possible to classify players as good or bad players (actually winners or losers) based on their profiles. The second goal is to develop concept descriptions, in order to be able to understand and analyze what separates winning and losing strategies.

A. Data preparation and explorative analysis

While collecting data, all six-handed tables with the limits \$0.50-\$1.00 and \$5-\$10 at a major online casino were monitored around the clock, for approximately three months. Hands played with less than four players were not added to the database. All data was imported into Hold'em manager, where a number of attributes were calculated for each player. All in all, a player profile, in this study, consists of 21 attributes describing the strategy used by the player, and one attribute measuring the player's success. In the following analysis and experimentation, only players having played more than 1000 hands were used. This resulted in two datasets consisting of 3262 (\$0.5-\$1.0) and 2555 (\$5-\$10) instances, i.e., players. In addition, two smaller datasets were generated, each consisting of players having played more than 10000 hands. These datasets have 270 (\$0.5-\$1.0) and 288 (\$5-\$10) instances. All attributes used in player profiles are described below.

- **Voluntarily put money in pot (VPIP):** The percentage of all hands that the player has added money to the pot, not counting the forced bets in the blinds. This attribute is a vital indicator for how tight-loose the player is.
- **Pre flop raise (PFR):** The percentage of all hands that the player raised preflop. This is a very important attribute indicating how aggressive the player is.
- **PFR/VPP:** This value, which aggregates the two previous attributes, shows how tight-aggressive the player is preflop.
- **3Bet PF:** Percentage of times a player reraises preflop, when facing a raise. Important for determining whether the player is passive or aggressive.
- **Won when saw flop (W\$WSF):** Percentage of hands where the player wins the pot when she sees the flop. This attribute also concerns the tight-loose dimension.
- **Aggression factor (AF):** An attribute trying to describe how aggressively a player plays. Since calls are considered passive plays, whereas bets and raises are aggressive, AF is defined as the number of all bets and raises divided by the number of all calls. As an example, a player having an AF value of 2.0, bets or raises twice as often as she calls.
- **Aggression percentage (AF%):** Percentage of betting rounds where a player bets or raises.
- **Flop continuation bet (CBetF):** Percentage of hands where the player is the preflop aggressor (i.e. put in the last raise) and then bets the flop. When playing short-

handed, a majority of all flops are taken heads-up (i.e. by two players only) so often both players fail to improve on the flop. With this in mind, it is important for a preflop raiser to often try to take down the pot with a bet on the flop even when missing it. A high CBet value, consequently, is an indicator of aggressive play.

- **Turn continuation bet (CBetT):** Percentage of hands where the player is the preflop aggressor and then bets both on the flop and on the turn. This attribute is related to both the tight-loose and the passive-aggressive dimension.
- **Check-raise (CR):** Percentage of time a player check-raises after checking the flop, turn or river. A check-raise is a very aggressive move, often associated with good players.
- **Cold call preflop (CC):** Percentage of times a player calls a raise (more than one bet) as her first action preflop. A high cold-call rate is associated with poor players, since poker theory says that you should almost always either fold or reraise against a raise.
- **Steal:** Percentage of times a player opens with a raise from late position when no previous player has entered the pot. Good players will often open with a raise in order to try to steal the blinds, even with fairly weak hands. This attribute, consequently, is an indicator of aggressive play.
- **Reraise preflop raise in blind (BBRR):** When playing short-handed, the play from the blinds is very important. Specifically, the blinds have to act first in all betting rounds after the flop, so calling a raise in the blinds without a good hand is a common but costly mistake. Naturally, good players in late position try to exploit this by steal-raising in order to win the pot immediately. Consequently, the big blind must defend his blind by sometimes reraising. This attribute, which is defined as the percentage of times a player reraises a preflop raise from the big blind, therefore relates mainly to the passive-aggressive dimension.
- **3bet:** Percentage of times a player reraises a raise. Reraising is an important tool to try to get the pot heads-up, typically by forcing players with drawing hands to fold. In addition, 3-bets is a way of building a large pot for a player who thinks she is ahead at the moment. A 3bet is, of course, an aggressive play.
- **4bet:** Percentage of times a player reraises a reraise.
- **PFR_B, PFR_UTG, VPIP_B, VPIP_UTG:** Position is extremely important in Hold'em, so a player must use her positional advantage to play more hands, more aggressively from late positions. These attributes represent how often a player enters and raises a pot when acting first (UTG) and last (Button).
- **Raise_pos, Play_pos:** Two aggregated attributes trying to capture the players' positional awareness. Raise_pos, which is defined as $(PFR_B - PFR_UTG) / PFR_UTG$, measures how much more often a player raises from the button, compared to when acting first. Similarly, Play_pos, defined as $(VPIP_B - VPIP_UTG) / VPIP_UTG$, indicates

how many more pots the player contests from the button, compared to when acting first. Naturally, the hypothesis is that a solid player should be more aware of how important the position is, and consequently have higher values for these two attributes.

- **Big bets won per 100 hands (BB/100):** This is the profit achieved by the player, measured as number of big bets won or lost, on average, over 100 hands. It must be noted that internet casinos take a fee (the rake) from all pots, so the winner does not get all the money put into the pot. In this study, player profits are actual profits; i.e., a player only wins the raked pot.

Table I below shows a summary of the \$0.5-\$1.0 dataset. The last column is the linear correlation between the specific attribute and the BB/100 attribute.

TABLE I
Summary of \$0.5-\$1.0 dataset (3262 players)

Attribute	Mean	Std. Dev.	Min	Max	Corr
VPIP	34.68	14.81	10.72	90.50	-0.60
PFR	13.47	6.40	0.00	49.15	0.31
PFR/VPP	47.20	26.63	0.00	91.38	0.50
3Bet PF	6.51	3.88	0.00	30.98	0.18
W\$WSF	39.75	3.60	26.59	54.40	0.47
CCPF	16.05	16.55	0.00	88.89	-0.61
AF	1.69	0.74	0.11	5.02	0.43
AF%	47.49	10.04	7.47	76.17	0.44
CR	8.55	4.71	0.00	35.69	0.30
CBetF	11.85	6.00	0.00	35.08	0.35
CBetT	9.43	5.01	0.00	31.88	0.30
Steal	24.36	12.52	0.00	62.68	0.43
BBRR	9.57	6.64	0.00	50.00	0.15
3Bet	8.45	4.17	0.00	41.28	0.14
4Bet	12.21	5.56	0.00	44.10	-0.05
Raise_Pos	38.23	86.78	-100.00	2698.08	0.13
PFR_B	16.41	8.41	0.00	53.08	0.37
PFR_UTG	12.69	6.87	0.00	68.28	0.20
Play_Pos	18.97	21.80	-77.38	75.09	0.36
VPIP_B	34.27	14.58	10.31	95.37	-0.59
VPIP_UTG	29.17	17.47	4.49	93.42	-0.61
BB/100	-3.02	6.25	-56.81	17.3	

Table I presents several interesting observations. First of all, it can be noted that all players in the dataset lose, on average, 3.02 BB per 100 hands. This is of course due to the rake and, if nothing else, should be reassuring for the Casinos. The overall picture is that a few attributes have a clear negative correlation with BB/100, while most other attributes are positively correlated with BB/100. Unfortunately, no specific attribute shows a very strong correlation with BB/100. On the other hand, only five attributes (3BetPF, BBRR, 3Bet, 4Bet and Raise_Pos) obtain correlations with BB/100 lower than 0.2.

Turning to individual attributes, it is very obvious that playing to loosely is the easiest way to lose money. VPIP, CCPF, VPIP_B and VPIP_UTG are all clearly negatively correlated with BB/100. Aggressive play, on the other hand, seems to be the key to successful play. AF, AF%, and Steal are all clearly positively correlated with BB/100. Looking at attributes focusing on positional awareness, the fairly high positive correlation between Play_pos and BB/100, indicates

that good players, at the very least should play significantly fewer hands from poor position. W\$WSF is a somewhat delicate attribute. It is no surprise that it is a good thing to win a high percentage of pots contested. Naturally, this attribute is more affected than the others by luck in individual hands. This is especially true for players having played relatively few hands. Table II below summarizes the \$5-\$10 dataset.

TABLE II
Summary of \$5-\$10 dataset (2555 players)

Attribute	Mean	Std. Dev.	Min	Max	Corr
VPIP	35.78	12.51	10.57	85.63	-0.55
PFR	16.69	6.36	0.06	50.43	0.18
PFR/VPP	51.82	22.33	0.15	96.60	0.45
3Bet PF	8.12	4.28	0.00	39.69	0.17
W\$WSF	41.90	3.44	28.38	54.55	0.48
CCPF	14.97	14.66	0.00	80.12	-0.57
AF	1.59	0.60	0.20	4.26	0.39
AF%	48.47	8.95	14.42	68.60	0.38
CR	12.38	5.68	0.00	33.74	0.36
CBetF	14.87	6.05	0.00	41.68	0.24
CBetT	11.48	5.05	0.00	40.74	0.16
Steal	30.89	11.88	0.00	80.80	0.30
BBRR	10.24	6.30	0.00	66.67	0.10
3Bet	9.72	4.45	0.35	35.92	0.15
4Bet	10.40	4.95	0.00	41.67	-0.02
Raise_Pos	41.07	131.60	-100.00	5311.11	0.10
PFR_B	20.75	8.35	0.00	58.98	0.27
PFR_UTG	16.58	8.19	0.00	73.60	0.02
Play_Pos	17.81	21.87	-59.57	72.77	0.34
VPIP_B	34.62	11.84	10.51	87.50	-0.53
VPIP_UTG	29.42	14.99	5.56	91.07	-0.56
BB/100	-2.92	5.70	-42.30	14.81	

Although the overall impressions are quite similar for this dataset, it is interesting to see that the mean values for attributes capturing aggressive play (e.g. PFR, PFR/VPIP, AF, AF%, CR, CBetF, CbetT and Steal) are all higher on this level. Despite this, most of these attributes are even now evidently positively correlated with BB/100; i.e., it is still beneficial to be more aggressive than the opponents.

In addition, it is obvious that, even on this level, a lot of players contest way too many pots, making all VPIP attributes clearly negatively correlated with BB/100. Finally, on this level, the loose-passive action CCPF is actually the most negatively correlated attribute.

B. Predictive modeling and concept description

When performing predictive classification, *accuracy*, i.e., the percentage correct predictions on novel data, is normally the prioritized criterion. Alternative metrics, especially for unbalanced datasets include *area under the ROC-curve* (AUC) and different information theoretic measures; e.g., the *F-measure*, which is the harmonic mean of precision and recall. While accuracy and the F-Measure are based only on the final classification, AUC measures the ability to rank instances according to how likely they are to belong to a certain class; see e.g. [8]. AUC can be interpreted as the probability of ranking a true positive instance ahead of a false positive; see [9].

Most high-accuracy techniques for predictive classification produce opaque models like artificial neural

networks (ANNs), ensembles or support vector machines. Opaque predictive models make it impossible to follow and understand the logic behind a prediction, which often must be considered a serious drawback. In practice, the usability of systems utilizing black-box prediction machines is often reduced, simply because decision makers have a hard time accepting recommendations without accompanying explanation of the underlying reasoning. Furthermore, opaque predictive models also make it impossible to inspect the model, looking for interesting patterns.

When models need to be interpretable (or ultimately comprehensible) accuracy is often sacrificed by using simpler but transparent models; most typically decision trees. This tradeoff between predictive performance and interpretability is normally called the *accuracy vs. comprehensibility tradeoff*. It must be noted, however, that decision trees often are so complex that their comprehensibility is limited. While it is certainly possible to trace the reasoning behind a specific prediction even in a very complex tree, understanding the overall relationship or discovering hidden patterns becomes quite prohibiting. With this in mind, it makes sense to distinguish between transparent and comprehensible models, and to measure comprehensibility (for tree models and rule sets) based on *model size*. Some typical choices are (for trees) *number of nodes* and (for rule sets) *number of rules* or *number of tests*.

The data mining task *concept description*, as defined by the CRISP-DM consortium [10] aims to generate understandable descriptions of concepts or classes. Consequently, the purpose is not to generate predictive models, but to gain insights. The results from a concept description project would most often be new information, typically presented as verbal descriptions or rules. Models produced by classification algorithms can, however, be regarded as concept descriptions, as long as they are comprehensible. In terms of standard performance measures, this corresponds to a transparent and relatively small model with good generalization ability, measured as high performance on unseen data, i.e., a comprehensible and accurate model. For the poker domain, a concept description would consist of a small model accurately capturing the differences between winning and losing strategies, or describing typical winners or losers.

The overall purpose of the predictive modeling was to find what strategies winning players in general apply, and similarly, to investigate the strategies of players losing money. With this in mind, we decided to use two pairs of binary classification experiments, at each level. In the first experiment, WIN, the task was to predict and explain what makes a player successful. The classes for this experiment thus became successful (*Winner*) and not successful (*No_Winner*). The second experiment (LOSE) was aimed at explaining what makes a player lose money. The two classes were *Loser* and *No_Loser*.

In both experiments, approximately one fourth of all instances were assigned to the targeted class; i.e., *Winner* for experiment 1 and *Loser* for experiment 2. As expected, the 25% players having the highest BB/100 were put in the

Winner class, while the 25% players having the worst BB/100 were put in the *Loser* class. Naturally, this setup makes the classes unbalanced, which may be a problem since many techniques tend to obtain pretty high accuracy, just by focusing on the majority class. With this in mind, we considered using cost-sensitive classifiers, but settled for using *oversampling*. More specifically, we applied the *SMOTE* [11] oversampling technique to produce artificial instances (based on five nearest neighbors), making the datasets balanced. Since there are two experiments and two levels, each with two differently sized datasets, we have altogether eight experiments; see Table III below.

TABLE III
EXPERIMENTS. (#INSTANCES INCLUDES ARTIFICIAL)

Experiment	Min #hands	Level	Win/Lose	#instances
1000:w	1000	\$0.5-\$1.0	Win	4888
1000:l	1000	\$0.5-\$1.0	Lose	4894
10000:w	10000	\$0.5-\$1.0	Win	403
10000:l	10000	\$0.5-\$1.0	Lose	401
1000:W	1000	\$5-\$10	Win	3803
1000:L	1000	\$5-\$10	Lose	3821
10000:W	10000	\$5-\$10	Win	431
10000:L	10000	\$5-\$10	Lose	429

All experimentation in this study was performed using the Weka data mining tool [12]. As mentioned above, we wanted to employ several different data mining techniques, but with different goals. First of all, we decided to use one technique producing opaque models. The intended use of such a black-box prediction machine is, of course, to categorize new instances (player strategies) into winners or losers. With this in mind, high predictive performance becomes the only important criterion for this technique, so we decided to use ANN ensembles. More specifically, we used the Weka meta technique *Bagging*, combining ten multilayer perceptron networks into an ensemble.

For the predictive modeling producing interpretable models, we evaluated altogether four techniques; *J48*, *Cart*, *JRip* and *Chipper*. *J48* is Weka's implementation of the tree inducer C4.5 [13]. The *Cart* implementation used is also from Weka, where it is called *SimpleCart*. This implementation follows the original *Cart* algorithm [14] closely. Both *Cart* and *J48*, produce decision trees. *JRip*, which is the Weka implementation of the RIPPER algorithm [15], on the other hand produces *ordered rule sets* or *decision lists*. For the standard algorithms, default settings were used in all experimentation.

Chipper is an algorithm built for concept description, introduced in [16] and since then implemented in Weka. *Chipper* produces decision lists by greedily formulating rules with high coverage, whilst maintaining acceptable accuracy. To allow the user to control the tradeoff between accuracy and comprehensibility, two parameters, called *ignore* and *stop*, are available. The *ignore* parameter sets the acceptable misclassification rate for each rule produced, given either as an absolute number of instances, or as a percentage of the remaining instances. The *stop* parameter controls how long the rule generation procedure should continue, by simply specifying the proportion of instances

that should be classified before formulating the default rule.

In this study, *Chipper* was used together with the Weka meta-procedure *CVParameterSelection*, which uses internal cross-validation to optimize settings for one or more parameters. Two different *Chipper* setups were evaluated, one favoring accuracy (*Chipper - A*), where *ignore* ranges between 0.5% and 2% and *stop* is between 96% and 99%, and another setting favoring concept description (*Chipper - C*), with *ignore* at 4% or 5% and *stop* between 75% and 95%.

For all experimentation, standard, 10-fold stratified, cross-validation was used. When measuring size, *total number of nodes* was used for *J48* and *Cart*, while *total number of tests* was used for *Jrip* and *Chipper*.

IV. RESULTS

Tables IV and V below show the results from the predictive modeling.

TABLE IV
RESULTS FOR ANN ENSEMBLE AND TREE MODELS

\$0.5-\$1	ANN			J48				Cart			
	Acc	Auc	F	Acc	Auc	F	Size	Acc	Auc	F	Size
1000:w	.758	.840	.750	.720	.749	.718	405	.719	.753	.718	85
1000:l	.858	.930	.860	.825	.836	.828	439	.825	.868	.824	75
10000:w	.811	.890	.800	.705	.720	.703	55	.710	.705	.710	51
10000:l	.815	.871	.815	.713	.751	.713	41	.678	.708	.677	55
\$5-\$10	ANN			J48				Cart			
	Acc	Auc	F	Acc	Auc	F	Size	Acc	Auc	F	Size
1000:W	.760	.840	.760	.705	.734	.704	419	.717	.739	.717	195
1000:L	.847	.920	.850	.809	.822	.809	291	.807	.853	.807	139
10000:W	.764	.830	.750	.708	.741	.700	57	.701	.712	.700	45
10000:L	.800	.890	.800	.718	.741	.717	41	.734	.744	.734	13
Mean:	.802	.876	.798	.738	.762	.737	218	.736	.760	.736	82

The results for the ANN ensemble clearly show that it is possible to obtain very accurate predictive models for the situation at hand. Over all eight datasets, the ANN ensemble obtained a mean accuracy just over 80%, and an AUC close to 0.9. The use of decision trees, as expected, resulted in significantly worse predictive performance. In addition, the induced trees are actually quite large, making them very hard to manually inspect or analyze. Comparing the different datasets, it is obvious that the easiest task is to find losers on the \$0.5-\$1.0 level (1000:l dataset). This was of course no surprise either since we would expect really poor players to play mainly low-limit.

TABLE V
RESULTS FOR RULE SET MODELS

\$0.5-\$1	JRip				Chipper - A				Chipper - C			
	Acc	Auc	F	Size	Acc	Auc	F	Size	Acc	Auc	F	Size
1000:w	.708	.728	.708	33	.705	.750	.690	47	.703	.740	.702	13
1000:l	.816	.842	.815	39	.808	.860	.810	36	.801	.855	.801	11
10000:w	.717	.737	.717	8	.767	.777	.766	33	.680	.726	.680	10
10000:l	.661	.696	.661	11	.711	.715	.710	31	.671	.685	.670	11
\$5-\$10	JRip				Chipper - A				Chipper - C			
	Acc	Auc	F	Size	Acc	Auc	F	Size	Acc	Auc	F	Size
1000:W	.693	.703	.693	67	.688	.732	.687	63	.686	.725	.685	16
1000:L	.806	.826	.806	18	.791	.842	.791	32	.795	.849	.795	13
10000:W	.664	.701	.664	8	.694	.719	.693	53	.640	.668	.639	13
10000:L	.709	.737	.708	6	.709	.723	.709	51	.713	.757	.713	11
Mean:	.722	.746	.722	24	.734	.765	.720	50	.711	.751	.711	12

When comparing the techniques producing rule sets, Chipper – A obtains the best predictive performance overall. As a matter of fact, both accuracy and AUC are comparable to J48 and Cart. Unfortunately, Chipper – A's decision lists tend to be quite lengthy, making their comprehensibility questionable. Both JRip and Chipper – C, on the other hand, generate mainly smaller models, even if there are a couple of exceptions for JRip. Naturally, the increased comprehensibility comes at a price of reduced accuracy, but in this study, the difference in predictive performance is actually quite small. Figures 1-3 below show some sample models from the different datasets.

```

IF CCPF >= 28.7 THEN Loser [1740/220]
IF W$WSF >= 41.1 THEN No_Loser [1201/141]
IF VPIP_UTG <= 16.18 THEN No_Loser [495/87]
IF CCPF >= 24.62 THEN Loser [254/65]
IF W$WSF >= 40.06 THEN No_Loser [205/54]
IF W$WSF >= 39.11 THEN No_Loser [173/44]
IF Steal <= 6.62 THEN Loser [120/37]
IF CBETT >= 12.78 THEN Loser [100/31]
IF W$WSF >= 38.42 THEN No_Loser [94/27]
IF 3Bet <= 4.3 THEN No_Loser [78/23]
DEFAULT: No_Loser [434/211]

```

Figure 1: Chipper – C rule set for 1000:l

In this rule set from \$0.5-\$1.0, the first rule singles out a large number of *Losers* based on their tendency to cold-call raises preflop. The fact that, on this level, cold-call turned out to be even more differentiating than, for instance, VPIP and AF, was somewhat unexpected, and could be a valuable finding for the poker domain. The second rule classifies quite a few players as *No_Losers* based on a high W\$WSF. Although this attribute is, as mentioned above, somewhat affected by luck in individual hands, it also shows the importance of only seeing the flop with hands that have a good chance of winning.

```

CCPF < 10.9
| W$WSF < 44.6
| | PFR < 19.2
| | | VPIP_B < 22.4: Loser(16/8)
| | | VPIP_B >= 22.4: No_Loser(59/24)
| | PFR >= 19.1
| | | AF < 2.3
| | | | PFR/VPP < 69.8: No_Loser(8/3)
| | | | PFR/VPP >= 69.8: Loser(46/13)
| | | AF >= 2.3: No_Loser(9/1)
| W$WSF >= 44.6: No_Loser(110/30)
CCPF >= 10.9: Loser(93/9)

```

Figure 2: Cart tree for 10000:L

This Cart tree is from the \$5-\$10 level and contains only players having played more than 10000 hands. Still, cold-call preflop is again used directly to accurately categorize a relatively large number of *Losers*. Here, however, cold-calling “only” 11% is found to be too much. W\$WSF is again very important for finding *No_Losers*, even when the number of hands played is as large as 10000. The rest of the tree tends to favor aggressive play, but the number of instances classified in each leaf is rather small and the variables used in the splits clearly interact, making the interpretation somewhat awkward.

```

(Steal >= 34.9)=> Winner(121/33)
(W$WSF >= 42.1) and (4Bet >= 10.4) and
(AF <= 2.5) and (PFR/VPP <= 78.7)=> Winner(49/7)
(PFR/VPP <= 72.9) and (VPIP_UTG <= 12.9) and
(VPIP >= 19.8)=> Winner(31/3)
=> No_Winner(202/43)

```

Figure 3: JRip rule set for 10000:w

This JRip rule set from \$0.5-\$1.0 starts by picking a large number of *Winners* based on a high Steal value. Again, this must be considered a surprising first rule, making it an interesting observation. The second and third rules are also somewhat unexpected, mainly because they seem to warn against a too aggressive play. The last conjunct in the last rule, finally, indicate that it is possible to play too tightly.

V. CONCLUSIONS

We have in this paper combined different data mining techniques to analyze data from online poker. The results show that it is possible to create very accurate black-box prediction machines (here ANN ensembles), clearly able to categorize players into *Winners* and *Losers*, based on their profiles. Examining decision trees and decision lists produced to obtain concept descriptions, it is obvious that these models are quite comprehensible, and still fairly accurate. An analysis of some concept descriptions also identified several rather unexpected findings. Most notably, cold-calling too many raises preflop turned out to be the most significant indicator for identifying *Losers*.

REFERENCES

- [1] S. Russel and P. Norvig, *Artificial Intelligence - A Modern Approach*, 2nd edition, Prentice Hall, 2003.
- [2] D. Sklansky and M. Malmuth, *Hold'em Poker for Advanced Players*, Two Plus Two Publishing, 1999.
- [3] E. Miller, *Getting Started in Hold'Em*, Two Plus Two Publishing, 2005.
- [4] E. Miller, D. Sklansky and M. Malmuth, *Small Stakes Hold'em – Winning big with expert play*, Two Plus Two Publishing, 2004.
- [5] L. Jones, *Winning Low-Limit Hold'em*, ConJelCo, 2000.
- [6] N. Grudzien and G. Herzog, *Winning in Tough Hold'em Games*, Two Plus Two Publishing, 2007.
- [7] <http://forumserver.twoplustwo.com/>
- [8] T. Fawcett, Using rule sets to maximize roc performance, *15th International Conference on Machine Learning*, pp. 445-453, 2001.
- [9] A. Bradley, The use of the area under the roc curve in the evaluation of machine learning algorithms, *Pattern Recognition*, 30(6):1145-1159, 1997.
- [10] The CRISP-DM Consortium, CRISP-DM 1.0, www.crisp-dm.org, 2000.
- [11] N. V. Chawla, K. W. Bowyer and W. P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, *Journal of Artificial Intelligence Research*, Vol. 16:321-357, 2002.
- [12] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 2005.
- [13] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
- [14] L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, 1984.
- [15] W. Cohen, Fast Effective Rule Induction, *Proceeding of 12th International Conference on Machine Learning*, pp. 115-123, 1995.
- [16] U. Johansson, C. Sönström, T. Löfström, and H. Boström, Chipper – A Novel Algorithm for Concept Description, *Scandinavian Conference on Artificial Intelligence*, pp. 133-140, 2008.

An Efficient Clustering Algorithm based on Sorting and Binary Splitting

Taewan Ryu

Dept. of Computer Science
California State University
Fullerton, California 92834

Jae Soo Yoo

Department of Information and
Communication Engineering
Chungbuk National University
Cheongju 361-763, Korea

Michael Allen Bickel

Rt 3 Box 1005
Dickinson, Texas 77539

Abstract—Clustering has always been an important data mining technique used in many different areas because of its wide range of applications in various disciplines. Accordingly, many different clustering algorithms have been proposed in the past. To successfully apply a clustering technique to real-world data mining problems, an efficient algorithm that requires minimum or no parameters is one of the key elements. Most existing algorithms, however, require various parameters such as the number of clusters and other criteria.

In this paper, we introduce an efficient clustering algorithm that does not require a similarity measure or any parameter but relies on sorting and binary splitting approach. The paper also discusses an implementation approach for the algorithm along with some application results. The algorithm will be further analyzed from the perspective of key desirable data mining features such as efficiency, scalability, insensitivity to the order of objects, ability to deal with noisy data, and minimal domain knowledge requirement or parameters.

Keywords: clustering algorithm, data mining

I. Introduction

Clustering is to group a set of objects in a data set into meaningful groups called clusters with similar features in the context of a particular problem [11, 13]. When a data set is successfully clustered, a set of objects grouped in the same cluster share similar characteristics but are dissimilar to objects in different clusters. Because of this nature of clustering results, clustering has been one of the popular data mining methods in many areas of different disciplines to understand the given data set or predict behavior or properties of objects in a data set based on group membership [8, 17].

The clustering methods can be divided into two basic types: partitioning and hierarchical methods [11]. Partitioning methods partition a set of objects into a specified number of disjoint k clusters with similar properties. Partitioning methods determine a partition of the

object set into k clusters that optimizes a criterion function. The criterion function is a function that evaluates the quality of clusters using a measure such as similarity or dissimilarity. The value of k may or may not be given. If the value k is not specified, the clustering methods will only depend on the given criterion function for clustering. Defining the function is highly dependent on problem parameters. Hierarchical methods construct a hierarchical group of the data. The hierarchical group is a sequence of nested partitions, which is naturally represented by the tree diagram (or dendrogram). The hierarchical representation of clusters enables the user to see how objects are being merged into clusters or split at successive levels of proximity. Both clustering methods have their appropriate domains of applications [11, 17]. For implementing partitioning and hierarchical methods, the following two approaches can be distinguished: agglomerative and divisive [13]. The agglomerative approach starts with each object as its own cluster and merges these elementary clusters into larger clusters. The divisive approach reverses the process by starting with all objects in one cluster and subdividing it into smaller clusters.

For effective and successful applications of clustering to real-world data mining problems, an efficient clustering algorithm is one of the most important components because most real-world data sets for data mining are voluminous. Some other desirable features for data mining methods include scalability, insensitivity to the order of objects in a given data set, ability to deal with noisy data, and minimal domain knowledge requirements [1, 12, 17].

This paper introduces and analyzes a partitioning and divisive clustering method called Fast Fuzzy Cluster (FFC), originally developed by one of this paper's authors, Michael Bickel [14] but was not fully studied and experimented for various applications. The algorithm utilizes sorting and binary splitting approach [16] but does not require any similarity measure and parameter such as the number of clusters, k for clustering.

The rest of the sections will describe the algorithm in detail, implementation approach, some application results, and analysis of the algorithm.

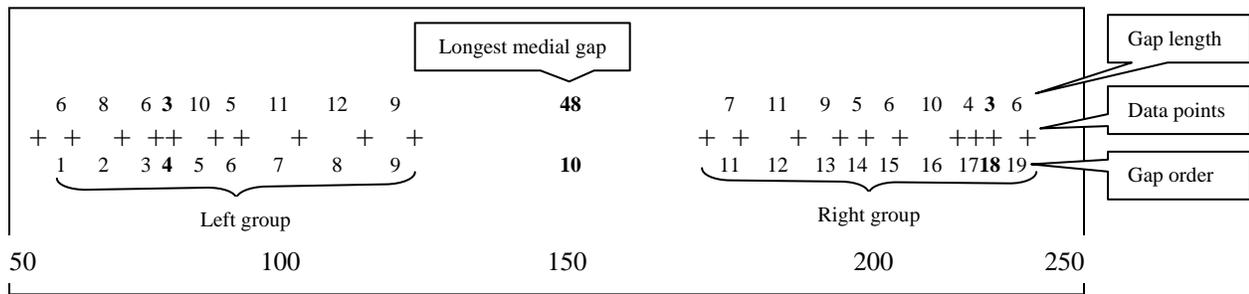


Fig. 1: A sample data set plotted on one-dimensional space

II. Algorithm

For a data set with obvious clusters, if the data set is plotted (or visualized), for example, on a two-dimensional space, a human can easily recognize clusters by the void gaps present among different clusters. FFC works similarly to the way that a human visually identifies clusters by gaps among clusters without relying on any similarity measure and parameter such as the number of clusters.

As a simple example to show how FFC algorithm works, let a data set with one feature be $D(x) = \{170, 52, 177, 58, 101, 231, 113, 188, 66, 197, 72, 208, 122, 218, 85, 222, 202, 75, 90, 225\}$. If we plot this data set where each data point is represented by a “+” sign as shown in Fig. 1, we can easily see clusters with a group of data points and gaps among them. The question is, however, how we can develop a computer algorithm to identify this obvious gap between the two groups or gaps among multiple groups in the case for multi-dimensional data sets. The interesting trick used in FFC is sorting because sorting can actually mimic the effect of data plotting and help identify the gaps among data points.

To give a sketch of the algorithm, FFC first sorts the data set by a selected dimension. To identify the gaps between clusters, FFC computes the gap length, g_{ij} , between each neighboring object pair, r_i and r_j , e.g., $g_{ij} = |r_i - r_j|$ as shown in Fig. 1. At this point, we can easily find the largest gap that divides the data set as clusters. However, simply selecting the largest gap may not necessarily give the gap that divides the data set into clusters. The algorithm may instead end up dividing every data point as a cluster, especially when the gap length for the largest gap is not big enough compared to other gaps. Therefore, the next important task is to distinguish the gaps in both sides of dense data points, which are candidate clusters, and the medial gap that is wide enough to separate data points into candidate clusters. For this purpose, the algorithm assigns an order to each gap, called gap order, o_i , and sorts the data set by gap length. Now, the dense data points in both candidate clusters will be located in the first certain x

percentage portion of the sorted data set, and the sparse data points for candidate medial gaps will be located in the last y percentage portion of the data set, where x and y are estimated percentages that represent an estimated data density in clusters and candidate medial gaps that separate both candidate clusters respectively. By default, FFC sets both x and y to 10, e.g., $x = y = 10\%$, which is a statistically significant value, but these percentage values can be set to a smaller or larger percentage depending on the data set and situation, for example, whether the data analyst wants a large number of clusters or a small number of clusters. The medial gap can be easily found by selecting the largest gap with the gap order, o_{med} , which is located in the last $y\%$ of the data set as shown in Table 1. Lastly, to determine whether the medial gap is located between two candidate clusters, we utilize the gap order as shown in Fig.1 by selecting minimum gap order, o_{min} and maximum gap order, o_{max} in the first $x\%$ of the data set that represent the two candidate clusters. The colored cells in Table 1 show this step. If the gap order of the largest gap, o_{med} is in between o_{min} and o_{max} , the gap is defined as the medial gap that meet the condition, $o_{min} < o_{med} < o_{max}$, and FFC divides the data set into two clusters based on the medial gap. The same process is repeated by taking the cluster as a new input data set until no more clusters are identified, e.g., when the division condition, $o_{min} < o_{med} < o_{max}$, o_{med} , is not met. Table 1 shows a data structure used for the algorithm for the sample data set, $D(x)$. The table stores the necessary information such as gap lengths and gap orders. By looking up data from this table, minimum gap, maximum gap, and the medial gap can be easily found and used for dividing the given data set.

Clustering a multi-dimensional data set can be done similarly by examining more than one dimension. For multi-dimensional data clustering, once all the dimensions (or features) or combination of several dimensions are examined, the algorithm identifies the largest gap from all the dimensions considered and splits the data set by the largest medial gap into clusters.

Table 1: A data structure that contains the key information for clustering. The colored cells on top and bottom represent the first $x\%$ and $y\%$.

$D(x)$	Sorted $D(x)$	Gap length	Gap order	Sorted gap	Gap order
170	52	6	1	3	18
52	58	8	2	3	4
177	66	6	3	4	17
58	72	3	4	5	6
101	75	10	5	5	14
...
222	208	10	16	11	7
202	218	4	17	11	12
75	222	3	18	12	8
90	225	6	19	48	10
225	231				

The clustering process halts when the medial gap is not found, which indicates that there are no more clusters found in the given data set. Otherwise, the process continues.

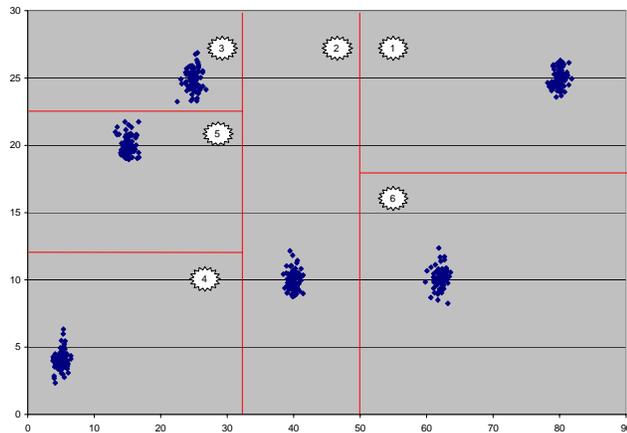


Fig. 2: A sample data set plotted on two-dimensional space

Fig. 2 shows an example data set plotted on two-dimensional space, visualizing the process of clustering. The red lines are the medial gaps used to divide the data set into clusters.

The following is the algorithm described in pseudo code. The only required parameter for the algorithm is the data set D . The percentage values of x and y that represents the density are set to 10%. The 10% is a selected heuristic value, but statistically important value to represent the

density of data points within a cluster and the density of medial gaps that divide the clusters. For clustering a multi-dimensional data set, only one feature can be selected with or without examining all features in the data set. Sorting can be performed only once for each feature by saving the intermediate data structure and reusing it.

Algorithm FFC(D):

```

{
  Input data set,  $D(a_1, a_2, \dots, a_n)$ 
   $x = y = 0.1$  //percentage of density
  largestGap = 0

  // $a_k$  is a feature that represents the dimension of  $D$ 
  // $n$  can be 1 based on the feature selected for clustering
  for  $k = 1$  to  $n$ 
  { //Gap finding
    Sort( $D$ )ak //sort the data set by a selected feature,  $a_k$ 
    GapLength( $D$ ) //compute the gap length  $g_{ij} = |r_i - r_j|$ ,  $i > j$ ,
                // $r_i$  and  $r_j$  are neighboring data points
    GapOrder( $D$ ) //assign a gap order to each gap,  $o_i$ 
    Sort( $D$ )g //sort the data set by gap length

    //to determine the medial gap
     $o_{max}, o_{min}, o_{med} = \text{GetMedialGap}(D)$ 

    If  $o_{min} < o_{med} < o_{max}$  then
    {
      largestGap =  $o_{med}$ 
      largestFeature =  $k$  //feature that has the largest gap so far
    }
  } // end of for

  If largestFeature > 0 then //if a medial gap is found
  {
    // $C1$  and  $C2$  are clusters
     $C1, C2 = \text{Split}(D, \text{largestGap}, \text{largestFeature})$ 
     $B = C1 \cup C2$ 
     $C = C \cup B$ 
    Cluster( $C1, \text{largestGap}, \text{largestFeature}$ )
    Cluster( $C2, \text{largestGap}, \text{largestFeature}$ )
  }
  return  $C$ 

  Cluster( $D, \text{largestGap}, \text{largestFeature}$ )
  {
    while  $B \neq \emptyset$  //repeat until no more clusters are found
    {
       $o_{max}, o_{min}, o_{med} = \text{GetMedialGap}(D)$ 
      if  $o_{min} < o_{med} < o_{max}$  Then
      {
         $C1, C2 = \text{Split}(D, o_{med}, \text{largestFeature})$ 
         $C = C \cup C1 \cup C2$ 
         $B = B \cup C1 \cup C2$ 
      }
    } else //no medial gap is found
       $B = B - D$ 
  } //end of while
} //end of Cluster()

} //end of algorithm

```

III. Implementation and Algorithm Analysis

The data structure shown in Table 1 that stores the necessary information for clustering is the main data structure used to implement the algorithm for clustering. In addition to this data structure, an efficient sorting algorithm such as the QuickSortis used in implementing this algorithm.

The theoretical time complexity of the algorithm is $O(n \log n)$ since it depends on the efficiency of the sorting algorithm. Most existing clustering algorithms take $\approx O(nkt)$ or $O(n^2)$, where k is number of clusters and t is number of iterations for clustering optimization [13].

Most clustering algorithms depend on similarity measures, the expected number of clusters, or some parameters estimation in advance [2, 9]. Accordingly, the major sources of computation for many existing clustering algorithms are the similarity measure computation and comparison among objects. On the other hand, FFC avoids the similarity measure computation and comparison in clustering a given data set. Instead, clustering is done by the biggest gap, called medial gap, between clusters and dividing the data set by the medial gap. To find the biggest gap and medial gap, sorting is used without requiring any additional information about the data beforehand, and the algorithm will stop when all the clusters are found.

Furthermore, for many real-world applications, clustering of large data sets should be finished in real-time and should not require many parameters, especially when a proper k is not available at the time of applications. The binary-splitting approach used in FFC makes the algorithm efficient because the entire data set is continuously scaled down into smaller data sets in each iteration. By splitting the data set and determining whether the data set must be re-examined or not, the algorithm avoids revisiting the same points and immediately stops the process when all clusters have been identified while most other less efficient algorithms continue to revisit points over and over again until the criteria of a given tuning parameter is met [2, 3, 9, 18]. Another aspect of efficiency is the method of assigning data sets to the split sets. This efficiency is realized by the implementation rather than by the algorithm itself. By assigning a number to a data set that has been split, it allows the algorithm to only work with those subsets and not the entire data set. The isolation of this data, only working with a smaller set of data points, reduces the number of iterations to a minimum.

The word "fuzzy" comes from the fact that some data points, such as outliers, will be isolated away from any cluster because they are deemed as not belonging to any cluster. This is based upon the assumption that gaps among data points in the center of a cluster will be small and data points with large gaps will lie between clusters or on cluster

boundaries. Since these points do not affect the clustering outcome, they can be handled specially by reassigning them to a "near" cluster or omitted from the clustering by treating them as outliers.

IV. Application Results

Several experiments were performed using the algorithm for different kinds of data sets to verify whether the algorithm could be applied to many different problems and for the purpose of performance benchmarking. Some of the problems selected include Automatic Speech Recognition (ASR) [5], Color Image Quantization (CIQ) [7], and unsupervised image segmentation for Spectral Analysis (SA) [7]. The algorithm was compared with other existing methods including Linde Buzo Gray (LBG) [20], Kohonen Artificial Neural Network (KANN) [19], K-Means (KM) [11], and Fuzzy K-means (FKM) [6].

ASR benchmarking result

ASR is the process of recognizing who is speaking on the basis of information contained in an individual's speech waves. In this experiment, we first sample speech patterns from different speakers and convert the wav files to ASCII values to create Codebooks with Codewords. Second, we use the same speakers to create testing samples, create Codewords again, and then match them with the Codewords created from the first step to identify the specific speaker or word. Two test results comparing with two algorithms, LBG and KANN for multiple-words to multiple words identification, and speaker and word identification are shown in Table 2 and Table 3, respectively. As shown in the results, FFC is fastest, and the accuracy of clustering is quite comparable to other algorithms.

Table 2: Multiple-words to multiple-words

	LBG	KANN		Average
Average VQ time	2.03	44.00		17.29
Average Test time	1.01	1.09		1.03
Average % Correct	77.08	70.31		74.13

Table 3: Speaker and word identification

	LBG	KANN		Average
Average VQ time	0.47	3.00		1.52
Average Test time	0.55	0.66		0.50
Average % Correct	85.70	86.70		90.80

CIQ benchmarking result

CIQ is a color reduction method for reducing colors from

an RGB image to an appropriate amount for display or printing device without losing image quality and integrity. In this experiment, we use PNG C library routines to sample two popular images in the area, Lena and Peppers shown in Image 1 and Image 2, respectively, with 16, 32, and 128 colors, save them into separate ASCII text files, perform the clustering, and finally use CIQ to regenerate the image base on the result of the clustering again. We then compare the performance with the quality of images as shown in Table 4.



Image 1: Lena



Image 2: Peppers

Table 4: CIQ for color reduction

	Quality	Color Depth		
		16	32	128
Lena	Best	FFC, LBG	LBG	LBG
	Worst	KM	KM	KM
	Smoothest	LBG, FKM	LBG	LBG, FFC
Peppers	Best	LBG	LBG	LBG
	Worst	KM	KM	KM
	Smoothest	FKM	LBG, FKM	FKM, FFC

As shown in table 4, the experimental results of CIQ using FFC are comparable to other methods like LBG and FKM.

SA benchmarking result

SA is a method used to reveal the underlying structure of the image data and segment the image into regions with similar spectral properties.

Table 5: SA for image segments

	Quality	Color Depth		
		4	8	12
Lena	Best	KM, FFC, LBG	FKM, KM	FKM, FFC
	Worst	FKM	FFC, LBG	KM, LBG
Peppers	Best	FKM, KM	FKM, FFC	FKM, FFC
	Worst	FFC, LBG	KM, LBG	KM

The results from SA experiments show that FFC is comparable to other methods. In terms of efficiency for CIQ and SA tests, FFC was relatively slower, especially for data sets with many clusters due to the fuzzy nature of the algorithm. In ASR, FFC proved to perform as well as the LBG algorithm and KANN. However, for single word speaker identification, FFC performance was not good compared to other methods, but when large training sets were used with large test sets, FFC performed as well as LBG, and better than KANN. However, performance of FFC was not good when small test sets were used with large training sets. In image processing, FFC was not quite as good as LBG in CIQ, but it outperformed LBG in SA on the average.

V. Summary and Conclusion

In this paper, we introduced an efficient partitioning and divisive clustering algorithm called FFC that clusters a data set based on sorting and binary splitting approach. Unlike many other existing clustering algorithms, FFC does not require any similarity measure or number of clusters as parameters. The algorithm clusters the data set by dividing it using the largest medial gap that is obtained by sorting the data set by the gap length. The algorithm was proven to be efficient from various application results as well as theoretical analysis. The efficiency of the algorithm is accounted mainly in three ways: first, the algorithm relying on the efficiency of sorting without requiring the comparison among objects; second, the fact that the data set is constantly being split; and third, the fact that the subsequent data sets are not re-visited. Additional efficiency was realized as a result of effective implementation. Several experiments for FFC were performed to see the applicability and performance of the

algorithm for different data mining problems such as ASR, CIQ, and SA. The application results show that FFC is relatively more efficient compared to most algorithms. In SA and ASR, the results of FFC were comparable to the well-known algorithm LBG.

However, like other algorithms, FFC shows both advantages and disadvantages. Some advantages of FFC include scalability, no requirement of domain knowledge to determine parameters, ability to deal with noisy data or outliers, and insensitivity to the order of input objects because it uses a sorting algorithm. Some disadvantages of FFC are: the algorithm is desirable for ordinal data clustering because it relies on sorting and clusters must be linearly separable. This problem can be somewhat relieved by rotating the data set in different angles.

Some future work for improvement includes investigation on how to resolve these disadvantages such as the ability to deal with non-ordinal data types and arbitrary cluster shape. In addition, more experiments are needed for evaluating the performance of fuzzy clustering and parallelization of the algorithm.

Acknowledgment

This work was partially supported by the Ministry of Education, Science and Technology Grant funded by the Korea Government (The Regional Research Universities Program/Chungbuk BIT Research-Oriented University Consortium).

References

- [1] R., Agrawal, T. Imielinski, and A. Swami, "Database mining: a performance perspective," *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, 1993.
- [2] T.W. Ryu and C.F. Eick, "A database clustering methodology and tool," *Information Sciences, an international journal*, vol. 171, pp. 29-59, 2005.
- [3] D. Wishart, "An Improved Multivariate Mode-Seeking Cluster Method," *General Applications Section and Multivariate Study Group Conference*, Royal Statistical Society, 1973.
- [4] Anderson, D., McNeil, G. 1992. "Artificial Neural Networks Technology." Rome Laboratory, Rome, NY
- [5] M. Do, "An Automatic Speaker Recognition System," Audio Visual Communications Laboratory, Swiss Federal Institute of Technology, Lausanne, Switzerland, 2000.
- [6] P. Dulyakarn and Y. Rangsanseri, "Fuzzy C-Means Clustering Using Spatial Information With Application To Remote Sensing," Faculty of Engineering King's Mongkut's Institute of Technology Ladkrabang, Bangkok, 2001.
- [7] P. Scheunders, "A Genetic C-means Clustering Algorithm Applied To Color Image Quantization," Vision Lab, Dept. of Physics, RUCA University of Antwerp, Belgium, 1997.
- [8] M.R. Anderberg, "*Cluster analysis for application*," New York, NY: Academic Press, 1973.
- [9] F.G. Ashby and N.A. Perrin, "Toward a unified theory of similarity and recognition," *Psychological review*, 95(1): 124-150, 1988.
- [10] P. Cheeseman, and J. Stutz, "Bayesian Classification (AutoClass): theory and results," *Advances in Knowledge Discovery and Data Mining*, U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, eds., Cambridge, MA: AAAI/MIT Press, pp. 153-180, 1996.
- [11] B.S. Everitt, "*Cluster Analysis*," Edward Arnold, Copublished by Halsted Press and imprint of John Wiley & Sons Inc., 3rd edition, 1993.
- [12] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "Knowledge Discovery and Data Mining: Towards a Unifying Framework," *In proc. the Second Knowledge Discovery and Data Mining conference*, Portland, Oregon, 1996.
- [13] A.K. Jain and R.C. Dubes, "*Algorithms for clustering data*," Englewood Cliffs, NJ: Prentice Hall, 1988.
- [14] M.A. Bickel, "Adaptive Fast Fuzzy Clustering System," U.S. Patent# 5,263,120.
- [15] R.A. Jarvis and E.A. Patrick, "Clustering using a similarity measure based on shared near neighbors," *IEEE Transactions on Computers*, C22, pp. 1025-1034, 1973.
- [16] G.N. Lance and W.T. Williams, "A general theory of classificatory sorting strategies: II. Clustering systems," *Computer Journal* vol. 10, pp. 271-277, 1967.
- [17] G. Piatetsky-Shapiro and W.J. Frawley, "Knowledge Discovery in Databases: An Overview," *Knowledge Discovery in Databases*, AAAI/The MIT press, pp. 1-27, 1991.
- [18] P. Smyth, "Clustering using Monte Carlo Cross-Validation," in Proc. *Second Knowledge Discovery and Data mining conference*, Portland, Oregon, 1996.
- [19] T. Kohonen, "Self-Organization and Associative Memory", Berlin: Springer-Verlag, 1984.
- [20] Y. Linde, A. Buzo, and R. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, vol. 28, pp. 84-94, 1980.

Perturbation Scheme for Online Incremental Learning of Features for Face Recognition

R. K. Agrawal and Ashish Chaudhary

School of Computer and System Science, Jawaharlal Nehru University, New Delhi-110067, India

Abstract - In this paper we have proposed a novel facial features extraction technique based on principal component analysis (PCA) for face recognition. The proposed scheme utilizes PCA in conjunction with perturbation theory. It is found on the basis of experimental results performed on four benchmark face image databases that our approach besides being in good agreement with the batch method and other incremental methods are also computationally more efficient.

Keywords: Statistical Pattern Recognition, Feature extraction, Face recognition, Principal Component Analysis, Perturbation technique.

1 Introduction

Feature extraction is one of the important phases while designing pattern recognition systems. It utilizes all the information of the given data to yield feature vector of the lower dimension and thereby eliminates redundant and irrelevant information. Principal Component Analysis (PCA) is one of the commonly used dimensionality reduction technique for extracting features in problems of pattern recognition [1]. The conventional implementation of PCA is carried out in batch mode which assumes that all observations for its computation are given in advance. PCA in batch method is computationally less efficient particularly when dealing with large-scale problems. In many real time applications such as time-series predictions and classification, we often come across difficult situations where a complete set of training sample is not available in advance. For example, in face recognition process human face undergoes facial variation due to different expressions (sad, happy, laughing face etc), lighting conditions, make up, and hairstyles etc. Hence, it is difficult to consider all facial variation when a human face is registered in a face recognition system, first time [2]. Hence, it is difficult to extract meaningful features only from previously available dataset and high performance in practical situations can hardly be expected with only a static data set. There is a need to develop adaptive face recognition system to take into account variation due to incoming new sample. One of the solutions for this situation is that we collect data whenever new data are presented and then construct a provisional system by batch learning [3]. However, such system will require large memory and high computational cost because the system would need to maintain a huge memory to store the data either previously learned, or newly presented, possibly without a limit.

Moreover, the system will not be able to utilize the knowledge acquired in the past, even if the learning of most of the data is finished, and will repeat the learning from the beginning whenever one additional sample is presented.

To address such situation several eigenspace model [4–6] have been proposed. Hall, Marshall and Martin [4] proposed Incremental PCA (IPCA) based on the updating of covariance matrix through a residue estimating procedure. Recently, Agrawal and Karmeshu [5] proposed perturbation scheme for online learning of features based on incremental principal component analysis. Memory size and complexity reduction provided by incremental PCA make them appropriate for different high dimensional online pattern recognition systems [7-8, 9-10]. It would be worth examining newer approaches which are computationally more efficient and easy to implement. The purpose of this paper is to propose and examine a new online feature extraction approach for face recognition, which is computationally more efficient in relation to the existing schemes when small incremental changes occur.

The paper is organized into five sections. The state-of-art of incremental PCA is given in section 2. The incremental perturbation scheme to compute the eigenvalues and eigenvectors up to first order from the variance-covariance matrix is described in section 3. In section 4, computational efficiency and performance issues of the proposed scheme for face recognition in relation to batch method and Ozawa et. al. method are examined. For this we have considered four publicly available face datasets [17-20]. Finally, we summarize this paper in section 5.

2 Incremental PCA

Often incremental changes in data occur in online pattern recognition applications. It is interesting to investigate the behavior of the system when a new sample is added to the system or the existing data undergoes a small incremental change. This results into small perturbation in variance-covariance matrix which requires updating the eigenvectors and eigenvalues. One of commonly used approaches [11-15] is based on construction of eigenspace model $\Omega = (\mu, U, \Lambda, N)$ by calculating the eigenvectors and eigenvalues of the variance-covariance matrix of $X_i \in R^n$ ($i=1,2,\dots,N$). Here, μ denotes the mean input vector, U represents $(n \times k)$ matrix

whose column vector corresponds to the eigenvector, and Λ denotes $(k \times k)$ matrix whose diagonal elements correspond to the eigenvalues and k gives the dimension of the reduced eigenspace. As a new sample \mathbf{X}_{N+1} is added to existing N samples with mean $\boldsymbol{\mu}$ and the variance-covariance matrix $\boldsymbol{\Sigma}_0$, then the new mean vector and the new variance-covariance matrix are respectively given by $\boldsymbol{\mu}^*$ and $\boldsymbol{\Sigma}^*$, i.e.

$$\boldsymbol{\mu}^* = \frac{1}{(N+1)} (N\boldsymbol{\mu} + \mathbf{X}_{N+1}) \quad (1)$$

$$\boldsymbol{\Sigma}^* = \frac{N}{(N+1)} \boldsymbol{\Sigma}_0 + \frac{N}{(N+1)^2} (\mathbf{X}_{N+1} - \boldsymbol{\mu})(\mathbf{X}_{N+1} - \boldsymbol{\mu})^T \quad (2)$$

It may be noted that the dimension of the new eigenspace may or may not change. Define a new residue vector \mathbf{h} [16] given by

$$\mathbf{h} = (\mathbf{X}_{N+1} - \boldsymbol{\mu}) - \mathbf{U}\mathbf{g} \quad (3)$$

where $\mathbf{g} = \mathbf{U}^T(\mathbf{X}_{N+1} - \boldsymbol{\mu})$.

Hall, Marshall and Martin [15] introduced the criterion for dimensional augmentation if the norm of the residue vector \mathbf{h} is larger than a threshold. The eigenvectors and eigenvalues should accordingly be updated based on the solution of the intermediate eigenvalue problem [15] given by

$$\left\{ \frac{N}{(N+1)} \begin{bmatrix} \Lambda & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} + \frac{N}{(N+1)^2} \begin{bmatrix} \mathbf{g}\mathbf{g}^T & \gamma\mathbf{g} \\ \gamma\mathbf{g}^T & \gamma^2 \end{bmatrix} \right\} \mathbf{R} = \mathbf{R}\boldsymbol{\Lambda}' \quad (4)$$

Where

$$\boldsymbol{\gamma} = \hat{\mathbf{h}}^T(\mathbf{X}_{N+1} - \boldsymbol{\mu})$$

$$\hat{\mathbf{h}} = \begin{cases} \mathbf{h}/|\mathbf{h}| & \text{if } |\mathbf{h}| > \eta \\ 0 & \text{otherwise} \end{cases}$$

Here, \mathbf{R} is $(k+1) \times (k+1)$ matrix whose column vectors correspond to the eigenvectors obtained from the above intermediate Eigen problem, $\boldsymbol{\Lambda}'$ is the new eigenvalue matrix, $\mathbf{0}$ is a k -dimensional zero vector. Using the solution \mathbf{R} , the new augmented $n \times (k+1)$ eigenvector matrix \mathbf{U}' is calculated as: $\mathbf{U}' = [\mathbf{U}, \hat{\mathbf{h}}] \mathbf{R}$.

Ozawa, Pang and Kasabov [2] have suggested that the criterion of the dimensional augmentation is not appropriate as a suitable threshold may be different depending on the

magnitude of input values. If the threshold is too small, one cannot get an efficient feature space with low dimensions. This results in degradation of performance and reduction in computational efficiency. Ozawa et. al. [2] have formulated an incremental update algorithm in terms of accumulation ratio

$$A(k) = \frac{N(N+1) \sum_{i=1}^k \lambda_i + N \|\mathbf{U}_k^T(\mathbf{X}_{N+1} - \boldsymbol{\mu})\|^2}{N(N+1) \sum_{i=1}^n \lambda_i + N \|\mathbf{X}_{N+1} - \boldsymbol{\mu}\|^2} \quad (5)$$

The advantage of their approach is that it requires no knowledge of the previous samples. However, the computational cost of evaluating the eigenvalues and eigenvectors in (4) is still high. It would be useful to devise an alternative approach, which employs perturbation theory to reduce computational costs and at the same time also increases its efficiency.

3 Incremental Perturbation Schème

Generally, in face databases, new datasets are different from the previous dataset such as images of same person may be different due to changes in face expressions. These small changes in data may be understood in terms of small perturbation, which allows us to use well-established perturbation theory [16]. Due to small changes in data, the variance-covariance matrix corresponding to the new data sets will not differ much from the one based on old data sets. The new variance-covariance matrix, $\boldsymbol{\Sigma}^*$ in terms of old variance-covariance matrix, $\boldsymbol{\Sigma}_0$ can be rewritten in terms of small parameter ε as

$$\boldsymbol{\Sigma}^* = \boldsymbol{\Sigma}_0 + \varepsilon \boldsymbol{\Sigma}_1, \quad \varepsilon = 1/(N+1) < 1 \quad (6)$$

$$\text{where } \boldsymbol{\Sigma}_1 = \left[\frac{N}{(N+1)} (\mathbf{X}_{N+1} - \boldsymbol{\mu})(\mathbf{X}_{N+1} - \boldsymbol{\mu})^T - \boldsymbol{\Sigma}_0 \right]$$

For small ε , the second term on right hand side of (6) can be regarded as perturbation term which is of $O(\varepsilon)$.

Assuming that the principal eigenvalues and eigenvectors of the new variance-covariance matrix do not differ much from $\boldsymbol{\Sigma}_0$ except for a few correction terms of different orders of ε , it may be appropriate to employ the powerful perturbation technique as developed in quantum mechanics [16]. According to the fundamental theorem from the theory of perturbation of Hermitian matrices [16], it follows that the eigenvalues and normalized eigenvectors of $\boldsymbol{\Sigma}^*$ can also be

expanded in power series in ε , converging to the respective eigenvalues and eigenvectors of Σ_0 in the limit $\varepsilon \rightarrow 0$. Let λ_i and ϕ_i denote the eigenvalue and normalised eigenvector of the new ($n \times n$) variance-covariance matrix Σ^* , i.e.

$$\Sigma^* \phi_i = \lambda_i \phi_i, \quad i = 1, 2, \dots, n \quad (7)$$

$$\phi_i^T \phi_k = 0, \quad i \neq k \quad (8)$$

In terms of small parameter ε , we can express λ_i and ϕ_i as a power series in ε

$$\lambda_i(\varepsilon) = \lambda_i^0 + \varepsilon \lambda_i^1 + \varepsilon^2 \lambda_i^2 + \dots \quad (9)$$

$$\phi_i(\varepsilon) = \phi_i^0 + \varepsilon \phi_i^1 + \varepsilon^2 \phi_i^2 + \dots \quad (10)$$

Where λ_i^0 and ϕ_i^0 correspond to i -th eigenvalue and normalized eigenvector of Σ_0 , while λ_i^j and ϕ_i^j ($j \geq 1$) are unknown coefficients that must be determined. This general result is valid regardless of the unperturbed eigenvalue λ_i^0 . However, in the case of the repeated eigenvalues, a particular basis of unperturbed eigenvectors must be selected for the power series expansions (9) and (10) to be valid.

In practice, the expansions (9) and (10) in terms of Σ_0 are truncated to yield tractable approximations. In particular, an n -th order approximation results from neglecting all terms of order ε^k with $k > n$. This type of approximation is justified on the basis that the perturbation series is convergent in some neighborhood of $\varepsilon = 0$. Hence, for sufficiently small ε , a low order approximation can be used to evaluate λ_i and ϕ_i with good accuracy. It is proposed to incrementally update the eigenvalues and eigenvectors of the covariance matrix Σ^* based on a first order approximation of (9) and (10). For low order approximations, perturbation methods require less computation time [12] as all the calculations are done with the same degree of accuracy without committing extra resources to achieve desired precision.

First order coefficients λ_i^1 and ϕ_i^1 in (9) and (10) can be given by [5],

$$\lambda_i^1 = (\phi_i^{(0)})^T \Sigma_1 \phi_i^{(0)} \quad (11)$$

$$\phi_i^{(1)} = \sum_{k=1}^r c_{ki} \phi_k^{(0)} \quad (12)$$

Where

$$c_{ki} = -(\phi_k^{(0)})^T \Sigma_1 \phi_i^{(0)} / (\lambda_k^0 - \lambda_i^0), \quad (13)$$

This enables complete determinations of $\phi_i^{(1)}$ in (12). The outline of algorithm based on IPS for face recognition is given below.

IPS Algorithm For Face Recognition

Given input a set of n Images $[X_1, X_2, \dots, X_n]$

1. Compute $\Sigma_0, \mu(n)$
2. Solve eigenvalue problem $\Sigma_0 \phi_i^{(n)} = \lambda_i^n \phi_i^{(n)}$, $i = 1, 2, \dots, n$.
3. For each tuple X_{n+1} do the following
 - a. Compute ε
 - b. Update mean vector $\mu^{(n+1)} = \frac{1}{(n+1)}(n\mu^{(n)} + X_{n+1})$
 - c. Compute $\Sigma_1 = \frac{1}{(n+1)}(X_{n+1} - \mu^{(n)})(X_{n+1} - \mu^{(n)})^T - \frac{n}{(n+1)}(\Sigma_0 - \mu^{(n)}\mu^{(n)T})$
 - d. Update eigenvalues and eigenvectors:

$$\lambda_i^{n+1} = \lambda_i^n + \varepsilon (\phi_i^{(n)})^T \Sigma_1 \phi_i^{(n)}$$

$$\phi_i^{(n+1)} = \phi_i^{(n)} + \varepsilon \sum_{k=1}^r c_{ki} \phi_k^{(n)}$$
 where $c_{ki} = -(\phi_k^{(n)})^T \Sigma_1 \phi_i^{(n)} / (\lambda_k^n - \lambda_i^n)$
 - e. Compute Updated Projection matrix W and Projected images Y_{n+1}

$$W = [\phi_1^{(n+1)}, \phi_2^{(n+1)}, \dots, \phi_r^{(n+1)}]$$

$$Y_{n+1} = W^T X_{n+1}$$

The above algorithm can be easily implemented.

4 Experimental Setup and Results

Experiments are carried out on four publicly available face databases: ORL [17], Yale [18], JAFFE [19] and Faces94

[20] to check the performance of the proposed perturbation scheme in relation to batch method and Ozawa et al methods. The performance is evaluated in terms of (1) classification accuracy (2) comparison of dominant eigenvalues and (3) inner product of eigenvectors. The number of eigenvector used in classification is determined on the basis of accumulation ratio. We used five different values of accumulation ratio viz 0.80, 0.85, 0.90, 0.95 and 0.99.

Table I Comparative results of classification accuracy

Dataset	AR(NV)	Perturbatio nmethod	Ozawa et al. method	Batch method
ORL	0.99(46)	0.74	0.74	0.82
ORL	0.95(38)	0.67	0.67	0.73
YALE	0.99(25)	0.82	0.82	0.82
YALE	0.95(22)	0.79	0.79	0.82
JAFFE	0.99(12)	0.98	0.98	0.98
JAFFE	0.95(11)	0.96	0.96	0.96
FACE94	0.99(16)	0.97	0.97	0.98
FACE94	0.95(15)	0.97	0.97	0.97

AR- accumulation ratio, NV- number of eigenvectors

The ORL database [17] consists of 40 different individuals with 10 images for each individual. The images from ORL database were cropped from 112×92 to 55×44 in our experiment. The Yale database [18] consists of 165 images which are made up of 16 different individuals with 11 images for each individual. The sizes of images were changed from 320×243 to 50×50 in our experiment. The JAFFE database [19] comprises 10 Japanese female images. Each person has seven facial expressions: "happy", "sad", "surprise", "angry", "disgust", "fearful", and "neutral." There

are three or four images for each facial expression of each female. The images from JAFFE database were cropped from 256×256 to 50×50 in our experiment. Faces94 database [20] has facial images of 153 individual (female-20, male-133). The speech is used to introduce facial expression variation (the subject sit at a fixed distance from the camera and are asked to speak while a sequence of image is taken). We choose 30 individual randomly from the database with 10 images for each individual for carrying out our experiment. The size of images were reduced from 180×200 to 50×55 .

For every experiment, first we constructed an initial feature space using 20% of the total samples in which at least one image from each class is ensured to be included. For carrying out incremental learning one sample is chosen randomly from the remaining training samples. We have used K-nearest neighbor classifier [21] for classification in our experiments. The "leave-one-out" strategy is adopted for testing and training.

The classification accuracy of perturbation scheme at final stage of learning as compared to batch method and Ozawa et al method for all the four databases is shown in Table 1 We observe that the accuracy of IPS is in full agreement with the corresponding results obtained in Ozawa et al method and is in good agreement with the corresponding results obtained in batch method for different threshold values of accumulation ratios.

The results for first 5 dominant eigenvalues and eigenvectors of different datasets with AR equal to 0.90 at final stage of learning are given in Table 2. The percentage relative error for different datasets shows that eigenvalues obtained by IPS are in good agreement with the corresponding eigenvalues obtained by batch method and Ozawa method. Table 2 also reveals the consistency of eigenvectors obtained by IPS with the corresponding eigenvectors obtained by batch and Ozawa et al methods as the inner product of the vectors is almost equal to 1. Similar results are also observed for other values of AR for all face datasets.

Table 2 Dominant eigenvalues and corresponding percentage relative error and inner product of eigenvectors for different face databases

Dataset	A.R	Perturbation Scheme	Ozawa et al. method	Batch method	Relative error % IPS	Relative error % Ozawa et al.	I^p	I^o
ORL	0.90	79538.3032	79538.3945	79533.7183	0.01	0.01	1.00	1.00
		52403.3668	52406.5309	52404.2010	0.00	0.00	1.01	1
		25798.9587	25807.8464	25807.6237	0.03	0.00	1.02	1
		23698.8667	23698.8666	23697.4644	0.01	0.01	1.00	1.00
		18686.3280	18686.3396	18685.2354	0.01	0.01	0.99	1.00
YALE	0.90	278661.4010	278663.8265	278647.6295	0.00	0.01	1.00	1.00
		201778.6547	201926.6577	201941.1530	0.08	0.01	0.99	1.00
		155125.0852	155125.0034	155115.9457	0.01	0.01	1.01	1.00
		100403.1069	100403.1198	100397.1802	0.01	0.01	1.00	1.00
		100308.8126	100328.6365	100321.0437	0.01	0.01	0.98	1.00
JAFFE	0.90	65246.3621	65256.0093	65254.4493	0.01	0.00	1.00	1.00
		43852.2510	43861.1954	43862.4335	0.02	0.00	0.96	1.00
		24481.4620	24489.3728	24489.4655	0.03	0.00	1.06	1.00
		21218.6602	21215.3639	21215.1393	0.02	0.00	1.05	1.00
		25954.2999	25947.0282	25955.7054	0.03	0.01	0.93	1.00
FACES94	0.90	136236.4237	136047.8890	136217.2319	0.14	0.01	0.96	1.00
		97881.0225	97893.0008	97886.4966	0.01	0.01	1.03	1.00
		57228.0777	57233.0391	57236.0906	0.01	0.01	1.00	1.00
		39890.0349	39754.4410	39864.6933	0.34	0.06	0.95	1.00
		25954.2999	25947.0282	25955.7054	0.03	0.01	1.03	1.00

I^p inner product of eigenvectors of IPS and batch method, I^o inner product of eigenvectors of IPS and Ozawa et al. method

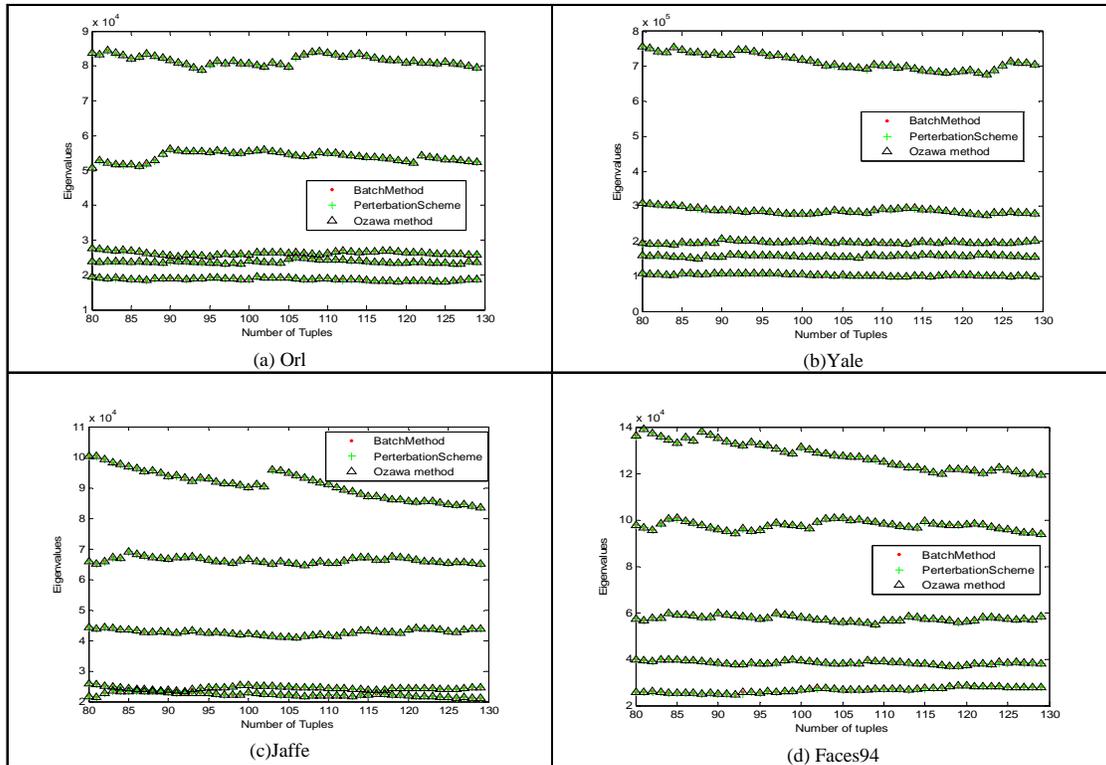


Fig.1. Variation of first five dominant eigenvalues with increment of tuple for accumulation ratio 0.9 for Face datasets (a) Orl, (b) Yale, (c) Jaffe and (d) Faces94

Fig. 1 shows variation of first five dominant eigenvalues obtained by batch method, perturbation scheme and Ozawa method for AR equal to 0.9 for all four face database. It can be observed from Fig. 1 that the eigenvalues are in good agreement with each other for all dominant eigenvalues when tuples is added one by one. Similar results are also observed for other values of AR for all face database.

In order to compare computational time required by batch method, IPS and Ozawa et al. method, experiment is performed on ORL face dataset. As incremental learning proceeds, new data is included in a random way to the initial eigenspace constructed on 20% of the total images. The comparison of average computational time of 10 runs to determine eigenvalues and eigenvectors with accumulation ratio=0.90 using IPS and batch method, and IPS and Ozawa et al. are shown in Fig. 2 and 3 respectively. Fig. 2 exhibits that computational time required by IPS is much less in comparison to batch method. Fig. 3 shows the gain in computational time of IPS scheme over Ozawa et al. method. While in case of IPS the computational time remains almost same at all stages of incremental learning, the computation time in case of Ozawa et al. method increases as the size of number of tuples increases.

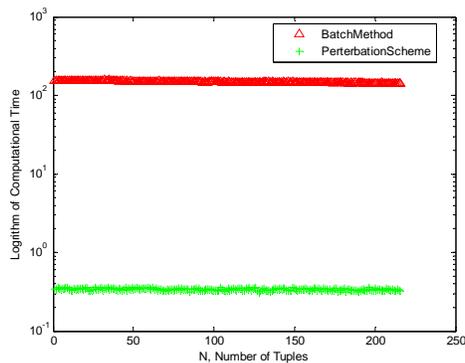


Fig. 2. Comparison of computation time (logarithmic scale) between batch method and perturbation scheme for OrL face dataset.

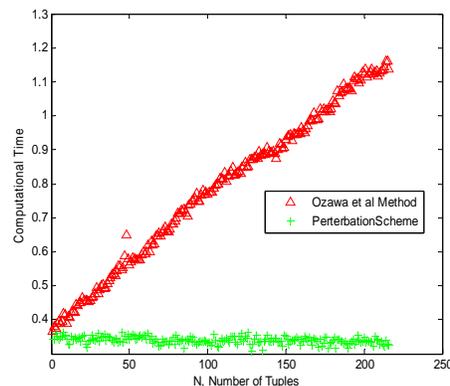


Fig. 3 Comparison of computation time between Ozawa et al. method and perturbation scheme for OrL face dataset.

5 Conclusion

This paper develops a feature extraction technique for face recognition based on incremental principal component analysis in conjunction with first order perturbation theory. In experimental results, it is shown that the proposed scheme IPS is computationally efficient over batch methods and other incremental method. This is due to the fact that the batch method and Ozawa et al. method involve intensive matrix operations unlike perturbative scheme. The computational gain of IPS becomes more obvious when we compare the time complexity of different methods. The time complexity of Ozawa et al. method is $O(n^3)$ as it requires solving an eigenvalue problem of a matrix of size $(n \times n)$ whereas the time complexity of IPS which involves multiplication of matrices is $O(n^2)$. The performance is evaluated in terms of (a) comparison of eigenvalues, (b) inner products of eigenvectors and (c) classification accuracy. For different face image datasets having high dimensional features, we note that the eigenvalues and eigenvectors obtained using the perturbative schemes are in excellent agreement with the results based on the direct method when the perturbation is small. The classification accuracy due to IPS, batch method and Ozawa et al. method are in complete agreement with each other. These experiments are a pointer toward the efficacy of the proposed scheme over others in the sense that besides providing excellent agreement in terms of performance measurements, they are computationally efficient and easy to implement.

6 References

- [1] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997
- [2] S. Ozawa, S. Pang, N. Kasabov, A modified incremental principal component analysis for online learning of feature space and classifier, in C. Zhang, H.W. Guesgen, W.K. Yeap (Eds.), *PRICAI : Trends in Artificial Intelligence LNAI*, Springer-Verlag (2004) 231-240.
- [3] S. Pang, S. Ozawa, and N. Kasabov, "Incremental Linear Discriminant Analysis for Classification of Data Streams" *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS*, VOL. 35, NO. 5, OCTOBER 2005
- [4] P. M. Hall, A. D. Marshall, R. R. Martin, "Incremental eigenanalysis for classification," in *Proc. Brit. Machine Vision Conf.*, vol. 1, pp. 286–295.

- [5] R. K Agrawal and Karmeshu, "Perturbation scheme for online learning of features ,Incremental Principal Component Analysis". Pattern Recognition, vol 41, no. 5, pp. 1452-1460, (2008).
- [6] S. Chandrasekaran, B. S. Manjunath, Y. F. Wang, J. Winkler, and H. Zhang, "An eigenspace update algorithm for image analysis," Graphical Models Image Process., vol. 59, no. 5, pp. 321–332, Sep. 1997.
- [7] B. Moghaddam, A. Pentland, Probabilistic visual learning for objec representation, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 696–710.
- [8] H. Murakami, B.V.K.V. Kumar, Efficient calculation of primary images from a set of images, IEEE Trans. Pattern Anal. Mach. Intell. 4 (5) (1982) 511– 515.
- [9] S. Chandrasekaran, B. Manjunath, Y. Wang, J. Winkler, H. Zhang, An Eigenspace update algorithm for image analysis, Graphical Models and Image Process. 59 (5) (1997) 321–332.
- [10] S. Chaudhuri, S. Sharma, S. Chatterjee, Recursive estimation of motion parameters, Comput. Vision Image Understanding 64 (3) (1986) 434–442.
- [11] Bunch, C. Nielsen, Updating the singular value decomposition, Numerische Math 31 (2) (1978) 131–152.
- [12] S. Chandrasekaran, B. Manjunath, Y. Wang, J. Winkler, H. Zhang, An Eigenspace update algorithm for image analysis, Graphical Models and Image Process. 59 (5) (1997) 321–332
- [13] R.D. DeGroat, R. Roberts, Efficient numerically stabilized rank-one eigenstructure updating, IEEE Trans. Acoust. Speech and Signal Process. 38 (2) (1990) 301–316.
- [14] P. Gill, G. Golub, W. Murray, M. Saunders, Methods for modifying matrix factorizations, Math. Comput. 28 (26) (1974) 505–535.
- [15] P.M. Hall, A.D. Marshall, R.R. Martin, Merging and splitting eigenspace models, PAMI 22 (9) (2000) 1042–1048.
- [16] A.H. Nayfeh, Introduction to Perturbation Techniques, Wiley Interscience, NY, 1981.
- [17] http://www.cl.cam.ac.uk/Research/DTGattarchive/pub/d/ata/att_faces.tar.Z/
- [18] <http://cvc.yale.edu/projects/yalefaces/yalefaces.html>
- [19] http://www.kasrl.org/jaffe_download.html
- [20] <http://cswww.essex.ac.uk/mv/allfaces/faces94.html>
- [21] Richard O. Duda ,Peter E.Hart , David G. Stork , Pattern Classification, 2nd Edition John Willey & Sons, INC

A Linear Integer Programming Approach to Objective Aware Feature Selection

Guangzhi Qu, Hui Wu, and Tao Xia

Computer Science and Engineering Department, Oakland University, Rochester, MI, USA

Email: {gqu, hwu, txia}@oakland.edu

Abstract—*Feature selection has been widely used and has played an important role in data preprocessing for most of data mining tasks, especially when dealing data with very high dimensionality. The resulting feature set directly affects the performance of the mining tasks, e.g., classification and clustering. In this work, we utilize the previously proposed metric (Decision Dependent Correlation : DDC) to quantify the information redundancy between features with respect to a specific learning task. Then we deploy a game theory based approach to formulate the feature selection problem into a linear integer programming problem. To evaluate the proposed approach, several UCI benchmark data sets are adopted in the experiments. Our feature selection method generates different feature sets from the other existing approaches (e.g., Weka) on the same data sets. Based on the selected feature set, we conduct classification tasks using C4.5 decision tree. Experimental results indicate that our method accomplishes the desired feature selection and classification performance simultaneously.*

Keywords: Feature Selection, Linear Integer Programming, Decision Dependent Correlation

1. Introduction

Data mining is a process of discovering previously unknown and meaningful knowledge from large data sets of complex systems, such as the Internet, social networks, protein networks, sensor networks, medical systems, high energy physics experiments and the stock market, where massive amounts of data is generated daily through observing real systems, scientific experimentation and simulation. For each collected data record, it may consist of a large number (tens of thousands) of attributes values. The data sets will be used extensively by collaborative researchers, besides the data creators for different scientific purposes. Hence, it is critical to design concepts and methods to provide useful and meaningful data for different objectives. In order to learn target concepts from these instances, learning algorithms are used to induce models that represent the knowledge space from the data. The learning processes are expected to be as accurate as possible, while maintaining simplicity and novelty. Thus, an important problem in data mining, usually called feature selection, is how to analyze the importance and relevance of the features. With careful feature selection,

both the computational time of inducing subsequent models and the quality of the those models will be improved.

Feature selection involves a process for determining which features are relevant and important, in that they explain the data and conversely, which features are redundant or provide little information [7]. Feature selection makes it easier to train learning models. Moreover, the resulting model may be simpler and easy to interpret the new observations in the knowledge space. Simple models often generalize better when applied to prediction and data visualization for human cognition. In this way, it allows scientists, researchers, educators to percept previously indiscernible abstract concepts, patterns, and important exceptions amidst vast data. By identifying an optimal set of features that are relevant to the decision making tasks, the subsequent models may provide valuable parametric, geometric, and topological structural information of the data.

We claim that the feature selection has a strong relationship with the learning task, i.e., for different decisions the feature selection procedure should identify different but appropriate features even on the same data set. In this work, we apply information theory based metrics to quantify the importance of a feature and the information redundancy between features with respect to the target learning task.

The rest of this paper is organized as follows: Section 2 details related work and approaches to feature selection. Section 3 describes the metrics used to evaluate the feature set quality and feature correlation, Section 4 formulates the feature selection problem into a linear integer programming problem, Section 5 evaluates the proposed approach on benchmark data sets. In Section 6, we discuss and conclude the paper.

2. Literature review

The literature on feature selection algorithms is extensive [5], [9], [8]. Basically, feature selection methods can be classified into two categories based on how the quality of features is evaluated: filters and wrappers [19], [6], [21]. The filters model is not dedicated to a specific type of data mining task in that the ranking of all features is produced before the learning algorithm is applied. The ranking could be done on weighting individual features [18], [4], [19] or searching a subset of features [7], [15]. Although feature

weighting algorithms usually are scalable to data sets with both a huge number of instances and a very high dimensionality, they cannot capture the redundancy among features. On feature subset level, exhaustive, heuristics, and random searching strategies are combined with different evaluation measure to form different algorithms. The time complexity is exponential in terms of data dimensionality for exhaustive search and quadratic or higher for heuristic search [7], [14], [15]. Hence, the existing subset search algorithms do not have good scalability to deal with high dimensional data. On the contrary, the wrappers model relies on the performance of the downstream learning to evaluate the quality of the selected set of features. Wrappers usually provide better accuracy than filters but are computationally more expensive in a sheer large dimensionality data environment [25].

When we think from a computational perspective, the feature selection problem is in fact a combinatorial optimization problem and generally difficult to solve. If we have n features, then the number of possible non-empty feature subsets is $2^n - 1$. Practically, heuristic searching strategies including exhaustive, best first search [19], hill climbing, A*, genetic algorithm [28], evolutionary search [17], simulated annealing, and greedy search elimination have been applied to help find the final feature set. However, these heuristic approaches scarify optimality for efficiency.

3. Correlation Evaluation Metrics

3.1 Motivation

The quality of the feature and relationship among features have been quantified using one or more metrics in many existing works. In the standard filters model, features are ranked according to their individual predictive power, which can be estimated by various means such as Fisher score [18], Kolmogorov-Smirnov test, Pearson correlation [23] or mutual information [1], [26]. However, selection based on such a ranking does not ensure weak dependency among features, and can lead to information redundancy. In order to remove the redundant information, consistency measure [7], correlation measure [14], [15], conditional decision trees, conditional mutual information maximization criterion [11], and predominant correlation [30] have been proposed and shown the effectiveness in removing irrelevant and redundant features in some scenarios.

But when there exists more than one learning objectives, the above metrics can not effectively quantify the redundancy among features with respect to the individual objective. For example, given the same large data set from Internet traffic monitoring, network traffic engineering team will focus more on the traffic spatial information together with traffic volume related information to guarantee the quality of service of the network operations. The security professionals, on the other hand, will pay more attention to the causality relationships among multiple transactions and

anomalous traffics to pinpoint the potential security breach. Previous work in [24] shows that for different learning objectives, the contribution of a feature or a feature set differs.

3.2 Objective Aware Metrics

Consider a vector variable $X = (X_1, \dots, X_i, \dots, X_n)$, where X_i is the i^{th} feature and n is the total number of features. Let X_i and X_j be two features. When there is no decision being considered with the features, the correlation between them is decision independent. *Decision Independent Correlation* (DIC) [24], denoted as $D(X_i, X_j)$, is defined as the ratio between the mutual information ($I(\cdot; \cdot)$) of these two features and the entropy ($H(\cdot)$) of the individual feature. For example,

$$D(X_i, X_j) = \frac{I(X_i; X_j)}{H(X_i)} \quad (1)$$

If $D(X_i, X_j) = 0$, it means the mutual information between these two features is equal to 0 and features X_i and X_j are uncorrelated. Scenario $D(X_i, X_j) = 1$ implies fully prediction between the two features.

When there is a decision Y being considered with the features, the correlation between the two features is decision dependent. We defined, in our previous work [24], *Decision Dependent Correlation* (DDC), to quantify the information redundancy between features as follows:

$$Q_Y(X_i, X_j) = \frac{I(Y; X_i) + I(Y; X_j) - I(Y; X_i, X_j)}{H(Y)} \quad (2)$$

The novelty of this objective aware metric is that it explicitly defines the redundancy between features given the target objective. We use the Venn diagrams in Figure 1 to illustrate the novelty of the new metric compared with the traditional metrics. In Figure 1(a), the correlation between X_i and X_j could be quantified as the mutual information shown as the shade area (a). When we need to determine the quality of feature with respect to the learning objective (e.g., classification), we can use the normalized mutual information as a good measurement reference. In Figure 1(b), Y and Z are different data mining tasks or knowledge discovering objectives, X_i , X_j and X_k are example features. By using Shannon's theoretic mutual information, we will have $I(Y; X_i) = a + b$ and $I(Y; X_j) = b + c$. We need also notice that the mutual information between the decision Y and features X_i and X_j is $I(Y; X_i, X_j) = a + b + c$ which is obviously less than $I(Y; X_i) + I(Y; X_j) (= a + 2b + c)$. Consequently, choosing both features X_i and X_j may include some redundancy ($b/H(Y)$ in amount) which could be quantified by the new objective aware correlation measure $Q_Y(X_i, X_j)$. In fact, suppose h is greater than c , compare with feature X_j , feature X_k will be a good candidate if we already chose feature X_i because $I(Y; X_i, X_k) = a + b + h > I(Y; X_i, X_j) = a + b + c$. Moreover, when we consider

another decision variable Z as shown in Figure 1(b), features X_i and X_j are not correlated. This fact cannot be captured by using the normalized the mutual information between two features, which is independent of the decision variable.

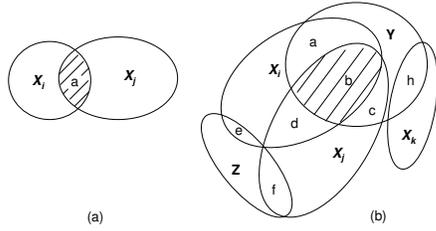


Fig. 1: Illustration of the feature correlations with respect to multiple objectives. (a) The objective independent correlation between two features X_i and X_j . (b) objective aware correlation for two features X_i and X_j with respect to two objective variables Y and Z .

4. Problem Formulation

Let $(\vec{x}_1, y_1), \dots, (\vec{x}_M, y_M) \in X \times Y$ be a set of training data records, where $\vec{x} \in X$ represents a vector-valued observation (measurement), $Y = \{y_1, \dots, y_C\}$ is a set of class labels. Let there be a total of C classes and a maximum of M data records in the training system. Let there also be a maximum of N available features composing a data record \vec{x} . We use X_i to denote the i^{th} feature, $i \in \{1, 2, 3, \dots, N\}$. So now the feature selection problem is a process to determine the 'best' r out of N features in order to learn equivalent amount of knowledge from the original high dimensionality space. In this work, the goal of feature selection is to identify the minimum set of features that are strongly related to the desired decision variable and has the least redundant information among them. In the following, we will show how to formulate the feature selection problem into a linear integer programming problem based on decision dependent correlation metric $Q_Y(X_i, X_j)$.

Define a complete graph $G = (V, E)$, where V is the set of vertex using to denote different features and E is the set of edges connecting feature pair. Let index set $I_N = \{1, 2, 3, \dots, N\}$. Given a set of N features $\{x_1, x_2, \dots, x_i, \dots\}$, $i \in I_N$, we want to choose a minimum set of the features with the goal of gaining maximum information for the final decision. We apply a game theory based model in formulating the question. This award-penalty strategy includes two components: the importance of the feature contributes award and the redundant information between features brings penalty. More specifically, if the i^{th} feature is selected, then a reward of G_i will be granted.

The reward of information gain, denoted as $G_Y(X_i)$, is the normalized the mutual information between the final decision (Y) and the individual feature (X_i) and is defined as follows:

$$G_Y(X_i) = \frac{I(Y; X_i)}{H(Y)} \tag{3}$$

Choosing two features X_i, X_j , the redundancy will cause penalty which is equal to the decision dependent correlation $Q_Y(X_i, X_j)$ (Q_{ij} for short) in quantity. Let z_i be a binary variable. Its value is 1 if the i^{th} feature is selected, otherwise it is 0. Then the feature selection problem (P^0) can be formulated as follows:

$$\begin{aligned} \min \quad & \sum_{i \in I_m} z_i \tag{P^0} \\ \text{s.t.} \quad & \sum_{i \in I_m} G_i \times z_i - \sum_{i \in I_m, j \in I_m} Q_{i,j} \times z_i \times z_j - \delta_2 \geq 0 \\ & z_i \in I_m, \forall i \in I_m \end{aligned}$$

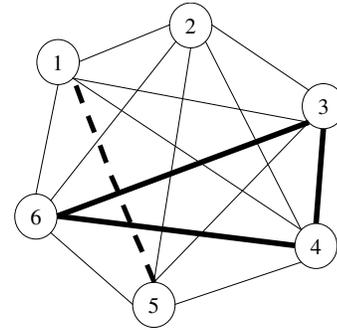


Fig. 2: Illustration of the feature selection algorithm

In Figure 2, we show a toy example of a six-vertex graph. If features 1 and 5 are selected, we have gained $G_1 + G_5 - Q_{15}$. If features 3, 4, and 6 are chosen, we gain $G_3 + G_4 + G_6 - Q_{34} - Q_{36} - Q_{46}$.

In the problem P^0 , the objective is to minimize the number of features. The constraints express the relevant information acquired by choosing the features and the redundancy introduced. We want to solve this problem to achieve a constant gain δ_2 , which is specified according to the user requirement in balancing the computational complexity and the feature set optimality. The above problem is a nonlinear integer programming problem due to the term of $z_i \times z_j$. The linearization of the original problem can be done through introducing new slack variables w_{ij} . Let w_{ij} denote the nonlinear term $z_i \times z_j$. The new linear integer programming formulation (P^1) can be used to help solve the original

problem.

$$\begin{aligned}
\min \quad & \sum_{i \in I_m} z_i && (P^1) \\
\text{s.t.} \quad & \sum_{i \in I_m} G_i \times z_i - \sum_{i \in I_m, j \in I_m} Q_{i,j} \times w_{ij} - \delta_2 \geq 0 \\
& z_i - w_{ij} \geq 0 \\
& z_j - w_{ij} \geq 0 \\
& -z_i - z_j + w_{ij} + 1 \geq 0 \\
& w_{ij} \geq 0 \\
& z_i \in \{0, 1\}, \forall i \in I_m
\end{aligned}$$

Table 1: Datasets from UCI repository

Dataset	Features #	Instances #	Classes #
Connect_4	42	67557	3
Covtype	54	581012	7
Shuttle	9	43500	7

Table 2: Data set Per-Class Distribution

Dataset	Class no.	Instances #	Percentage
Connect_4	1	44473	65.83%
	2	16635	24.62%
	3	6449	9.55%
Covtype	1	211840	36.46%
	2	283301	48.76%
	3	35754	6.15%
	4	2747	0.47%
	5	9493	1.63%
	6	17367	2.99%
	7	20510	3.53%
Shuttle	1	34108	78.41%
	2	37	0.09%
	3	132	0.30%
	4	6748	15.51%
	5	2458	5.65%
	6	6	0.01%
	7	11	0.03%

5. Evaluation and Results

The proposed feature selection approach was implemented on a PC with a 2.6Ghz Intel Core 2 Duo processor, 2GB RAM, Ubuntu operating system and the C/MySQL development environment. In our experiments on a variety data sets from UCI repository [3], we use proposed method and the peer implemented from Weka [12], [27] to select features separately. We then compare the performance of classification based on the feature sets from different methods (i.e., our approach and Weka).

5.1 Datasets

The following datasets, taken from the UCI machine learning repository, were used to evaluate the performance of the proposed feature selection approach: *Connect_4*, *Covtype*, and *Shuttle* [3]. Tables 1 and 2 summarize the characteristics of these datasets including the number of

features, the number of records, the number of classes in each data set, and per-class data distribution. In choosing these data sets as the candidates for experiments, we took into consideration the number of features (from 9 to 54), the number of records (from 43500 to 581012), and the number of classes (from 3 to 7) that these data sets can be representatives for a generic family of scientific data.

5.2 Evaluation Metrics

In evaluating the merit of the proposed feature selection approach, we compare the classification performance based on the feature set chosen by our approach with the other feature selection scheme (e.g., Weka). For a fair comparison, we used the C4.5 provided from Weka as the classification tool with the same parameters for all the testing. Traditionally, the terms true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) are used to compare the given classification of a record (the class label assigned to the item by a classifier) with the desired correct classification (the class the item actually belongs to). Since the data sets have multiple classes, the deduced metrics precision and recall for i^{th} class classification are defined as follows.

$$P_i = \frac{TP_i}{TP_i + FP_i} \quad (4)$$

$$R_i = \frac{TP_i}{TP_i + FN_i} \quad (5)$$

F-measure that combines Precision and Recall is the weighted harmonic mean of precision and recall. It is defined as follows.

$$F_i = \frac{2 \times P_i \times R_i}{P_i + R_i} \quad (6)$$

The classification performance (precision and recall) on the whole data sets is evaluated by using *macro_averaging* mechanism that gives equal weight to each class. Hence the *macro_averaged* metrics can reflect the classification performance even for the unbalanced data sets. The corresponding definitions are as follows.

$$P_{macro} = \frac{1}{|C|} \times \sum_{i=0}^{|C|} \frac{TP_i}{TP_i + FP_i} \quad (7)$$

$$R_{macro} = \frac{1}{|C|} \times \sum_{i=0}^{|C|} \frac{TP_i}{TP_i + FN_i} \quad (8)$$

$$F_{macro} = \frac{2 \times P_{macro} \times R_{macro}}{P_{macro} + R_{macro}} \quad (9)$$

Table 3: Feature selection and classification precision (%) on UCI datasets - connect-4

	Class	Feature set	Precision(%)	Recall(%)	F-Value(%)
Weka	c_0	{1,15,19,20,21,37}	55.7010	43.9852	43.0642
	c_1	{1,15,19,21,37}	73.2454	88.8899	80.3129
	c_2	{1,15,19,20,21,37}	64.1971	29.8467	40.7485
	c_3	{4,8,15,19,20,21,22,27,32,38}	00.0000	00.0000	00.0000
Our Approach	c_0	{1,2,7,8,14,15,19,20,21,22,26,27,32,37,38}	61.9247	53.0518	53.6494
	c_1	{1,7,14,15,19,21,37}	75.4001	88.8786	81.5864
	c_2	{1,15,19,20,21,37}	64.1971	29.8467	40.7485
	c_3	{21}	00.0000	00.0000	00.0000

Table 4: Feature selection and classification precision on UCI datasets - covtype

	Class	Feature set	Precision(%)	Recall(%)	F-Value(%)
Weka	c_0	{1,9,14,16,18,26,27,31,35,36,37,40,48,49,50,51,52,53,54}	67.5296	47.8386	51.4273
	c_1	{1,14,18,24,25,26,35,36,37,42,48}	65.8298	72.0667	68.8072
	c_2	{1,14,21,26,27,35,36,40,42,48,49,51,52,53}	72.3099	75.4233	73.8338
	c_3	{1,14,16,17,18}	68.8491	76.9676	72.6824
	c_4	{14,17,28,30,31}	54.9811	15.8719	24.6328
	c_5	{1,27,31,44}	55.3731	3.90814	7.30099
	c_6	{14,16,24,28,29,30,31}	82.9167	1.14585	2.26046
	c_7	{1,13,34,49,50,51,52,53,54}	76.4238	52.0137	61.8991
Our Approach	c_0	{1,4,6,10}	80.3509	73.6553	76.5828
	c_1	{1,4,6,10}	87.7259	85.6935	86.6978
	c_2	{1,10,11,36}	74.8353	78.4759	76.6124
	c_3	{1,4,5,8}	72.2328	66.6023	69.3034
	c_4	{1,4,5,7}	72.8407	57.4081	64.2101
	c_5	{1,6,8,9,10,44}	81.9484	60.8764	69.858
	c_6	{1,7}	54.6589	8.07278	14.0678
	c_7	{1,4,6,10,11}	91.5947	85.2755	88.3222

5.3 Results

Results in Tables 3, 4, and 5 show that our methods outperform peer feature selection schemes (e.g., Weka) in terms of the size of selected feature set and classification performance with respect to each class of the testing datasets and the whole data sets themselves. In presenting the feature selection and classification results, we use $c_i, i \in \{0, 1, \dots, K\}$ to denote the i^{th} class, where K is the total number of classes in the data set. c_0 means the whole data set. In following, we compare our approach with Weka implementation in the aspects of *Feature set size*, *Feature set quality*, and *Objective oriented feature set*.

- *Feature set size*: the number of features selected by our proposed method is less than what is chosen by Weka for the same classification performance. For example, as shown in Table 4 when compared on the data set *covtype* with a total number of 54 features, our proposed method chose 4 features

of {1, 4, 6, 10}. The classification performance using C4.5 shows *precision*=80.3509%, *recall*=73.6553%, and *F*=76.5828%. Comparatively, Weka chose 19 features with *precision*=67.5296%, *recall*=47.8386%, and *F*=51.4273%.

- *Feature set quality*: the quality of the feature set can be quantified as the classification performance with the almost the same size of the feature set. It shows from the experimental results that our method will help choose better quality feature set. As shown in Table 5, for the data set *Shuttle*, for the whole data set, Weka will choose features {1, 7, 9}, however, the performance of the C4.5 classification based on these features shows *precision*=69.5476%, *recall*=64.8498%, and *F*=66.5053%. When use our proposed method, features {2, 5, 9} were chosen. Based our feature set, C4.5 classification results (*precision*=95.5268%, *recall*=87.5481%, and *F*=90.9435%) improve a lot from Weka based classification performance results.

Table 5: Feature selection and classification precision (%) on UCI datasets - shuttle

	Feature Set	Precision	Recall	F-Value
Weka	$c_0: \{1,7,9\}$	69.5476	64.8498	66.5053
	$c_1: \{1,9\}$	99.7543	99.9971	99.8755
	$c_2: \{2,6,8\}$	90.2439	100	94.8718
	$c_3: \{2,9\}$	94.964	100	97.417
	$c_4: \{1\}$	76.3224	99.2146	86.2758
	$c_5: \{1,7\}$	99.9187	100	99.9593
	$c_6: \{2\}$	0	0	0
	$c_7: \{2,4\}$	71.4286	90.9091	80
Our Approach	$c_0: \{2,5,9\}$	95.5268	87.5481	90.9435
	$c_1: \{2,3,5,8,9\}$	99.9736	99.9736	99.9736
	$c_2: \{3,5,7,9\}$	94.4444	91.8919	93.1507
	$c_3: \{2\}$	80	100	88.8889
	$c_4: \{3,7,8,9\}$	99.852	99.9852	99.9185
	$c_5: \{7\}$	99.9187	100	99.9593
	$c_6: \{2,8,9\}$	100	100	100
	$c_7: \{2\}$	73.3333	100	84.6154

- *Objective oriented feature set*: the experimental results also demonstrate that for different classes (decisions), our method will choose different feature sets, which maintain the maximized relevancy to the decision while eliminating the redundancy among the features with respect to that decision. For example, in selection features for classes c_2 and c_6 of data set *Shuttle*, our approach will choose feature sets $\{3, 5, 7, 9\}$ and $\{2, 8, 9\}$, respectively. The Weka implementation will choose $\{2, 6, 8\}$ and $\{2\}$. The following classification (C4.5) on class c_6 performance shows that based feature set $\{c\}$, the classification precision, recall, and F-score are all 0 as shown in Table 5. Based on our method, the corresponding results are all 100%.

6. Conclusion

This study proposed a linear integer programming approach to solving the feature selection problem. In this work, we utilize the previously proposed objective aware metric to quantify the feature's relevancy and redundancy. Hence, the resulting feature set considers both relevancy and dependency among the features with respect to the given data mining tasks. Our analysis shows that the correlation relationship among features depends on the decision task and thus they display different behaviors as you change the decision task. To evaluate the proposed approach, several UCI benchmark data sets are adopted in the experiments. Our feature selection method generates different feature sets from the other existing approaches (e.g., Weka) on the same data sets. Based on the selected feature set, we conduct classification tasks using C4.5 decision tree. Experimental results indicate that our method accomplishes the desired feature selection and classification performance simultaneously.

In the future, we plan to test other public datasets and real world problems to verify and extend our approach.

References

- [1] Battiti, R. *Using mutual information for selection features in supervised neural net learning*, IEEE Transactions on Neural Networks, Vol. 5, 1994
- [2] Bradley PS, Fayyad UM, Mangasarian OL. *Mathematical programming for data mining: formulations and challenges*, INFORMS Journal on Computing 1999;11(3):217-38.
- [3] Blake, C.L., Merz, C.J., 1998. UCI repository of machine learning databases.
- [4] Blum, A. and Langley, P. *Selection of relevant features and examples in machine learning*, Artificial Intelligence, 1997, 245-271.
- [5] C.M. Bishop, *Neural Networks for Pattern Recognition*, New York: Oxford Univ. Press, 1995.
- [6] Das, D. Filters, *Wrappers and a boosting-based hybrid for feature selection*, Proceedings of the 8th International Conference on Machine Learning, 2001, pp. 74-81.
- [7] Dash, M., Liu, H., Motoda, H. *Consistency based feature selection*, Proceedings of the 4th Pacific Asia Conference on Knowledge Discovery and Data Mining, 2000, pp. 98-109. Springer-Verlag.
- [8] M.Dash and H.Liu, *Feature Selection for Classification*, Intelligent data Analysis, vol. 1, pp. 131-156, 1997.
- [9] R.O.Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, second Ed. New York: John Wiley and Sons, Inc., 2001.
- [10] Dy J. and Brodley, C.E. *Feature selection for unsupervised learning*, JMLR., 2004: 5, 845-889.
- [11] FLEURET, F. *Fast binary feature selection with conditional mutual information*, JMLR., 2004: 5, 1531-1555.
- [12] Stephen R. Garner, *WEKA: The Waikato Environment for Knowledge Analysis*, in Proc. of the New Zealand Computer Science Research Students Conference, 1995.
- [13] Guyon, I. and Elisseeff, A. *An introduction to variable and feature selection*, JMLR., 2003: 3, 1157-1182.
- [14] Hall, M. *Correlation based feature selection for machine learning*, Doctoral dissertation, University of Waikato, Dept. of Computer Science, 1999.
- [15] Hall, M. *Correlation-based feature selection for discrete and numeric class machine learning*, Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 359-366.
- [16] He X., Cai, D., Niyogi, P. *Laplacian score for feature selection*, In INPS, 2005, Mit Press.
- [17] Kim YS, Street WN, Menczer F. *Feature selection in unsupervised learning via evolutionary search*, In: Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining (KDD-00); 2000. p. 365-9.
- [18] Kira, K. and Rendell, L. *The feature selection problem: Traditional methods and a new algorithm*, Proceedings of the 10th National Conference on Artificial Intelligence, 1992, pp. 129-134). Menlo Park: AAAI Press/ The MIT Press.
- [19] Kohavi, R. and John, G. *Wrappers for feature subset selection*, Artificial Intelligence, 1997: 97(1-2) : 273-324.
- [20] Liu H. and Yu, L. *Toward integrating feature selection algorithms for classification and clustering*, IEEE TKDE, 2005: 17, 491-502.
- [21] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer Academic, Boston (1998).
- [22] Mitra, P., Murthy, C.A., Pal, S. K. *Unsupervised feature selection using feature similarity*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, Vol. 24, No. 3, 301-312.
- [23] Miyahara, K. and Pazzani, M.J. *Collaborative filtering with the simple Bayesian classifier*, Proceedings pacific Rim International Conference on Artificial Intelligence, 2000, pp. 679-689.
- [24] Qu G., Hariri, S., and Yousif, M. *A new dependency and correlation analysis for features*, IEEE TKDE, 2005: 17(9), 1199-1207.
- [25] Raman B, Joerger TR. *Instance based filter for feature selection*, Journal of Machine Learning Research, 2002;1-23.
- [26] Torkkola, K. *Feature extraction by non-parametric mutual information maximization*, JMLR., 2003: 3, 1415-1438.
- [27] Ian H. Witten and Eibe Frank (2005) *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

- [28] Yang J, Honavar V. *Feature subset selection using a genetic algorithm*, In: Motoda H, Liu H, editors. *Feature selection, construction, and subset selection: a data mining perspective*. New York: Kluwer; 1998.
- [29] Yoon, H., Yang, K., and Shahabi, C. *Feature subset selection and feature ranking for multivariate time series*, IEEE TKDE, 2005:17(9), 1186-1198.
- [30] Yu, L. and Liu, H. *Feature selection for high-dimensional data: a fast correlation-based filter solution*, In Proceedings of the 12th international conference on machine learning, Washington DC, 2003.
- [31] Zhao, Z. and Liu, H. *Spectral feature selection for supervised and unsupervised learning*, In Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, 2007.

Data Mining in the Real World: Experiences, Challenges, and Recommendations

Gary M. Weiss

Department of Computer and Information Science, Fordham University, Bronx, NY, USA

Abstract - *Data mining is used regularly in a variety of industries and is continuing to gain in both popularity and acceptance. However, applying data mining methods to complex real-world tasks is far from straightforward and many pitfalls face data mining practitioners. However, most research in the field tends to focus on the algorithmic issues that arise in data mining and ignores the human element and process issues that are often the cause of these pitfalls. While there are some papers on data mining experiences and lessons learned, they are quite rare, especially in the research community. The purpose of this paper is to begin to fill in the “gap” between data mining methods and the practice of data mining. This paper describes many of the experiences of the author as a data mining practitioner, highlights the issues that he encountered while in industry, and provides a number of strategies and recommendations for dealing with these issues. This paper should benefit both practitioners and researchers but hopefully, more than anything, it will foster discussion on the practical data mining issues that are all too often ignored.*

Keywords: data mining applications, data mining lessons

1 Introduction

Data mining is fundamentally an applied field, driven more by a class of problems (e.g., classification, clustering, etc.) than by a specific set of methods. Nonetheless, most published work in the field focuses almost exclusively on data mining methods and algorithms. Although an increasing amount of work is beginning to focus on the characteristics of real-world problems that makes data mining difficult (class imbalance, unequal misclassification costs, etc.) there is still relatively little work that describes the many practical issues that arise when addressing real data mining problems. This paper takes one small step toward remedying this deficiency by discussing issues that the author has encountered during his years as a data mining practitioner. The hope is that a discussion of these issues will be useful to both data mining practitioners and researchers. In addition to raising a variety of issues, the paper also provides a number of strategies and recommendations. However, it should be noted that this paper focuses on the author’s personal experiences, which certainly may differ from the experiences of other practitioners. Furthermore, this paper also tends to focus more on organizational and process issues than on technical issues.

As just mentioned, this paper is not meant to be a comprehensive accounting of all of the issues that may arise when tackling difficult, real-world, data mining problems. Rather, it focuses on the author’s experiences as a data mining practitioner. Because of this, the author’s work history is relevant and is briefly described here. Prior to moving into academia the author worked for over eighteen years at AT&T Bell Labs and AT&T Labs, of which the last nine years were spent either doing data mining or in activities related to data mining. In one position the author designed and implemented an expert system to maintain central off switching equipment [15] and was able to supplement the knowledge acquired from domain experts with knowledge extracted mined directly from telecommunication alarm data [14]. In his next position the author served as a technology expert, promoting and educating others on advanced technologies, including data mining technology. In his last position at AT&T, the authors spent five years in a market analysis group applying data mining methods to solve business and marketing problems. The author has also gained some industry experience since joining Fordham University, via a grant from a large multinational financial institution to investigate the use of data mining in that domain. Useful insights have also been gained from graduate students who work in industry as professional data analysts.

This paper shares the author’s experiences and describes many of the issues that he has faced. These issues and experiences are organized around several topics. Organizational issues that impact data mining, including various “political” issues, are described in Section 2. There are many issues involved with obtaining suitable data and Section 3 covers this issue. Section 4 describes general issues related to the process of data mining, including the selection of data mining tools. Finally, Section 5 summarizes the issues and experiences described throughout this paper as well strategies and recommendations for dealing with these issues.

2 Organizational Issues

There are many organizational and non-technical issues—such as political issues—that may arise in a data mining project. The first thing that a data mining practitioner needs to be aware of is the organizational context in which he operates. Are the data mining practitioners an integral part of the organization that is initiating the project, are they in another part of the same company (acting as internal consultants), or are

they acting as outside consultants? Typically the closer the data mining practitioners are to the initiators of the data mining effort, the greater the support they can expect—but that is not always the case and there can certainly be resistance to data mining even within one's own organization. Thus, what matters is not just the organizational context, but the support that the data mining practitioners receive from the organization sponsoring the work. As one group of data mining practitioners explains [2] “never underestimate the power of politics and turf battles,” a sentiment this author certainly agrees with.

The author has encountered a variety of organizational issues during his tenure in industry. However, it is also important to understand the underlying *causes* for these issues, so that one can predict when these issues are likely to arise and possibly develop strategies for addressing or avoiding them. A key organizational issue relates to obtaining the data necessary for data mining (Section 3 covers the non-organizational issues related to data acquisition). Because the meaning of the data fields is often not fully documented or is only understandable to domain experts, it is usually critical to have access to these domain experts. Data mining expertise simply cannot make up for a lack of domain knowledge—something the author has found out the hard way. Furthermore, getting sufficient access to the domain experts is often even more difficult than getting access to the data—even when the domain expertise resides in the same company. There are a number of reasons for this—all of which in the author's case were probably exacerbated by the fact that his company was in a consistent “downsizing” mode during his years as a data mining practitioner. These reasons include:

- **Job security:** *sharing* data and expertise may make you or your organization less critical to the business.
- **Power:** as the adage goes, “knowledge is power” and the more knowledge you control the more powerful you are. Most groups within a business want to retain or increase their power and not dilute it by sharing critical information. Organizations also tend to want to keep work for themselves and increase their “head count” (i.e., number of employees).
- **Limited Time:** Time is a valuable resource and sharing knowledge really does require significant time. Also, the most knowledgeable domain experts—which are the ones you really need access to—are typically the most valued and in demand.
- **Lack of Budget:** For many data mining projects there is either no budget or a very small budget to reimburse the domain experts for their time. Thus, the organization with the necessary information may not be fairly compensated for access to their domain experts.
- **Pride:** A group often believes that they are doing the best job possible and that there is no benefit in bringing in outsiders, potentially with “new-fangled” and (in their eyes) unproven methods.

While some of these reasons may not be consistent with the best interests of the business, they are rational from the perspective of the individual workers and should not be underestimated or ignored. In the author's experience these issues arise in virtually every project. These issues are certainly not specific to data mining and in fact the author has encountered these same issues when working on expert systems [15], which, like data mining, requires a lot of domain expertise and can lead to a loss of control of critical knowledge and lost jobs within an organization. As a concrete example, in one project the author needed to mine the call detail data that describes every individual phone call that traverses the telecommunications network, but many of the fields were hard to interpret and poorly documented, so additional domain expertise was required. The person with that domain expertise was located in another organization within the same company and had previously functioned as a consultant to the author's organization—and had received funding for that. Given that “downsizing” was occurring, the domain expert was not motivated to share his domain expertise, acquired painstakingly over many years, and thus it was difficult to obtain the necessary information.

The organizational issues just mentioned may not have easy solutions, but there are certain actions that can be taken. First, assess the amount of organizational support before starting the project and, if at all possible, get formal commitments for the support before starting the project. Also, make sure that the necessary domain experts are allocated to the project and funded at the appropriate level, and, if feasible, transferred into the organization, even if only on a temporary basis.

There are a variety of other organizational and political issues that can also arise. One issue relates to a bias that is sometimes found against data mining. This bias is probably due to our field being relatively new and the unjustified hype that occurred as data mining began to become popular. While the author found relatively little resistance to data mining while at his former employer, many of the graduate students from the Economics department, who take his data mining course, have specifically noted this bias when working in industry. This is undoubtedly due to the fact that Economics has traditionally been dominated by Statistics and the influence of Computer Science has been relatively minimal, at least until recently. Some of these students have received very negative comments about data mining, especially as they try to utilize their new skills at work, including the fact that “it just does not work.” The best way to combat the bias against data mining is through the deployment of successful applications and education. It certainly is a good idea for data mining practitioners to be aware of the commonalities and differences between data mining and statistics and this can be done by reading articles on this topic by Friedman [5] and Hand [6].

Given these and other issues, for data mining to succeed it often needs to yield significant, non-incremental, improvements. Otherwise there may be too much inertia in the organization to deploy new processes. The author has found this to be true, especially when promoting other “new” technologies

such as expert systems and object-oriented programming and design. Many projects the author has been involved in were never completed because the benefits were not clear enough to warrant continued work. Other data mining practitioners have also found this to be true [2], as evidenced by the statement “The data mining algorithm has to perform demonstrably and considerably better than an existing system.” Thus it is important to assess the potential upside of a data mining project early on and determine whether the potential benefit is sufficient to ensure adoption of the project. It is also important to balance short-term and long-term needs since short-term benefits can help maintain support for a project. This should be kept in mind as the work is being planned and because of this a phased approach is often best. This approach is advocated by one group of data mining practitioners [8] when they recommend that one should “crawl, walk, run.” As they point out and as the author noted for himself, in many cases the customer may not even require true data mining but instead just needs some fancy reports. In this case it is generally best to satisfy their immediate needs and then move on to a more complex analysis of the data, if necessary.

Many projects, however, are dropped not due to inertia, but due to an even more significant issue—the project was never *bought into* by the people in charge. This might sound silly in a business environment, but it occurs all of the time. It is especially prevalent when dealing with new technologies and in companies that are technology driven, where the technology experts (as in the author’s case) often reside in separate organizations, or *laboratories*. In this situation it is often not the case that the business customer recognizes a need and seeks expert advice. Rather, a more typical scenario is that the technology expert actively tries to find a customer and business problem that matches his expertise.

The practice just described may seem like a poor business model—and perhaps it is—but there is some justification for it when the customer is not knowledgeable about the new technology and thus may not even recognize suitable business problems. This is especially true for data mining. In organizations where this practice exists the data mining practitioner may need to spend a significant amount of time educating the customer, trying to elicit good business problems, and selling his ideas. Thus, a data mining practitioner should have good communication skills and also be a good educator. Unfortunately it is often difficult to get “buy in” for a project early on, and thus one has a tendency to just jump into the project, without a firm commitment from the customer. The hope is that one can achieve good results quickly and use this to then get the necessary commitment. This often leads to a lack of resources (such as time and expertise from the customer) which may ultimately harm the project. But there are benefits to this model for data mining practitioners and their organizations. In the author’s organization if you achieved good results and managed to get the customer to buy into a project, then you could repeat the analysis periodically and receive substantial funding with relatively little effort. This funding could then be used in turn be used to subsidize the exploratory phase of other data mining projects.

There are many other organizational issues that could be discussed. One issue involves who should perform the data mining analysis—should it be done “in house” or by an external consultant. At the start of the author’s career it was perceived that his employer had the “not invented here syndrome”, where everything that could be done internally was done internally (e.g., one organization even developed its own operating system). However, as competitive pressures mounted and corporate cultures changed, the company and most other companies moved away from that philosophy. Now the “don’t reinvent the wheel” philosophy seems to dominate, where any work that can be outsourced is outsourced. This certainly includes data mining, since many companies cannot support their own data mining organizations or experts. One should be aware of the philosophy of their organization and its implications. The author recently acted as a consultant for a large multinational bank for the initial stages of a project, which could have been done internally, but the decision was made to outsource it all—at great expense. While there are advantages to outsourcing data mining work, there are many disadvantages, some of which have already been mentioned (e.g., the larger the distance to the customer the more resistance). However, one issue that should not be forgotten is that domain expertise is very important and difficult to acquire. Thus, if the knowledge can be kept internally, maintained, and reused, it may be less costly in the long run.

The author’s industry experience involved working with data analysts from very different background. Most were statisticians or, more specifically, econometricians. Interacting with people who have diverse backgrounds is very useful and provides a good environment for learning. This is especially useful when some problems are more suited to one discipline over another. For example, much of the data that the author has mined in industry is time series data and very often statistical techniques are more appropriate for handling this type of data. Thus, it can be beneficial for organizations that are involved with data analysis to employ people with diverse backgrounds, rather than isolating them in different groups—which is a common practice.

3 Obtaining and Preparing Data

Data acquisition is one of the key stages in the data mining process [4] and in the author’s experience this is one of the most problematic stages. Section 2 focused on some of the organizational and political issues that may complicate data acquisition but in this section we focus on some additional issues that complicate this step. To appreciate the importance of the data acquisition phase it is important to understand that data acquisition is typically the most time-consuming part of the data mining process. The author’s experiences—which are generally consistent with that of other data mining practitioners—indicate that significantly more than half of the overall data mining effort is spent obtaining, preparing, pre-processing, and transforming the data. It is critical that new data mining projects budget for this time and provide the

resources necessary to acquire the data. For those who are interested in learning more about data preparation, an entire text is dedicated to this topic [9].

The data that one would like to have for data mining often is not available. In some cases it may be possible to modify business processes to acquire the data, but often that is not practical. For example, when analyzing telecommunication network data the author sometimes found that useful information was not stored, but since modifying the software on telecommunication switches is both costly and risky, it was not even feasible to start collecting the desired data on an ongoing basis. The best practice is to carefully consider what data may potentially be useful when new data sources becomes available and store as much as possible, or, as Kohavi et. al [3] suggest, "Collect the right data, up front." Identifying the "right data" is not easy since it is difficult to envision all potential uses for data, but corporations are finally taking the time to do this because they now recognize the business value of data.

Having data stored is important, but it is equally important—if not more important—to make sure that the data is readily *accessible*, given the many organizational issues mentioned in Section 2. There has been a shift toward easier access to data in industry over the past decade, as many corporations have moved from hundreds of separate, independent databases to a few large, carefully maintained, data warehouses with simple, uniform, (often web-based) user interfaces. For large corporations with many legacy systems this was extremely costly to implement, but ultimately is well worth the effort. Unfortunately, much of the data stored in typical data warehouses is summarized data, whereas in data mining one typically requires access to the lowest level, non-aggregated, data. Until recently it was difficult or impossible to permanently store these huge amounts of data (e.g., the AT&T call record database contains over 1.9 *trillion* records) and as a consequence one would only have access to the full data for a limited time period, which can be problematic if one is trying to track changes over time or needs to model phenomena that occur only occasionally (e.g., phone traffic on Mother's day). However, over the past few years it has finally become feasible to store hundreds of Terabytes of data, so now even non-aggregated data can often be kept in its entirety. Having this data, clearly documented and easily accessible, can dramatically reduce the time required during the data acquisition phase of data mining.

One problem that the author has encountered repeatedly, which is often overlooked, relates to combining data from multiple sources. This combination requires a common key, but such a key is not always available. In particular, in a large company organized into many independent business units (some of which may have formerly been independent companies), different databases often have different ways of identifying customers. Thus even the most basic merging of data can be problematic. For example, a customer like IBM might have many variations of its name and the contact information, which is often used as a secondary key to uniquely identify a customer, may vary. This "name matching" problem was also

encountered by the author when working with a large financial institution. In each instance where this problem was encountered by the author, the problem required months or years of effort to address and the development of specialized name-matching software. This problem is significant and can sometimes be avoided by developing consistent terminology within a company and carefully reviewing database schemas, but even this is not sufficient to avoid all problems due to unanticipated changes such as mergers and acquisitions.

One of the key steps in the data mining process [4] involves preparing the data so that it is suitable for data mining. Very often the raw data must first be transformed before it is useful and this was especially true in the author's experience, largely due to the fact that much of the data he mined was temporal in nature (i.e., data streams) and could not be fed directly into conventional data mining prediction algorithms (e.g., decision trees, rule-based). For instance, call detail records are essentially generated in real-time by a telecommunication network as a data stream, but in order to mine information about individual customers (represented by phone numbers) all of the calls associated with a phone number must be aggregated into a single record. This is typically done by introducing a host of summary features, such as *average call duration* and *average number of calls per day*. The aggregation process must be done carefully to ensure that the information most critical for data mining is not lost and this is where domain expertise and insight can be critical—and where creativity may enter the process via the construction of new features. As an example, one data mining problem that involved trying to identify telemarketers based on their calling behavior. To help address this problem we introduced a feature that measured the *dispersion* of a user's phone calls with respect to different area codes, since most traditional residential and business customers call only a relatively few area codes. This feature turned out to be very useful in building the predictive model. In general, feature construction or feature engineering is a critical step which can greatly impact the success or failure of a data mining project. The issues with data streams are also not unique to the telecommunications industry and arise in many different domains, such as business (daily stock prices) and medicine (cardiac monitoring).

Data quality is another key issue and it is critical that a data mining practitioner validate the data and clean it as necessary. The author has often found errors in the data and has also found that the documentation that describes the data is often incorrect or out of date. Examining the distribution of each data field is critical and can quickly identify errors—as well as outliers. As an example, one project the author participated in involved analyzing the behavior of "800" number calls, which incur no cost to the call originator. After sorting these 800 numbers by total number of calls per month, it turned out that a few of these numbers accounted for a large percentage of the total calls (a few of the 800 numbers were associated with millions of calls per month). As it turned out, these numbers were calling card numbers owned by the telecommunication company and anyone who used one of its calling cards had to call this number. We then removed these

numbers from the analysis since we were only interested in predicting the phone usage of 800 numbers owned by “true” customers. The key lesson is to understand your data and ultimately the best way to do this is to spend time looking at the data records and the distribution of values for each field.

Another data-related issue that the author has repeatedly encountered concerns class imbalance [13]. Many prediction problems, including the prediction of telecommunication equipment failures [14], are extremely rare. The issue of dealing with “unbalanced data” is not at all uncommon in data mining, but yet standard data mining algorithms, which are geared toward maximizing predictive accuracy, usually perform poorly in these cases. Partially for this reason the author developed a custom tool, Timeweaver [12], which was designed to perform well for unbalanced classification problems, but due to the inherent complexity of this problem the author was forced to tackle the easier problem of predicting circuit pack failures rather than the much rarer catastrophic failures of entire switches. Fortunately there has been a great deal of work in recent years in mining unbalanced data, including two workshops on this topic [1] [7]. One thing that is important in these cases is to try to obtain accurate cost information, so that cost-sensitive learning can be used to compensate for the class imbalance. Since in this case the cost of a false negative will almost always be higher than the cost of a false positive (where the rare class is the positive class), cost sensitive learning will tend to cause more of the rare cases to be predicted, which is usually desirable. Unfortunately, it is often difficult to obtain accurate cost information, which is why this information is usually not provided. Nonetheless, every attempt should be made to obtain this cost information, even if one is only able to obtain rough estimates for these costs.

The acquisition of data from external sources (e.g., census data, data about business from D&B, etc.) represents an opportunity that is often overlooked. In many cases the data may not directly address the underlying data mining problem in an obvious way, but may nonetheless lead to significant improvements. This also provides an opportunity for the practitioner to be creative. As one example, many of the author’s analyses have involved businesses at specific locations. Based on data taken from the U.S. Census, the author has been able to supplement this data with aggregate data associated with the businesses’ location. For example, depending on the specific data mining problem, it might be useful to know if the geographic area has a higher than normal percentage of high-tech companies or people with graduate degrees. Practitioners should thus be open to the use of all sources of data, many of which can be acquired without cost.

Data, however, is often acquired only with a cost, even if that cost is not in the form of payments to a third party. As discussed earlier, it may take time to acquire the data, clean the data, and transform the data. Thus, one decision that the author encountered regularly was how much data to acquire, clean, and process. This topic is not something that is often discussed in the research literature, although it has been studied [16]. While one could generate learning curves by sam-

pling different training set sizes from the currently available training data, a simpler but effective strategy is to just generate a few points on the learning curve below but close to the current training set size, in order to estimate the benefit of obtaining additional training data. Then, depending on the slope of the learning curve near the current training set size and the cost of acquiring additional data, one can determine whether further data acquisition is warranted.

4 Data Mining Process

This section describes issues related to the general process of data mining and the selection of tools for performing data mining. The data mining process is an iterative process and experience has shown that it is important to take this into account. Very often data mining will not go as expected and one must respond based on the feedback gained along the way. One strategy is to build a quick prototype, even though this generally requires that a variety of data quality issues must (initially) be ignored. Generating *some* results quickly, however, is very useful since this will ensure that you have a well defined problem and well thought out evaluation criteria. The reasons for building quick prototypes in data mining are largely the same ones for building fast prototypes for software systems, where the “waterfall” model of software development often fails because it does not easily accommodate feedback and changing requirements. Kohavi et al. [8] essentially advocate this strategy when they advise “build simple models first.”

Another reason for building quick prototypes is to establish a baseline against which to measure future improvements. If complex modeling or highly engineered features fail to yield improvements over simple methods and features, then the complex ones should not be employed. It is also important to ask if the initial model—which may not even have been generated using data mining—is sufficient for solving the problem. Past experience has demonstrated that quick prototypes sometimes are satisfactory. In one case the business problem was to determine the fraction of phone lines and phone minutes associated with voice, data, and fax. The initial plan was to build a training set for various usage segments (where the segments were based on the number of monthly phone minutes), train a predictive model for each segment, and then apply the predictive model to the “universe” of phone lines. As it turned out, the segments with very large phone minutes were small enough to be sampled completely and our automated tool, which automatically dialed the lines to determine the type of phone usage, could, in the time allotted, label 10,000 lines from each of the segments with lower phone usage. Given all of this data and our random sample for each segment, no predictive model was required. Instead, we just scaled up the numbers to account for the known number of lines in each usage segment. In other cases experience has shown that only very simple predictive models, sometimes using only a few variables, are necessary.

There are many different data mining tools that are available and care should be taken when choosing a tool to use.

Because the author has not performed an extensive evaluation of these tools, comprehensive guidance as to which tool is best can not be provided. However, the author has used a number of different tools and while in industry used both SAS Enterprise Miner [10] and SPSS Clementine [11], two of the more powerful and widely used commercial data mining packages. These tools are comprehensive data mining suites that support a large number of data mining methods. Based on experience it is preferable to use suites such as these rather than stand-alone data mining tools that only support a single method. The advantage of using one of these (or similar) tools is that they allow one to easily generate multiple models, compare the performance of different models, and combine multiple models via an ensemble. These packages also provide basic support for data exploration and visualization, as well as methods for data transformation. In short, these tools provide at least some support for all parts of the data mining process [4]. While these commercial suites can be expensive, free alternatives do exist, such as WEKA [17], an open source data mining package from the University of Waikato that has many of the features of Clementine and Enterprise Miner. Others, including some of the author's students, have also had good experiences with R (<http://www.r-project.org/>), an open source statistical programming environment that contains a large number of data mining methods.

5 Summary and Recommendations

This paper discussed the personal experiences of the author while acting as a practitioner of data mining in an industrial setting. A number of issues were raised—issues that often are not discussed in research papers on data mining. Hopefully this discussion will provide some insight into the challenges one may encounter when using data mining to solve complex real-world problems. This paper also outlined strategies for addressing the issues and challenges that were raised and these challenges and recommendations are summarized in Appendix 1, provided at the end of this paper after the references. Hopefully this paper will stimulate further discussion, and research, on the *practice* of data mining.

6 References

- [1] N. Chawla, N. Japkowicz and A. Kolcz, (editors), in *Proceedings of the ICML Workshop on Learning from Imbalanced Data Sets*, 2003 [online]. Available: <http://www.site.uottawa.ca/~nat/Workshop2003/workshop2003.html>.
- [2] T. Dasu, E. Koutsofios, and J. Wright, "Zen and the art of data mining," in *Proc. of the KDD 2006 Workshop on Data Mining for Business Applications*, 2006, pp. 37-43.
- [3] T. Dasu and G. M. Weiss, "Mining data streams," in *Encyclopedia of Data Warehousing and Mining, second edition*, J. Wang Ed. Kluwer Academic Publishers, 2008, pp. 1248-1256.
- [4] U. Fayyad, G. Piatesky-Shapiro and P. Smyth, "From data mining to knowledge discovery in databases," *AI Magazine*, vol. 17, no. 3, pp.37-54, Fall 1996.
- [5] J. Friedman, "Data mining and statistics: what's the connection," *29th Symposium on the Interface: Mining and Modeling Massive Data Sets in Science, Engineering, and Business*, 1997.
- [6] D. Hand, "Data mining: statistics and more?" *American Statistician*, vol. 52, no. 2, pp. 112-118, May 1998.
- [7] N. Japkowicz (editor), *Proceedings of the AAAI'2000 Workshop on Learning from Imbalanced Data Sets*, AAAI Tech Report WS-00-05, July 2000.
- [8] R. Kohavi, L. Mason, R. Parekh, and Z. Zheng, "Lessons and challenges from mining retail E-commerce data," *Machine Learning*, vol. 2, no. 1-2, pp. 83-113, Oct.-Nov. 2004.
- [9] D. Pyle, *Data Preparation for Data Mining*. San Diego: Morgan Kaufmann, 1999.
- [10] SAS Corporation, "SAS Enterprise Miner." [<http://www.sas.com/technologies/analytics/datamining/miner>]
- [11] SPSS Corporation, "Clementine." [<http://www.spss.com/clementine>].
- [12] G. M. Weiss, "Timeweaver: a genetic algorithm for identifying predictive patterns in sequences of events", in *Proc. of the Genetic and Evolutionary Computation Conference*, 1998, pp. 718-725.
- [13] G. M. Weiss, "Mining with rarity: a unifying framework," *SIGKDD Explorations*, vol. 6, no. 1, pp. 7-19, June 2004.
- [14] G. M. Weiss and H. Hirsh, "Learning to predicting rare events in event sequences," in *Proc. of the 4th International Conference on Know. Disc. And Data Mining*, 1998, pp. 359-363.
- [15] G. M. Weiss, J. P. Ros, and A. Singhal, "ANSWER: Network Monitoring Using Object-Oriented Rules," in *Proc. of the 10th Conference on Innovative Applications of Artificial Intelligence*, 1998, pp. 1087-1093.
- [16] G. M. Weiss and Y. Tian, "Maximizing classifier utility when there are data acquisition and modeling costs," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 253-282, Oct. 2008.
- [17] I. H. Witten and E. Frank, *Data Mining: practical machine learning tools and techniques, 2nd Edition*, Morgan Kaufmann, San Francisco, 2005.

Appendix 1: Summary of Issues and Recommendations

The Table below lists the main issues discussed in this paper and then provides a summary of the recommendations associated with each of these issues.

<p><i>Issue: lack of access to data and domain expertise (due to concerns with job security, pride, power and limited resources)</i></p> <ul style="list-style-type: none"> • Provide sufficient budget and resources for acquiring the data and include time with domain experts. • If possible, have domain experts assigned to your organization, even if temporarily.
<p><i>Issue: bias/fear of data mining</i></p> <ul style="list-style-type: none"> • Education and understanding. Avoid hype. Read Friedman [5] and Hand [6].
<p><i>Issue: organizations have inertia and are often reluctant to adopt new methods and technologies</i></p> <ul style="list-style-type: none"> • Evaluate projects carefully and focus on ones that may yield clear and substantial improvements. • Used a phased approach that will provide some short-term “wins.”
<p><i>Issue: may often be no up-front “buy-in” from those who are responsible for funding the project.</i></p> <ul style="list-style-type: none"> • Data mining practitioners should be a good educator, communicator, and salesman.
<p><i>Issue: data may have errors and/or documentation can be out-of-date or just incorrect</i></p> <ul style="list-style-type: none"> • Examine data distribution for each field and determine whether values make sense; investigate a sample of the outliers. • Select some examples at random and verify that the data values make sense.
<p><i>Issue: it is often it is difficult to merge information from multiple data sources due to lack of unique key</i></p> <ul style="list-style-type: none"> • Try to adopt company-wide standards (this is often not sufficient due to mergers and acquisitions). • Develop tools to perform the matching process (unfortunately this can be expensive and time consuming).
<p><i>Issue: data may not be represented at the level needed for data mining</i></p> <ul style="list-style-type: none"> • Transform the data to the appropriate level. If aggregation is necessary, preserve the most critical information. • Be creative and use domain knowledge and/or exploratory data analysis to come up with useful features.
<p><i>Issue: class distribution of data is often highly unbalanced</i></p> <ul style="list-style-type: none"> • Use appropriate evaluation metrics (avoid accuracy). Perhaps use ROC analysis if cost information is not available. • Try to acquire accurate cost information or at least reasonable estimates and then employ cost-sensitive learning.
<p><i>Issue: good training data is costly to obtain</i></p> <ul style="list-style-type: none"> • Generate learning curves to determine whether it may be profitable to obtain additional training examples. • Try to incorporate secondary data sources (some of which may be free), such as U.S. Census data.
<p><i>Issue: the data mining problem and/or the evaluation criteria may not be well defined.</i></p> <ul style="list-style-type: none"> • Build a quick prototype in order to get quick feedback. Customers respond best given something concrete to evaluate.
<p><i>Issue: what data mining methods and tools should be used?</i></p> <ul style="list-style-type: none"> • Data mining suites that support multiple data mining methods and the full data mining process are generally best. • No one data mining method is best, so try a variety of methods. Prefer simple methods.

Privacy Preserving Sharing of Data

Sreenivasa Rao, S. Ram prasad Reddy, KV Ramana, V. Valli Kumari, KVSVN Raju

Abstract— Organizations share their data about customers for exploring potential business avenues. This sharing of data has posed several threats leading to individual identification. Owing to this, privacy preserving data publication has become an important research problem. The main goals of this problem are to preserve privacy of individuals while revealing useful information. An organization may implement and follow its privacy policy. But when two companies share information about a common set of individuals, and if their privacy policies differ, it is likely that there is privacy breach unless there is a common policy. One such solution was proposed for such a scenario, based on k-anonymity and cut-tree method for 2-party data. This paper suggests a simple solution for integrating n-party data using dynamic programming on subsets. The solution is based on thresholds for privacy and information gain based on k-anonymity.

Index Terms—Privacy preserving, data mining, k-anonymity

I. INTRODUCTION

WITH numerous organizations collecting customer data there exists a possibility of data sharing for exploring interesting data about behavior of customers [22, 24]. This leads to identification of customers which can be treated as a privacy threat according to HIPAA[1] and EU directives[2]. These acts insist that anonymity should be guaranteed if the customers wish so.

A customer data normally contains attributes like SSN, name, age, zipcode, date of birth and gender. This data enables identification of the individuals even though information like SSN and Name suppressed. This was first identified in [3]. The solution proposed k-anonymity property to be applied to the data before release. k-anonymity demands that every tuple in a micro data table released be indistinguishably related to no fewer than k-tuples. Subsequently several solutions were published. Most of them addressed issues related to preserving privacy of individuals related to a single organization [18, 21, 22]. Even though the data is anonymised, this data when integrated with other organizations data might reveal information about a specific individual. This paper discusses an approach to protect

privacy when anonymized data of two or more organizations is integrated.

Section II discusses the related work, Section III formalizes the problem. Section IV discusses the architecture for implementing our approach. Section V analyses our approach when compared to other published work.

II. RELATED WORK

The organizations share their data with many other research institutes for various uses. Today technologies are providing easy way of information sharing. However sharing the data with outsiders should not reveal the individual identification of a person [4]. Care must be taken to provide the privacy for the person specific data at the time of publishing personal information for research purposes. The objective of privacy preserving mining is that this data, when published should not link back to the individual.

In *k*-anonymised data, privacy is achieved through generalization and suppression. Suppression of directly identifiable attributes, like name, SSN (Social Security Number) is done by not publishing them. The whole data set is divided into equivalence classes. Each equivalence class has a distinct tuple occurring *k*-times, which is called generalization. Thus, generalization means replacing a tuple with a more generalized tuple, which is indistinguishable from several other tuples in the equivalence class. This is also called as anonymization. But several problems are identified with *k-anonymity* [5]. A *k*-anonymous table may allow an adversary to derive the sensitive information of an individual with 100% confidence. There is considerable information loss from the data. And it does not take into account personalized anonymity requirements.

k-anonymity allows an attacker to discover the value of sensitive attributes, when there is little diversity in the sensitive attributes [6]. The concept of *k-anonymity* tries to capture, on the private table to be released, one of the main requirements that has been followed by the statistical community and by agencies releasing the data, and according to which the released data should be indistinguishably related to no less than a certain number of respondents. The set of attributes included in the private table, are also externally available and therefore are exploitable for linking, is called quasi-identifier. The requirement just stated is then translated in the *k-anonymity* requirement, which states that every tuple released cannot be related to fewer than '*k*' respondents. To counter the problems stated above, another scheme called *l*-diversity[6] was proposed. *l*-diversity provides privacy even

when the data publisher does not know what kind of knowledge is possessed by the adversary. It ensures that, all tuples that share the same values of quasi identifiers should have *l-diverse* values for their sensitive attributes. Even *l-diversity* is prone to attacks by an adversary, as it guarantees a low breach probability [5, 7].

Anatomy is another *l-diversity* specific method. Though it does not violate the *l-diversity* property, it confirms that a particular individual is included in the data. *t-closeness* is another scheme, which recommends table-wise distribution of sensitive attribute values to be repeated within each anonymised group [8]. In other words, it formalizes the idea of global background knowledge by requiring that the distribution of a sensitive attribute in any equivalence class is close to the distribution of the attribute in the overall table (i.e. the distance between the two distributions should be no more than a threshold t). To find the distance between values of sensitive attributes, they use the Earth Mover Distance metric to measure the distance between the two distributions.

Personalized privacy preservation is another method which allows each sensitive attribute in a record in the table to have a privacy constraint [5, 23, 25]. The paper states the problem that all the methods exert the same amount of privacy preservation for all persons, without catering for their concrete needs. The consequence is that insufficient protection is offered to a subset of people, while excessive privacy control is applied to another subset. The paper proposed a generalization framework based on the concept of personalized anonymity. The advantage is that large amount of information can be retained. The main drawback is its complexity, both time and space wise. The computational effort is too high as generalization has to be done also on the sensitive attribute column. Personalized privacy preservation uses a tree based approach, for personalized privacy, greedy algorithm is used and hence is not optimal, so does not achieve minimal loss. *p-sensitivity k-anonymity* is almost similar to *l-diversity*[6].

Extended *p-sensitivity k-anonymity* is a scheme that extends *p-sensitive k-anonymity* property that which is similar to the personalized privacy method, where in the protection is offered at different levels in the taxonomy for the sensitive attribute [9]. Another scheme in [10] assumes hierarchy in each QI attribute, and that all partitions in a general domain should be at the same level of hierarchy.

Genetic algorithm framework is used to transform the data while satisfying the privacy constraints [11]. The data transformation was done by generalizing or suppressing the potentially identifying content of the data. Usage based metric was defined to quantify loss of information due to the transformations. Top down specialization based solution was

offered in [12] for handling both categorical and continuous attributes. The data structure called TIPS was developed which had every parent node representing a generalized record, every child node representing a specialization of the parent on exactly one attribute. The approach produces progressive generalization process which used a metric to step through to determine and decide trade off of privacy and accuracy. Another method for anonymizing classification data was proposed in [13]. Their goal was to find the *k-anonymization* of data which preserves the classification structure. The paper argues that minimizing distortion in the training data is not relevant to the classification goal that requires extracting the structure of predicting on future data. The paper presents a top down approach to iteratively refine data from general state into special state, guided by maximizing data between information and anonymity. A solution for information sharing by two parties who wish to integrate their private databases who achieve the common goal beneficial to both provided the privacy requirements are satisfied. The privacy requirement is none other than the *k-anonymity* requirement of integrated databases. This paper also used the top-down specialization technique.

Privacy preserving in clustering using fuzzy sets [14] addresses the problem of preserving privacy of individuals when data is shared. Sharing the entire data not only provides irrelevant information to the miner but also makes the data vulnerable to privacy violation. Dimensionality of the data is reduced as per the request for data, which is a small quantity of the entire database. Later confidential attributes in the data to be released are transformed to fuzzy so as to meet privacy requirements and simultaneously preserving the meaning of the data and also accuracy of the results.

Privacy preserving in clustering by categorizing attributes using privacy and non privacy disclosure sets [15] proposes a categorization process of attributes into privacy and non privacy disclosure sets based on the business rules. The non privacy disclosure set is categorized to have sensitive and non sensitive attributes. If miner requests sensitive attributes along with the attributes of non privacy disclosure sets, it leads to privacy violation. In such cases, the required sensitive attributes are transformed into fuzzy for preserving privacy. Before transforming the data, dimensionality of the data is reduced to ensure relevant data is released. This claims that the proposed categorization ensures less information loss, this is because when the miner requests data that contains no attribute combinations that violate individuals' privacy and then no attributes are transformed into fuzzy.

In the paper a novel approach for privacy preserving publication of data [16], the solution is based on fuzzy and taxonomy tree where optimal and useful knowledge about individual's information can be retained while preserving

privacy of that individual. The basic approach allows owner of the data to prescribe privacy levels and disclosure levels.

Our solution is based on the problem and scenarios stated in [8].

III. SYSTEM ARCHITECTURE

This model (Figure 1) primarily has two objectives: preserving privacy while revealing useful information for sensitive attributes and to find an integrated table without loss of information. This involves the following steps:

1. Dimensionality reduction: Suppressing the unnecessary attributes.
2. Identifying sensitive attributes through business rules
3. Categorizing the attributes (Categorizer)
4. Use anonymizer for preserving linkage of individual with sensitive categorical attributes.
5. Release the anonymised data and announce the joint anonymity property.
6. Perform data integration without revealing any sensitive information and its associated individual.

Dimensionality reduction

Dimensionality reduction and attribute selection aim at choosing a sub set of attributes sufficient to describe the data set. The goal of the methods designed for dimensionality reduction is to map d -dimensional objects into k -dimensional objects, where $k < d$. Dimensionality reduction is beneficial only when the loss of information is not critical to the solution or the problem, or if more information is gained by the visualization of the problem than what is lost. Reduction from dimension d to k ($k < d$) reduces complexity, reduces communication cost and provides privacy since extra data given may help in re-identifying individuals or loss of sensitive information vulnerable for second use.

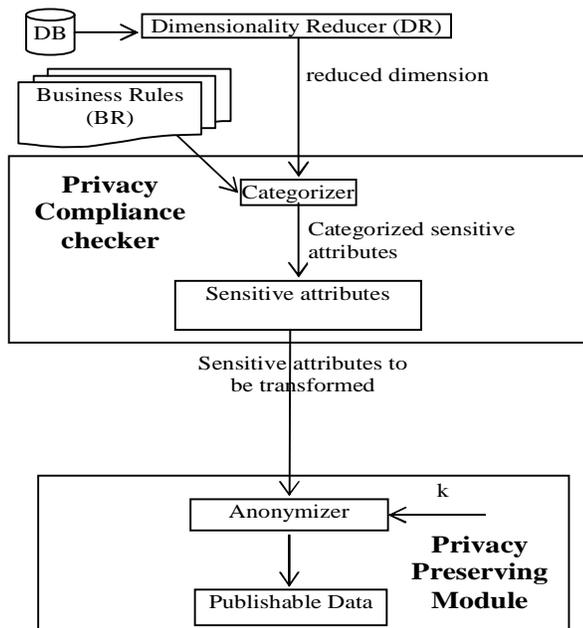


Figure 1. System Architecture

Categorizing attributes

The attributes in the reduced table are classified as identifier attributes, sensitive attributes and quasi identifiers.

1. *Identifier Attributes*: These attributes uniquely identify the individual associated with the tuple, as anonymization requires that the data be disassociated with the identifiers. One specific example is the name attribute.
2. *Sensitive attributes*: These attributes should not be disclosed to the public or may be disclosed after disassociating its value with an individual's other information.
3. *Quasi Identifier (QID)*: These values may be published, but it so happens that with a combination of these attributes an individual may get identified. For instance, age and gender might disclose the identity.

Anonymizer

Using the taxonomy trees of categorical and numerical attributes, we generate the k -anonymised data on each table.

IV. PROBLEM DEFINITION

Consider the data in Table 1 and taxonomy trees in Figures 2, 3 and 4 [17]. Party A and Party B own $T_A(SSN; Gender; \dots; Class)$ and $T_B(SSN; Education; Age; \dots; Class)$ respectively. After joining the two tables on SSN, the "female, mechanical" on $(Gender; Education)$ becomes unique, therefore, vulnerable to be linked to sensitive information such as *Age*. To protect against such linking, we can generalize *Civil* and *Mechanical* to *Non-Computers* so that this individual becomes one of many female professionals. However we preserve information as the basic classification is not changed.

Definition 1. (k-anonymity)

Consider p quasi-identifiers QID_1, \dots, QID_p on T . Let r_i denote the number of records in T that share the value qid_i on QID_i . The anonymity of QID_i , denoted r_i , is the smallest r_i for any value qid_i on QID_i . A table T satisfies the anonymity requirement $\{ \langle QID_1; k_1 \rangle, \dots, \langle QID_p; k_p \rangle \}$ if $r_i \geq k_i$ for $1 \leq i \leq p$, where k_i is the anonymity threshold on QID_i [8]. If QID_j is a subset of QID_i , where $i > j$, and if $k_j \leq k_i$, then $\langle QID_j; k_j \rangle$ is implied by $\langle QID_i; k_i \rangle$, therefore, can be removed.

Table 1: Compressed Table

Shared		Party A		Party B			
SSN	Class	Gen der	...	Education	Age	...	# of Recs
1-3	0Y3N	M		Life Sc.	17		3
4-7	0Y4N	M		Physical Sc.	18		4
8-12	2Y3N	M		Fine Arts	19		5
13-16	3Y1N	F		Journalism	24		4

17-22	4Y2N	F		Computers	26		6
23-25	3Y0N	F		Computers	27		3
26-28	3Y0N	M		Civil	27		3
29-31	3Y0N	F		Civil	27		3
32-33	2Y0N	M		Mechanical	27		2
34	1Y0N	F		Mechanical	27		1
Total	21Y13N						34

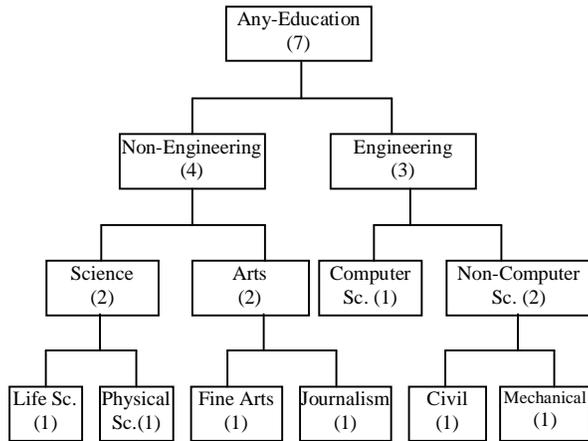


Figure 2. Taxonomy for Education

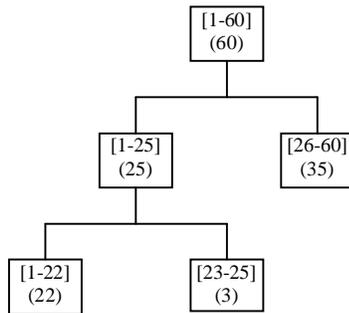


Figure 3. Taxonomy for Age

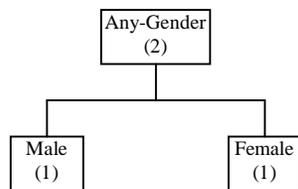


Figure 4. Taxonomy for Gender

V. PRIVACY PRESERVING DATA INTEGRATION

The k-anonymity policies of an organization define information access threshold for its database. The threshold is the minimum amount of generalization required for giving information. Given two or more organizations who want to share their data without revealing information, the privacy

preserving integration is to determine an optimal combination of attributes to disclose information while preserving privacy. Thus each organization may define privacy strength for each of their data sources and a joint anonymity requirement (Figure 5).

Generalization preserves the privacy where as specialization makes it more informative.

Definition 2. (Secure data integration [17])

Given two private tables T_A and T_B , a joint anonymity requirement $\{ \langle QID_1, K_1 \rangle, \dots, \langle QID_P, K_P \rangle \}$ and a taxonomy tree for each attribute, the secure data integration is to produce a generalized integrated table T^* such that it satisfies the joint anonymity requirement and retains as much information as possible.

Each node in the taxonomy is associated with a privacy strength value. The vale varies for categorical data and continuous data. For continuous data the difference between ranges is taken as the privacy strength of the nodes. For categorical data the leaf nodes are given values '1' and for the internal nodes the privacy strength is the number of leaf nodes that each node has.

Privacy strength (P) of a given node is the number of leaves in the in the sub tree with this node as the root.

Informativeness (I) is given as the ratio between the path length from root to the current node to sum of the path length from root to the current node and length of the tree with current node as the root.

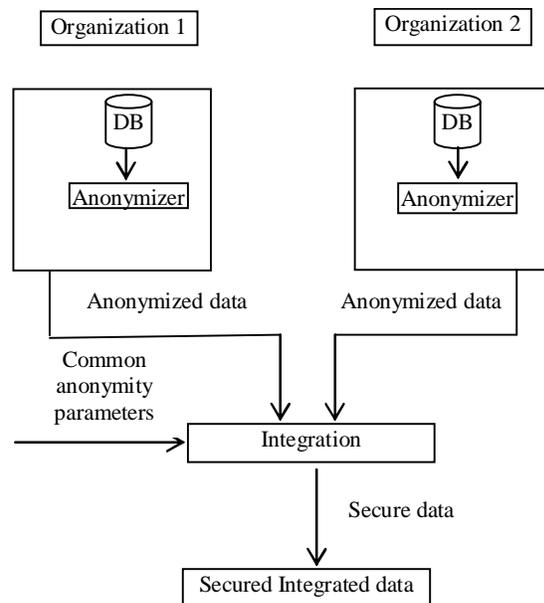


Figure 5. Secure data integration

VI. ANALYSIS

Dynamic Programming is a method for efficiently solving a broad range of search and optimization problems which exhibit the characteristics of overlapping subproblems and optimal substructures.

The principle of optimality is the basic principle of dynamic programming, which was developed by Richard Bellman: that an optimal path has the property that whatever the initial conditions and control variables (choices) over some initial period, the control (or decision variables) chosen over the remaining period must be optimal for the remaining problem, with the state resulting from the early decisions taken to be the initial condition.

In this paper, we use the principle of knapsack model using dynamic programming. Dynamic programming is mainly used to tackle optimization problems that are solvable in polynomial time. Dynamic programming is chosen to produce an optimal solution. The optimal solution is selected from the subsets that satisfy the threshold by using the principle of knapsack model.

Dynamic programming is process of solving problems where one needs to find the best decisions one after another. The method takes much less time than naive methods.

The computation of privacy strengths for the taxonomies discussed above are performed as shown below:

The privacy strength is shown in the parenthesis.

Education={ Any-Education(7), Engineering(3), Non-Engineering(4), Science(2), Arts(2), Computer Science(1), Non-Computer Science(2), Life(1), Physical(1), Fine Arts(1), Journalism(1), Civil(1), Mechanical(1)}

Age= {[1-60](60), [1-25](25), [26-60](35), [1-22](22), [22-25](3)}

Gender= {Any-Gender(2), Male(1), Female(1)}

Suppose if the QID is {Gender, Education} then the possible and valid subsets would be as below:

{ (Any-Gender, Any-Education), (Any-Gender, Non-Engineering), (Any-Gender, Engineering), (Any-Gender, Science), (Any-Gender, Arts), (Any-Gender, Computer Science), (Any-Gender, Non-Computer Science), (Any-Gender, Life), (Any-Gender, Physical), (Any-Gender, Fine Arts), (Any-Gender, Journalism) (Any-Gender, Civil), (Any-Gender, Mechanical), (Male, Any-Education), (Male, Non-Engineering), (Male, Engineering), (Male, Science), (Male, Arts), (Male, Computer Science), (Male, Non-Computer Science), (Male, Life), (Male, Physical), (Male, Fine Arts), (Male, Journalism) (Male, Civil), (Male, Mechanical),

(Female, Any-Education), (Female, Non-Engineering), (Female, Engineering), (Female, Science), (Female, Arts), (Female, Computer Science), (Female, Non-Computer Science), (Female, Life), (Female, Physical), (Female, Fine Arts), (Female, Journalism) (Female, Civil), (Female, Mechanical) }

For each subset, k-anonymity requirement is verified. If the required threshold is satisfied [19, 20], the privacy strength(P), informativeness(I) are computed. These values are recorded along with the respective subset and the threshold value. After considering all the possible subset, the subsets that have been recorded are taken into consideration. From these subsets, the supersets if any, are identified and discarded from the set. From the remaining subsets, the subset that produces the optimal solution is taken and the data is published satisfying the required threshold.

Algorithm: anonymization

Input: Integrated table that contains data of both the parties.

Output: Optimal anonymized table.

Step1: identify all the QID sets. { QID₁, QID₂, , QID_n }

Step2: for each QID_i that belongs to QID set, generate the power set.

Step3: for each qid_j combination in the power set of QID_i

Step4: generate the corresponding equivalence classes

Step5: compute the relative privacy strength(P), informativeness(I)

Step6: end for

Step7: end for

Step8: discard the equivalence classes that does not satisfy the threshold

Step8: find the equivalence classes that provide optimal solution by considering privacy strength and informativeness.

Step9: Publish the anonymized data.

For instance see table 2, let us consider one combination and observe the resultant dataset for a threshold value k=3. Here the qid is <Any-Gender, Any-Education>.

Table 2. Anonymized table

Class (Shared)	Gender	Education	Age	# of Recs
0Y3N	Any-Gender	.	Any-Education	17		3
0Y4N	Any-Gender		Any-Education	18		4
2Y3N	Any-Gender		Any-Education	19		5
3Y1N	Any-Gender		Any-Education	24		4
4Y2N	Any-Gender		Any-Education	26		6
12Y0N	Any-Gender		Any-Education	27		12

P = {2, 7} I = {0, 0} No. of Equivalence classes = 6, k = 3

Since the threshold is satisfied the particular combination is considered for analysis. The same is repeated for remaining combinations. The combinations that satisfy the threshold value would only be considered. From these sets, a combination value is selected such that it produces an optimal value where it provides better privacy and more information.

The dataset for the combination <Gender, Education> is given in table 3.

Table 3. Subsets satisfying the threshold values

QID	PS(P)	Info(I)	Anony(ki)	# Recs
Any-Gender, Any-Education	(2,7)	(0,0)	3	34
Any-Gender, Non-Engineering	(2,3)	(0,0.33)	3	16
Any-Gender, Engineering	(2,4)	(0,0.33)	6	18
Any-Gender, Science	(2,2)	(0,0.66)	3	07
Any-Gender, Arts	(2,2)	(0,0.66)	3	09
Any-Gender, Computers	(2,1)	(0,1)	3	09
Any-Gender, Non-Computers	(2,2)	(0,0.66)	9	09
Any-Gender, Life Sc	(2,1)	(0,1)	3	03
Any-Gender, Physical Sc	(2,1)	(0,1)	4	04
Any-Gender, Fine Arts	(2,1)	(0,1)	5	05
Any-Gender, Journalism	(2,1)	(0,1)	4	04
Any-Gender, Civil	(2,1)	(0,1)	6	06
Any-Gender, Mechanical	(2,1)	(0,1)	3	03
Male, Any-Education	(1,7)	(1,0)	3	17
Male, Non-Engineering	(1,3)	(1,0.33)	3	12
Male, Engineering	(1,4)	(1,0.33)	5	05
Male, Science	(1,2)	(1,0.66)	3	07
Male, Arts	(1,2)	(1,0.66)	5	05
Male, Computers	(1,2)	(1,0.66)	5	05
Male, Life Sc	(1,1)	(1,1)	3	03
Male, Physical Sc	(1,1)	(1,1)	4	04
Male, Fine Arts	(1,1)	(1,1)	5	05
Male, Civil	(1,1)	(1,1)	3	03
Female, Any-Education	(1,7)	(1,0)	4	17
Female, Non-Engineering	(1,3)	(1,0.33)	4	04
Female, Engineering	(1,4)	(1,0.33)	6	13
Female, Arts	(1,2)	(1,0.66)	4	04

Female, Manager	(1,1)	(1,1)	3	09
Female, Computers	(1,2)	(1,0.66)	4	04
Female, Journalism	(1,1)	(1,1)	4	04
Female, Civil	(1,1)	(1,1)	3	03

From the generated subsets, we select the subsets for which we get the maximum privacy strength value and maximum informativeness value. According to this the highlighted rows give the subset combinations that are to be selected. The correlation between privacy strength and informativeness is shown in figure 6.

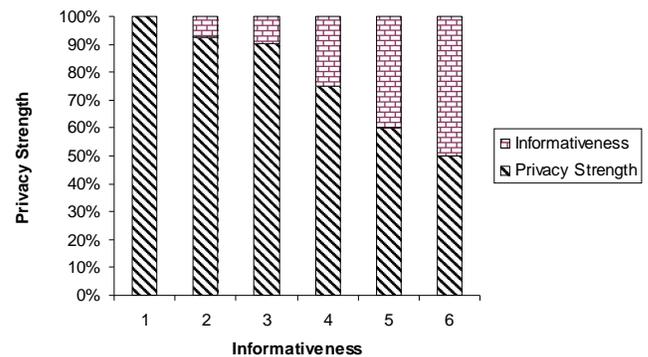


Figure 6. Privacy strength vs Informativeness

VII. CONCLUSION

The paper proposed a solution for achieving anonymity when data from two organizations with common privacy policy are integrated. The solution is a simple and effective method as it uses cost effective algorithms for achieving anonymity. The solution proposed in [17] is based on tree data structure called TIPS. We base our solution on subset generation and selecting the most relevant subset. We are currently examining the feasibility of this approach for achieving anonymity on the fly in dynamically growing databases.

REFERENCES

- [1] Centers for Medicare & Medicaid Services. The Health Insurance Portability and Accountability Act of 1996 (HIPAA). Online at <http://www.cms.hhs.gov/hipaa/>, 1996.
- [2] Council Directive (EC) 2001/29/EC of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society.
- [3] L. Sweeney. k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5):557-570, 2002.
- [4] R. Agrawal and R. Srikant. Privacy-Preserving Data Mining. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Dallas, Texas, May 2000.
- [5] X. Xiao and Y. Tao. Personalized privacy preservation. In ACM SIGMOD, June, 2006.

- [6] Machanavajjhala A., Gehrke J., Kifer D., "l-diversity: Privacy Beyond k-Anonymity". Proceedings of the 22nd IEEE Intl. Conf. on Data Engineering, 2006.
- [7] K. Wang, B. C. M. Fung, and P. S. Yu. Handicapping attacker's confidence: An alternative to k-anonymization. Knowledge and Information Systems: An International Journal, 2006.
- [8] Ninghui Li, Tiancheng Li and Suresh V. "t-Closeness: Privacy beyond k-Anonymity and l-Diversity". ICDE 2007, 23rd IEEE Intl. Conf. on Data Engineering, 2007.
- [9] Truta T.M., Bindu V. (2006), Privacy Protection: P-Sensitive K-Anonymity Property", Proceedings of the Workshop on Privacy Data Management, In Conjunction with 22th IEEE International Conference of Data Engineering (ICDE), Atlanta, Georgia.
- [10] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. In IEEE Symposium on Research in Security and Privacy, May 1998.
- [11] V. S. Iyengar. Transforming data to satisfy privacy constraints. In ACM SIGKDD, pages 279–288, 2002.
- [12] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In ICDE, pages 205–216, 2005.
- [13] Benjamin C.M. Fung, Ke Wang, Philip S. Yu, "Anonymizing Classification Data for Privacy Preservation," IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 5, pp. 711-725, May 2007, doi:10.1109/TKDE.2007.1015
- [14] P. Kusuma Kumari, KVSVN Raju, S.Srinivasa Rao, Privacy Preserving in Clustering using Fuzzy Sets, WORLDCOMP'06 The 2006 International Conference on Data Mining (DMIN'06), pp 290-295, JUNE 26-29, Las Vegas, USA, 2006.
- [15] K .Sridevi, KVSVN Raju, V.Valli Kumari and S.Srinivasa Rao. Privacy Preserving in Clustering by Categorizing Attributes using Privacy and Non Privacy Disclosure Sets, WORLDCOMP'07, The 2007 Intl. Conf. on Data Mining, pp301-307, June 2007, Las Vegas, USA.
- [16] V.Valli Kumari, S.Ram Prasad Reddy, M.Aruna Sowjanya, B.Jhansi Vazram, KVSVN Raju, "A novel approach for privacy preserving publication of data", in Proceedings of International Conference on Data Mining, DMIN'08, CSREA Press, Las Vegas, USA, in July, 2008.
- [17] K. Wang, B. C. M. Fung, and G. Dong. Integrating private databases for data analysis. In IEEE ISI, May 2005.
- [18] K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: A data mining Solution to privacy protection. In ICDM, pages 249–256, 2004.
- [19] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In IEEE ICDE, 2006.
- [20] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In PODS, 2004.
- [21] K. Wang, B. C. M. Fung, and P. S. Yu. Template-based privacy preservation in classification problems. In IEEE ICDM, pages 466–473, November 2005.
- [22] Oliveira, S.R.M., Zaiane, O.R.: Privacy preservation when sharing data for clustering. In: Proc. Workshop on Secure Data Management in a Connected World.(2004) 67–82.
- [23] Dakshi Agrawal and Charu C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. ACM, 2001.
- [24] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.
- [25] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee. Toward Privacy in Public Databases. In Proceedings of the Theory of Cryptography Conference, pages 363–385, 2005.

A Decision Support System Based on Data Mining for Pediatric Cardiology Diagnosis

Paulo J. L. Adeodato^{1,2}, Tarcisio B. Gurgel¹, and Sandra S. Mattos³

¹Center for Informatics, Federal University of Pernambuco, Recife, PE, Brazil

²NeuroTech Ltd., Recife, PE, Brazil

³Maternal-Fetal Cardiac Unit (UCMF), Royal Portuguese Hospital, Recife, PE, Brazil

Abstract - *This paper presents a decision support system based on data mining for helping diagnose cardiac diseases on infants. The Maternal-Fetal Cardiac Unit of the Royal Portuguese Hospital of Recife (Brazil) has provided the data and characterized the problem with the objective of identifying patients who need surgical treatment for priority care. The solution development involved databases integration, data cleaning and preprocessing and data transformation to embed medical experts' knowledge. Decision trees and induction of classification rules extracted the knowledge in explicit human language while the multi-layer perceptron neural network estimated the need of each patient for surgical treatment. The results in technical performance metrics were fine and the staff cardiologists assessed that the knowledge expressed by the induced rules was consistent with the medical understanding and that the success rate in surgical risk estimation recommended the solution as a valuable tool for supporting the patients diagnoses.*

Keywords: Automatic medical diagnoses, Decision support systems, Data mining.

1 Introduction

Being Medicine complex and not yet well understood in all its aspects, it has a lot to benefit from other technical fields in which the concepts of correct functioning and better performance are theoretically established [1]. Data Mining as an application of Artificial Intelligence together with several other such fields has, in recent years, contributed in creating medical tools that have caused a strong impact in the delivery of health services through improvements in the diagnosis of diseases and reduction of the time pressure on doctors and nurses [2].

The computerization of the various activities related to health has increasingly generated huge volumes of data [3]. The use of this material, however, is far below its potential. Evidences about the data are obtained, in most cases, by mere statistics, but knowledge intrinsic to data remains hidden in them. A real mine for Data Mining.

Cardiovascular diseases are among those which most lead to patient's death in Brazil and in the world [4]. Despite being a serious problem, the majority of congenital heart diseases can have good prognosis if diagnosed and treated early [5]. The Maternal-Fetal Cardiac Unit (UCMF) is a hospital unit specialized in the diagnosis and treatment of fetuses and children carrying heart diseases. The data base of medical consultations and examinations of echocardiograms in this unit had not yet been systematically explored to support decision-making. The data base had not even been organized for such purpose.

The work reported here aimed at applying data mining for developing a decision support system in pediatric cardiology care at UCMF. The system has used Artificial Intelligence techniques for expliciting knowledge (decision trees and classification rules), and Artificial Neural Networks for estimating risks.

The application of data mining tools in cardiology has previously been made but with a different focus from this. Podgorelec *et al.* [6] have used data from pediatric cardiology to evaluate a new algorithm for the induction of classification rules. Matousek and Aubrecht [7] focused on unifying heterogeneous data sources for the construction of a consolidated cardiology database where data mining could be more efficient. This work focuses on the decision support system for diagnosing heart diseases in infants.

This paper is organized in four more sections. Section 2 describes the decision process at UCMF. Section 3 presents the databases and their contents. Section 4 describes the techniques of extraction of knowledge and the results of their application here. Section 5 presents final remarks on the work carried out and its potential application in other fields of medical diagnosis.

2 Decision Making

At UCMF nowadays, diagnoses are made without the aid of decision supporting tools. The doctor evaluates the patient's general state based only on their expertise and either

diagnoses the illness or prioritizes the care for this potentially more critical patient.

The decision support system proposed in this work encompasses two main objectives described below and illustrated in Fig. 1:

- the discovery of new knowledge, which would feed a knowledge base for assisting the doctor in the decision-making process;
- the construction of classifiers, which could be used, for example, as a second opinion, that for either confirming the doctor's opinion or suggesting a reassessment of the case.

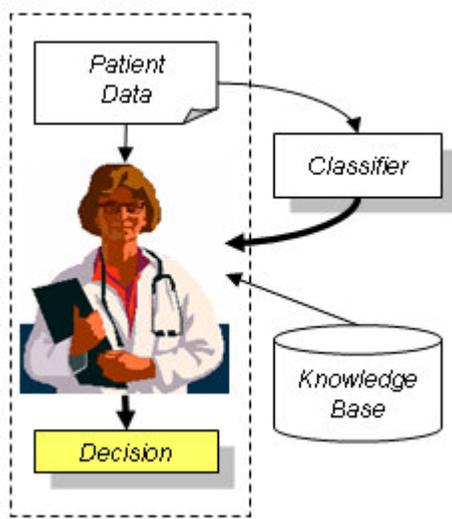


Fig. 1. Decision making at UCMF, nowadays (inside the dashed rectangle) and the supported decision (with knowledge base and risk estimator).

3 Data

The primary database at UCMF is the clinic database formed by several relational tables from which only two were used in this work: “Patients”, containing all patients’ registration data, and “Consultations” containing all information on their consultations and results of their examinations. This database consists mostly of categorical fields.

Another source of data available is the set of spreadsheets of echocardiograms; one file per examination with no direct link to the clinic database. Thus this data source had to be converted to a relational table. The spreadsheets fields consist, mainly, of the measurements and observations obtained from the two types of bi-dimensional echocardiograms: M mode and Doppler. Heart rate, left ventricle size at diastole and maximum blood speed in the aorta are examples of these numerical measurements. There

are also some categorical fields that indicate either properties of the heart or the integrity of its parts such as the ventricles and venous drainage.

Pre-processing of these databases involved records and fields filtering; recovery non-standardized fields (*e.g.* age, in the consultation database and categorical fields in the echocardiograms database). The most important data transformation was the simplification of “diagnosis” variables to represent only the presence or absence of abnormalities in that heart feature. The patient’s initial diagnosis was the target variable for the decision support system thus requiring a more detailed treatment. The UCMF was interested in two types of decisions as illustrated in Table 1. One decision process involved 3 classes with 3 different associated actions: surgical patient, ill patient (non-surgical) and healthy patient. The other decision process which is the focus of this paper was a binary decision process for identifying if the patient needed surgery.

TABLE 1
INITIAL DIAGNOSIS CLASSES FROM THE CONSULTATION DATABASE AND THEIR DERIVED CLASSES.

7 original classes	3 derived classes	2 derived classes
Simple Congenital Heart Disease	Surgical anomaly	Surgical
Complex Congenital Heart Disease		
Abnormal (Other)	Clinical anomaly	Non-surgical
Acquired Heart Disease		
Risk Factor for Coronary Heart Disease		
Arrhythmia		
Normal	Normal	

Among the difficulties in recovering data, two deserve mentioning. First, there were plenty of records for which the link between the fetus (recorded by their mother’s name) and the child name after they were born had been lost. Second, there was no key for linking the databases’ records for a large number of patients.

After data transformation, there were around 4,000 records of useful data collected along 30 months at UCMF.

4 Modeling Techniques

This work followed approaches previously used by experts in the application of artificial intelligence techniques to a variety of medical fields for improving decision-making [3], [8], [9]. Decision Trees, [10] and Classification Rules [11] techniques were used for knowledge extraction in explicit language. Multi-Layer Perceptron (MLP) Artificial Neural Networks [12] were used for risk estimation. MLP neural

networks were chosen for their high performance in classification tasks, classification rules were chosen for explaining the MLP decisions and for forming the knowledge base, and the decision trees were chosen for being an alternative way to human decision making, easily explicable.

4.1 Decision trees

The algorithm J4.8 [14] was used for generating the decision trees. It is a version of the algorithm C4.5 [13] implemented in the software package Weka [14]. The decision trees were generated with KNIME (<http://www.knime.org>), extensible software for data mining that allows the use of Weka implementations.

The decision trees were generated separately for each database. The expert found the trees generated from the consultations data base consistent with reality. Fig. 2 illustrates the decision tree generated for the echocardiograms data base. This tree shows the integrity of intra-cardiac flow as the most relevant information for classifying the state of a patient as Normal or Abnormal. For being structural lesions, in most cases, congenital heart diseases lead to disturbances in the pattern of intra-cardiac flow. Despite being logical, this reasoning is not the approach taken by the heart expert for infant diagnosis; it is only usual for diagnosing adults.

In Fig. 2, the angle of the circular sector represents the confidence of the selection (ratio between the first and second number) while the level of the vertical bar represents the selection of the sample (ratio of the second number and size of the sample).

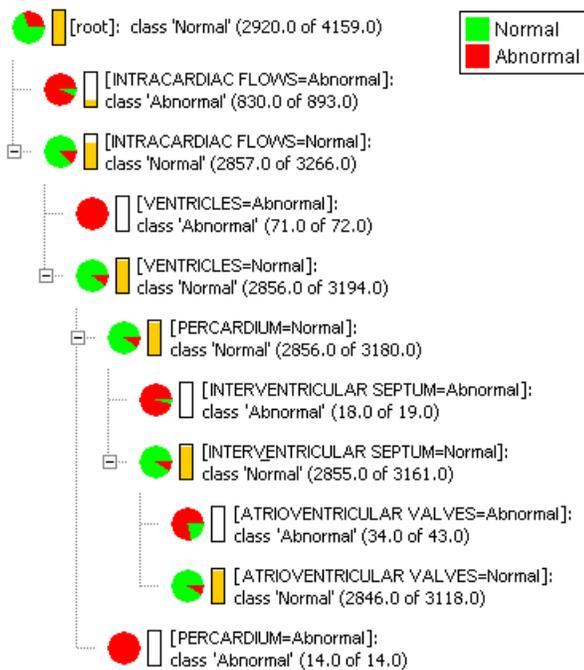
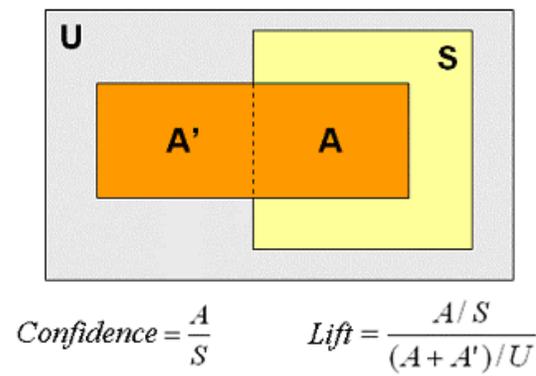


Fig. 2. Echocardiogram decision tree with categorical fields.

4.2 Classification rules

The classification rules were induced by the *a priori* algorithm, one of the most used algorithms for associative rule induction [11] [15]. The rules were then filtered for preserving only those which had the target variable as consequent which characterize the classification rules. Further filtering discarded rules with statistically insignificant *lift*.

The relevance of the rules was assessed by two metrics: *lift* and *confidence*. The term “*selection*” refers to the hypercube defined by the rule conditions (combined clauses) upon which the rule is evaluated. It is important for verifying the statistical validity of the rule and the fraction of the database it affects. “*Confidence*” measures the quality of the rule by the relative frequency of examples of the desired class among those selected. The “*lift*” is the ratio of the relative frequencies of the desired class examples in the *selection* and in the population. This metrics indicates the increase of concentration of the desired class examples in the selected niche compared to the population average. Statistical significance of the *lift* is measured based on the confidence interval of the relative frequency measured as a proportion [16], with $\alpha=0.9$ confidence. Fig. 3 illustrates these concepts.



U = Patient population
 S = Selected patients
 A = Selected target patients
 A' = Non-selected target patients

Fig. 3. Metrics for classification rules' quality assessment.

The classification rules are presented in tables with these concepts summarized for niches defined with either a single clause or the combination (intersection) of two clauses. The most significant results obtained from these rules were:

1. Observed on their own, the presence of typical heart disease symptoms increase the risk of potentially surgical cardiopathy;
2. The presence of the murmur is a risk factor for the existence of a surgical cardiopathy;

- Murmur on its own has moderate relevance for decision making;

The first result is obtained from the joint interpretation of the rules in Table 2. The variable "Other" indicates the presence of any other symptoms that are considered non-relevant to cardiology. According to the rules, the absence of such symptoms increases the risk of the patient having a more serious heart disease. This means that if the patient presents only the characteristic symptoms of heart diseases and no other, it is more likely that he has a more serious heart disease, which usually requires a surgical procedure. This interesting observation is plausible, since the presence of these other symptoms is evidence of the existence of other non-cardiac diseases of, maybe, lower risk.

TABLE 2
CLASSIFICATION RULES FOR THE FIRST CASE REPORTED IN THE CONSULTATIONS DATA BASE.

Condition	Confidence	Lift	Selection
Asymptomatic = No	33.7%	1.98	23%
Asymptomatic = No and Other = No	41.9%	2.46	17%

The second result indicates the presence of murmur as an increasing risk factor for the existence of surgically treatable diseases. This conclusion comes from the six rules in Table 3, grouped in pairs of a single-clause rule followed by that combined with the presence of murmur as a second clause.

TABLE 3
CLASSIFICATION RULES FOR THE SECOND CASE REPORTED IN THE CONSULTATIONS DATA BASE.

Condition	Confidence	Lift	Selection
Cause1 = Previous Cardiopathy	79.5%	4.67	11%
Cause1 = Previous Cardiopathy and Murmur = Yes	87.7%	5.16	6%
Cyanosis = Yes	70.1%	4.12	2%
Cyanosis = Yes and Murmur = Yes	80.4%	4.73	1%
Forwarded by = Cardiologist	58.3%	3.42	8%
Forwarded by = Cardiologist and Murmur = Yes	73.8%	4.34	3%

The third result obtained from the classification rules induced from the consultations data base was also related to the presence of murmur. Considered a common symptom among patients of the clinic, murmur was not expected by the heart specialist to be, alone,

valuable for decision making. In general, a large number of children presenting either harmless or physiological murmurs are a cause of frequent referrals for evaluating the infant's heart. The rule in Table 4, however, is the sixth most important among the single clause rules induced from the consultations data base. This rule presented interesting knowledge to the specialist team.

TABLE 4
CLASSIFICATION RULES FOR THE THIRD CASE REPORTED IN THE CONSULTATIONS DATA BASE.

Condition	Confidence	Lift	Selection
Murmur = Yes	37.0%	2.17	27%

The classification rules for echocardiograms data base are also organized in the same format as illustrated in Table 5. It lists only the single clause rules with high lift. Differently from the decision tree where the attribute "intra-cardiac flows" was the most relevant variable for decision making, the classification rule with the same attribute does not present the highest lift. However, the niche defined by this attribute (*selection*) is at least three times larger than those which produce higher lifts, thus turning it an important attribute for decision-making.

TABLE 5
CLASSIFICATION RULES WITH A SINGLE CLAUSE AND HIGHEST LIFTS IN THE ECHOCARDIOGRAMS DATA BASE

Condition	Confidence	Lift	Selection
Ventricles = Abnormal	99.6%	3.34	7%
Interventricular Septum = Abnormal	99.6%	3.34	6%
Atrioventricular Valves = Abnormal	95.3%	3.20	6%
Intracardiac Flows = Abnormal	93.0%	3.12	21%
Interatrial Septum = Abnormal	73.2%	2.46	12%

4.3 Artificial neural networks

Two classifiers were used for supporting the cardiologist's decision in two different scenarios: in scenario 1, for assessing the heart health of the patient in its consultation; in scenario 2, for evaluating the cardiac health of the patient after the results of echocardiograms. The response of the neural network for scenario 1 indicates whether or not the patient has a critical profile, one that would need an operation. In scenario 2, the response of the neural network indicates whether or not the patient's heart is in normal state. The data bases used were, for scenario 1, the consultations while, for scenario 2, the echocardiograms.

The classifiers were constructed with Multi-Layer Perceptron (MLP) artificial neural networks [12], for their high performance in binary classification problems [17], and robustness [18]. The MLP networks had three neurons in the hidden layer and two neurons in the output layer merged in a single score $y = (y_1 + y_2 + 1) / 2$. Training has used the algorithm error back-propagation, with a learning rate of 0.001. Each data base used was partitioned into three sets: training data, with 50% of examples for adjusting the weights; validation data with 25% of the records for stopping the training, and test data with the remaining 25% of the examples, for evaluation of performance of the network.

The performance achieved with the neural networks for each case was considered good by the specialist in the field. The main performance metrics used were the statistical test of Kolmogorov-Smirnov (KS2) [19] and measures of Error Type I and Error Type II, which in this work represent, respectively, the percentage of normal individuals classified as abnormal, and the percentage of abnormal individuals classified as normal. In general, in Medicine, the metrics of sensitivity and specificity are used for analyzing the performance of diagnoses [20]. These measures are related to measures of error by the formulas: sensitivity = 1 - Error Type II, and specificity = 1 - Error Type I.

Tables 6 and 7 show these various performance measures for decisions thresholded at the point of maximum KS2, for scenarios 1 and 2 respectively.

TABLE 6

MLP NEURAL NETWORK PERFORMANCE ASSESSMENT, ON THE CONSULTATION DATA BASE, THRESHOLDED AT THE MAXIMUM KS2.

		Classified	
		Negative	Positive
Actual	Negative	72.9%	10.1%
	Positive	4.1%	12.9%
Error I	Error II	MaxKS2	
12.1%	24.1%	0.641	

TABLE 7

MLP NEURAL NETWORK PERFORMANCE ASSESSMENT, ON THE ECHOCARDIOGRAMS DATA BASE, THRESHOLDED AT THE MAXIMUM KS2.

		Classified	
		Negative	Positive
Actual	Negative	66.6%	3.6%
	Positive	5.7%	24.1%
Error I	Error II	MaxKS2	
5.1%	19.0%	0.761	

ROC curves (Receiver Operating Characteristic) [21] graphically show the tradeoff between the Type I error and sensitivity, or, respectively, the rate of false positives and the

rate of true positives [3]. False positives are measured along the abscissas axis, while the true positives, the along vertical axis. The ROC curves can be used by the expert in the field to assist defining the best decision threshold as presented in a medical internet site with user controlled threshold and illustrated examples in health domain [22].

Fig. 4 illustrates the curves obtained for the neural networks in scenarios 1 and 2. The areas under these ROC curves (AUC_ROC) are equal to 0.91, for scenario 1, and 0.94 for the scenario 2, close to the maximum 1.00.

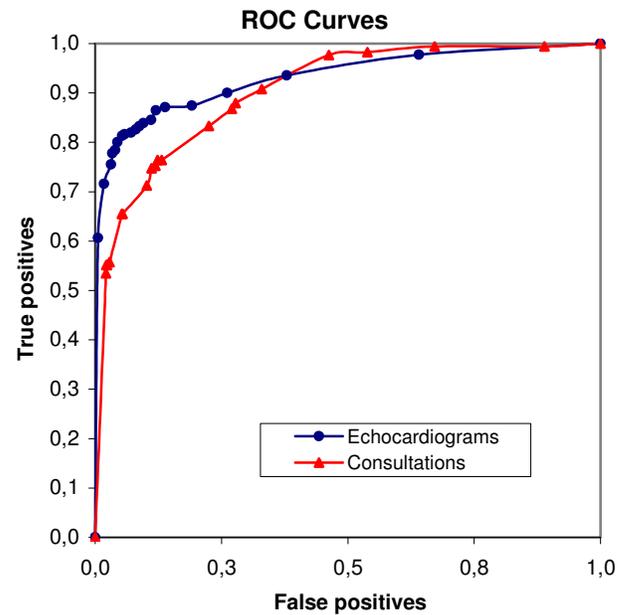


Fig. 4. ROC Curves for the MLP responses on both scenarios: consultations and echocardiograms.

5 Concluding Remarks

This paper has presented a data mining based decision support system for diagnosing infants' heart diseases at the UCMF hospital unit. It has used 3 techniques for knowledge extraction.

The knowledge represented by the decision trees' structure and the classification rules induced confirmed the specialists expertise and, in some cases, attracted their attention for a different reasoning for reaching the diagnostic. The cardiologists also found the results obtained by the classifiers so reliable that they have stated they will consider the system's response as a second opinion on the medical diagnosis. In case of different diagnoses, the patient would be reassessed by the doctor, thus reducing the chance of wrong decisions. The high accuracy of the neural network reinforces its application as an important tool for the estimation of the need of surgical intervention.

The quality of the solution developed and its acceptance by heart medical specialists showed the feasibility of using this tool for diagnostic decision support at the Maternal-Fetal Cardiac Unit (UCMF). Currently, the interface for data entry is being refined for minimizing input errors during the form filling procedure. The lack of data standards is one of the hardest obstacles for automating the work presented here in its application to the early stages of the clinical examination.

6 References

- [1] A. Horn. "AI in medicine on its way from knowledge-intensive to data-intensive systems"; *Artificial Intelligence in Medicine*, Vol. 23, Issue 1, pp. 5—12, 2001.
- [2] R.B. Rao, R. Rosales, S. Niculescu, S. Krishnan, L. Bogoni, X.S. Zhou, and B. Krishnapuram. "Mining Medical Records for Computer Aided Diagnosis"; In: *Demo in KDD conference*, Philadelphia, 2006.
- [3] N. Lavrac. "Data Mining in Medicine: Selected Techniques and Applications". *Proceedings of the Second International Conference on the Practical Applications of Knowledge Discovery and Data Mining (PADD'98)*, London, pp. 11—31, 1998.
- [4] World Health Organization. "The world health report 2003". Available: <http://www.who.int/whr/2003/en/> [Accessed: Dec 04, 2006].
- [5] A.J. Marelli, A.S. Mackie, R. Ionescu-Ittu, E. Rahme, and L. Pilote. "Congenital heart disease in the general population: changing prevalence and age distribution"; *Circulation*, Vol. 115, Issue 2, pp. 167—172, 2007.
- [6] V. Podgorelec, P. Kokol, M.M. Stiglic, M. Heričko, and I. Rozman. "Knowledge discovery with classification rules in a cardiovascular dataset"; *Computer Methods and Programs in Biomedicine*, Vol. 80, pp. 39—49, 2005.
- [7] K. Matousek, and P. Aubrecht. "Data Modelling and Pre-processing for Efficient Data Mining in Cardiology"; *International Special Topics Conference on Information Technology in Biomedicine*, CD-ROM, Piscataway, 2006.
- [8] C. Ordonez, C.A. Santana, and L. Braal. "Discovering Interesting Association Rules in Medical Data"; *ACM DMKD Workshop*, pp. 78—85, 2000.
- [9] A. Silva, P. Cortez, M. F. Santos, L. Gomes, and J. Neves. "Multiple Organ Failure Diagnosis Using Adverse Events and Neural Networks". *Proceedings of the International Conference on Enterprise Information Systems IV*, Porto, 2004, Springer, pp. 127—134, 2006.
- [10] Tom M. Mitchell. "Machine Learning". McGraw Hill, 1997.
- [11] Jiawei Han, and Micheline Kamber, "Data Mining: Concepts and Techniques". Morgan Kaufmann Publishers, 2006.
- [12] Simon Haykin. "Neural Networks: A Comprehensive Foundation". Prentice Hall, 1998.
- [13] J.R. Quinlan. "Improved Use of Continuous Attributes in C4.5"; *Journal of Artificial Intelligence Research*, Vol. 4, pp. 77—90, 1996.
- [14] Ian H. Witten, and Eibe Frank. "Data Mining: Practical machine learning tools and techniques". Morgan Kaufmann, 2005.
- [15] David J. Hand, H. Mannila, and P. Smyth. "Principles of Data Mining". The MIT Press, 2004.
- [16] Raj Jain. "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling". Wiley-Interscience, 1991.
- [17] P.J.L. Adeodato, G.C. Vasconcelos, A.L. Arnaud, R.F. Santos, R.C.L.V. Cunha, and D.S.M.P. Monteiro. "Neural Networks vs. Logistic Regression: a Comparative Study on a Large Data Set". *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)*, Cambridge, UK, 2004.
- [18] M.Y. Kiang. "A comparative assessment of classification methods"; *Decision Support Systems*, Vol. 35, pp. 445—454, 2003.
- [19] W.J. Conover. "Practical Nonparametric Statistics". Wiley, 1999.
- [20] K.J. Cios, and G.W. Moore. "Uniqueness of medical data mining"; *Artificial Intelligence in Medicine*, Vol. 26, Issue 1-2, pp. 1—22, 2002.
- [21] T. Fawcett. "An introduction to ROC analysis," *Pattern Recognition Letters*, Vol. 27, Issue 8, pp. 861—874, 2006.
- [22] The Anaesthetist. Last accessed on April 2009: <http://www.anaesthetist.com/mnm/stats/roc/>.

Name Entity Recognition and Classification in Medical Text Documents

Yinghao Huang
Computer and Information Science
University of Michigan – Dearborn
Dearborn, MI 48128
yinghaoh85@gmail.com

Yi Lu Murphey
Electrical and Computer Engineering
University of Michigan – Dearborn
Dearborn, MI 48128
yilu@umich.edu

Naeem Seliya
Computer and Information Science
University of Michigan – Dearborn
Dearborn, MI 48128
nseliya@umich.edu

Roy B. Friedenthal
Central Orthopedics
820 S. White Horse Pike
Hammonton, NJ 08037
roy@comcast.net

Abstract — This paper presents a name entity recognition and classification system (called MD_NER_NCL) for medical text document processing and analysis. The system involves a document segmentation process, the use of the OpenNLP application, and a statistical reasoning process. We present an innovative segmentation algorithm, called *HBE segmentation*, to segment a medical text document into the **Heading, Body and Ending** parts. The knowledge base used for the statistical reasoning process contains three entity lists: **people name prefix list, people name suffix list, and false positive prefix list**. Each prefix and suffix on these lists is accompanied by a risk factor derived from a training data set. Our empirical case study results show that the proposed system, MD_NER_NCL, makes a significant improvement over OpenNLP for people name entity detection.

Keywords: named entity recognition, data encryption, medical informatics, document segmentation, natural language processing

I. INTRODUCTION

Human medical data is one of the most rewarding, yet very difficult, data to mine and analyze. According to Cios and Moore [1], about three-quarter billions of people living in North America, Europe, and Asia have at least some of their medical information collected in an electronic form. These documents contain rich knowledge that is useful for medical research and clinical study. On the other hand, there are ethical, legal, and social constraints imposed on medical data collection, distribution, and analysis.

This research focuses on the medical documents written by physicians. Such documents often contain large amounts of information about patients, such as medical records, medical history, symptoms, diagnoses, etc. Much of this information is essential for making medical and/or administrative decisions based on patients' physical conditions. However, these medical documents always contain names of physicians, patients, provider organizations, etc. In order to protect the

privacy of patients, physicians, and other personnel and organization information, it is important to detect these name entities and encrypt them before the documents are being distributed for further processing. The limitation of data privacy and other issues concerning medical data has very much limited data mining research in the medical domain.

Name entity recognition is a preprocess in many information retrieval and natural language processing (NLP) applications for a variety of reasons including privacy protection and correlating name entities with specific groups of people. Various techniques of name entity recognition (NER) have been developed.

McCallum and Li [4] introduced Conditional Random Fields for NER. Chieu and Ng [18] presented a Maximum Entropy-based NER, while Bender and Josef in [19] also describe a system that applies the Maximum Entropy Model of NER. Mayfield and McNamee [14] presented an approach to NER that uses Support Vector Machine (SVM). Florian and Ittycheriah investigated the combination of a set of different statistic NER, including the transformation-based learning classifier, a Hidden Markov Model (HMM) classifier, a robust risk minimization classifier, and a Maxent classifier [17].

However, most of the NER related works are applied to text which are well organized, generally error free, and with a relatively formal format. Although Chang and Sung [7] explored Maxent and CRF in NER for processing informal text including email message and newsgroups postings, in which a list of specific features is retrieved from the text data and combined to find the most efficient ones, there is still not much work about NER for informally represented text.

The name entity recognition problem in this application domain serves as a preprocessing stage for a data encryption process. The basic requirements for the encryption process are as follows:

- All names, persons or organizations, should be encrypted.

- The same names should be encrypted with the same encoding.
- Different names should be encrypted with different encoding.
- The encryption should indicate different categories of names, so knowledge can be accurately discovered at later processes. Table I shows the major categories of names that need to be identified and the pre-code for each category. There are seven categories of names and the date line entity that need to be detected and discriminated.

Table I: Categories of names in medical correspondence documents

Type	Pre-Code
Person: Patient	PP#
Person: Writer	PW#
Person: Receiver	PR#
Person: Doctor	PD#
Person: Other	PO#
Organization: Hospital	OH#
Organization: Other	OO#
Date	DT#

The above encryption requirements dictate the performance requirements of the NER system we are developing, and they are:

- minimizing the missing rate
- minimizing partial recognition of entities
- minimizing false positives
- Classifying recognized names into the pre-code categories

We present a name entity detection and classification system, called MD_NER_NCL. The system consists of a segmentation process, and NER, and a statistical reasoning process to accurately detect and classify name entities. We will show that the proposed algorithm provides much improved recall and precision for name detections in the medical text documents of our case study.

The remainder of this paper is organized as follows: Section II gives a detailed description on the general content and format of the medical documents and the proposed system, MD_NER_NCL, Section III presents the case study and the empirical results, and Section IV concludes the paper and provides some suggestions for future research.

II. PROBLEM DESCRIPTION

In this research, we focus on the medical documents in the form of correspondences between physicians and patients or various third parties such as insurance companies, patients' employers, and attorneys. These documents vary in style and format, and their names vary with many different spellings. Figure 1 shows an example of such a document – the actual

names and organizations have been replaced with fictitious ones for data privacy reasons. Although these documents have large variations in format, they can be described in the general frame work shown in Figure 2. In general, a document file may contain several letters. So the format shown in Figure 2 can be repeated within the same document file multiple times.

We propose a name entity detection and classification system, MD_NER_NCL, for medical document processing. Figure 3 shows the computational steps of the system. Since the header of a document can contain a variety of names, e.g. person names, organization names, street names, city names, etc., it poses a difficult challenge for name detection. In particular, we noticed that OpenNLP often missed names in the header section of a given document. Furthermore, since the sentences in the header are not separated by punctuations as those in the body part, the entire header section can be mistakenly interpreted as a sentence if not processed properly.

May 15, 2002

Traverwood Property and Casualty Companies
 26010 White Fox Road, P.O. Box 1500
 Monseer, Michigan 12891
 ...

ATTN: Terry Mjollokny
 Claims Representative

RE: Marian Jerr
 FILE #: 000 A 12 11 85-1/333

Dear Ms. Mjollokny:

HPI: Ms. Jerr describes sustaining injury ... on August 17, 1995
 She states that she went to City Center Hospital She came under the care of Dr. Greene.... She saw Dr. Andrews for her knees

PMH:
 ...

MEDICAL RECORDS:

Report of Dr. Richard of June 11, 1996 indicates a past medical history of prior motor-vehicle-accident was noted. ... A note of March 16, 1995 indicates that Marian Jerr was treated for injuries sustained in a motor-vehicle-accident on March 15, 1995. These appeared to be the notes of Dr. John Doe.

ASSESSMENT: At the present time there are no objective physical findings to indicate ongoing disability. It is my opinion that Ms. Kerr has recovered from any injuries sustained

 It may be of value to obtain more detailed records from Dr. Gilbert insofar as pre-existing complaints are concerned.

Sincerely yours,
 James K. Finley, M.D.

Figure 1: An example of medical correspondence

Date Line (e.g. August 13, 1998)

Address Lines (e.g. CXX PXX and CXXX Companies,
XXX WXX HXX Road, P.O Box XXXX,
VXX, Michigan, 00000)

Receiver Lines (e.g. ATTN: MXX MXXX
Claims Representative)

Patient Lines (e.g. RE: KXX KXXX
FILE#: 000 A 00 00 00-0/000)

Body Part (e.g. Line begins with "Dear Mr./Mrs. MXXX")

Writer Lines (e.g. Sincerely yours, RXX F. FXX, M.D.)

Figure 2: A general format of medical documents

A document segmentation algorithm, named HBE, is developed to segment a document file (of our case study) into sequences (parts) of Heading, Body, and Ending parts. A Heading part includes the Date Line and the name and address of the received entity. The Body part immediately follows the Heading and ends before the solution line. The Ending part includes the solution line and the sender's name.

The HBE algorithm is developed based on a line projection strategy. The line projection of a document, *D*, is defined as the histogram of characters for line entities in the document *D*. We define a line entity as all characters between two line breaks. Figure 4 shows an example of the line projection of a document. In the Heading and Ending parts, a line entity is actually a line in the common sense. In the Body parts, a line entity is an entire paragraph. Therefore the line projection has the number of characters ranging from less than 10 to thousands.

Qualitatively speaking, the Heading and Ending parts contain very short line entities, while the Body parts contain mostly long line entities. The HBE segmentation algorithm (shown below) is designed to find the beginning and ending of each part. In order to set a quantitative threshold to separate the line entities in the Body parts from the Heading and Ending parts, we measured the number of characters in a full line in a word document of commonly used font sizes. The number of characters in a full line in a word document with font size equal 9 is about 119, and for font size 12 it is about 86. Since there are many different font types, it is safe to set the maximum length of a full line in a document to be less than 150 characters. We denote this quantity as

max_length. If the length of a line entity is greater than max_length, the line entity is a paragraph that contains multiple document lines. We developed the following algorithm for segmenting a document into a sequence of Heading, Body and Ending parts.

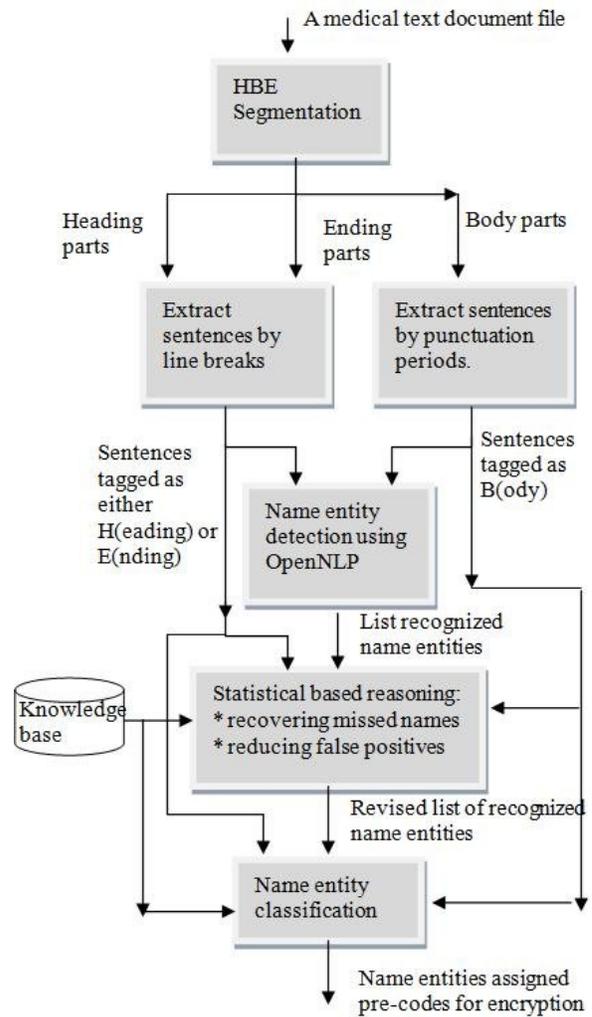


Figure 3: Computational steps in MD_NER_NCL.

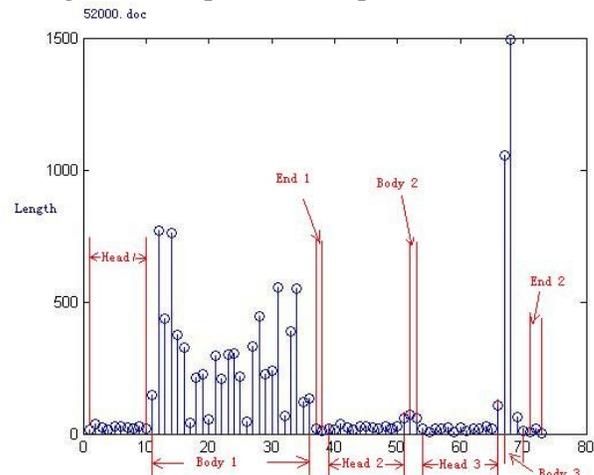


Figure 4: Illustration of line projection of a document

HBE Segmentation Algorithm

1. Read in the content of a word document.
2. Set up a threshold of the maximum number of characters in a line for the document, and denote it as `max_length`.
3. Segment the text into "line entities" by line breaks
4. Calculate the number of characters in each "line entities".
5. Search the line entities according to their order in the document.
6. If the current line entity has no period at the end and its length is less than 50% of `max_length`, it is a heading line.
 - i. If the previous line entity is also a heading line, add the current line to the current Heading.
 - ii. If the previous line entity is not a heading line, generate a new Heading and mark the current line entity as the beginning of the new heading.
 - iii. Make the next line entity in the document as the current line entity and goto Step 6.
7. If the current line entity ends with a comma and has the keyword "sincerely" and its length is less than 50% of the `max_length`, it is an ending line entity.
 - i. Mark the line entity as the beginning line of a new ending.
 - ii. Mark the last line entity as the closing line of the current body part.
 - iii. Make the next line entity in the document as the current line entity and add it to the Ending part. Note the line after the "Sincerely" line is usually the name and title of the sender.
 - iv. Make the next line entity in the document as the current line entity and goto Step 6.
8. If the current line entity does not meet the conditions stated in step 6 and 7, it belongs to the Body part
 - i. Add it to the current body part.
 - ii. Make the next line entity in the document as the current line entity and goto step 6.

The line projection provides rich knowledge about the Heading, Body and Ending parts in the medical correspondence documents. Since the line projection in Figure 4 shows multiple sequences of "head", "body" and "end", it implies that the document contains several letters.

After the segmentation process, the sentence entities in the Heading and Ending parts are extracted by searching for line breaks. For example, in Figure 1 the Heading part starts from the dateline and ends at the line above the "Dear..." line. Each line in this section ends with a line break; therefore, it is a sentence entity. The sentence entities in the Body part are those between two periods.

The name entity detection process uses the OpenNLP Maxent package to detect name entities in the sentence entities generated by the HBE segmentation algorithm. The OpenNLP was implemented based on the Generalized Iterative Scaling and Maximum Entropy model [18]. It generates MUC-style

name entities. The OpenNLP software provides a number of MUC-style named entity finder models.

The original name finder interface provided by OpenNLP package only allows user to type in text into console. The way it processes the text and detects name entity has the following steps.

1. The text is sent to the SentenceDetector to be divided into sentences. A blank line is treated as a paragraph boundary.
2. The text of one sentence was split into tokens based on whitespaces in that line.
3. Generate name tags for this sentence based on tokens and outcome maps generated from previous lines.
4. Find name entities and add labels to them based on three kinds of name tags "start", "other", "cont" which represent the position of the token within the name entity.
5. Repeat from Step 2 until all the sentences are processed.

Notice that we need to extract text from medical documents, the OpenNLP did not provide ways to handle with such problem. Hence, the code has been changed to receive the retrieved text from word documents, while the text is originally split into sentences by reasonable "sentence ends" predicted by the SentenceDetector.

However, in processing text of the medical documents which has "Header", "Body" and "End" part in an "email format", the way OpenNLP process the text produces a poor detection rate and high false alarm in detecting name entities, which will also be discussed in detail in part III. Moreover, by using this approach the entire header section will be mistakably interpreted into one sentence as stated in part II. The possible reason is that the SentenceDetector of OpenNLP is also used a Maxent model that is trained on English data from Wall Street Journal text. To that trained model, the line break after each short line in the "Header" parts may not be considered as reasonable end of a sentence based on the training data.

As a result, alternative approaches are required in order to improve the OpenNLP performance in processing text with specific format. The HBE segmentation algorithm appears to be one of the possible solutions. Furthermore, we could also benefit from the segmentation approach if we want to retrieve specific content from the documents and conduct more data mining research work on it in the future. Another possible approach is to train another SentenceDetector using our medical text data as training data. It is also part of the future works together with training our own people and organization name entity detection models using medical text data by adding more specific features.

Name entity classification is conducted based on the name entity detection results. Patients' names are identified in the Heading part, and often occur with prefix "RE". The receiver's name is identified in the Heading part, and the receiver's name is identified in the Ending part. Doctors' names are identified if they have prefixes: "Dr." or "Drs." A name entity containing the word "Hospital" or "Medical Center" is assigned as the class of hospital names. The name entities often occur with

prefixes such as "Mr.", "Mrs.", "Ms.", and "Miss." The name entities not in the above classes are classified as organization names. The dateline is detected by searching each line in the Heading part for the word that matches one of the twelve names of months in a year.

The statistical reasoning process involves a model training to recognize more people name entities and reduce false positives. It uses the knowledge base that contains the three lists generated through the following training process. The training process relies on a training data set that has all name entities labeled. The first list is the true people name prefix list, PNP_list. For every true people name entity in the training data set, we extract the prefix of the name from the document and add it into the PNP_list. Then we calculate the risk factor of each prefix on the list by the following formula:

$$\alpha_i = \frac{nf_i}{np_i + nf_i}$$

where nf_i is the number of false name entities after the i^{th} prefix, and np_i is the number of true name entities after the i^{th} prefix. Similarly, we generate the true people name suffix list, PNS_list, from the training data set and calculate the risk factors using the same formula above. The third list is a false positive list, FP_list, which is generated as follows. We apply the segmentation algorithm followed by the OpenNLP process to the training data set, and all the false positives in the training data are put in the FP_list. For each false positive, we calculate the risk factor for the i^{th} prefix on the FP_list as follows:

$$\beta_i = \frac{np_i}{np_i + nf_i}$$

These prefix and suffix lists, along with the risk factors, are stored in the knowledge base as shown in Figure 3. The statistical reasoning process has the following computational steps.

1. For every prefix, p , in the true people name prefix list PNP_list, if its risk factor is low, search the test data text for p , and extract and label the name entity after p as a people name entity.
2. For every suffix, s , in the true people name suffix list PNS_list, if its risk factor is low, search the test data text for s , and extract and label the name entity before s as a people name entity.
3. For every false positive prefix, fp , in the FP_list, if any name entity detected by the system has fp as its prefix,

then this is a false positive; hence, remove it from the name entity list.

III. CASE STUDY

We used a set of 100 medical document files that have been labeled manually with different name entity classes. These documents represent Independent Medical Examination reports of patients with orthopedic related ailments. Each document also contains several correspondences. We decomposed these document files into training and test data sets. The training data set contains 708 people names and 53 organization names. The test set contains 894 people names and 53 organization names.

The training data were used to generate the statistical information of the prefixes and suffixes, which is shown in the following Table II. In order to evaluate our system, three experiments were conducted. In the first experiment, we applied the OpenNLP process to the entire test data without any preprocessing. Those results are summarized in Table III. The false positive is very high, while the precision for people name entities is about 45.7% and 62% for organization name entities. The recall is about 71% for people name entities and 76% for organization name entities. In addition, 76 people name entities and 15 organization name entities were partially recognized.

In the second experiment, we applied the segmentation algorithm followed by the OpenNLP process to the test data, and those results are summarized in Table IV. The segmentation algorithm has helped reduce false alarms significantly, and has also increased the name entity recognition accuracy. As a result, the precision has increased to 97.5% and the recall to 74.7% for people name entity.

In the third experiment we applied the proposed MD_NER_NCL system, shown in Figure 3, to the test data, and those results are shown in Table V. The precision on people name entities recognition has increased to 97.76% and the recall to 83.69%. The statistical reasoning process has picked 75 more people name entities without generating any additional false positives. The recognition on the organization name entity has not improved, since the statistical reasoning process only improves the recognition of people name entities.

Table II: Prefix List

	Prefix content	Frequency		
Truth Positive Prefixes	Attention:	4		
	ATTN:	55		
	Dr.	397		
	Drs.	1		
	Mr.	107		
	Ms.	121		
	RE:	39		
			Document name	False Positive Entity
False Positive Prefixes	PMH:	2	51722.doc	PMH: History of ...
	S.	2	52000.doc	S. White Horse Pike
	SYMPTOMS:	1	52095.doc	SYMPTOMS: Currently
	Mt.	1	51897.doc	Mt. Holly, NY

Table III: Performance of OpenNLP on Test Data

	All	Correct	Partial	Missed	False positive	Precision	Recall
People Name	834	593	76	165	704	45.72%	71.1%
Organization	123	93	15	15	57	62%	75.6%

Table IV: Performance of the proposed segmentation algorithm followed by OpenNLP on Test Data

	All	Correct	Partial	Missed	False positive	Precision	Recall
People Name	834	623	58	153	16	97.5%	74.7%
Organizations	123	91	17	15	37	71.1%	74%

Table V: Performance of the proposed MD_NER_NCL on Test Data

	All	Correct	Partial	Missed	False positive	Precision	Recall
People Name	834	698	58	78	16	97.76%	83.69%
Organizations	123	91	17	15	37	71.1%	74%

V. CONCLUSION & FUTURE WORK

This paper presents an effective name entity recognition and classification system, called MD_NER_NCL, for processing medical text documents. The system consists of a document segmentation process (HBE segmentation algorithm), the OpenNLP process, and a statistical reasoning process.

Our empirical case study results show that the proposed system made a significant improvement over the OpenNLP system. More specifically, for the people name entity recognition, MD_NER_NCL yielded 97.76% precision and 83.69% recall, while the OpenNLP process provided 45.7% precision and 71.1% recall for the same case study data.

REFERENCES

- [1] Krzysztof J. Cios and G. William Moore, Uniqueness of Medical Data Mining, *Artificial Intelligence in Medicine*, 26(1-2):1-24, Sept-Oct., 2002.
- [2] Diana Maynard, Valentin Tablan, Cristian Ursu, Hamish Cunningham, and Yorick Wilks, Named Entity Recognition from Diverse Text Types, In *Recent Advances in Natural Language Processing 2001 Conference*, Ed. Tzigov Chark, 2001.
- [3] Hideki Kozima, Text Segmentation Based on Similarity between Words. In *Meeting of the Association for Computational Linguistics*, pages 286-288, 1993.
- [4] Andrew McCallum and Wei Li, Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons, In *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL)*,

2003.

[5] Jay M. Ponte and W. Bruce Croft, Text segmentation by Topic, In *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries*, 1997. URL: <http://citeseer.ist.psu.edu/105763.html>.

[6] Gerard Salton, Amit Singhal, Chris Buckley, Mandar Mitra, Automatic text decomposition using text segments and text themes, In *UK Conference on Hypertext*, pages 53-65, 1996.

[7] Yu-Shan Chang and Yun-Hsuan Sung, Applying Name Entity Recognition to Informal Text. URL: <http://nlp.stanford.edu/courses/cs224n/2005/SungChang.pdf>

[8] Vangelis Karkaletsis, Constantine D. Spyropoulos, and Georgios, Named Entity Recognition from Greek texts: the GIE Project, In *Advances in Intelligent Systems: Concepts, Tools and Applications*, Ed. S. Tzafestas, pages 131-142, 1999. Kluwer Academic Publishers

[9] GATE: General Architecture for Text Engineering, Natural Language Processing Group, University of Sheffield. URL: <http://gate.ac.uk/>

[10] Vangelis Karkaletsis, Georgios Paliouras, Georgios Petasis, Natasa Manousopoulou, and Constantine D. Spyropoulos, Name-Entity Recognition from Greek and English Texts. *Journal of Intelligent and Robotic Systems*, Volume 26, Issue 2, Pages 123-135, Hingham, MA, October 1999. Kluwer Academic Publishers.

[11] Valentin Tablan, Cristian Ursu, Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Oana Hamza, and Yorick Wilks, Software Architecture for Language Engineering, In *EuroLan 2001*, Dept. of Computer Science, University of Sheffield, 2001. URL: <http://gate.ac.uk/talks/eurolan2001/>

[12] Ross Wilkinson, Effective Retrieval of Structured Documents, In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 311-317, New York, NY, 1994.

[13] James P. Callan, Passage-level evidence in document retrieval, In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302-310, New York, NY, 1994.

[14] James Mayfield, Paul McNamee, and Christine Piatko, Named Entity Recognition using Hundreds of Thousands of Features. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, Edmonton, Canada, 2003.

[15] Carvalho, Victor and William W. Cohen, Learning to extract signature and reply lines from email. In *Proceedings of the Conference on Email and Anti-Spam, Mountain View, California*, 2004.

[16] Einat Minkov, Richard C. Wang, and William W. Cohen, Extracting Personal Names from Emails: Applying Named Entity Recognition to Informal Text, In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 443-450, Morristown, NJ, 2005. Association for Computational Linguistics.

[17] Florian and Ittycheriah, Named Entity Recognition through Classifier Combination. In *Proceedings of CoNLL-2003*, Edmonton, Canada, 2003.

[18] Hai L. Chieu and Hwee T. Ng, Named Entity Recognition: A Maximum Entropy Approach Using Global Information, In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1-7, Morristown, NJ, 2002. Association for Computational Linguistics.

[19] Oliver Bender, Franz Josef Och, and Hermann Ney, Maximum Entropy Models for Named Entity Recognition. <http://www.aclweb.org/anthology-new/W/W03/W03-0420.pdf>

[20] Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel, Nymble: A high-performance learning name-finder, In: *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP-97)*, San Francisco, CA, 1997. Morgan Kaufmann Publishers.

Distributed Frequent Pattern Mining Using Time-Slice Method

Fan Wu, Ya-Han Hu, and Che-Wei Yang

Department of Management Information System, National Chung Cheng University, Chiayi, Taiwan, ROC

Abstract -Mining frequent pattern itemsets in databases is popular in the mining for association rules. There were many methods such as Apriori-like or FPGrowth-like algorithms proposed to find frequent patterns. But both of them cannot be applied nowadays because database grows extremely fast that even the FP-tree cannot entirely fit into main memory. Hence, many parallel and distributed algorithms were later proposed to solve such problems. But they require obtaining infrequent patterns by scanning the disk. Hence, the Tidset method based on Parallel FP-tree algorithm was then proposed. With Tidset instead of scanning database, the number of times of occurrences can be obtained directly. Although Tidset improves the efficiency of information exchange, lots of paper including this one didn't study on how much process time spent on scanning the database in its first scan. This method proposes time-slice method in the stage when frequent itemsets are collected to the central node. Furthermore, in the tree exchange stage, local nodes obtain the frequent items and counts from central node, and obtain infrequent items from other local nodes. In this case, we not only improve the central node's utility rate but also avoid broadcasting transactions cost.

Keywords: Frequent pattern, Tidset, Parallel.

1 Introduction

In 1994, Agrawal, R. and R. Srikant [1] proposed an idea – apriori algorithm. At the start it was proposed to find the hiding information that might bring business. Unexpectedly, it becomes a powerful influence on database discipline. However, some problems exist in apriori algorithm. First, it needs to generate colossal candidate itemsets. Second, the algorithm needs to scan database again and again. Hence, Frequent Pattern tree algorithm is proposed to solve the problems. But both of them cannot be applied nowadays because database grows up extremely fast that even the FP-tree can not entirely fit into main memory. Furthermore, the processing on single computer is costly. Therefore, parallel data mining method was proposed to deal with immense data over the last decade. Some algorithms like Count Distribution algorithm [2] and Fast Distributed Mining algorithm [3] had been proposed for distributed computing. CD algorithm uses broadcast mechanism to overcome distributed communication problems, the other one chooses one of these distributed machines as the central to let it control distributed

communication problems. The main problem is similar since both are apriori-like algorithms that face the repeat scanning colossal candidate problems. On the other hand, Zhou and Yu proposed Tidset based Parallel FP-tree algorithm [4] that adopted the concept of FP-tree. They not only avoided generating colossal candidate sets but also overcame two problems, namely, when database is too large or minimum threshold is too low, the execution time will increase rapidly. Distributed association data mining, if based on FP-tree algorithm has three main steps, namely, scans of database for frequent itemsets and construction of FP-growth and communication among each computer. The time-slice method that we do in scanning database stage, it will be useful in collecting frequent patterns stage, which is our main concept that transmits tree without broadcasting.

TPFP-tree algorithm constructs a matrix to sum each item, called Tidset, which can get local transaction occurrence times directly instead of scanning database again. Tidset improves efficiency of communication, but lots of paper include this one didn't mention how much time they spend on scanning database in their first scan. It's like common sense that could be ignored. However, we can do something more during the first scan of database such that the total scanning database time doesn't increase, but the total time spending on latter data mining will be reduced. Furthermore, lots of parallel distributed algorithms choose broadcast method in exchange tree stage, when more local computers involve for distributed computing, the cost of communication increase as exponent.

FP-tree algorithm scans database once and gets local frequent itemsets without sorting then stores in F, the frequent itemsets order by support descending store as L. Central node collects local frequent itemsets after their FP-tree is constructed. Our method proposes time slice method in collecting frequent itemsets stage, namely, the frequent transaction itemsets in local nodes will be pushed to central node during the time pass. Furthermore, in the tree exchange stage, local nodes obtain others local frequent items counts from central node and obtain local infrequent items from other local nodes instead of broadcasting items to all nodes. Generally, this idea can avoid the broadcast problem that we mentioned before.

2 Related Work

Please use the styles contained in this document for: Title, Abstract, Keywords, Heading 1, Heading 2, Body Text, Equations, References, Figures, and Captions. Do not add any page numbers and do not use footers and headers (it is ok to have footnotes).

Let $I = \{a1, a2, a3, \dots, am\}$ be a set of items; $T = \{T_1, T_2, T_3, \dots, T_n\}$ be a set of transactions, where T_n is a combination of elements in a , which means each transaction is a subset of I . Support is defined as the number of transactions that contains certain itemsets. A frequent itemset is a set of items whose support is higher than some user-specified minimum support threshold, ξ . For example, Table I shows the transactions and itemsets. Let's focus on items A and D. The support of the AD is 2/5. It is frequent itemsets as long as the threshold is not higher than 2/5.

TABLE I
TRANSACTIONS WITH THEIR OWN ITEMSETS

TID	Item ^a
T100	ACDE
T200	CDE
T300	BE
T400	AD
T500	CE

2.1 Apriori Algorithm

Apriori algorithm [1] is the most important algorithm in data mining. The algorithm needs to generate lots of candidate itemsets and prune them recursively until the largest frequent itemsets have been obtained. For example, there are 1-itemsets $T = \{A, B, C, D\}$. Their candidate 2-itemsets are $T = \{AB, AC, AD, BC, BD, CD\}$. If there are 10 thousands of frequent 1-itemsets in T , the algorithm will generate more than 107 candidate itemsets, which waited for pruning tactically. Afterward, the frequent 2-itemsets are obtained. The disadvantages of apriori-like algorithms make scanning database repetitively and processing enormous candidate itemsets.

2.2 Frequent Pattern Growth

Han et al. presented a frequent pattern growth (FP-growth) algorithm[5, 6], which avoiding generating candidate itemsets. Moreover, this algorithm develops dense data structure which compresses storage space much smaller than original database. The main advantages of FP-growth algorithm are: First, the algorithm scans complete database only twice. One is to find all 1-frequent patterns and the other one is to construct FP-tree phase. Second, the algorithm creates a descending header table which stores 1-itemset and their counts only. Then, it builds FP-tree structure by scanning complete database with header table. The maximum frequent

itemsets will be obtained by using FP-growth algorithm with 1-itemset database instead of storing k-itemsets in database. The steps are as follows, in the first scans, the algorithm constructs the descending header table that consisted of the 1-itemsets' counts and its link. Afterward, it scans again for creating FP-tree. When FP-tree has been built, FP-growth algorithm starts to mine the relation of frequent itemsets. We take Table I as an example to describe how FP-tree algorithm works. First, we define the threshold as 40 percent. Table II shows the items which are sorted after the FP-tree algorithm discards the items whose value are below 40 percent.

TABLE II
DESCENDING FREQUENT ITEMS AFTER FIRST SCANS

TID	Original Items	Sorted Items
T100	A,C,D,E	E,C,D,A
T200	C,D,E	E,C,D
T300	B,E	E
T400	A,D	D,A
T500	C,E	E,C

The FP-tree algorithm gets frequent 1-itemset $\{(E:4), (C:3), (D:3), (A:2)\}$ where items are sorted in frequent descending order. After that, FP-tree algorithm creates header table and constructs FP-tree by scanning database second time. The header table consists of two elements: Item names and head of node-link, which means the first occurrence of element. It uses dotted lines to link the same item. As Fig 1.

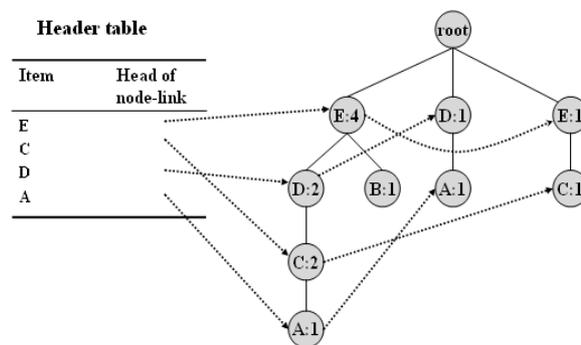


Fig. 1. Header table with FP-tree

The 1-itemsets FP-tree has constructed. FP-growth algorithm is going to run for finding all frequent itemsets' relation that based on 1-itemsets FP-tree. First, FP-growth algorithm constructs conditional pattern bases, E, C, D and A. Second, the algorithm scans FP-tree and constructs conditional FP-tree. The result shows in Table III. Frequent patterns are. $\{(AD), (CDE), (CE), (DE)\}$

TABLE III
CONDITIONAL ITEMSETS AND CONDITIONAL FP-TREE

PREFIX FREQUENT ITEMS	Conditional Itemsets	ConditionalFP-Tree
A	{(E:1,D:1,C:1),(D:1)}	{(E:1,D:1,C:1),(D:1)}
C	{(E:2,D:2),(E:1)}	{(E:2,D:2),(E:1)}
D	{(E:2)}	{(E:2)}
E	\emptyset	\emptyset

2.3 Parallel Data Mining

Because the increase of data mining time is caused by the increase of database, neither the apriori-like algorithms nor the FP-growth like algorithms are suitable for solving this problem. Ideas of parallel data mining were proposed. Agrawal et al. [2] proposed parallel algorithms, called count distribution (CD), which adopts apriori algorithm but can derive frequent itemsets in distributed system. This algorithm divides into two steps. First, it divides database into n smaller database, where n is equal to the number of computing computers. Afterward, each computer tries to get local frequent itemsets that is obtained by apriori algorithm. Second, these computers transfer information to each other for mining global frequent patterns. By the way, the transfer information method adopts broadcast mechanism so lots of candidate itemsets will be sent. Cheung et al. proposed another algorithm, called FDM[3] (Fast Distributed Mining of association rules), which also adopts apriori-like method. The FDM algorithm main differ from CD algorithm in it need to choose a computer as center computer. The center computer collects all local frequent itemsets from n local computers and constructs global frequent pattern itemsets. There are some different versions of FDM, such as FDM-LUP or FDM-LPP[3]. The basic rule of FDM we just mentioned has another name, called FDM-LP. We don't describe this method of algorithm because it's the same as FDM. FDM-LUP adopts Local pruning and Upper-bound Pruning. The upper-bound pruning method is efficient in apriori-like algorithm. If the candidate sets are not global frequent patterns after calculation, the candidate sets will be pruned. On the other hand, the candidate sets will be kept and waited for broadcast. For example, we assume that the threshold is 10% and each computer has 20 transactions in DB. The counts of local frequent patterns show in Table IV. We take (A, B) for example. If P2 (A, B) support is 2, then minimum support is $10\% * 20 = 2$. It is frequent patterns in P2. We need to check it is global candidate sets or not. The calculation is $P(A, B) = 2 + \min P1(A, B) + \min P3(A, B) = 2 + 2 + 1 = 5$, but it is below the upper bound $10\% * 60 = 6$. The (A, B) cannot be global candidate sets, we prune this candidate sets. Let's take another example P2(C, D). If P2(C, D) support is 2, the minimum support is $10\% * 20 = 2$. It is frequent patterns, too. The calculation is $P(C, D) = 2 + \min P1(C, D) + \min P3(C, D) = 4 + 2 + 2 = 8$, it's higher than upper bound 6. This candidate sets is possible global candidate sets, so we keep them and wait for broadcast. The last one method is FDM-LPP. It adopts local pruning with polling-site pruning. It's almost the same as

FDM-LUP which we mentioned. The only different between these two methods is FDM-LPP chooses a computer as polling site then perform upper-bound pruning method instead of pruning in each local computer.

All of them try to reduce communication cost or increase data mining efficiently by using their method. It is worth nothing that as long as distributed algorithms adopt apriori-like one, it can't avoid generating candidate itemsets.

TABLE IV
LOCAL FREQUENT PATTERNS COUNTS

P1		P2		P3	
Items	Support	Items	Support	Items	Support
A	5	A	3	A	1
B	2	B	2	B	4
C	4	C	2	C	2
D	6	D	2	D	5

2.4 Parallel FP-Tree Algorithm

Javed et al. [7] proposed parallel algorithms for multiprocessor systems. The research is the FP-tree algorithm applied in a distributed system. The steps are as follows: First, each processor is assigned partial local database and then constructs local header table, respectively. Second, the master processor obtains all local header tables from local processors and combines them as global header table. Afterward, the algorithm divides global header table to the same amount of items and then assigns them to local processors. The principal characteristic is that the algorithm gets global frequent itemsets from local processors by exchanging local FP-tree. The main problem is that some frequent itemsets are handed to be found since they might be infrequent itemsets in other local processors. Hence, local processors need to exchange their own sub-trees to each other. However, it is costly because infrequent itemsets need to be got by scanning database, please note that FP-tree only keeps frequent patterns.

2.5 Tidset Based FP-Tree Algorithm

Zhou et al. [4] first performed a simulation of parallel FP-tree in each stage. Zhou found that parallel FP-tree algorithm spends a lot of time in exchange stage. Hence, Zhou et al. proposed a method, which called Tidset based parallel FP-tree algorithm mining, which gets infrequent itemsets without scanning database again. Notice that the scanning database time doesn't list in the paper, but its execution time is larger than most of stages. Our idea is to develop a time slice method which is used in scanning database stage.

We briefly express our summary. The method of exchange tree which adopts in Apriori or FP-growth gives the sign “—” in Table V, because they are not developed on distributed environment.

TABLE V
CHARACTERS OF DIFFERENT ALGORITHMS

character \ algorithm	Generate candidate	Scan database	Method of exchange tree	Process data capacity	Method of determining infrequent patterns
Apriori	Yes	numerous	—	low	—
FP-growth	No	2	—	low	—
Parallel mining	Yes	numerous	broadcast	high	Scan DB
Parallel FP-mining	No	2	broadcast	higher	Scan DB
Tidset based FP-tree mining	No	2	broadcast	higher	Scan Tidset

3 Method

3.1 Basic Notation

Let $DB_i = \{T_{i1}, T_{i2}, T_{i3}, \dots, T_{im}\}$ be a set of transactions in DB_i , where $DB_i (i \in [1, 2, \dots, n])$ means the local database in node i . Note that the union of all local databases is the whole database, i.e., $\bigcup_{i=1}^n DB_i = DB$; and the intersection of any two local database is null. Namely, $DB_j \cap DB_k = \emptyset$, which j and k are in $[1, 2, \dots, n]$. Let N_0 be the central node, and N_i be the local node.

Definition 1. Let F_i be frequent patterns obtained from DB_i . Counts or items will increase during the database scanning.

Definition 2. Time-slice means periodical time. The frequent patterns will be transmitted to N_0 in the end of one periodical time. The interval length of periodical time can be adjusted.

Definition 3. Let $R_i \{a_i:m\}$ be the request message. Note that m is the item number and a_i is the node number. The request message stores the items that one or more nodes inquire.

3.2 Time Slice Transmitting

3.2.1 Transmitting to central node

Our goal is to transmit F_i to N_0 . First, FP-tree algorithm scans database and stores frequent patterns in N_i . We decompose N_i by periodical time and transmit frequent itemsets. The frequent patterns are showed in fig 2. For example, DB_1 scans the first period and then gets frequent itemsets $\{a1, a4, a5\}$. Subsequently, we transmit these 3 items to N_0 . In the following period, we get $\{a5, a6, a8\}$ and

transmit them again. We might encounter the problem that we get the same frequent patterns we recorded before. Hence, we use a flag to identify which items we have sent last time. Furthermore, we do not need to worry about whether we get the same frequent pattern during scanning database because their counts will be accumulated instead of inserting new items in N_i . For example, we transmit $\{a1, a4, a5\}$ to N_0 , then we will scan N_i again after periodical time passes. If the local node gets $\{a1, a4, a6\}$, we will insert a new item $a6$ to N_i and only transmit $a6$ to N_0 because $\{a1, a4\}$ just needs to update the counts. At the same time, for the other local nodes whose N_i does not contain the frequent patterns of N_0 , the central node N_0 transmits the request message of frequent patterns to them. Each local node transmits their frequent patterns counts to N_0 after it scans complete database. Therefore, N_0 derives all frequent patterns counts.

3.2.2 Getting request message

The next step describes how to transmit request message of frequent patterns to local node. First, N_0 gets frequent patterns of first period of N_1, N_2 and N_3 , and then N_1, N_2 and N_3 get the request messages from N_0 . In the following period, when local node transmits frequent patterns which is requested in request message, the local node will prune those items that also exist in request messages. Then N_i transmits to N_0 which remains in F . The method of transmitting and pruning will process until finishing scanning database.

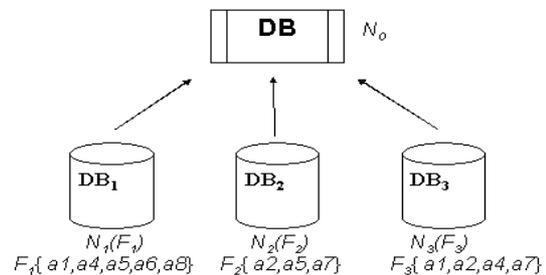


Fig. 2. Local database with its frequent patterns

For instance, we assume minimum threshold as 3, and N_i transmits frequent patterns each period. Their frequent patterns counts are $N_1:F_1\{a4:5+3, a1:5+2, a5:6, a6:7, a8:4\}$, $N_2:F_2\{a5:3, a2:3, a7:4\}$, $N_3:F_3\{a1:5+2, a2:3+4, a4:4, a7:4\}$. The antecedent of plus sign means frequent pattern counts that are derived from first period. On the contrary, the consequent of plus sign is derived from second period. Therefore, N_1 sends $\{a4, a1\}$ in first period then sends $\{a5, a6, a8\}$ in second period. Finally, N_0 gets $F_0\{a1, a2, a4, a5, a6, a7, a8\}$.

We will present how to obtain request message. In first period, the objective is to transmit local frequent patterns to N_0 . In fig 2, $N_1 \{a4, a1\}$, $N_2 \{a5\}$, $N_3 \{a1, a2\}$ transmit to N_0 . Table VI shows each local node and what frequent patterns they send to N_0 . The second step is N_i gets request messages. We can see table 4 which space is null, then N_0 will transmit request

messages to N_i , so N_1 gets $R_1 \{a_3:2, a_2:5\}$, $N_2:R_2 \{a_{13}:1, a_3:2, a_1:4\}$ and $N_3:R_3 \{a_1:4, a_2:5\}$. Note that the items in R_1, R_2 and R_3 are all frequent patterns existing in at least one node, but R_1, R_2 and R_3 are infrequent patterns in their nodes respectively.

TABLE VI
N₀ WITH FREQUENT PATTERNS IN FIRST PERIOD

Item \ Node	N ₁	N ₂	N ₃
a1	*		*
a2			
a4	*		
a5		*	

Note: The symbol * means item has been transmitted to N₀

In second period, local node N_1 obtains $F_1 \{a_5, a_6, a_8\}$, N_2 gets $F_2 \{a_2, a_7\}$ and N_3 gets $F_3 \{a_4, a_7\}$. Afterward, N_0 obtains $F \{a_2, a_4, a_5, a_6, a_7, a_8\}$. Each node compares F_i with R_i . For example, N_1 transmits $F_1 \{a_5, a_6, a_8\}$ and later request messages of N_1 are $R_1 \{a_3:2, a_2:5\}$. We prune a_5 that stores in request message because its attribute transforms into frequent pattern, so the status of request messages are $N_1:R_1 \{a_3:2\}$. Let's take a look at Table VII. N_0 transmits the request message $\{a_6, a_7, a_8\}$ to assigned nodes. When second period come, $N_1:R_1$ gets $\{a_{23}:7\}$, $N_2:R_2 \{a_1:6, a_1:8\}$, $N_3:R_3 \{a_1:6, a_1:8\}$. The final request messages are $N_1:R_1 \{a_3:2, a_{23}:7\}$, $N_2:R_2 \{a_{13}:1, a_1:4, a_1:6, a_1:8\}$ and $N_3:R_3 \{a_2:5, a_1:6, a_1:8\}$. The example shows in Table VIII.

TABLE VII
N₀ WITH FREQUENT PATTERNS IN SECOND PERIOD

Item \ Node	N ₁	N ₂	N ₃
a1	-		-
a2		*	-
a4	-		*
a5	*	-	
a6	*		
a7		*	*
a8	*		

Note: The symbol - means item had been processed in former period ; The symbol * indicates item has been processed in this period.

3.3 Modification of request messages

The request messages need to be adjusted because it is not absolutely correct. After second period, N_0 gets $F \{a_2, a_4, a_5, a_6, a_7, a_8\}$ where a_2, a_4, a_5 we have processed in first period. Hence, N_0 scans those items obtained in former period again. In this case, we get $N_2:F_2 \{a_2\}$, $N_3:F_3 \{a_4\}$ and $N_1:F_1 \{a_5\}$, then N_0 scans in order to find out the items whose space is null; and further, it transmits request messages to those(?) are inquired nodes. Therefore, N_0 transmits request messages $\{a_2:2\}$ to N_1 , $\{a_3:4\}$ to N_2 , $\{a_1:5\}$ to N_3 . The adjusted request messages are $N_1:R_1 \{a_{23}:2, a_{23}:7\}$, $N_2:R_2 \{a_{13}:1, a_{13}:4, a_1:6, a_1:8\}$ and $N_3:R_3 \{a_{12}:5, a_1:6, a_1:8\}$.

After local nodes finish their scanning job, the N_0 obtains all frequent patterns counts. Furthermore, each node has request messages that inquired infrequent patterns. Take N_3 for example, the frequent patterns of N_3 are $F_3 \{a_1, a_2, a_4, a_7\}$, so N_3 inquires frequent patterns from N_0 . The other infrequent patterns will be sent from N_i , namely, $N_1:R_1 \{a_3:2, a_{23}:7\}$, $N_2:R_2 \{a_{13}:1, a_{13}:4\}$.

3.4 Transaction sets

After scanning database phase finishes in the first time, the transaction sets are created by each node. The content of table stores whole transaction relation. Table IX indicates the Tidset that can get transactions relation without scanning database again. The function of this table focuses on dealing with communication of infrequent patterns. It can derive infrequent patterns efficiently instead of scanning complete database.

Our method creates 1-itemset frequent patterns in N_0 and request messages in N_i , and our method's utilities can avoid transaction broadcasting to each other. When we reduce more, the rate of building frequent pattern tree of each node will be faster. Thus, local node constructs FP-tree respectively. Afterward, FP-tree of each node transmits to N_0 . Thus, N_0 combines all FP-tree then constructs global FP-tree by FP-growth algorithm.

TABLE VIII
TRANSACTION SETS

Item \ Trans.	E	C	D	A	B
1	*	*	*	*	
2	*	*	*		
3	*				*
4			*	*	
5	*	*			
sum	4	3	3	2	1

Note: The symbol * indicates the item which occur in N_i

TABLE IX
THE TIME-SLICE ALGORITHM, PROCESS IN N_i

Input	1. DB: transaction database 2. support : minimum support threshold 3. time interval: the user defined a time for a period, namely sec in pseudo code.
Output	all 1-itemset frequent patterns, Transaction sets table

Procedure:

1. Scan the DB with support. In the first period, we check the support and transmit all items whose supports are higher than user defined. In addition, we create a table that stores char 1 in those items whose use for identification.
2. In next period we repeat the former phase.

The pseudo code of time-slice

Function time-slice

```
t = clock();
while (1){
scan DB and create transaction sets table at the same time;
if (feof(DB))
    break;
else
    sec = SEC_SINCE(t);
    if (sec <= user defined ){
    output 1-frequent itemsets;
    else
    {
    output 1-frequent itemsets;
    }
}
}
```

TABLE X
THE TIME-SLICE ALGORITHM, PROCESS IN N_0

Input	1. period of 1-itemset frequent patterns 2. global minimum support threshold
Output	all items' request messages with N_i

Procedure:

1. Scan array and add value to two way array which indicates the number of times of item occurrence in N_i
2. Output request messages to N_i .

```
while (1){
scan array and create two dimensional table which in each row
stores item id and each column stores node id;
For(i=0; i< two dimensional array; i++){
if ( array[i] == '\0')
    break;
else
    gives the value 1 to array;
}
For(i=0; i< row; i++){
for(column=0; k< column; k++)
output each row's request messages to  $N_i$ }
```

TABLE XI
THE REQUEST MESSAGE ALGORITHM, PROCESS IN N_i

Input	1.request message
Output	transaction to correspond with request message

Procedure:

1. Scan request messages and transmit transactions to other nodes.
2. Get the transactions from other nodes , then run FP-growth algorithm with their own database and transactions we mentioned, the global FP-Tree
for(i=0; i < item id; i++){
if (request message [i] !=0)
output transaction to correspond node;
else
obtain transactions from N_0 }

4 Discussion

This study require the local nodes have to obtain frequent patterns from central node and obtain infrequent patterns from the other nodes. We discuss the following conditions. First, we assume a patterns tree T_i , if the threshold below the ξ , then any supersets of T_i must not be frequent patterns. We do not obtain any frequent patterns from central node, but we transmit T_i to other nodes which require. Second, if T_i 's threshold above the ξ , We transmit T_i to central node only. Third, if T_i contains partial frequent items and infrequent items, We transmit T_i to central node only. When a node require such T_i , the node obtain this tree from central node. Hence, the complete information exchange stage is finished. This simulation focuses on variety characters. The first one is total execution time, we compare Tidset execution time on broadcast method with our method. And another one is to adjust threshold and compare the memory usage. Finally, we show the total execution time on different nodes.

Our datasets generated by IBM's Quest Synthetic Data Generator code which can compile in windows system. The C program which we have written adopt visual c++ compiler. The method of information exchange of tree has been completed by java. This study adopts array instead of scanning the disk, and adopt the algorithm instead of broadcasting. Thus, it is reasonable that the study increases efficiency.

5 Conclusion

It is necessary to use distributed systems nowadays because the growth rate of database is faster than a PC. In this study, we adopt Tidset method that can directly derive transactions in memory instead of scanning the disk. Moreover, we enhance the performance that can lower down the information exchange cost of nodes and rise central node's utility rate. In the future, we can combine Tidset that applied to our algorithm with utility or weight issues.

6 References

- [1] Agrawal, R. and R. Srikant, "Fast Algorithms for Mining Association Rules." VLDB, 1994: p. 489-499.
- [2] Agrawal, R. and J.C. Shafer, "Parallel Mining of Association Rules." IEEE Transactions on Knowledge and Data Engineering 1996. 8(6): p. 962-969.
- [3] David W. Cheung, Jiawei Han, Vincent T. Ng, Ada W. Fu and Yongjian Fu, "A Fast Distributed Algorithm for Mining Association Rules." in Proceedings of the fourth international conference on on Parallel and distributed information systems 1996.
- [4] Zhou, J. and K.-M. Yu, "Tidset-Based Parallel FP-tree Algorithm for the Frequent Pattern Mining Problem on PC Clusters in Advances in Grid and Pervasive Computing." 2008. p. 18-28.
- [5] Han, J., J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation." ACM SIGMOD Record 2000. 29(2): p. 1-12.
- [6] Jiawei Han, Jian Pei, Yiwen Yin and Runying Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach" Data Mining and Knowledge Discovery, 2004. 8(1): p. 53-87.
- [7] Javed, A. and A. Khokhar, "Frequent Pattern Mining on Message Passing Multiprocessor Systems" Distributed and Parallel Databases, 2004: p. 321-334.
- [8] Bo He, Yue Wang, Wu Yang and Yuan chen, "Fast Algorithm for Mining Global Frequent Itemsets Based on Distributed Database", in Rough Sets and Knowledge Technology. 2006. p. 415-420.
- [9] Schuster, A. and R. Wolff, "Communication-Efficient Distributed Mining of Association Rules," Data Mining and Knowledge Discovery, 2004. 8: p. 171-196.
- [10] Dora Souliou, Aris Pagourtzis and Nikolaos Drosinos, "Computing frequent itemsets in parallel using partial support trees." Journal of Systems and Software, 2006. 79(12): p. 1735-1743.

A Novel and Efficient Distributed Data Mining Algorithm Based on Frequent Pattern-Tree

Fan Wu, Ya-Han Hu, and Tz Ke Wu

Department of Management Information System, National Chung Cheng University, Chiayi, Taiwan, R.O.C

Abstract - In this paper, we proposed a novel algorithm which is implemented on the distributed system that can efficiently solve the problem of FP-tree. The algorithm adopts divide and conquer strategy to split a huge database into several sub-databases. With data division, the algorithm can reduce the total execution time. The algorithm doesn't construct the whole FP-tree. Instead, with intermittence, a time division mechanism, the algorithm can also efficiently reduce the execution time. We also compress the data into a one dimensional array while transmitting, which can reduce the communication cost.

Keywords: FP-Tree, large database, frequent pattern mining

1 Introduction

Many enterprises today use databases to store their daily transactions. On analyzing these data, they find patterns among the transactions to form association rules which can make strategy more precisely. An association rule is the concept that people usually buy a specific combination of different products together, given a simple example, bread and milk usually bought together.

Some traditional pattern-mining algorithms, such as Apriori [1] and FP-tree [2], are applied to find frequent patterns. The Apriori and Apriori-like algorithms generate candidate itemsets and scan databases as many times as the length of the longest frequent itemsets. In contrast, the FP-tree and FP-tree-like algorithms only need to scan database twice. In the first scan, the algorithm counts the number of occurrences for each item. Then, the algorithm prunes items with their counts smaller than the minimum threshold and sorts the rest frequent items as a header table. In the second scan, the algorithm uses the sequence of header table to construct FP-tree. That is, while scanning each transaction, the algorithm adds a new node to the FP-tree if the item is not in the tree; otherwise, it increases the count of the node by one in the tree. By exploring FP-tree, the FP-mining algorithm can discover a complete set of frequent patterns. With only twice of database scans, FP-tree is consider to be more efficient than the Apriori.

Although FP-tree is considered an efficient method, some problems exist with FP-tree. FP-tree algorithm becomes inefficient while the minimum threshold is small and/or the size of database is huge.[3] These two situations can increase

both the depth and the width of FP-tree. Thus, it will take much more time to mine frequent patterns by FP-mining algorithm. In addition, the large FP-tree may not be able to put into memory, as will make it inefficient to swap data from disk and memory.

1.1 Existing Solutions

To deal with large database mining problems, several researches have been developed. Among them, distributed system is considered as one of a good ways. Mafruz (2003) introduced two kinds of methods, parallel association rule mining (PARM) and distributed association rule mining (DARM).[4] In. PARM divides a database into several local databases, and uses parallel multiprocessor shared-nothing environment to mine local databases. The processors need to communicate with each other for the global counts. On the other hand, DARM considers association rules discovering from the geographically distributed data sets. The main challenge of DARM is the communication cost, so the aim of subsequent researchers focus on minimizing communication costs. Table 1 shows the comparisons of the main existing algorithms of DARM and PAEM with our method.

Table 1: Comparison of the main existing algorithms of DARM and PAEM with our method

Algorithm	PFP-tree[3]	CD[5]	FDM[6]	DDM[7]	Our method
Environment	Parallel	Parallel	Distributed	Distributed	Parallel
Based on	FP-tree	Apriori	Apriori	Apriori	FP-tree like
Focus in	Minimize Communication overheads	Minimize communication cost	Minimize communication cost	Minimize communication cost	Reduce communication cost and execution time
Memory usage	efficiently	inefficient	*	efficient	n/a
Communication cost	Each processor communicate with other processor iteration	$O(C ^*n)$ C : the size of candidate itemsets r n: number of datasets	$O(Cp ^*n)$ Cp : potentially frequent candidate itemsets n: number of sites	$O(P_{rabove} * C * n)$ P_{rabove} : probability of a candidate itemset that has support greater than the support threshold	n/a

There are also many extensions of PARM and DARM, Yu [8] tried to deal with load balancing issue. In [9], Chung proposed DMA to generate much smaller candidate itemsets in order to reduce communications, Chung also proposed PMM[10] to look ahead at each pass of Apriori and prunes more candidate itemsets by checking the frequencies of their supersets. Gregory Buehre etc. [11] developed an encoding method, called *strip marshaling*, to reduce the communication cost by transferring the FP-tree into an one way array before transmitting to other sites. [12] used the concept of metadata to generate global frequent itemsets. With hash concept, HPA in [13] could eliminate the cost of broadcasting.

1.2 Our methods

DARM and PARM are developed based on the Apriori or FP-tree algorithm, respectively. They use these two algorithms to mine association rules on distributed sites. The main challenge is the share of memory and communication cost, but they did not try to solve the inherent problems, such as tremendous candidate itemsets and huge size of FP-tree. In this paper, we proposed a novel algorithm which is implemented on the distributed system that can efficiently solve the problem of FP-tree.

The algorithm adopts divide and conquer strategy to split a huge database into several sub-databases. With data division, the algorithm can reduce the total execution time of algorithm. In this paper, we don't construct the whole FP-tree. Instead, with intermittence, a time division mechanism, the proposed algorithm can efficiently reduce the execution time. We also compress the data into a one dimensional array while transmitting, which can reduce the communication cost.

2 RELATED WORKS

Before discussing the related works, some notations are introduced first. Let I denote the set of items i , i.e., $I = \{i_1, i_2, \dots, i_n\}$, where n is the number of items in the database. Assume there is a database DB with m transactions related to the itemset I , DB represents the database and DB contains a set of transactions T . That is, $DB = \{T_1, T_2, \dots, T_m\}$, where transaction $T_j \subseteq I$. The support of an itemset A means the number of transactions in DB contains itemset A within it, denoted as $Sup_{DB}(A)$. Let ξ be the minimum support threshold. An itemset(A) is said to be frequent if $Sup_{DB}(A) \geq \xi$.

2.1 Frequent Pattern-tree Algorithm

Frequent pattern tree algorithm includes two sub-parts, FP-tree construction and FP-growth, and we briefly summarized them as follows.

2.1.1 FP-tree construction

The FP-tree construction scans the database twice. In the first scan, the algorithm gets the support counts of each

item and prunes items with support smaller than the minimum threshold. The algorithm also sorts these frequent items in the descending order and stores them as a header table. The table has two columns. One stores the item name; the others, called header of node-links, stores the pointer of the link list that chains all the transactions containing the corresponding item in it. As shown in Fig. 1, for example, a database DB has 5 transactions and the minimum support threshold is 3. Items with supports equal to or more than 3 are illustrated in bolder alphabets. After scanning every transaction in DB , we sort the items in each transaction according to the their frequencies, as shown in the right column of Fig. 1

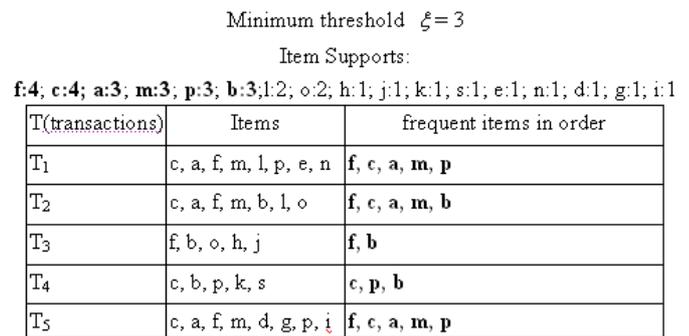


Fig. 1. An example of DB

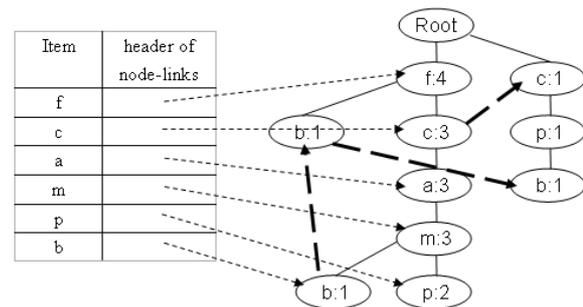


Fig. 2. Header table and FP-tree of DB

In the second scan, the algorithm scans the database transaction by transaction. In the mean time, the algorithm starts to construct the FP-tree. At first, the root node is produced. While scanning each transaction, the algorithm will construct a linked list, according to the order of sorted header table, from root to leaves. Notice that if the front part of itemset of a scanning transaction matches the prefix of some list in the tree, no brand new linked list is created. Instead, the count of the nodes in the matched prefix of the list is increased by one. The unmatched latter part of itemset of that transaction will deviate a branch from the matched list. As mentioned before, each header of node link stores the pointer linking to the list of all the transaction containing the corresponding node. When a new node is created, the algorithm lengthens the list to link to the new node. As the bolder arrow-line in **錯誤! 找不到參照來源**, take node b as an example. Once a node of item b is created, the node contains a node-link linking to the next node b . After all the transactions containing item b are scanned, all the node of

item b in the FP-tree are linked together by the node-links. The header of node-links in header table records the link to the first node of item b .

The FP-tree contains information about frequent items. Such as, the tree can tell the pattern counts of each item. As shown in the right most vertical link in **錯誤! 找不到參照來源**, item $\{p\}$ co-occurs with item $\{c\}$ for one time, the middle vertical link, item $\{p\}$ co-occurs with items $\{f, c, a, m\}$ for two times. Furthermore, the horizontal list connecting the same item in the all vertical links can be used to mine patterns about that item in the database. These two properties can be used in FP-mining.

2.1.2 FP-growth

After FP-tree was constructed, the second part, FP-growth, will start. With the two properties mentioned before, the FP-growth generates *conditional FP-trees* for every item to mine frequent patterns. To generate an item k 's conditional FP-tree, the FP-growth will form the item k 's *conditional pattern base* first. An item k 's conditional pattern-base contains a set of items associated with item k which is collected by exploring the FP-tree from node k to root. Fig. 3 shows the conditional pattern base and conditional FP-tree of DB. Take the item p for example, the algorithm can get the position of first node p by checking the header of node-links of item p . The algorithm explores the FP-tree from the position to the tree root. As we can see, the first p node is at the leaf of the middle branch of FP-tree. Start from the node p , the algorithm can get the itemset $\{(f:2, c:2, a:2, m:2)\}$ while exploring the FP-tree to the root. Because of the node p co-occurs 2 times with itemset $\{(f:2, c:2, a:2, m:2)\}$ in the DB, the count of every item in the itemset are recorded 2 in the conditional pattern base as well. The right branch of DB's FP-tree also contains a node p , the position can be easily checked by the node link from previous node p . After the exploring, the algorithm can get the itemset $\{(c:1)\}$ which co-occurred with item p once in the DB. With no other node p , the whole conditional pattern base of item p is $\{(f:2, c:2, a:2, m:2), (c:1)\}$, as shown in the first row of conditional pattern base in Fig. 3.

The conditional pattern base of p can form a conditional FP-tree of p in reverse direction (i.e., from bottom to top). The summation of each item in a conditional pattern base must larger than or equal to the minimum threshold and the item is said to be frequent with item in the left most column, and thus can form a conditional FP-tree with respect to item in left most column. The conditional FP-tree is used to grow frequent pattern. Continue with item p , only item c under the conditional pattern base of p has its count being equal to the minimum threshold 3, so the conditional FP-tree of p is $\{(c:3)\}P$. With $\{(c:3)\}P$, the frequent pattern is $\{(c:3).(p:3)\}$.

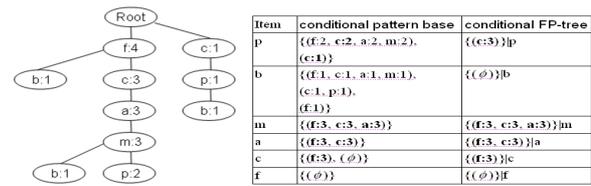


Fig. 3. Conditional pattern base and conditional FP-tree of DB

To illustrate more clearly, here is another example for item m , its conditional pattern base is $\{(f:3, c:3, a:3)\}$, and its conditional FP-tree is $\{(f:3, c:3, a:3)\}M$. Thus, the frequent patterns related to item m are $\{(fm:3)\}$, $\{(cm:3)\}$, $\{(am:3)\}$. Because of the order of frequencies, there is more information contained in the conditional FP-tree. For items am , the conditional FP-tree is $\{(f:3, c:3)\}AM$, and the frequent pattern related to items am is pattern $\{(cam:3)\}$ and $\{(fam:3)\}$. In addition, the conditional FP-tree of items cm is $\{(f:3)\}CM$. There for, the frequent pattern of items cm is $\{(fcm:3)\}$. Furthermore, the itemset cam 's conditional FP-tree is $\{(f:3)\}CAM$ and can mine pattern $\{(fcam:3)\}$ from the tree. For summary, the frequent pattern related to item m is $\{(m:3), (am:3), (cm:3), (fm:3), (cam:3), (fam:3), (fcam:3), (fcm:3)\}$.

With FP-growth algorithm, FP-algorithm can mine frequent patterns by only scanning database twice. Comparing with Apriori algorithm, FP-tree doesn't generate candidate itemsets.

3 OUR METHOD

In this section, we present an efficient parallel algorithm based on FP-tree. We assume that a database is divided into many partitions. The distributed database in our environment is a horizontally partitioned database, which means that the schema of all the database-partitions is the same. We also assume that one item can only appear once in a transaction.

3.1 Notations

In this paper, we developed a novel algorithm based on FP-tree construction-process and use a *data-batch concept* while transferring data in a centralized distributed system. We first give some notations and definitions before talking about our method.

Notation 1: C : The center site responds for the computing the center header table and do the FP-growth.

Notation 2: L_i : Local sites. Assume there are k local sites in a distributed system.

The algorithm is implemented on a distributed environment with one center (Notation 1) and n local sites, denoted as (Notation 2). That is, the whole system can be represent as $C \cup (\bigcup_i^n L_i)$.

Notation 3: DB_T : A database with T transactions.

Notation 4: DB_i : A DB is partitioned into n parts to every L_i , each part has T/n transactions. The union of DB_i equivalents to DB.

In our environment, all the DB_i have the same schema of DB, which is called horizontal partition.

Notation 5: CH: Center header table. The table records all the frequent items and sorts the items by counts in DB with descending order.

Notation 6: LH_i : Local sites' header table. Each LH_i records corresponding frequent items with same order as CH.

Notation 7: Suppose there are k different kinds of items and $fre(k)$ represents the number frequent items.

Figure 4 shows an example of distributed system with one center C and three local sites L_1, L_2 and L_3 with the assigned DB_1, DB_2 and DB_3 .

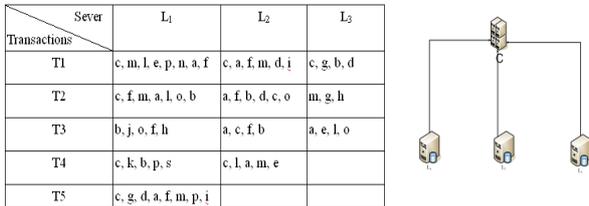


Figure 4: An example of one center and three local sites with assigned database.

3.2 Proposed algorithm

In this section, we illustrate our algorithm. Our algorithm also scans database twice. In the first scan, make each local site knows the global view of frequent items. Every L_i scan it's own DB_i to get all the item counts and construct LH_i , then transfer LH_i to C . The C (center) accumulates the same items' counts to construct CH (center header table) and prunes the infrequent items. After the pruning, C send the final CH to every L_i to update the LH_i . In the second scan, the algorithm records the frequent itemsets in each L_i into a two-dimensional array to reduce the communication cost while transmitting. Further, the algorithm also uses the *intermittence concept* to decide when to transfer the frequent itemsets information to the center for the latter process of pattern-growth. With intermittence, C can construct conditional pattern base and each L_i can scan DB_i simultaneously. In the following, we will discuss our method in detail.

3.2.1 The first scan

In the first scan, each L_i scans its assigned database DB_i and creates its own header table LH_i . All the LH_i records three information: item name, item counts and the last transaction contains the item, denoted as LST.

Item name	c	f	a	m	d	i	b	e	l	o
Item count	4	3	3	2	2	1	2	1	1	1
LST	T4	T3	T4	T4	T2	T1	T3	T4	T4	T2

Figure 5: Header table of L_2, LH_2

Figure 5 shows an example of LH_2 , the third row records the last transactions contains the corresponding item in the first row. As for example, transaction T_3 in L_2 has four items $\{a, c, f, b\}$. The item b and item f appears in this transaction for the last time, so the LST of the two items is T_3 . With LST, our algorithm can decide when to transfer the information to C in the second scan, we called the concept as *intermittence*.

After all the transactions in each DB_i are scanned, each L_i gets the counts of all items in the DB_i . All L_i then transfer the item name and item counts in LH_i to the center site, C , to build the center header table, CH . C accumulates every item's counts from all LH_i . For example, Figure 6 shows every LH_i contains all the item counts in each DB_i after the first scan of DB_i . Each LH_i will be then forwarded to site C to form CH as mentioned before. Figure 6 also shows that the item counts in CH are the summation of the three LH_i .

Header table	Item counts
LH_1	c:4,f:4,a:3,m:3,p:3,b:3,l:2,o:2,h:1,j:1,k:1,s:1,e:1,n:1,d:1,g:1,i:1
LH_2	c:4,f:3,a:3,m:2,d:2,i:1,b:2,e:1,l:1,o:1
LH_3	g:2,b:1,c:1,d:1,h:1,m:1,a:1,e:1,l:1,o:1
CH	f:7;a:8;c:9;d:4;g:3;i:2;m:6;p:3;b:6;l:4;o:4;h:2;j:1;k:1;s:1;e:3;n:1

Figure 6: LH_i of each L_i and accumulated result of CH without sorting

After CH is constructed, site C can prune the items with counts smaller than the minimum support threshold and sorts the remaining items, i.e., the frequent items, with the counts of each item. Site C then sends back the sorted frequent item table CH to every L_i . Each L_i then compares the CH with its own LH_i , picking out the same items as CH and sorting them as the order of items in CH . Figure 7 shows the sorted CH and updated LH_i with minimum threshold = 3. The blank slot in Figure 7 means the L_i doesn't has the item in DB_i . By keeping the items appear in CH in each LH_i , we can assure the completeness of all the rules after the pattern-growth.

With minimum threshold = 3											
Header table	Item counts										
CH	c:9	a:8	f:7	m:6	b:6	d:4	l:4	o:4	g:3	p:3	e:3
LH_1	c:4	a:3	f:4	m:3	b:3	d:1	l:2	o:2	g:1	p:3	e:1
LH_2	c:4	a:3	f:3	m:2	b:2	d:2	l:1	o:1			e:1
LH_3	c:1	a:1		m:1	b:1	d:1	l:1	o:1	g:2		e:1

Figure 7: Sorted CH and updated three LH_i

3.2.2 The second scan

After the update process of LH_i , local site L_i starts the second scan. By doing the scan, the algorithm will encode the data into a two-dimensional array and transmit the partial array to C by *intermittence concept*.

3.2.2.1 Encode data

While scanning a transaction, the algorithm sorts the frequent items in the transaction with the order of items in LH_i .

After the sorting, the algorithm reads the sorted frequent items one by one. Whenever got an item, the algorithm identify the position of the items in a two-dimensional array. The two-dimensional array is showed in Figure 8.

mim = 3	4	4	3	2	2	1	2	1		1	
c	0	0	0	0	0	0	0	0	0	0	0
a	0	0	0	0	0	0	0	0	0	0	0
f	0	0	0	0	0	0	0	0	0	0	0
b	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0
g	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0

Figure 8: The two-dimensional array of L_2

The leftmost column records the item name and the top grid records the minimum threshold. The first row records the counts with the sequence of updated LH_2 , which means the first grid of first row is count of item c , the second grid of first row is count of item a and so on. We use an example to illustrate the position of an frequent item in a sorted transtraction. Continue with the former example, transtraction T_1 in L_2 has six items $\{c, a, f, m, d, i\}$. In second scan, our algorithm first picks out the frequent items and sort them according to the order of LH_i , resulting in $\{c, a, f, m, d\}$. Then, we read the reduced frequent itemset one by one. For item c , find the row with leftmost grid is c , and put at the column of item c 's count. The count of item c will reduced by one as shown in Figure 9.

mim = 3	4	4	3	2	2	1	2	1		1	
c	1	0	0	0	0	0	0	0	0	0	0
a	0	0	0	0	0	0	0	0	0	0	0
f	0	0	0	0	0	0	0	0	0	0	0
b	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0
g	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0

Figure 9: The two-dimensional array of L_2 after reduced and sorted T_1 item c is scanned

As for the second item a , find the grid within the row begin with item a in the left most column and the column of item a 's count to increase by one. Because item c in this frequent itemset appears in front of item a , the left grid of item c 's count column with the same row is increased by one, too. Figure 10 shows the result of item a is got.

mim = 3	3	3	3	2	2	1	2	1		1	
c	1	0	0	0	0	0	0	0	0	0	0
a	1	1	0	0	0	0	0	0	0	0	0
f	0	0	0	0	0	0	0	0	0	0	0
b	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0
g	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0

Figure 10 : The two-dimensional array of L_2 after reduced and sorted T_1 item a is scanned

Take one more example, when item f is picked, the column of item decreased by one and the grid in row begin with item f and column of item f 's count increased by one. The prior two items c 's and a 's column with same row also increased by one, Figure 11 is the result of item f is scanned.

mim = 3	3	3	2	2	2	1	2	1		1	
c	1	0	0	0	0	0	0	0	0	0	0
a	1	1	0	0	0	0	0	0	0	0	0
f	1	1	1	0	0	0	0	0	0	0	0
b	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	0	0	0	0	0	0	0
o	0	0	0	0	0	0	0	0	0	0	0
g	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0

Figure 11 : The two-dimensional array of L_2 after reduced and sorted T_1 item f is scanned

3.2.2.2 Intermittence transmitting

In this paper, we use a *intermittence* concept while transferring data between C and L_i . The concept of intermittence indicates that while transmitting data from local sites L_i to the canter C , algorithm doesn't transmit the entire encoded data of whole database at one time. Instead, the algorithm transmits the encoded data practically according to the LST in the header table LH_i .

We transfer partial of the two dimensional array when the number of scanned transactions satisfies the minimum LST in the LH_i . For example, consider Figure 5 again. At first, the minimum LST is T_2 , while finishing the scan of T_2 , L_2 will transmit the intermittent item d 's and o 's row to C . which means transfer $\{1, 1, 1, 0, 1, 0, 1\}$ and $\{1, 1, 1, 1, 0, 0, 1, 1\}$ to C , as shown in Figure 12

mim = 3	2	2	2	1	1	1	0	0		1	
c	2	0	0	0	0	0	0	0	0	0	0
a	2	2	0	0	0	0	0	0	0	0	0
f	2	2	2	0	0	0	0	0	0	0	0
b	1	1	1	1	0	0	0	0	0	0	0
m	1	1	1	0	1	0	0	0	0	0	0
l	0	0	0	0	0	0	0	0	0	0	0
d	1	1	1	0	1	0	1	0	0	0	0
o	1	1	1	1	0	0	1	1	0	0	0
g	0	0	0	0	0	0	0	0	0	0	0
e	0	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	0	0	0

Figure 12 : The two-dimensional array of L_2 after T_2 is scanned

The *LST* indicates the last transaction containing the frequent item. After the *LST*, there won't be any transaction containing intermittent item, thus, we can transmit frequent item and its related preceding order items' counts to *C*.

After transferring, *C* analyses the information from each LH_i and puts the same frequent item's preceding counts together. With the same frequent item, the equivalent preceding order items count will be added up. All frequent items' count can construct as the conditional pattern base in the FP-tree algorithm.

TABLE 2: (A) INTERMITTENT ITEM B AND PRECEDING ORDER ITEMS' COUNTS
(B) CONDITIONAL PATTERN BASE OF B
(C) CONDITIONAL FP-TREE OF B

(a)frequent item b and preceding order items	(b)conditional pattern base of item b	(c)the conditional FP-tree of b
In sever 1: {2, 1, 2, 2}	{2, 1, 2, 2}	{(c:5), (a:3), (f:4)}b
In sever 2: {2, 2, 2, 2}	+{1, 0, 0, 2}	
In sever 3: {1, 0, 0, 2}	----- {5, 3, 4, 6}	

Table 2 shows how to form conditional FP-tree from one dimensional array. The first column shows the frequent item b and preceding order items' counts of each LH_i . Second column shows the summarized of preceding order items' count of item b. As we can see, the conditional FP-tree of b is constructed in the third column and the correspond count is {5, 3, 4} Thus, we can find patterns with similar way as FP-growth.

4 DISCUSSION

The completeness and correctness of our algorithm are discussed in this section.

Lemma 1 *By keeping the same form of two diminational array, which means left most column record the frequent item*

in same sequence and the first rowis the local count of frequent items,all the information can be found in site C.

Rationale. In each L_i , the row of every frequent item in the two dimianational array records the times appear with other frequent items. As notation 3 and 4, a $DB_T = \bigcup_1^k DB_i$. Thus, when all the two-diminational arrays are accumulated in *C*, the center site can get all the relationships among each frequent items in the hole *DB*.

Lemma 2 *As mentioned in Lemma 1, C has all the relationship between frequent items.By checking every row to generate the conditional pattern base, we can get all the conditional FP-tree.*

Lemma 3 *The execution time is of each L_i bounded by the $fre(k)$ and number of transtraction T/n .*

Rationale. In each L_i , the worst case will be that every transtraction has all frequent items, than the reduced transtraction will be $fre(k)$. By scanning one frequent item, the algorithm has to find the position of column and row. So the *big O* of execution is $O(fre(k)*T/n)$.

Lemma 4 *The total communication cost is*

$$O(n *(fre(k)^2 +k))$$

Rationale. In the first scan, every L_i sends all the items in DB_i to *C*. The worst case is that all L_i has all k kinds items, $n*k$. In the second scan, the cost is $1/2*fre(k)$ of the helf of two diminational array with n local sites. Thus the *big O* is $O(n *(fre(k)^2 +k))$.

5 CONCLUSION

In this paper, we proposed a novel algorithm which is implemented on the distributed system that can efficiently solve the problem of FP-tree. With intermittence concept and data-compression technique, the algorithm can reduce the communication cost. In addition, our algorithm may be more efficient compared to the algorithm based on DARM.

6 REFERENCES

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proceedings of the 20th VLDB Conference Santiago, Chile, 1994.

[2] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proceedings of the 2000 ACM SIGMOD international conference on Management of data, Dallas, Texas, United States.

[3] A. JAVED and A. KHOKHAR, "Frequent Pattern Mining on Message Passing Multiprocessor Systems," Distributed and Parallel Databases, Volume 16, 2004.

- [4] M. Z. Ashrafi, D. Taniar, and K. Smith, "ODAM: An Optimized Distributed Association Rule Mining Algorithm," *IEEE Distributed Systems Online* 1541-4922, IEEE Computer Society, vol. 5, No. 3, 2004.
- [5] R. Agrawal and J. C. Shafer, "Parallel Mining of Association Rules," *IEEE Transactions on Knowledge and Data Engineering* vol. 8, No. 6, 1996.
- [6] D. W. Cheung, J. Han, V. T. Ng, A. W. Fu, and Y. Fu, "A Fast Distributed Algorithm for Mining Association Rules," *Proceedings of the fourth international conference on Parallel and distributed information systems*, Miami Beach, Florida, United States, 1996.
- [7] A. Schuster and R. Wolff, "CommunicationEfficient Distributed Mining of Association Rules," *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, Santa Barbara, California, United States, 2001.
- [8] K.-M. Yu, J. Zhou, and W. C. Hsiao, "Load Balancing Approach Parallel Algorithm for Frequent Pattern Mining," V. Malyskin (Ed.): *PaCT 2007*, LNCS 4671, 2007. © Springer-Verlag Berlin Heidelberg 2007.
- [9] D. W. Cheung, V. T. Ng, A. W. Fu, and Y. Fu, "Efficient Mining of Association Rules in Distributed Databases," *IEEE Transactions on Knowledge and DATA Engineering*, VOL. 8, NO. 6, 1996.
- [10] S. M. Chung and C. Luo, "Parallel Mining of Maximal Frequent Itemsets from Databases," *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03)*, Sacramento, California, 2003.
- [11] G. Buehrer, S. Parthasarathy, S. Tatikonda, T. Kurc, and J. Saltz, "Toward Terabyte Pattern Mining An Architecture-conscious Solution," *Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming*, San Jose, California, USA, 2007.
- [12] J.-F. ZHANG, H. SHI, and L. ZHENG, "A Method and Algorithm Of Distributed Mining Association Rules in Synchronisms," *First International Conference on Machine Learning and Cybernetics*, Beijing, November 2002.
- [13] T. SHINTANI and M. KITSUREGAWA, "Hash Based Parallel Algorithms for Mining Association Rules," *Fourth International Conference on Parallel and Distributed Information Systems*, Miami Beach, FL, USA, 1996.

Factors Which Influence the Recovery of Alcohol Addicts: A Second Follow Up Study

K. Burn-Thornton¹ and T. Burman²

¹School of Health Sciences & Social Care, Brunel University, Uxbridge, MIDDLESEX, UB8 3PH, UK

²Department of Computer Science, University of Durham, South Rd, DURHAM, DH1 3LE, UK

Abstract - *In this paper we describe further investigations into how patient, and counselor, characteristics appear to influence the degree (and speed) of recovery of alcohol addicts who are undergoing a program of therapeutic intervention . We compare these results with investigations carried out ten years ago and those which were carried out in parallel with this study.*

We show that the fastest recovery will be exhibited by a single addiction male, white, client who married. The counselor performing the intervention should also be female and white. In contrast the recovery of a multiple addiction female, non-white, client who is single, is on sickness benefit whose counselor is female and black, will be the slowest.

Keywords: *Addiction Recovery Rate.*

1 Introduction

This paper describes the results of a second follow up study into whether the results of an investigation into the factors which influence alcohol addiction recovery still hold ten years after the initial investigations was completed. There were two aims of this investigations.

The first was to determine if the original factors continued to impact upon the recovery of alcohol addicts and, if they did not, to determine what the probable cause, or causes, of the change in their impact. The second was to determine whether the results of this second follow up study are concomitant with those of the first follow up study which was conducted in parallel with this investigation.

We begin by describing the intervention process which is used for the alcohol addicts used in the sample and the means by which data for this process was recorded. This is followed by a description of the nature of the data used for the investigations and how it was cleaned prior to the investigations. The methodology used in the investigations is then described, followed by a discussion of the results obtained and conclusions which may be drawn. Future avenues for extending this work are then outlined.

2 Intervention Process

Recovery from alcohol addiction is thought to involve movement through recovery stages – this movement may be often recursive and iterative [2]. Figure 1 gives an indication of the recovery patterns for a rapidly recovering, and slow recovering, patients. More information regarding the process is given in reference 2.

The recovery process is supported by the attendance at counseling sessions. Counseling sessions are used to support recovery and counselors endeavor to determine which recovery stage the patient is currently occupying and hence gain an indication of the recovery pattern which the patient might be following.

The counseling sessions, from which this data was captured, usually involve one patient and one counselor but, to ensure a uniformity of ‘staging’ amongst the counselors (uniformity of application of the chosen intervention process), sessions with more than one counselor and more than one patient are often held. However, this process may differ in different organizations and different countries.

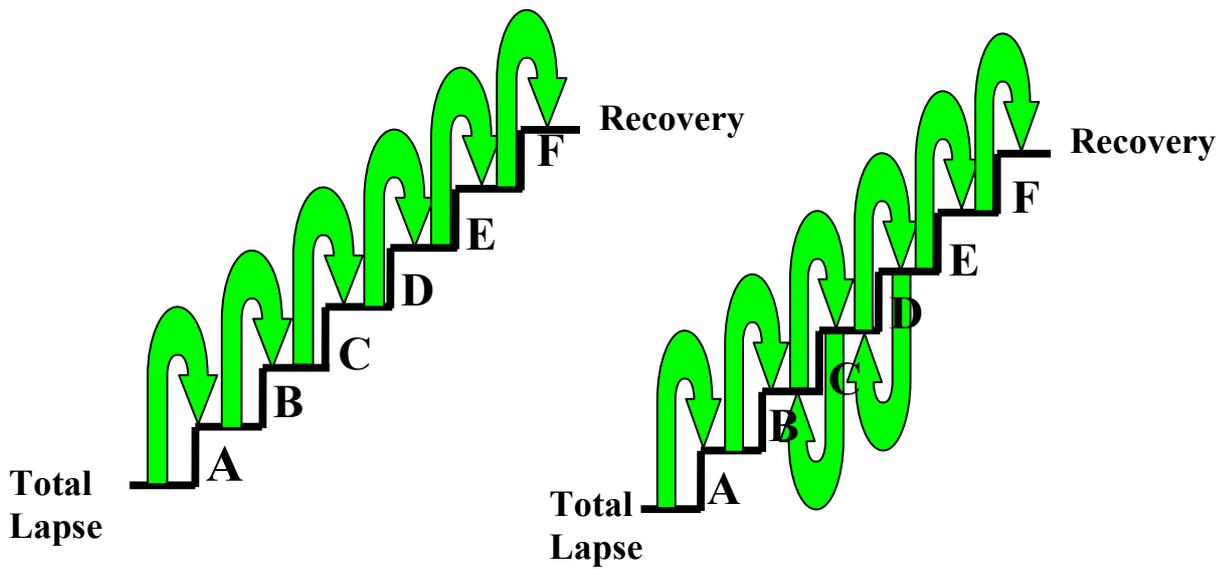
The following section describes how the data recorded during these counseling sessions, and other data, was used in these investigations.

3 Data Collection

The stage data recorded by the counselors during each session is ‘warehoused’ in a central data repository using human data input from the counselors recording/notes.

The nature of this method of data input may result in data entry errors, missing entries and incorrect data entries. However, such errors are minimized by the use of a ‘mask’ for data entry and appropriate checking of data input type. Although this does not prevent misreading of the original data by those who are inputting the data to the system. In addition, a set of allowable data values are predetermined and the entry checked against a value from this predefined set.

At each counseling session the apparent recovery stage reached by the patient is recorded by the counselor using a reference model of patient functionality against recovery



Example progression of 'recovery' Example progression of 'relapse'

Figure 1 – Example progression during recovery from alcohol addiction

Variable	Type	Nature	Set of allowed values
Unique Client Identification Number	Numerical	Quantitative	{0... infinity }
Current Stage of Recovery	Numerical	Qualitative	{1,2,3,4,5,6,7,8,9}
Addiction Type - Presenting	Categorical	Qualitative	{Alcohol, Drugs, Drugs/Alcohol}
Addiction Type - Apparent	Categorical	Qualitative	{Alcohol, Drugs, Drugs/Alcohol}
Location	Categorical	Quantitative	{Location 1.... Location?}
Year	Numerical	Quantitative	{2001.....2005}
Sex	Categorical	Quantitative	{M,F}
Marital status	Categorical	Quantitative	{S,M,D,B}
Ethnicity	Categorical	Quantitative	{COC,NONCOC}
Employment	Categorical	Quantitative	{company1, company?}
Counsellor	Categorical	Quantitative	{name1...name?}
Week No.	Numerical	Quantitative	{1....52}
Age	Numerical	Quantitative	{?...?}
Counsellor/Client Relative Ethnicity	Numerical	Quantitative	{1,0}
Counsellor/Client Relative Sex	Numerical	Quantitative	{3,4,6, 8} ==={MM,MF,FM,FF}

Figure 2 – Natures of the Variables used.

stage reached. An example of this reference model has already been described in reference [2].

This means that, in addition to the stage reached, for each counseling session, a minimum of twelve associated variable values were available for analysis. These were the minimum number of variables which were routinely, accurately, recorded for each patient at each location. Although most locations routinely accurately recorded many more variables this was not the case at all locations and this resulted in the reduction in variables used for these investigations. The variables used were also those used in the initial investigations.

This variable information included the current recovery stage (STAGE), attending addiction type (AAT), presenting addiction type (PAT), district (LOC), time in programme in terms of year (YR) and week number (WN), age (AGE), sex (SEX), marital status (MS), ethnicity (ETH), employment (EMPLOY) and counselor (COUNS). Information regarding key attributes of the counselor was also recorded.

Figure 2 shows the nature of the variables which were available for each patient. The figure shows that, of the thirteen variables recorded only three are qualitative - the recovery stage, AAT and PAT.

The information from the counseling sessions and the other patient information was merged into one database to facilitate these investigations. The merged data contained records from 12,112 counseling sessions for nearly 2,500 clients. The data set was the same size of that used in the first follow up study, the study which was carried out in parallel with this study but, as in the case of the first follow up study, the data set was a third of the size of the data set used in the original investigations. However, it contained same proportion of clients with regard to sex, location, employment, marital status, age, ethnicity, stage density, counselors, counselor/client sex, and ethnicity, ratio and session density.

The following section describes pre-processing, and data cleaning, which was carried out on the data prior to the investigations.

4 Data Cleaning & Pre-processing

Few records contained missing data, nearly 140, so that a decision was made to remove this small number of records from the data set because they constitute less than one percent of the total data available and/or less than one percent of the total data from one location(s). There was a proportionate reduction in missing data when compared to both the first follow up study and the original investigations.

In addition, as before, all categorical fields were replaced by numerical substitutions to aid in the analysis which requires numerical data. For consistency the same

substitutions used in the other investigations were used. No further modification of the data was carried out.

The substitutions for the LOC, AAT, PAT, MS, SEX, EMPL, ETH, EMPL and COUNS fields was achieved by sorting the possible categorical values in alphabetical order and then replacing the categorical value by a number representing the alphabetical order of the value i.e. the categorical value first in the alphabetical order was replaced by. This is described in detail in reference [2].

The following substitutions for AGE were made; 24 and under:1, 25-34:2, 35-44:3, 45-54:4, 55-64:5 and 65+:6. These substitutions are the same which were used for the age variable in the Isle Of Man section of the GENACIS study [5]. This will provide us with an opportunity to compare the results directly with current pan-European investigations.

In addition to this four new complex variables were created. Two were related time and two related to the client. The former were number of weeks on programme (NWP) and number of hours undergoing counseling (NHUS) which was later renamed to counseling sessions (SS). NHUS includes sessions which the patient has undergone on a one to one basis with a counselor and those attended with other 'recovers'. It is unclear whether these sessions should be regarded as the same but the group sessions constitute less than 1% of the sample so should have negligible impact on the recovery model. The latter were relative sex of client and counselor and relative ethnicity of client and counselor. These variables were assigned four, and two, values respectively.

The data was investigated using the methodology described in the following section which is that used in the first follow investigation and contains, as a sub-set, the methodology used in the original investigations.

5 Methodology

Using the merged data set as a starting point, for each client session the time in and time to each stage (in session number and weeks) was calculated from the entry point to the programme. Once the calculations had been carried out survival analysis was carried out for the data using time to all stages as the survival time.

Survival plots, and analysis, were carried out as a function of all the variables. As for the other investigations. results were obtained for all stages but this paper focuses upon the results for Stage9. This part of the methodology was identical to that of the original investigations.

In addition, regression analysis was carried out in order to determine which variables may be utilized to most accurately in order to describe the recovery behavior to Stage 9. This additional part of methodology was that also carried out in the first follow up investigations. However, the results of this

will not be discussed here and will be discussed elsewhere in another paper which is currently under revision.

6 Results

6.1 Survival Analysis

The survival plot for all clients reaching stage 9 is shown in Figure 3. Figure 3, a Kaplan Meier survival estimate, shows that the median time to reach Stage 9 is 29 weeks.

The following sub-sections provide a description of the specific dependencies of the recovery times upon the variables, and their recovery models. Unless stated it should be assumed that group into which the clients are divided is equally populated.

6.1.1. Gender

The male clients take seventy percent less time than the female clients to reach Stage 9 with male clients taking less than half a year of counseling to reach Stage 9.

This difference in recovery could be a consequence of the effect of the counselor/client characteristics referred to later in section 6.1.

6.1.2. Age

Clients in the age group 25-34 have a recovery time of 13 weeks which is approximately three times less than those aged 24 and under - the age band with the longest time, to reach Stage 9.

This longer recovery time could be a consequence of the fact that those in the younger age group significantly less ready to engage in the therapeutic intervention than those aged 25-34.

The recovery time is approximately equal across the other age groups with a recovery rate approximately half that of ages 25-34.

6.1.3. Location

The recovery of clients at location 4 is an order of magnitude less than that for the other locations. This could be a result of the route in which clients are referred such that they may be more ready to fully engage in the process than those from other locations.

No clients from locations 3,7, 9 or 10 reach Stage 9. This could be a consequence of the way in which the counsellors determine the stage for these clients and also the route by which the clients are referred.

6.1.4. Marital Status

Those of unknown marital status recover an order of magnitude faster than the other groups, the greatest difference is shown between this marital status and those who are single.

This suggests that although the group represent those of unknown marital status the clients may belong to an

unnamed group who have a impetus to recover. It could be reasonable to assume that this group includes those who are temporarily separated from their spouse whilst they endeavor to 'complete' the therapeutic intervention process. The group could also contain those who status is not defined on the list of assignable values for input. As this group could contain patients of different marital status it is better to choose the marital status with the next fastest recovery, married, which is 0.95 that of 'unknown'

6.1.5. AAT

Alcohol addicts, appear to have a recovery rate nearly twice as fast as that of multiple addicts The multiple addict group is taken to be a group which ranges from multiple drug addiction to drug and alcohol addiction.

6.1.6. Ethnicity

Non whites takes more than three times as long to reach stage 9, and hence progress on the recovery path than whites. This could be a direct consequence of the counselor/client ethnicity comparison but other factors may also be influencing this.

The results of the analysis taking to account relative ethnicity and sex of clients/counselors, described in later sub-sections, suggest that the former may be the case.

6.1.7. Employment

Those who are employed, with a median of 25 weeks of intervention, are the fastest to reach stage 9 with those who are disabled taking more than twice the time to reach stage 9, a median time of 55 weeks.

6.1.8. Counselor

It appears that counselor has a great impact upon recover in that there is an order of magnitude of recovery rate between the 'slowest' and 'fastest' counselors.

6.1.9. Relative Client/Counselor Ethnicity

The same ethnicity of client, and counselor, whether it be ethnic group 0/ ethnic group 0 of ethnic group 1/ ethnic group 1, represents the fastest recovery rate being 1.6 times faster than a ethnic group 0/ ethnic group 2 paring.

6.1.10. Relative Client/Counselor Gender

A M:F relative sex of client, and counselor, represents the fastest recovery rate being approximately 1.5 faster than a F:F paring.

It is worth noting that both the F:M and M:M parings resulted in no clients reaching stage 9.

6.1.11. Counseling Session Number

There is a direct relationship between the frequency of counseling sessions (Number of sessions with time (weeks)) and the time taken to reach Stage 9 in that those who experience a higher frequency of sessions take a shorter time to reach stage 9.

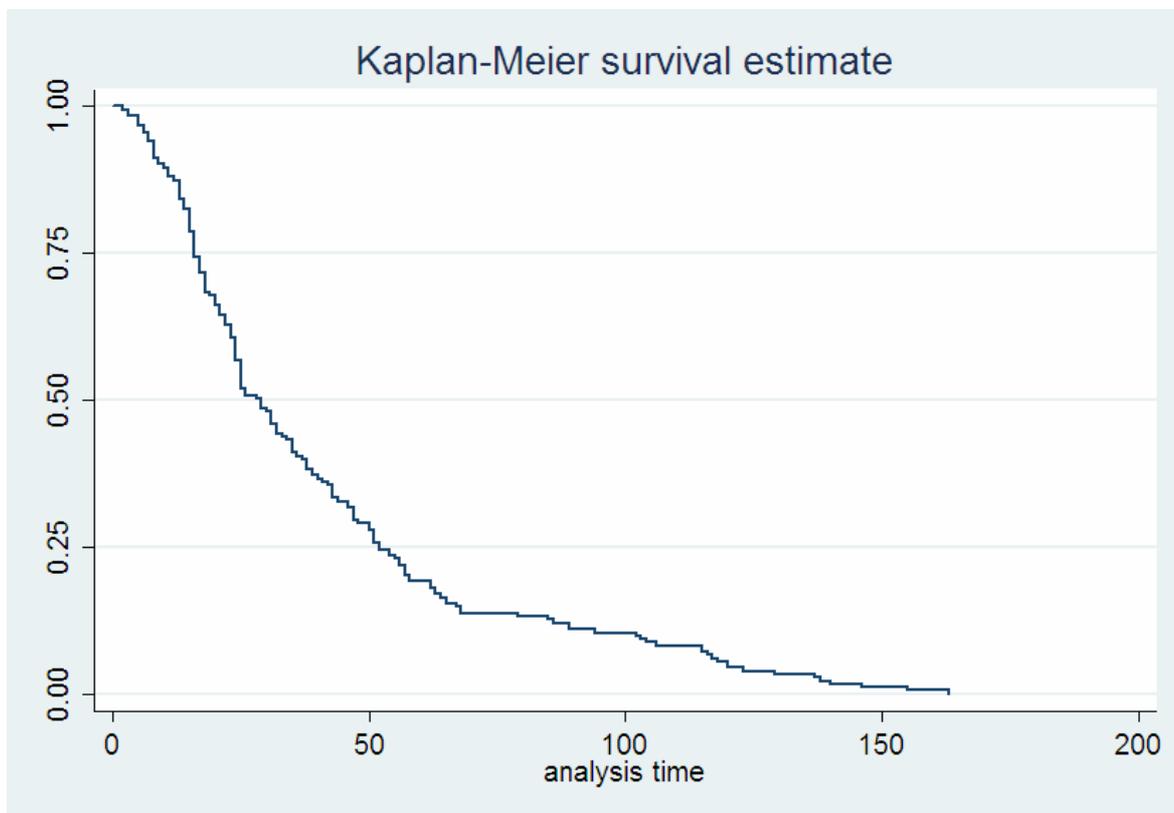


Figure 3 – Survival Analysis of Time to Stage 9 (Time – weeks)

7 Comparison of Results with Initial Investigations (& Follow Up) Investigations

The results of this follow up analysis show that the fastest recovery will be exhibited by a single addiction male, white, client who married. The counselor performing the intervention should also be female and white. In contrast the recovery of a multiple addiction female, non-white, client who is single, is on sickness benefit whose counselor is female and black, will be the slowest.

For the worst recovery, these results are in agreement with the results of the initial investigations but for the best recovery they are in agreement with the exception of the marital status.

This lack of agreement results solely from the unknown marital status of those achieving best recovery, this could, of course, include married but it has not been possible to find the true value of the marital status of the whole of this group with certainty. If the next fastest group of known marital status, married, is included there is agreement.

However, all results are in agreement with those obtained in the parallel investigations, the first follow up investigations.

Thus, within the accuracy of data capture and variable value assignment, there appear to agreement of these follow up results with the results of the initial investigations. The exact agreement between the two follow up investigations, carried out from data collected over the same time period, suggests that any lack of agreement in the models, and observed behaviors, may be caused by relaxing in protocols for staging, and data capture, over the ten year period.

8 Conclusions

We have shown that not only is the recovery of addicts dependent upon client sex, marital status, ethnicity, employment type, number of sessions & presenting type but that it is also dependent upon the ethnicity and sex of the counselor performing the intervention.

The fastest recovery will be exhibited by a single addiction male, white, client who married. The counselor performing the intervention should also be female and white. In contrast the recovery of a multiple addiction female, non-white, client who is single, is on sickness benefit whose counselor is female and black, will be the slowest.

These results are important because they describe the results of a different group of counselors following the same protocols for the intervention staging process but for a different groups of clients with the same ethnic, employment, location and marital status proportions of the original group.

The results confirm the results of the initial investigations in that they confirm the suggestion of the initial investigations that the values assigned to session number (SS), relative counselor/client sex (CCSEX), employment (EMPL) and counselor (COUNS) for a specific client can be used to gain an indication of whether, or not, the client will have a fast, or slow, recovery or recover at all. This is especially so for the clients at certain locations where no clients have been deemed to reach Stage9.

9 Future Work

A natural progression of this work will be to investigate how the recovery model changes, for best and worst cases, when other the effect of other variables such as age, previous intervention, social class and nuclear family group are taken into account.

This information was not available during these investigations but it is currently being sourced from disparate locations within the survey area.

It is also envisaged that future work will include the investigation of the use of emerging Data Mining approaches – many of which were not available to us at the beginning of the initial investigations.

We are currently investigating the use of Markov models to explain the inter-stage behavior in an analogy with patient care management in institutions.

10 References

- [1] M.L. Plant, P. Miller and M.A. Plant, 'Spreading out or concentrating weekly consumption: Alcohol Problems and other consequences within a UK population sample', *Alcohol & Alcoholism*, 40: 461-468, 2005.
- [2] K.E.Thornton, G.Smerdon & M.Findlay, 'Discovering Recovery Patterns using Data Mining', *Expert Update - Knowledge based Systems & Applied Artificial Intelligence*, vol.3 no. 3, p25-30, ISSN 1465-4091, January 2001.
- [3] K.E. Burn-Thornton & A. Lazzarini, 'Achieving more accurate classification of alcohol data using pre-processing', in *Proc. Workshop: IEEE ICDM International Conference on Data Mining*, Dec 2002, Japan.
- [4] K.E. Burn-Thornton & A. Lazzarini, 'The use of Reliability Techniques to predict recovery rate of alcohol addicts', *JASIST*. 56(4), pp. 356-363, 2005.
- [5] M.L. Plant, P. Miller and M.A. Plant, 'The relationship between alcohol consumption and problem behaviours: gender differences among British adults', *Journal of Substance Use*, 10:2005.

11 Acknowledgements

The authors would like to thank the Royal Academy of Engineering for the grant which supported this work and ADDCATION for their help in data provision and knowledge of the intervention process.

Supervised DAG based Data mining model for DNA Sequence Analysis and Pattern Discovery

Syed Shams-ul-Haq¹, Muhammad Nadeem²

¹ & ²Department of Computing, SZABIST, Karachi, Sindh, Pakistan

¹shams@szabist.edu.pk, ²nadeem@szabist.edu.pk

Abstract – The DNA sequencing and annotation is by nature computationally intensive task. Exponential growth of bio-sequences in public genomic databanks on the web enforces the need for efficient information extraction methods. The large volume of full scale genomic sequences in databases prohibits homology based extrinsic and data mining algorithms with running time $O(n^2)$ or higher order to perform exhaustive search. This research presents a novel Directed Acyclic Graph base data mining approach for common patterns identification, discovery and matching in large data sets. A practical $O(n)$ pattern matching algorithm is presented. The algorithm employs dynamic Nucleic-Quad Trees, efficient hashing and histogram based techniques to find all possible matches among arbitrary number of bio-sequences in linear time. The proposed mining model enables discovery of complex deep gene structures and patterns in an effective way.

Keywords: Directed Acyclic Graph, Nucleic-Quad Trees, Data Mining, DNA Sequence Analysis, Gene Annotation, Pattern matching and discovery.

1. Introduction

The Computational Molecular Biology [1], an interdisciplinary field of study, is concerned with diversified subjects such as molecular biology, computer sciences, statistics, mathematics, physics and chemistry. The central dogma of molecular biology states that “Deoxyribonucleic acid (DNA) acts as a template to replicate itself, DNA is also transcribed into Ribonucleic acid (RNA), and RNA is translated into protein.”

Deoxyribonucleic acid (DNA) is defined as a linear polymer made up of individual chemical units known as nucleotides or bases. The four deoxyribonucleotides that make up the DNA sequences of known living things are Adenine, Guanine, Cytosine, and Thymine denoted by A, G, C, and T, respectively. The order of deoxyribonucleotides in linear DNA sequence determines the organism function and contains instructions that build an organism. The complete DNA sequence that codes for a living thing is called its Genome. The Genome is divided into individual genes.

A genome give static view of sequence, whereas micro-array experiments yields gene expression patterns that offers dynamic information about cell function. Microarray technologies have been utilized to evaluate the level of expression of thousands of genes in breast, colon and blood cancer treatment [2]. The tremendous volume of the data prohibits view of large microarray clustering result on a 2D or 3D visual display. The need for efficient interactive visualization tools to facilitate pattern extraction from microarray datasets is growing day by day.

This research aims to provide a novel approach for DNA sequencing. It applies supervised directed acyclic graph based pattern matching algorithm for discovery of unique or common frequently occurring patterns among arbitrary number of bio-sequences. Our algorithm improves upon the running time of widely applied homology search and data mining algorithms enabling researchers to analyze more complex structural similarities and functionality of genes in different species.

2. Literature Review

The state of the art in the field of bioinformatics is presented. The overview of widely used tools, techniques and methodologies along with challenges in the area of sequence analysis and biological data mining are discussed.

2.1 Gene Annotation Systems

The AbXtract [3] is one of the early attempts to characterize the function of genes and proteins based on information extracted from Bio-Literature text. The system follows a rule-based approach where keyword's frequency determines relevant keywords to be assigned to protein families. Another system proposed by [4] performs automated gene annotation with keywords dig from different bio-literature abstracts relying on mappings between different ontologies. BioRAT [5] is rule based system relies on large number of rules derived exclusively from patterns inserted by user. It searches and highlights most pertinent facts from abstracts or full texts against a user query. Textpresso [6] is also rule based system that searches and mark documents with terms from a built-in ontology knowledge base. BioIE [7] another

rule based system that primarily uses Natural Language Processing (NLP) techniques, sense disambiguation and syntactic dependent rules to extract biological interactions from texts and annotate them with Gene Ontology (GO) terms. A morphology based system presented by [8] annotates gene, protein, and families with Gene Ontology (GO) terms extracted from texts.

A case-based text-mining system proposed by [9], identifies relationships among biological entities such as protein, gene or cell cycle. MeKE [10] extracts protein functions from Bio-Literature based on sentence alignment and statistical classifier. A helper tool, GOAnnotator [11] assists the gene ontology annotation of UniProt [12] entries. The extraction of GO terms is performed by FiGO [13] that takes text as input and returns the detected GO terms.

2.2 Bioinformatics Tools

A number of data mining, machine learning, and statistical analysis tools have been developed for bio-data exploration and analysis. [14], [15], [16], [17] discusses bioinformatics tools and data mining methods. The pioneering researchers in 1960s developed methods to predict phylogenetic relationships of the related protein sequences during evolution [15]. PHYLIP and PAUP [18] use probabilistic models to construct phylogenetic trees.

The primitive dynamic programming methods of finding the best alignment between two sequences of lengths m and n runs in $O(mn)$ [19], [20]. This technique becomes infeasible due to high time and space complexity. Heuristics-based searches that employ hash tables outperform traditional dynamic programming methods. BLAST a pair-wise alignment tool and ClustalW [21], a multiple alignment tool are based on dynamic programming algorithm [22]. Some of the important tools that employ hash table based approach are BLAST [23], Fast Alignment (FASTA) [24], PSI-BLAST [25], MegaBLAST [26], BL2SEQ [27], SENSEI [28], FLASH [29], PipMaker (BLASTZ) [30], BLAT [31], GLASS [32], and PatternHunter [33]. These tools primarily build a hash table on either the query sequence or the database sequence (or both) for all possible substrings of a predefined size δ .

A number of homology search tools that use suffix trees [34] approach, include MUMmer [35], QUASAR [36], AVID [37], and REPuter [38]. A suffix array is constructed by QUASAR on one of the bio-sequences, the number of exactly matching seeds are then counted based on this suffix array. A region is searched by the BLAST only in case if the number of exactly matched seeds is greater than predefined threshold. Prokaryotic genomes can be accurately predicted by number of bioinformatics tools such as GeneMark [39] and Glimmer [40] using HMM and other Markov models. Similar approaches were taken to develop eukaryotic gene prediction tools such as GeneScan [41] and GRAIL [42].

2.3 Biological Data Mining Tools

A number of web based bioinformatics data-mining systems, such as MAGPIE [8] and GeneQuiz [43] have integrated different functionality and aspects of bio-data. These tools employ web based server-side architectures and uses web browsers as their primary interface to interact with users. MAGPIE focuses on genome sequencing project management, ORF (Open Reading Frame) identification, and metabolic pathways. It provides output in HTML (HyperText Markup Language) format. GeneQuiz makes completely automatic gene function predictions, store output in a database and provide HTML views of the output tables. GeneMine [44] focuses on gene or protein structure-function. It combines information such as disease associations, genomic mapping, protein functional features, and structural conservation patterns. The visualization and modeling component of GeneMine is (LOOK) [45].

3. Common Data Formats

A survey of common bio-data file formats used for storing bio-sequences and their suitability particularly for the large-scale sequence analysis applications is evaluated.

3.1 HTML or ASCII Text

ASCII Text files are loosely tagged texts, represented by HTML on the web [48]. Web-based applications work by taking simple text queries as input and apply homology search on bio-sequences stored as large strings in relational databases. The lexicographical comparison complexities of bio-sequence in HTML or Text format makes pattern matching task computationally intensive and limits number of concurrent comparisons. It is human readable, extendable and unstructured tagged data format containing meta-sequence information, tags, comments, and bases as ASCII characters in a sequence.

3.2 SCF Data Format

SCF format files are used to store data from DNA sequencing instruments. Each file contains the data for a single reading and includes: its trace sample points, called sequence, the positions of the bases relative to the trace sample points, and numerical estimates of the accuracy of each base. Comments and "private data" can also be stored. This format is machine independent, for details see [46].

3.3 ZTR Data Format

The ZTR [47] is a binary based bio-data format. The basic structure of a ZTR file is (header, chunk*) – i.e. header followed by zero or more chunks. The header consists of an 8 byte magic number, followed by a 1-byte major version number and 1-byte minor version number. The first byte of the data consists of a format byte. The most basic format is zero - indicating that the data is "as is". Other values are used to encode various filtering and compression techniques.

4. AN-DNA Data Format Model

This section presents AN-DNA Data format model, a new SIMD optimized binary data file format proposed by authors. The format is specifically designed for optimal usage by the modern processors. New file format improves upon the existing file formats [46], [47], [48]. It outperforms other file formats in disk storage and memory footprints, along with L1 and L2 processor cache friendly structure. The DNA strand consists of long array of [A / T / G / C] nucleotides. Mathematically it can be expressed as follows:

$$\gamma_i^p = (a | t | g | c)_i^p, \quad \ddot{C}_i = [\gamma^0, \gamma^1, \gamma^2]_i \quad (1,2)$$

$$DNA_n = \langle \ddot{C}_1, \ddot{C}_2, \ddot{C}_3, \dots, \ddot{C}_n \rangle \quad (3)$$

The codon \ddot{C}_i comprises of three nucleotides (1), (2). DNA_n sequence is defined in terms of series of codons (3).

$$DNA_n = \left\langle \left[\gamma^0, \gamma^1, \gamma^2 \right]_1, \left[\gamma^0, \gamma^1, \gamma^2 \right]_2, \left[\gamma^0, \gamma^1, \gamma^2 \right]_3, \dots, \left[\gamma^0, \gamma^1, \gamma^2 \right]_n \right\rangle \quad (4)$$

The information contained in a DNA strand is either presence or absence of nucleotide [A/T or G/C] at given position in chain [A/T] and [G/C] are complementary so they can be encoded into binary for processing. This encoded binary sequence can be compacted and stored in a more efficient manner as shown in (5).

$$AT_i^p = (1|0)_i^p, \quad GC_i^p = (-AT_i^p)_i^p \quad a|t=1, g|c=0 \quad (5)$$

The nucleotides γ_i^p in DNA binary sequence (6) indicate presence or absence of γ_i^p . Obtained from putting A/T rule value given by (5) in (4).

$$DNA_n = \left\langle \left[\underbrace{1^0, 0^1, 1^2}_{c_1} \right]_1, \left[\underbrace{1^0, 0^1, 0^2}_{c_2} \right]_2, \left[\underbrace{1^0, 1^1, 1^2}_{c_3} \right]_3, \dots, \left[\underbrace{1^0, 0^1, 1^2}_{c_n} \right]_n \right\rangle \quad (6)$$

4.1 Format Specifications

The binary file header starts with 1-byte integer padding, 1-byte data padding and 4-byte data sequence length identifying number of 32-bit integers containing the bit pattern in a data file. The integer padding byte identifies number of padding bits (zeros) appended in bit-pattern due to 32-bit alignment, valid value range for integer padding is from 0 to 31. The second data padding byte determines number of bits padded due to incomplete DNA sequence. This value should be zero in normal circumstances. Three periodicity [49] property of DNA sequence implies that the DNA strand should be divisible by 3. The non-zero value indicates incomplete original data and number of zeros appended in bit pattern in order to make it divisible by 3, the

valid value range is from 0 to 2. The beginning tag and end mark is used to uniquely identify file format and its version. Figure 1 illustrates the file structure of the v2 bio-data file.

8 bytes Identity tag			
V No.1	V No.2	V No.3	V No.4
AT	Unused	D Pad	I Pad
4 bytes sample length			
4 bytes Data 1			
... ..			
4 bytes Data η			
4 bytes End Mark			

Figure 1: AN-DNA Data format structure

The format type is a binary mode file that is not human readable. The size of the file is kept aligned with 8 byte boundary for optimal disk and memory access. Mathematically AN-DNA v2 bio-sequence is represented as:

$$DNA_n^{\nu 1} = \langle \ddot{C}_0[\eta], \ddot{C}_1[\eta], \ddot{C}_2[\eta] \rangle \quad (7)$$

$$\ddot{C}_0[\eta] = [\gamma_1^0, \gamma_2^0, \gamma_3^0, \gamma_4^0, \gamma_5^0, \dots, \gamma_{\eta-1}^0, \gamma_\eta^0] \quad (8)$$

$$\ddot{C}_1[\eta] = [\gamma_1^1, \gamma_2^1, \gamma_3^1, \gamma_4^1, \gamma_5^1, \dots, \gamma_{\eta-1}^1, \gamma_\eta^1] \quad (9)$$

$$\ddot{C}_2[\eta] = [\gamma_1^2, \gamma_2^2, \gamma_3^2, \gamma_4^2, \gamma_5^2, \dots, \gamma_{\eta-1}^2, \gamma_\eta^2] \quad (10)$$

4.2 Attribute Selection

Attribute selection of bio-sequences may involve specification of particular tagged regions or genes in the entire sequence, rather than choosing entire bio-sequence for mining purpose. For example, a tagged region is represented by the base-pair offsets. Nucleotides falling in the range starting from index 301 to 833 inclusive will be considered for sequence analysis. It is expressed as:

REFERENCE 1 (bases 301 to 833)

The DNA_η , comprises of codons $\ddot{C}_s^1, \ddot{C}_s^2, \ddot{C}_s^3, \dots, \ddot{C}_s^\eta$ expressed by (4). A region of interest is defined as a subset DNA_η^s of entire bio-sequence DNA_η .

$$DNA_\eta^s \subset DNA_\eta \quad \because DNA_\eta^s = DNA_\eta - \overline{DNA_\eta^s} \quad (11)$$

The starting index is denoted by α_η^{si} to ending index β_η^{si} in original sequence.

$$\overline{DNA_\eta^s} = \langle \ddot{C}_s^1, \ddot{C}_s^2, \dots, \ddot{C}_s^{\alpha-1}, \ddot{C}_s^{\beta+1}, \dots, \ddot{C}_s^\eta \rangle \quad (12)$$

Subtracting $\overline{DNA_\eta^s}$ (12) from original sequence (6) we get:

$$DNA_\eta^s = \left\langle \left[\gamma^0, \gamma^1, \gamma^2 \right]_\alpha, \dots, \left[\gamma^0, \gamma^1, \gamma^2 \right]_\beta \right\rangle \quad (13)$$

$$DNA_s^\eta = \langle \ddot{C}_s^\alpha, \dots, \ddot{C}_s^\beta \rangle \quad (14)$$

Different selected regions in the DNA sequence to be analyzed are given by:

$$D\tilde{N}A_\eta^\delta = \left[DNA_\eta^{s1}, DNA_\eta^{s2}, \dots, DNA_\eta^{s\delta} \right] \quad (15)$$

4.3 Data Cleansing

Once attributes and sequences are selected, the data preprocessing phase begin with identification and then removal of outliers from selected bio-sequences, the outliers usually are unselected regions (12) in the DNA sequences or integer padding bits appended at the tail of sequence in order to make it 8 byte boundary aligned. The transformation part constitutes the assembling of bio-data bit-patterns in memory or disk, to be used by mining algorithm in the next phase. The bit-patterns are split into three codon ith nucleotide array sequences. These arrays contain interleaved nucleotides of each codon in the DNA sequence.

5. DAG Mining Model

The heart of DNA sequence analysis using extrinsic algorithms [50] is homology search for gene annotation, the identification and prediction of protein coding regions in bio-sequences. The data mining task of pattern matching can be applied to discover the similarities between two or more bio-sequences, or even finding repetitive patterns contained inside a sequence itself.

5.1 Nucleic-Quad Trees

We present a novel technique for representing bio-patterns using special tree based structure named ‘‘Nucleic-Quad Trees’’. All possible combination nucleotides configurations can be represented by the nucleic-quad tree.

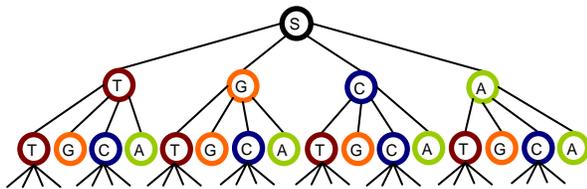


Figure 2: Nucleic-Quad Tree Expansion

A depth of the tree records the length of the pattern, the tree grows as $O(4^n)$, where $(n+1)$ is the total number of levels of the nucleic-quad tree containing 4^n Nodes. Growth of the tree is only dependent upon the found patterns in sequence during the first time traversal. It is represented as $NQ(\eta)$ -Tree, where η is the tree depth. A set of all possible two nucleotide configurations are generated as: $N = \{AA, AT, AG, AC, GA, GG, GT, GC, CC, CT, CA, CG, TA, TG, TT, TC\}$ these are encoded as one nibble. The tree is build dynamically while traversing the bio-sequence. Sub-patterns found in sequence are captured and stored in Histograms.

5.2 Sub-Pattern Histograms

The histogram is constructed based upon all possible generated permutation of the two nucleotides that contains the occurrence frequency of particular nucleotide pair configuration. Formally,

$$\tilde{H}g_\gamma^s[\eta] = \left\langle \hat{f}_\gamma^1, \hat{f}_\gamma^2, \hat{f}_\gamma^3, \dots, \hat{f}_\gamma^{\eta-1}, \hat{f}_\gamma^\eta, \sum_{v=1}^{\eta} \hat{f}_\gamma^v \right\rangle_s \quad (16)$$

$$\tilde{H}g^s[\eta] = \left(\tilde{H}g_A^s[\eta], \tilde{H}g_T^s[\eta], \tilde{H}g_G^s[\eta], \tilde{H}g_C^s[\eta] \right)_s \quad (17)$$

A histogram (17) is represented as set of frequencies of distinct nucleotides (16) sub-histogram of length η each.

Table 1: Histogram for nucleotide frequency in bio-pattern

γ_i^s	$\hat{f}_\gamma^s[1]$	$\hat{f}_\gamma^s[2]$		$\hat{f}_\gamma^s[\eta-1]$	$\hat{f}_\gamma^s[\eta]$	$\sum \gamma_i^s$
A_c	\hat{f}_a^1	\hat{f}_a^2		$\hat{f}_a^{\eta-1}$	\hat{f}_a^η	$\sum_{v=1}^{\eta} \hat{f}_a^v$
T_c	\hat{f}_t^1	\hat{f}_t^2		$\hat{f}_t^{\eta-1}$	\hat{f}_t^η	$\sum_{v=1}^{\eta} \hat{f}_t^v$
G_c	\hat{f}_g^1	\hat{f}_g^2		$\hat{f}_g^{\eta-1}$	\hat{f}_g^η	$\sum_{v=1}^{\eta} \hat{f}_g^v$
C_c	\hat{f}_c^1	\hat{f}_c^2		$\hat{f}_c^{\eta-1}$	\hat{f}_c^η	$\sum_{v=1}^{\eta} \hat{f}_c^v$

The table 1 shows the histogram record for a bio-pattern of length η . For each nucleotide that contains the occurrence frequency of particular nucleotide at all possible positions.

$$DNA_s^\eta = \left\langle \gamma_s^1, \gamma_s^2, \gamma_s^3, \dots, \gamma_s^{\eta-1}, \gamma_s^\eta \right\rangle \quad (18)$$

$$\hat{\rho}_s = MIN(\eta, 2^m), m \in \mathbb{Z}^+ \quad (19)$$

In bio-data (18) there can exist only (19) sub patterns. For large η , all of sub patterns can not be stored in memory. The problem can be solved by building a disk cache for storing the partial sequences or sub-patterns.

5.3 Frequent Patterns Discovery

The discovery of frequently occurring motifs, a kind of sequential and structured patterns such as biochemical structures could uncover important information essential for the data analysis. If the pattern is frequently occurring in a bio-sequence it must be performing some meaningful functions and ought to play important role. In our approach all possible permutation of small frequently occurring patterns are stored into hashed array, containing the pattern ID as key and frequency of occurrence as value. Potential new patterns and structures in the bio-data are identified by the occurrence frequency of the sub-pattern.

$$\tilde{P}_n^\sigma = \left[S_n^1, S_n^2, S_n^3, \dots, S_n^{\sigma-1}, S_n^\sigma \right], \forall \tilde{P}_n^\sigma[i] \subset DNA_s^\eta \quad (20)$$

$$Pf_s^\sigma = \left[\hat{f}_s^1, \hat{f}_s^2, \hat{f}_s^3, \dots, \hat{f}_s^{\sigma-1}, \hat{f}_s^\sigma \right] \quad (21)$$

Common frequent patterns in the class of sub-patterns (20) of length σ (sigma) are determined by finding highest frequencies in (21).

Table 2: Histogram for nucleotide frequency in bio-pattern

PID	$DNA_s^\eta = \langle \dots, \overset{\cdot\cdot}{C}_s^1, \overset{\cdot\cdot}{C}_s^2, \dots, \overset{\cdot\cdot}{C}_s^\eta \rangle$	Freq
1	$S_n^1 = \langle \gamma_1^1, \gamma_2^1, \gamma_3^1, \dots, \gamma_{n-1}^1, \gamma_n^1 \rangle$	\hat{f}_s^1
\vdots	\vdots	\vdots
σ	$S_n^\sigma = \langle \gamma_1^\sigma, \gamma_2^\sigma, \gamma_3^\sigma, \dots, \gamma_{n-1}^\sigma, \gamma_n^\sigma \rangle$	\hat{f}_s^σ

Table 2 shows all possible permutations of nucleotide configurations in the bio-pattern of fixed length η with their respective occurrence frequencies. The Table 2 provides critical information about potential new frequent as well as rare (not so frequent or unique) patterns.

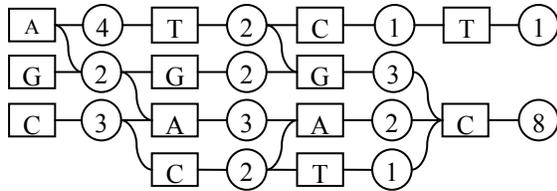


Figure 3: DNA frequency Directed Acyclic Graph
 $DNA[s] = \{ \dots ATGCGAACAACAGGCATCTCAACCCTCCCACGGGC \dots \}$, 4-tuple ($\eta = 4$) Patterns captured by the Directed acyclic graph are given by set $P = \{ATCT, ATGC, AGGC, AAAC, GGGC, GAAC, CAAC, CCAC, CCTC\}$ as shown in figure 3. The homology search with gaps is performed by dropping nodes from the DAG. Pattern alignment is achieved by skipping nodes from the start.

6. Results

The experimental results are based on the AN-BIO Miner v1.0, a Directed Acyclic Graph mining tool developed by authors. It implements Nucleic-Quad Tree based pattern matching and AN-DNA Data model presented in this paper. AN-DNA v1.0 and v2.0 data format are supported by mining application for DNA strand processing and storage. A data conversion tool that import and export DNA trace data in different formats is also developed as part of the research work. The web based content mining application architecture [51] and SIMD optimizations [52] can be applied to construct scalable web interface.

6.1 Data Model Comparative Analysis

Figure 4 plots a graph of bio-data stored in three different formats. The “Bio-Data File Size” bars demonstrate the file size differences in selected formats. The lowest file size

offers the highest advantage in terms of saving of disk space, lower value is better in first bar series. The “Size Reduction Gain” Bars plots the gain offered due to disk space savings, the greater value shows higher gain.

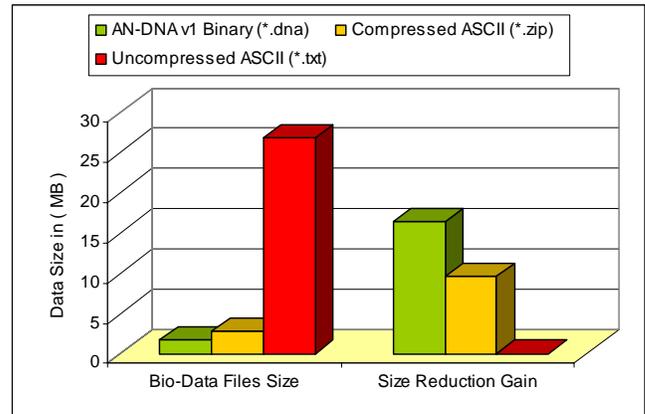


Figure 4. Data formats size comparison

6.2 Gene Sub-Pattern Matching

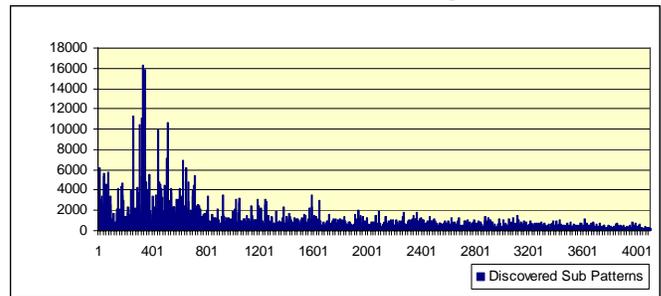


Figure 5. Frequency plot of all unique Sub-Patterns
 The Figure 5, shows the all (more than 4000) distinct sub-patterns of length ($\eta = 6$) found in a bio-sequence. The y-axis plots the frequency of each unique sub-pattern and x-axis represents each distinct sub-pattern by id. Bio-sequence bases were not shifted in NQ-Tree construction, so the base offset value remains zero.

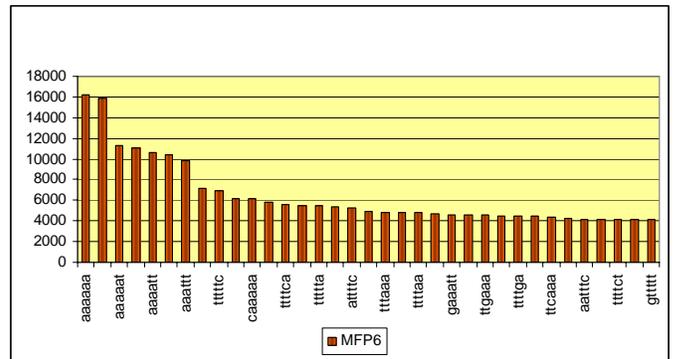


Figure 6: Most Frequent Sub-Patterns $\eta = 6$.

The Figure 6 plots 35 top most frequently occurring sub-patterns of length 6 in the descending order. The y-axis plots the frequency of each unique sub-pattern and x-axis represents each distinct sub-pattern.

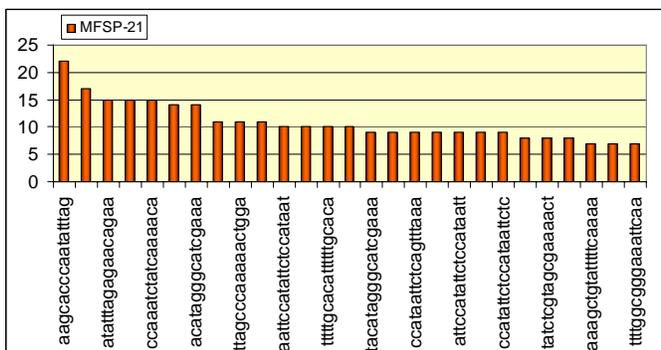


Figure 7: Most Frequent Sub-Patterns $\eta = 21$.

Similarly Figure 7 shows the bar graph of 27 most frequently occurring sub-patterns having length ($\eta = 21$). The frequency of occurrence of larger sub-patterns decline exponentially. There were more than 10000 sub-patterns of $\eta = 6$ at an average, whereas there are only 10 to 15 sub-patterns of $\eta = 21$ at an average in bio-data.

6. Conclusion

Our approach to data mining of biological data for common patterns identification, discovery and comparison presented a fast linear time $O(n)$ solution for finding and discovering patterns based on the dynamic Nucleic-Quad Trees, efficient hashing algorithms and histograms. We have presented a CPU register and cache friendly data model where data is encoded into the bit-pattern for fast CPU register processing, specifically optimized for vertical SIMD operations.

The DAG mining model enables discovery of complex deep gene structures and sub-patterns within a bio-sequence in an efficient manner, further it is used for fast linear time homology search among arbitrary number of large bio-sequences. The DAG Mining model is particularly suited for comparison and analysis of millions of large scale genomic sequences. The techniques presented in this paper can be further explored and applied to solve other computationally intensive bio-informatics problems.

7. References

- [1] Salzberg, S. L., Searls, D.B., and Kasif, S., "Computational Methods in Molecular Biology", Elsevier Sciences B. V., Amsterdam, 1998.
- [2] "Special Issue on Bioinformatics, Part I: Advances and Challenges," IEEE Proceedings, vol. 90, Nov. 2002.
- [3] Andrade, M. and Valencia, A. "Automatic extraction of keywords from scientific text: Application to the knowledge domain of protein families", Bioinformatics, vol. 14, issue 7, pp. 600-607., 1998.
- [4] Pérez, A., Perez-Iratxeta, C., Bork, P., Thode, G., and Andrade, M. "Gene annotation from scientific literature using mappings between keyword systems", Bioinformatics, vol. 20, issue 13, pp. 2084-2091, 2004.

- [5] Corney, D., Buxton, B., Langdon, W., and Jones, D. "BioRAT: Extracting biological information from full-length papers", Bioinformatics, vol. 20 issue 17, pp. 3206-3213, 2004.
- [6] Müller, H., Kenny, E., and Sternberg, P. "Textpresso: An ontology-based information retrieval and extraction system for biological literature", PLOS Biology, vol. 2 issue 11, E309, 2004.
- [7] Kim, J., & Park, J. "BioIE: Retargetable information extraction and ontological annotation of biological interactions from literature", Journal of Bioinformatics and Computational Biology, vol. 2, issue 3, pp. 551-568., 2004.
- [8] Koike, A., Niwa, Y., and Takagi, T. "Automatic extraction of gene/protein biological functions from biomedical text", Bioinformatics, vol. 21, issue 7, pp. 1227-1236, 2005.
- [9] Palakal, M., Stephens, M., Mukhopadhyay, S., and Raje, R., "Identification of biological relationships from text documents using efficient computational methods", Journal of Bioinformatics and Computational Biology, vol. 1, issue 2, pp. 307-342., 2003.
- [10] Chiang, J., and Yu, H. "MeKE: Discovering the functions of gene products from biomedical literature via sentence alignment", Bioinformatics, vol. 19, issue 11, pp. 1417-1422., 2003.
- [11] Rebholz-Schuhmann, D., Kirsch, H., and Couto, F., "Facts from text - is text mining ready to deliver? ", PLOS Biology, vol. 3, issue 2, e65, 2005.
- [12] Apweiler, R., Bairoch, A., Wu, C., Barker, W., Boeckmann, B., Ferro, S., et al., "UniProt: The universal protein knowledgebase", Nucleic Acids Research, (Database issue), vol. 32, pp.115-119, 2004.
- [13] Couto, F., Silva, M. and Coutinho, P., "Finding genomic ontology terms in text using evidence content", BMC Bioinformatics, vol. 6, sup. 1, s21, 2005.
- [14] Han, J., and Kamber, M., "Data Mining: Concepts and Techniques", Morgan Kaufmann, San Francisco, 2001.
- [15] Mount, D., "Bioinformatics: Sequence and Genome Analysis", Cold Spring Harbor Laboratory Press, Woodbury, NY, 2001.
- [16] Hastie, T., Tibshirani, R. and Friedman, J., "The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer-Verlag, New York, 2001.
- [17] Lopresti, D. P., and Tomkins, A., "Block edit models for approximate string matching", Theoretical Computer Science, vol. 181, issue 1, pp. 159-179, 1997.
- [18] Felsenstein, Phylip, J., "phylogeny inference package 3.2", Cladistics, vol. 5, pp. 164-166, 1989. <http://evolution.genetics.washington.edu/phylip.html>.
- [19] Needleman, S. B. and Wunsch, C. D., "A general method applicable to the search for similarities in the amino acid sequence of two proteins", Journal of Molecular Biology, vol. 48, pp. 443-453, 1970.
- [20] Smith, T. F. and Waterman, M. S., "Comparison of biosequences", Adv. Appl. Math., v2, pp.482-489, 1981.

- [21] Higgins, D. G. and Sharp, P. M., "Clustal: a package for performing multiple sequence alignment on a microcomputer", *Gene*, vol. 73, pp. 237-244, 1988.
- [22] Smith, T. F. and Waterman, M. S., "Identification of common molecular subsequences", *Journal of Molecular Biology*, vol. 147, issue 1, pp.195-197, 1981.
- [23] Altschul, S. F., Gish, W., Miller, W., Myers, E. W. and Lipman, D. J. "Basic Local Alignment Search Tool", *Journal of Mol. Biology*, vol. 215, pp. 403-410, 1990.
- [24] Pearson, W. R. and Lipman, D. J., "Improved tools for biological sequence comparison", *Proceedings of the National Academy of Sciences of the USA*, vol. 85, pp. 2444-2448, 1988.
- [25] Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D., "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs", *Nucleic Acids Research*, vol. 25, issue 17, pp. 3389-3402, 1997.
- [26] Zhang, Z., Schwartz, S., Wagner, L. and Miller, W., "A greedy algorithm for aligning DNA sequences", *Journal of Computational Biology*, vol. 7, pp. 203-214, 2000.
- [27] Tatusova, T. A. and Madden, T. L., "BLAST 2 sequences, a new tool for comparing protein and nucleotide sequences", *FEMS Microbiology Letters*, pp. 247-250, 1999.
- [28] States, D. J. and Agarwal, P., "Compact encoding strategies for DNA sequence similarity search", *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, pp. 211-217, 1996.
- [29] Califano, A., and Rigoutsos, I., "FLASH: fast look-up algorithm for string homology", of *International Conference on Intelligent Systems for Molecular Biology*, pp. 56-64, 1993.
- [30] Schwartz, S., Zhang, Z., Frazer, K. A., Smit, A., Riemer, C., Bouck, J., Gibbs, R., Hardison, R., and Miller, W., "PipMaker a web server for aligning two genomic DNA sequences", *Genome Research*, vol. 10, issue 4, pp. 577-586, 2000.
- [31] Kent, W. J., "BLAT: The BLAST-like Alignment Tool" *Genome Research*, vol. 12, issue 4, pp. 656-664, 2002.
- [32] Batzoglu, S., Pachter, L., Mesirov, J. P., Berger, B., and Lander, E. S., "Human and mouse gene structure: comparative analysis & application to exon prediction", *Genome Research*, vol. 10, pp. 950-958, 2000.
- [33] Tromp, M. Ma, J., and Li, M., "PatternHunter: faster and more sensitive homology search", *Bioinformatics*, vol. 18, pp. 1-6, 2002.
- [34] Gusfield, D., "Algorithms on Strings, Trees and Sequences, Computer Science and Computation Biology", Cambridge University Press, New York, 1997.
- [35] Delcher, A. L., Kasif, S., Fleischmann, R. D., Peterson, J., White, O., and Salzberg, S. L., "Alignment of whole genomes", *Nucleic Acids Research*, vol. 27, issue 11, pp.2369-2376, 1999.
- [36] Burkhardt, S., Crauser, A., Ferragina, P., Lenhof, H.-P., Rivals, E., and Vingron, M., "q-gram based database searching using a suffix array (QUASAR)", *International Conference on Research in Computational Molecular Biology Proceedings*, pp. 77-83, Lyon, 1999.
- [37] Bray, N., Dubchak, I., and Pachter, L., "AVID: a global alignment program", *Genome Research*, vol. 13, issue 1, pp. 97-102, 2003.
- [38] Kurtz, S., and Schleiermacher, C., "REPuter fast computation of maximal repeats in complete genomes", *Bioinformatics*, vol. 15, issue 5, pp.426-427, 1999.
- [39] Borodovsky, M., and McIninch, J., "GeneMark: parallel gene recognition for both DNA strands", *Computational Chemistry*, vol. 17, pp. 123-133, 1993.
- [40] Salzberg, S., Delcher, A., Kasif, S., & White, O. "Microbial gene identification using interpolated Markov models", *Nucleic Acids Research*, vol. 26, pp. 544-548, 1998.
- [41] Burge, C. B., and Karlin, S., "Finding the genes in genomic DNA", *Curr. Opin. Struct. Biol.*, vol. 8, pp.346-354, 1998.
- [42] Uberbacher, E. C., and Mural, R. J., "Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach", *Proceedings of the National Academy of Sciences of the USA*, vol. 88, pp. 11261-11265, 1991.
- [43] Gish, W., and States, D.J., "Identification of protein coding regions by database similarity search", *Nature Genet.*, pp. 266-272, 1993.
- [44] Mueller, M., and Lee, C., "The GeneMine System for Automated Gene Function Analysis and Rich Structure-Function Annotation", *Molecular Applications Group, Palo Alto, CA*, 1995.
- [45] Lee, C., "LOOK: A Software System for Integrated Macromolecular Sequence-Structure Analysis and Modeling", *Molecular Applications Group, Palo Alto, CA*, 1993.
- [46] Dear, S., and Staden, R., "A standard file format for data from DNA sequencing instruments", *DNA Sequence*, vol. 3, pp. 107-110, (1992).
- [47] Bonfield, J. K., and Staden, R., "ZTR: A new format for DNA sequence trace data", *Oxford journal of Bioinformatics*, Oxford, vol. 18, pp. 3-10, 2002.
- [48] National Centre for Biotechnology Information (NCBI), [Online]. Available at: <http://www.ncbi.nlm.nih.gov/> (Accessed: jan-01-2009).
- [49] Chechetkin V. R., and Turygin, A. Y., "Size-dependence of Three-periodicity and Long-range Correlations in DNA Sequences," *Physics Letters A*, vol. 199, pp. 75-80., 1995.
- [50] C. Burge., "Identification of genes in human genomic DNA", Ph.D. dissertation, Stanford University, May 1997.
- [51] Nadeem, M., and Shams, S., "Guided Web content mining approach for automated meta-rules extraction and Information Retrieval", *Proceedings of DMIN-08*, pp. 619-625, 2008.
- [52] Shams, S., Adeel, Y., and Shoab, A. K., "A Novel SIMD Optimized Superscalar Framework for Fast Prediction of Exons", Un-published research, 2009.

Performance Monitoring and Evaluation of Software Developers in an Information Technology Company using Data Mining Techniques

Chandrani Singh and Dr.Arпита Gopal

Dept of Computer Applications, Affiliated to University of Pune, Sinhgad Institute of Business Administration and Research, Kondhwa (Bk), Pune-411048, India

Abstract - Assessment as a continuous process generates data, which acts as an indicator for measuring the performance of an individual and subsequently impacts on the decision making of the various stakeholders of the organization. The data mining methodology while extracting meaningful patterns from the organization's database related to the performance of the software developers, contributes to generate results based on which employees ensure maximizing their quality, potential and productivity. A framework has been designed based on the idea proposed in this paper by taking into consideration a number of parameters essential for the derivation of performance indicators to evaluate and assess the software developers in an IT organization. The aim is to enable higher level authorities to take decisions and understand certain patterns of employment motivation, satisfaction, growth and decline and also to depict the detailed performance pattern to the individual for further enhancements. For software developers, performance measurement has to be carried out in two phases as described in the paper. The data mining techniques will work on the measures of performance in these two phases to generate the results with substantial accuracy.

Keywords: Evaluation, Performance, Parameters, Score

1 Introduction

The Applications of Data Mining in the field of business have unearthed the fact that typical data mining questions of the business world can help evaluate and answer not only queries related to the overall growth of the organization but can monitor and assess the prime assets (i.e. employees) within the organization [2]. The justification of the above statement can be strengthened by the fact that commercial companies target at winning customer satisfaction and broaden up customer base and they do this after extensive analysis on customer behavior where Data Mining plays an important role. Similar requirement has arisen to mine employee data from various stakeholders' perspective to steer the organization towards improved productivity, consistency and quality. The performance evaluation of

software developers is being emphasized on, keeping in mind that relatively new entrant to an organization show significant improvements in their performances and companies can enthruse them to be more quality compliant in order to achieve quality goals. Eventually a system is needed to be designed for the same. The methodology adapted to design the system comprises of *Phase- I and Phase II.*

Phase-I

Finding the key parameters needed for the assessment and evaluation of the software developers across the verticals. Developing a model to extract their competency scores. Generating a mathematical model for computation of the final scores.

Phase-II

Applying data mining techniques to the derived composite scores to find:

- Employees technical performance,
- The hidden trends in their performances and
- The patterns of performances of the software developers across the segments.

The next step within phase II is to apply the algorithms on the cumulative score to generate the final results.

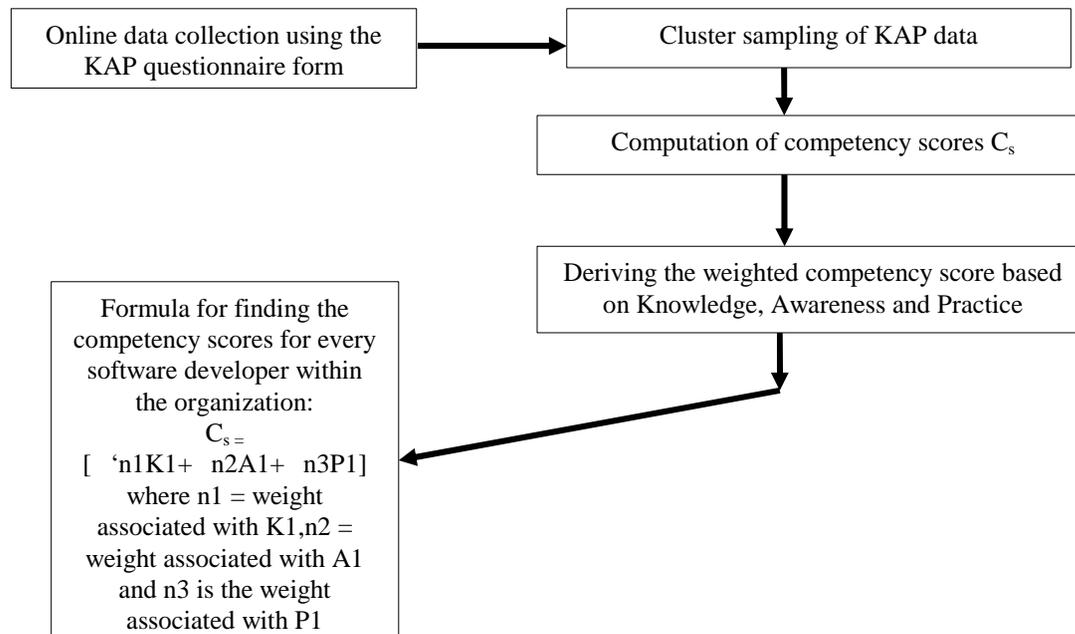
2 METHODOLOGY

Phase-I-First step is to generate the competency score based on the KAP model [3]. The model proposed is given in the next page (Fig1). The competency score will be used for evaluating the performances of the software developers along with other significant parameters for assessment of the employees. The parameters, which have to be taken into consideration, are the following:

I. Administrative & Organizational Skills

A. Organization/Planning: An employee carries out functions in an organized and thorough manner from start to end. Orchestrates staff to accomplish the greatest productivity with the least effort. They will be marked in the scale of 1-10 in categories like N/A, Unacceptable, Needs Improvement Meets Expectations, Exceeds and Excels and same procedure will be followed below.

Fig 1. Composite Score Model



B. Follow-through & Thoroughness: An employee stays with a project to ensure its completion in a timely and thorough manner.

C. Professional/Technical Knowledge: He is knowledgeable and able to serve as internal consultant. Stays current with professional field and is able to analyze and use statistical reports to assist in planning and operating the department.

D. Crisis Management: Employee remains calm and responsive during times of crisis and is able to demonstrate positive leadership and management skills when under stress and knows how to systematically solve problems.

2. Organizational Climate

A. Loyalty: Employee is supportive of leadership in word and deed.

B. Team orientation: S/W developers work cooperatively with other department staff.

C. Coaching orientation: Personal involvement in training and interest in employee's progress.

D. Change Orientation: Employee welcomes new ideas, supports and promotes positive change.

E. Management Style: He is open, participative encouraging others to take responsibility and can make decisions.

3. Communication

A. Listening Skills: Employee knows how to listen actively/attentively, demonstrating effective clarification skills.

B. Verbal/Written: Demonstrates an effective ability to communicate orally and in writing.

C. Inter-Departmental Communication: Keeps others informed in and out of immediate department as appropriate.

4. Customer Service

A. Customer Service Orientation: Views internal and external relationships as customer partnerships.

B. Customer Focus: Puts customers first and trains other employees to do the same.

C. Service Focus: Seeks input from customers and employees for continuous quality improvement.

D. Balanced Perspective: Knows how to balance his superior's needs and demands with business realities and limitations.

5. Performance Skills

A. Fiscal Management: Understands financial reports to some extent.

B. Problem Resolution: Demonstrates effective problem resolution skills.

C. Productivity: Pushes himself and his colleagues towards higher performance levels.

6. Leadership

A. Initiative: Demonstrates an ability to convert ideas to action.

B. Decision Making: Demonstrates ability and willingness to make small scale decisions in a timely manner.

C. Modeling: Practices what is preached.

7. Mission Orientation

A. Organizational Knowledge: Understands the organizations business, mission and values.

B. Commitment: Demonstrates support in word and deed for the company's goals and objectives.

C. Trust Worthiness: Acts in a manner, which engenders trust from employees and management.

D. Organizational Courage: Demonstrates character, strength & appropriateness in addressing questionable business practices.

E. Equality Focus: Emphasizes the highest quality performance as a way of meeting the projects mission and values.

Taking the above parameters into consideration a Total Composite Score will be calculated to find out the performance of that employee(s) [10]. This will be calculated taking three weighted factors into consideration: Technical (C_s), Managerial (C_M) and Social (C_O). So total composite score for performance evaluation will be:

$$Total\ Comp\ Score = W_1TC_s + W_2TC_M + W_3TC_O \quad (1)$$

Under the technical aspect the three factors of Knowledge, Awareness and Practice for a subject is taken into consideration. Under managerial aspect Leadership, Organizational and Administrative skills and Mission orientation is taken into account and under Social aspects communication, acceptability, team spirit, integrity and punctuality will be considered. In the above we have sighted a few example parameters from a set of 70 for assessment of employees. Appropriate weights will be assigned to T_M, T_S and T_O. The total composite score will be shown in a tabular form as represented below:

Table 1: Total Composite Score table.

Verticals	No of employees appeared	Min Score	Max Score	Score range
Banking	25	11	25	1-30
Healthcare	15	13	26	1-30
Telecom	15	11	24	1-30

Phase – II – Designing an Employee Performance Monitoring and Evaluation Mining Model (EPMES)

In Phase 2 data mining algorithms will work to find the trends and patterns of the total composite score. A model (Fig 2) has been proposed which shows a broader framework of the plan, which constitutes a series of steps to be executed right from data assimilation to designing of the mining model. The EPMES model designed (Fig 3) will contain rule database and employee database where in the rules will be designed to act on the employee data to answer questions related to their performances. In the broader sense first the problem has to be defined in the first of the six steps.

Defining the problem which encompasses the following:

What types of relationships are we trying to find?

Does the problems solution reflect the policies or processes of a Company?

Do we want to make predictions from the data-mining model, or just look for interesting patterns and associations?

How are the columns related and the tables related?

How is the data distributed?

Does the data accurately represent the processes of the organization?

To answer these questions an extensive study on the available employee and performance data, across the company, has to be performed.

Collecting and Preparing Data - Data can be scattered across the companies and stored in different formats, or may contain inconsistencies such as incorrect or missing entries.

Data cleaning is not just about removing bad data, but about finding hidden correlations in the data, identifying the most accurate data sources, and determining which columns are the most appropriate for use in analysis.

Data Analysis –This will encompass categorization and balancing of data and dividing it into training, testing and validation data.

Exploring Data - Exploration techniques include calculating the minimum and maximum values of performance, calculating mean and standard deviations in the performance, and looking at the distribution of the data.

Building EPMES Model - Before the model is processed, it has to be analyzed. This model acts as a container that specifies the columns used for input, the attribute that we are predicting, and parameters that tell the algorithm how to process the data. Processing a model is also called training. Training refers to the process of applying a specific mathematical algorithm to the data in the structure in order to extract useful patterns. The patterns that are found in the training process depend on the selection of training data, the algorithm chosen, and how the algorithm has been configured. The algorithms, to be used are:

Classification algorithms which predict one or more discrete variables, based on the other attributes in the dataset.

Segmentation algorithms divide data into groups or clusters of items that show similarities.

Association algorithms find correlations between different attributes in a dataset. The most common application of this kind of algorithm is for creating association rules, which are stored in rule database and can be used in analysis. Permutation and combination of these rules will help in generating the indicators.

Exploring and Validating the Model – After building the EPMES model data is separated into training and testing datasets so that it is possible to accurately assess the performance of the model on the same dataset.

Creating prediction queries will make proper usage of the training dataset to build the model, and the testing dataset to test the accuracy of the model. The validation data will validate the model.

Update the models after review and analysis - The two models shown below depict the underlying concept of the proposal. Updating the models dynamically, as more data comes into the organization, and making constant changes to improve the effectiveness of the solution will be an expensive and time consuming task. The proposed methodology used above is to finalize on the model proposed for the assessment monitoring and evaluation of the employees. The prediction target processes will summarize all the qualities of assessment and performance monitoring of employees which will hold information that will answer questions using a matrix of 1500 employee records multiplied by some number of derived variables. Employee details will be taken from the two companies as has been finalized.

Deploying and Refining the Model - Use the EPMES model to create predictions, which can then be used to make business decisions. Queries are created to retrieve statistics, rules, or formulas from the model and data mining functionality embedded directly into the application designed. A package will be created in which a mining model can intelligently separate incoming data into multiple tables in the database. A report will be generated that lets stakeholders directly query the mining model. This is a very important step for continuously improvement on the performance of the model and the algorithm.

3. DATA MINING ALGORITHMS (Classification, Clustering and Association algorithm execution for the proposed model)

3.1 CLASSIFICATION ALGORITHMS

In analyzing the above data it is also very important for the classification model obtained to be user friendly, so that management can make decisions about the employees. In general, models obtained using categorical data are more comprehensible than when using numerical data because categorical values are easier for a stakeholder to interpret. Below are some classification algorithms as proposals to be executed on the Employee Performance Monitoring and Evaluation Model (Fig 3). *Decision trees* are considered easily understood models because reasoning can be associated with each conclusion.

Decision trees don't lose data of the previous node at any point of time. However, if the tree obtained is very large (a lot of nodes and leaves) then the complexity to reach a conclusion increases and they become less comprehensible. A decision tree can be directly transformed into a set of IF-THEN rules that are one of the most popular forms of knowledge representation. Hence decisions generating from C4.5 and CART algorithms are simple for management to

understand and interpret. Aim is to run the above algorithms with proper training, testing and validation dataset.

Rule induction algorithms are normally considered to produce comprehensible models because they discover a set of IF-THEN classification rules that are a high level knowledge representation and can be used directly for decision making and some algorithms such as *GGP* have a higher expressive power allowing the user to determine the true performance pattern.

Fuzzy rule algorithms obtain IF-THEN rules that use linguistic terms that make them more comprehensible/interpretable by human beings. So Fuzzy rule algorithms are intuitive, understandable and easily interpreted by the stakeholders [9]. Some good performance algorithms are *GAP* and *LogistBoost*, which can be used on the EPMES model.

3.2 CLUSTERING ALGORITHMS

Clustering is a division of data into groups, which possess similar characteristics. Representing the data by fewer clusters loses certain minute details, but achieves simplification. From a machine learning perspective clusters correspond to hidden patterns whereas from a practical perspective clustering plays an outstanding role in data mining applications such as scientific data exploration, information retrieval. Clustering algorithms are of basically two types: Hierarchical, Partitioning and Categorical clustering methods. Hierarchical algorithms are further categorized into agglomerative (bottom-up) and divisive (top-down) [1]. Here we will restrict ourselves to use the hierarchical clustering algorithms to represent similar clusters. Under hierarchical methods are:

Agglomerative Algorithm-Chiu et al. proposed a conceptual or model-based approach to hierarchical clustering. The model associated with a cluster covers both numerical and categorical data. With every cluster C , we associate a logarithm of its (classification) likelihood:

$$l_c = \sum_{x_i \in C} \log(p_i | C) \quad (II)$$

where θ is a manifold parameter. Here logarithm has been applied to purposefully show a decrement in more number of cluster formulations and bridging the gap between clusters which do not differ significantly. The algorithm uses strong chance estimates for parameter θ . The distance between two clusters is defined as a decrease in log-likelihood given by the equation below:

$$d(C_1, C_2) = l_{C_1} + l_{C_2} - l_{C_1, C_2} \quad (III)$$

caused by merging of the two clusters under consideration. The agglomerative process continues until the stopping criterion is satisfied. This algorithm has the commercial implementation (in SPSS Clementine) [2] and can be a probable algorithm for finding employee segments and

Fig 2. Proposed Model – A Broad Framework

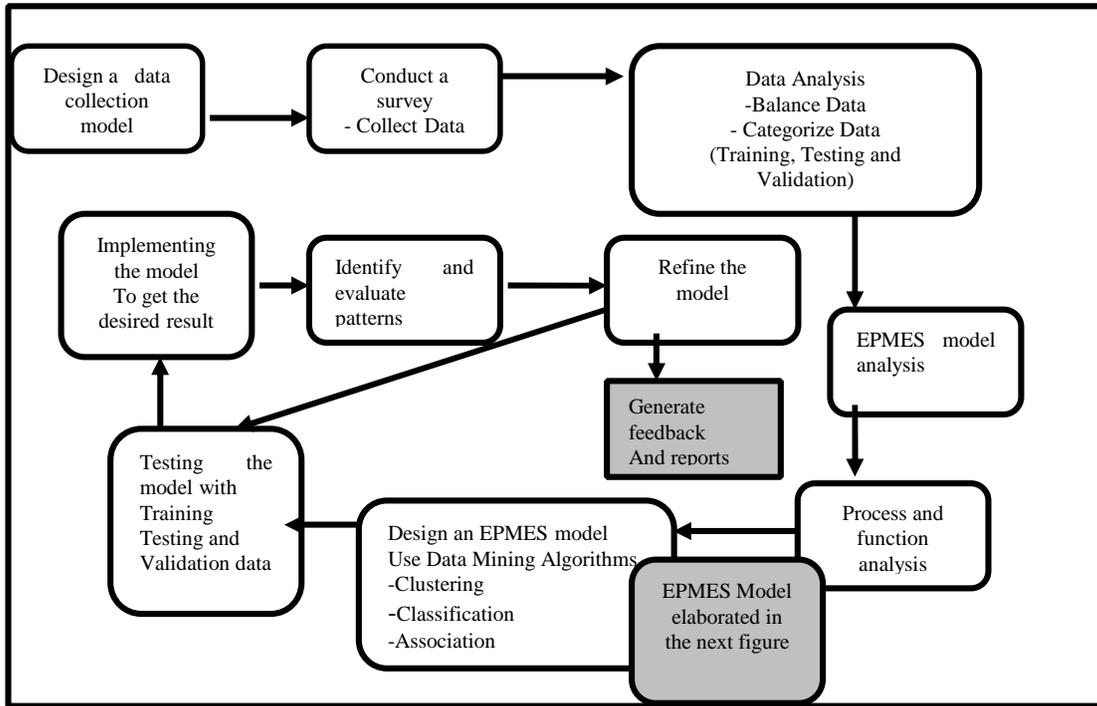
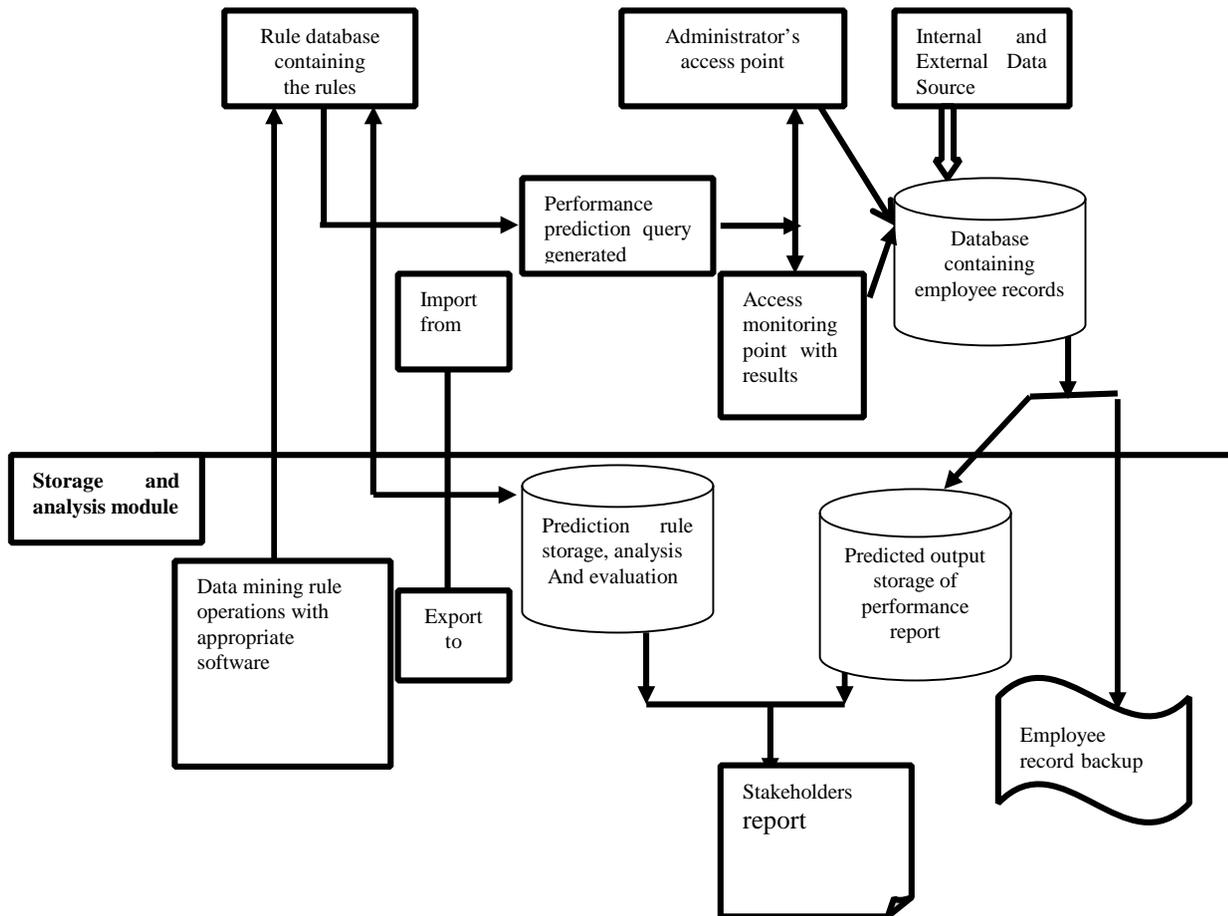


Fig 3. EPMES Model



merging the segments if the distance between the clusters shows a decrease in log likely hood.

Methods Based on Co-Occurrence of Categorical Data

Another probable clustering algorithm, *CACTUS* (Clustering Categorical Data Using Summaries) proposed by Ganti et al [2] looks for clusters at certain intervals in point-by-attribute data with categorical attributes and broader data region. It can also be used to identify clusters in employee database using summary data.

Another probable algorithm is *ROCK* (Robust Clustering algorithm for Categorical Data) proposed by Guha et al. [3] which deals with categorical data and has many common features of Hierarchical Clustering. Some of the common features are: (1) it is a hierarchical clustering approach; (2) The combination of clusters follows a bottom up approach until specified number k of clusters is constructed, and (3) it uses data sampling. *ROCK* defines a neighbor of a point x as a point y such that $\text{sim}(x, y) \geq \theta$ for some threshold value of θ , and then only proceeds to link(x, y) where x and y have equal number of common neighbors. Clusters formulated here consist of points with a high degree of connectivity and higher number of links. Agglomeration is found to be highly predominant here. *ROCK* utilizes a specific objective function for clustering.

3.3 ASSOCIATION ALGORITHMS

The algorithm *Apriori* and *AprioriTid* can be used to find strong association rules from the frequent datasets. The problem of association rule mining is to efficiently find all rules with support $> \text{Minsup}$, confidence $> \text{Minconf}$. [2]. Here support and confidence refer to the relative frequency and conditional probability of appearance of item sets in a particular data base. The inputs for the *Apriori* and *AprioriTid* algorithm are the following: *Input*: D is the database of m transactions, and a minimum support, mins , is represented as a fraction of m . *Output*: frequent item sets, L_1, L_2, L_k . *AprioriTid* uses the database only once and shows better performance than *Apriori* algorithm in later passes, as shown in the figure 4. So probable algorithm can be *AprioriTid* for generating frequent item sets to find strong association rules within employee data and segments.

Frequent-pattern tree algorithm

Apriori based algorithms generates and test a very large number of candidate item sets. Example: with 1000 frequent 1-items, *Apriori* would have to generate 2^{1000} candidate 2-itemsets. To decrease the number of sets the FP-growth algorithm was devised. This algorithm eliminates the generation of a large number of candidate item sets and generates a compressed version of the database for finding frequent item sets. Above two algorithms can stand the chance for finding the association rules within employee database. For the training, testing and validation of the model the system will be trained with 60% of the data, tested with 20% data and to improve upon the accuracy, validated by 20% data.

A proposed algorithm for mining frequent item sets from employee database

This efficient algorithm works by singly scanning the database for mining complete frequent item sets. This algorithm works well without any tree construction. It is fast and efficient and traces the occurrences of items by performing Logical AND operation [9].

Step 1:

Scan the Transactional database to construct the support count (SCT) table and Bit Form Table (BFT)

Step 2:

//In the table SCT, consider the group of nodes which satisfies the minimum support count. Find the group of nodes which are fully connected (G) with each other.

Step 3:

For all G,

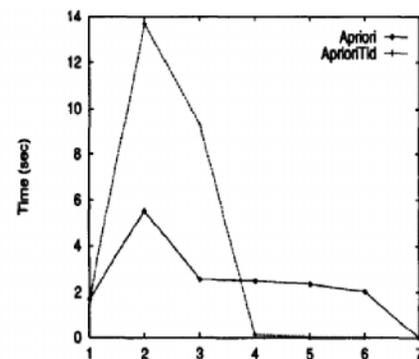
(i) Perform Logical 'AND' operation on BFT.

(ii) Find the total value of the resultant value of Logical 'AND' operation.

(iii) If the total value of Logical 'AND' operation satisfies the min_support count, and then add 'G' to the frequent item sets LK

This is algorithm will help to mine item sets from employee data to extract frequent occurring patterns from huge employee database.

Fig 4. *AprioriTid* beats *Apriori* in later passes



4. CONCLUSIONS

The proposal put forth in the paper justifies that Data Mining can provide effective monitoring tool for employee performance with considerable accuracy. Once derived variables are refined it will improve rule quality and various reporting tools will serve mainly to compare changes over time in performances as may be affected by the different rules that are available plus other well chosen variables will expose systematic structures required to improve performance monitoring and evaluation Above paper is a proposal for implementing the said concept and in our future research we will be coming up with the results of the implementation.

5. REFERENCES

- [1] A.K. Jain and R. C. Dubes. [1988]. Algorithms for Clustering Data, Prentice Hall, NJ, 1988.
- [2] R Agrawal, R Srikant[1994]. Fast Algorithms for Mining Association rules in Large Database by Proceedings of the VLDB.
- [3] A Study of Employee Competency in Software Process Management Siew Hock Ow & Mashkmi Hj. Yaacob, Faculty of Computer Science & Information Technology University of Malaya 50603 Kuala Lumpur, Proceedings of the 3rd International Software Engineering Standards Symposium (ISESS '97) 1082-3670/97 \$10.00 © 1997 IEEE
- [4] Ganti, V., Gehrke, J. and Ramakrishnan, R. [1999a]. CACTUS-Clustering Categorical Data Using Summaries, In Proceedings of the 5th ACM SIGKDD, 73-83, San Diego, CA.
- [5] GUHA, S., RASTOGI, R., and SHIM, K. [1999]. ROCK: A robust clustering algorithm for categorical attributes, In Proceedings of the 15th ICDE, 512-521, Sydney, Australia.
- [6] Zaki, M.J.[2000]. Scalable algorithms for association mining Knowledge and Data Engineering, IEEE Transactions on Volume 12, Issue 3, May/June 2000 Page(s):372-390, Digital Object Identifier 10.1109/69.846291
- [7] Chiu, T., Fang, D., Chen, J., and Wang, Y. [2001]. A Robust and scalable clustering algorithm for mixed type attributes in large database environments, In Proceedings of the 7th ACM SIGKDD, 263-268, San Francisco, CA.
- [8] Luan J. [2002] "Data Mining and Knowledge Management in higher Company" Presentation at AIR Forum, Toronto, Canada.
- [9] S.P. Latha, N. Ramaraj, [2007] "Algorithm for Efficient Data Mining," iccima, pp.66-70, International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007).
- [10] A Decision Support System for Performance Appraisal by Dr.P.D.D.Dominic Department of Computer and Information Sciences Universiti Teknologi PETRONAS 31750 Tronoh, Perak Malaysia, Izzatdin Abdul Aziz Department of Computer and Information Sciences Universiti Teknologi PETRONAS 31750 Tronoh, Perak Malaysia., K.N.Goh Department of Computer and Information Sciences Universiti Teknologi PETRONAS 31750 Tronoh, Perak Malaysia.

Are Decision Trees Always Greener on the Open (Source) Side of the Fence?

Samuel A. Moore¹, Daniel M. D'Addario², James Kurinskas³, and Gary M. Weiss⁴

¹⁻⁴Department of Computer and Information Science, Fordham University, Bronx, NY, USA

Abstract - This short paper compares the performance of three popular decision tree algorithms: C4.5, C5.0, and WEKA's J48. These decision tree algorithms are all related in that C5.0 is an updated commercial version of C4.5 and J48 is an implementation of the C4.5 algorithm under the WEKA data mining platform. The purpose of this paper is to verify the explicit or implied performance claims for these algorithms—namely that C5.0 is superior to C4.5 and that J48 mimics the performance of C4.5. Our results are quite surprising and contradict these claims. This is significant because existing work that is based on these claims (e.g., J48 being equivalent to C4.5) may be misleading.

Keywords: classification, decision trees, evaluation.

1 Introduction

This paper compares the performance of three popular decision tree algorithms: C4.5, C5.0, and J48. These algorithms are all related since C5.0 is an updated, commercial version of C4.5 and J48 is an implementation of C4.5 under the WEKA data mining platform. Specific claims have been either explicitly or implicitly made about the relative performance of these algorithms and the goal of this paper is to assess these claims and, in particular, determine if:

- C5.0 is superior to C4.5
- J48 and C4.5 perform similarly

If C4.5 and J48 do not perform similarly then this may impact existing research which assumes that they perform similarly; conclusions made for those papers may not generalize to C4.5, which for many years was the standard decision tree classification algorithm used in the machine learning and data mining communities. Also, it is important to assess the performance of C4.5 since, although it is quite old and has not been updated in many years, it is still frequently used in research. If its performance is far inferior to commercial decision tree algorithms, the research community should be clearly aware of this and that any conclusions based on this old algorithm may be suspect. This paper also investigates the claims made by Rulequest Research that their C5.0 decision

tree algorithm performs better than C4.5, especially on larger data sets.

2 Methodology

The three decision tree learners that are evaluated in this paper are all related. The C4.5 learner is an open source, free version of the decision tree rule creation algorithm created by Ross Quinlan [2]. C5.0 is the commercial and updated version of C4.5 from Rulequest Research, which is supposed to be superior to C4.5 (see www.rulequest.com for more details). The J48 learner is a version of the C4.5 algorithm implemented as part of the WEKA data mining platform [3]. The terms WEKA and J48 may be used interchangeably in this paper.

The data sets used in our analysis are listed in Table 1. Most of the data sets are from the UCI repository [1]. Table 1 shows the size of each data set as well as the degree of class imbalance by showing the percentage of examples belonging to the minority class. In the remainder of the paper data sets are referred to using the data set number provided in the first column of Table 1.

TABLE 1:
DATA SET DETAILS

#	Name	Size	% Minority
1	adult	5,000	24%
2	band	538	42%
3	breast-wisc	699	35%
4	echocardiogram	74	32%
5	weather	1,000	40%
6	car	1,728	30%
7	hepatitis	155	20%
8	blackjack	150,000	36%
9	contraceptive	1,473	27%
10	hypothyroid	3,163	48%
11	horse-colic	300	36%
12	liver	345	42%
13	sonar	208	47%
14	vote	300	39%
15	hungarian-heart	294	36%

All experiments were run for C4.5, C5.0 and J48 using the default decision tree settings and ten-fold cross validation. The default parameters were used for each of the classifiers tested. The data partitioning for the cross validation was done external to the three classifier induction programs to ensure that each of the three classifiers were induced using precisely the same data partitions. In addition to tracking the accuracy and error rate of each decision tree (based on test set performance) we also tracked the F-measure, which balances the importance of precision and recall.

A confusion matrix for a two class problem is shown in Table 2. Accuracy is the fraction of examples classified correctly and is calculated as $(TP+TN)/(TP+TN+FP+FN)$, with error rate (ER) equaling $1 - \text{accuracy}$.

TABLE 2:
CONFUSION MATRIX

		Predicted Class	
		P	N
Actual Class	P	TP (True Positive)	FN (False Negative)
	N	FP (False Positive)	TN (True Negative)

Accuracy is a poor measure when there is substantial class imbalance in a data set and for that reason we also track the F-measure, which balances the performance of the two classes. The F-measure is defined below, where recall and precision are defined as:

$$\text{Recall} = TP/(TP + FN)$$

$$\text{Precision} = TP/(TP + FP)$$

$$\text{F-Measure} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$$

Finally, we used a t-test to determine if the observed differences in classifier performance are statistically significant. The results of the t-tests are reported, along with the performance results, in Section 3.

3 Results

The results for all three decision tree algorithms, for both F-measure and error rate, are shown in Table 3. For each data set and each performance metric, the best value is highlighted by underlining it (note that higher F-measure values are better). From the results in Table 3 it is clear that C4.5 consistently yields the best value for both F-measure and error rate. Furthermore, C5.0 outperforms J48, although the relative differences are much smaller.

TABLE 3:
STATISTICS FOR C4.5, C5.0 AND J48 (WEKA) ALGORITHMS

#	F-measure			Error Rate		
	C4.5	C5.0	J48	C4.5	C5.0	J48
1	<u>.922</u>	.908	.908	<u>.119</u>	.146	.144
2	.486	<u>.705</u>	.447	.286	<u>.243</u>	.303
3	<u>.981</u>	.958	.958	<u>.024</u>	.054	.054
4	<u>.989</u>	.970	.970	<u>.013</u>	.040	.040
5	<u>.699</u>	.626	.633	<u>.209</u>	.302	.289
6	<u>.996</u>	.958	.954	<u>.004</u>	.057	.063
7	<u>.957</u>	.858	.902	<u>.071</u>	.232	.161
8	<u>.509</u>	.480	.489	<u>.271</u>	.276	.278
9	<u>.525</u>	.410	.396	<u>.201</u>	.214	.214
10	<u>.946</u>	.916	.920	<u>.005</u>	.007	.007
11	<u>.925</u>	.888	.886	<u>.100</u>	.150	.150
12	<u>.735</u>	.563	.587	<u>.194</u>	.336	.313
13	<u>.952</u>	.692	.697	<u>.043</u>	.266	.288
14	<u>.975</u>	.958	.947	<u>.030</u>	.050	.063
15	<u>.919</u>	.902	.842	<u>.105</u>	.125	.214
Ave.	<u>.834</u>	.786	.761	<u>.112</u>	.168	.172

These results are surprising. First, C4.5 outperforms C5.0, the newer and more advanced commercial version of C4.5. While most of the claimed superiority of C5.0 over C4.5 has to do with large data sets (and we have relatively few truly large data sets), there is no acknowledgement that C5.0 does *worse* on small data sets—yet that is exactly what we see. Perhaps the biases built into C5.0 that permit it to do well on large data set cause it to perform poorly on smaller data sets. But if this were true, it should be possible to use different biases based on training set size or simply use C4.5 for small data sets. We also see that J48 does not perform similarly to C4.5 and in fact performs much more similarly to C5.0 than to C4.5. We cannot explain this given that J48 is supposed to be a reimplement of C4.5.

Table 3 clearly shows which method performs best but the magnitudes of the differences are only apparent with careful study. Figures 1 and 2 depict these differences visually. Figure 1 provides a scatter plot for F-measure performance and Figure 2 a scatter plot for error rate performance, where each data point corresponds to a data set and two different learning methods. To avoid overcrowding, each figure only compares the performance of C4.5 versus C5.0 and C4.5 versus J48 and does not directly compare C4.5 and C5.0. If two methods perform identically all points would be clustered around the line $y=x$. For F-measure, points below the line $y=x$ indicate that C4.5 performs better while for error rate points above the line indicate that C4.5 performs better. Figures 1 and 2 indicate that C4.5 consistently performs better for both measures and that these differences are quite substantial since the data points tend to fall quite far from the line $y=x$.

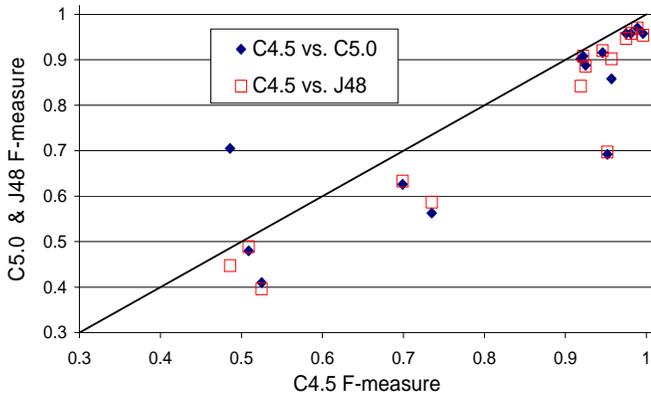


Figure 1: Comparison of F-measure Results

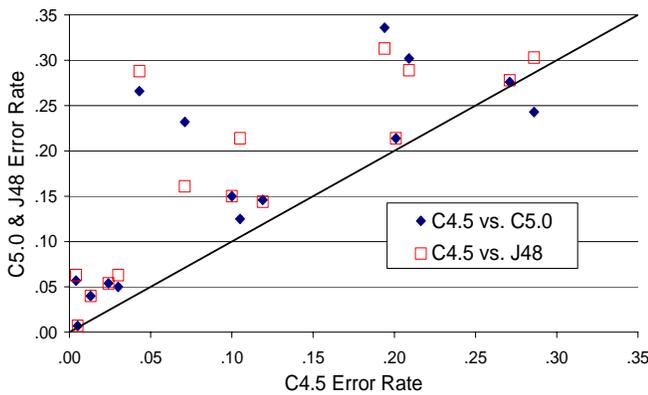


Figure 2: Comparison of Error Rate Results

We ran a t-test to do pairwise comparisons of the three methods and the results are shown in Table 4. The first column specifies which algorithms are being compared while the second column specifies which algorithm appears to perform best. The last two columns provide the confidence levels for this observation. The higher the confidence the more sure we can be that the observed differences are real and not due to chance. Note that a common threshold for confidence is 0.90 and based on that we cannot conclude with a high level of confidence that all of the differences are statistically significant.

TABLE 4:
T-TEST RESULTS

Algorithms	Best	Confidence	
		F-meas	ER
C4.5 vs. C5.0	C4.5	.90	.90
C4.5 vs. J48	C4.5	.68	.72
C5.0 vs. J48	C5.0	.84	.88

Based on the results in Table 4 we can be confident that for both F-measure and error rate C4.5 outperforms C5.0. The results also indicate that the differences between C5.0 and J48 may be statistically significant, but not at the 0.9 confidence threshold.

In order to analyze the claim that C5.0 works better on larger data sets, we partitioned the data sets into two groups based on their sizes and then analyzed the results. One group, which we refer to as “smaller”, contains the nine data sets with fewer than one thousand records and the remaining six data sets are referred to as “larger.” The performance for these two partitions, for both F-measure and error rate, are shown in Table 5.

TABLE 5:
AVERAGED PERFORMANCE BASED ON SIZE

Data Sets	C4.5		C5.0		J48	
	F-meas	ER	F-meas	ER	F-meas	ER
All	<u>.835</u>	<u>.112</u>	.787	.167	.770	.172
Smaller (<1000)	<u>.873</u>	<u>.108</u>	.826	.178	.798	.188
Larger (>1000)	<u>.767</u>	<u>.135</u>	.717	.168	.717	.166
Abs. diff	.106	.027	.109	.010	.081	.022

If we look at the absolute difference between the two partitions, C5.0 does have a bigger absolute difference between the two partitions for error rate (.027 vs. .010), but not F-measure. However, error rate was the only metric considered in the original claims for C5.0 being superior to C4.5. If we look at the differences between C4.5 and C5.0 or between C5.0 and J48, we see the same patterns as before, although the superiority of C4.5 over C5.0 is less for the larger partition. It is also helpful to look at the data set with the largest number of records. The blackjack data set (#8) has 150,000 records and for this C4.5 only narrowly beats C5.0 and J48. In fairness we should point out that our “larger” data sets are not all that large and we are currently in the process of evaluating much larger datasets to extend these results.

4 Conclusion

Our results indicate that C4.5 performs consistently better than C5.0 and J48 (at least on relatively small data sets) and appears to perform as well on the “larger” data sets, although no very large data sets were evaluated. In many cases the differences appear to be substantial, for both error rate and F-measure. It appears that the inductive bias for C5.0 is such that it is not well suited to small data sets. If this is in fact the case, it might be possible to adjust this bias based on the data set size so as to improve its performance on small data sets without necessarily harming its performance on larger data sets. Our analysis would benefit from additional large data sets (i.e., with more than 100,000 records) and we are currently in the process of evaluating such data sets. Rulequest research also claims that C5.0 is much more efficient than C4.5 in terms of memory and computation time and we hope to comprehensively evaluate this claim in the future. However, we have performed some preliminary tests and based on these the claims seem accurate. We did a preliminary analysis

of the *forest-covertype* data set, which contains 581,012 examples, and found that on the same computer C5.0 ran in 3.5 seconds whereas C4.5 took about an hour and a half (Rulequest Research quotes a similarly impressive speedup for this data set on their website).

Perhaps even more surprising and significant, however, is the results from our comparison of J48 and C4.5. We expected J48 to perform similarly to C4.5 since J48 is supposed to be a reimplementation of C4.5, but our results clearly indicate that J48 consistently performs worse than C4.5 on the data sets that we evaluated and actually performs much more similarly to C5.0. We do not know the reason for this surprising behavior, but it certainly warrants further study and raises questions about any research that has assumed that J48 is equivalent to C4.5. On the positive side, C4.5 is still frequently used for research and our results suggest that its performance is still impressive, since it performs competitively

with C5.0, a recently updated, commercial decision tree learner.

5 References

- [1] D. J. Newman, S. Hettich, C. L. Blake and C. J. Merz. *UCI repository of machine learning databases* [<http://www.ics.usi.edu/~mlearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science. 1998.
- [2] Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.
- [3] I. H. Witten and E. Frank, *Data Mining: practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

Action Selection in Customer Value Optimization: An Approach Based on Covariate-Dependent Markov Decision Processes

Angi Rösch and Harald Schmidbauer

Abstract— Typical methods in CRM marketing include action selection on the basis of Markov Decision Processes with fixed transition probabilities on the one hand, and scoring customers separately in pre-defined segments on the other. This points to a gap in the usual methodology insofar as customer scoring implies the explicit use of customer-specific information (covariates), while transition probabilities of Markov chains are conceived of as averages, without reference to the peculiarities of the customer to be addressed. Trying to unite both approaches, we suggest a model for customer transitions which allows transition probabilities to depend on covariates. Our model can be seen as an effort to focus on one-to-one marketing methods, permitting customer-specific action selection with the overall goal of customer value optimization. We show how to maximize the objective function subject to budget constraints. Our approach is motivated by the needs of a major European insurer. A numerical example with a realistic structure illustrates the capabilities of our approach.

I. INTRODUCTION

A. The problem setting

Marketing in CRM typically involves different forms of customer contacts and marketing interventions, e.g. catalogs, simple or sophisticated mailing, product- or relationship-oriented actions. Traditional segmental marketing approaches deal with the allocation of marketing efforts to different segments of customers, where segments are built on customers' characteristics, past customer relationship, or heterogeneity with respect to response (e.g. [3]; an overview can be found in [7]). On the other hand, a "one-to-one marketing" has been promoted as the ultimate form of CRM ([4]), demanding different treatment for different customers. Personalized marketing interventions could explicitly take into account personal responsiveness. In ([7]), it is shown that the heterogeneity of response can be partially explained by customer's characteristics and past behavior. In order to optimize next-period change in profitability at the customer level, the authors propose a hierarchical model for the shift in gross profit due to marketing allocations to the customer, introducing a customer-specific response parameter vector which is estimated from a linear model with covariates.

In recent years, Markov chain approaches have increasingly gained popularity in CRM marketing, introduced by ([5]). They can accommodate situations of both retention and migration of customers in a probabilistic way, and enable future prospect for customer relationship in terms of expected customer value and customer equity. The idea of a Markov

chain approach is to model the route of a typical customer across customer segments (states) from one time period to the next as governed by transition probabilities. Probabilities are supposed to depend on the current state of the relationship only (this property is called the Markov property). They can be estimated from transactional data. Marketing-action-specific data can then be used to model the intermediate- and long-term impacts of marketing in Markov Decision Processes and to find the optimal allocation of marketing interventions to customer segments ([8], [3]; [1]).

B. Our Objective

In this paper we present an approach for marketing action selection which is optimal with respect to expected intermediate- or long-term customer value at the customer level rather than for customer segments. To this goal we employ a covariate-dependent Markov Decision Process. It incorporates transition probabilities that are based on the customer's characteristics and past behavior, in particular being appropriate to shed light on the customer's inclination to respond to different types of mailing contact or at least implying a moderating effect on it. Transition probabilities are estimated using multinomial logistic regression models. As we have in mind an application to a major European insurance company, we estimate the risk of loss which diminishes customer value.

C. Available Data

Apart from transactional customer data, our present study requires realistic company data on demographic and past behavioral customer characteristics, e.g. age, relationship duration, type and number of contracts, loss events, form and number of mailing contacts. Microgeographic data on living, social and economic environment add to the set of covariate data.

D. Outlook

This paper is organized as follows. Section II introduces the model which we use in our study. How we proceed to optimize personalized customer values with respect to action selection and empirical findings are provided in Sections III and IV. Finally, Section V gives a brief discussion of our approach. — All computations were carried out in R [6].

II. THE MODEL

We relate the observed behavior of a customer to the outcome of a stochastic model which is governed by latent behavioral attitudes and marketing actions. We suppose that

Angi Rösch is with the FOM University of Applied Sciences, Study Centres Munich, Germany, and Taian, China (email: angi@angi-stat.com).

Harald Schmidbauer is with the Department of Business Administration, Bilgi University, Istanbul, Turkey (email: harald@bilgi.edu.tr).

each time epoch t from a finite set $\mathcal{T} = \{0, 1, 2, \dots, T-1\}$ is a decision epoch for both the customer and the marketer on how to proceed until $t+1$. Thereon, we define a Markov Decision Process (MDP) featuring the following:

A. States

Customers are classified according to states which reflect the hierarchy of potential needs, e.g. standard customer, standard customer plus simple product, and plus comprehensive product respectively. By introducing an additional state for new or former customers, we can take into account situations of both retention and migration of customers.

Then, at time epoch t , a customer may decide to move from state s to s' from a finite set \mathcal{S} of n states. Her decision may be driven not only by personal preferences but also by the marketing action she experienced at that time.

B. Actions

At each time epoch t , the marketer decides about which actions to applied to each of her customers. Action $a \in \mathcal{A}$ is selected according to its desirability, which depends on the targeted customer state. The action set \mathcal{A} comprises three basic categories: no action, simple mailing, sophisticated mailing.

C. Transition probabilities

Suppose that customer k is in state s at time epoch t . Then, the probability that she will switch to state s' at time epoch $t+1$ under the regime of marketing action a is denoted by

$$p_{t,k}(s, s'|a, X_{t,k,s,a}).$$

The characteristic feature of our approach is that we allow transition probabilities to depend on customer-specific covariates $X_{t,k,s,a}$, specifying the covariates of a customer k who is exposed to action a when being in state s at time t , to switch to s' until time $t+1$. Estimates of transition probabilities are obtained using a multinomial logistic regression model with mean function

$$p_{t,k}(s, s'|a, X_{t,k,s,a}) = \frac{\exp(X_{t,k,s,a} \cdot \beta)}{1 + \exp(X_{t,k,s,a} \cdot \beta)} \quad (1)$$

Then, for a given sequence of actions a , the sequence of states which customer k visits within the time-horizon \mathcal{T} is the realization of a Markov Decision Process with state space \mathcal{S} , action space \mathcal{A}^n and transition probabilities $p_{t,k}(s, s'|a, X_{t,k,s,a})$.

D. Reward function

The magnitude of reward obtained from the application of action a to a customer depends on the target state, and may as well be affected by customer attributes and the outgoing state, e.g. elder customers or customers in certain states receive special offers. The expected reward by customer k moving from state s to s' under the regime of action a at time epoch t is denoted by

$$R_{t,k}(s, s'|a).$$

This may include expected damages produced by an insurant when applying the model in the context of insurance industry.

E. Costs

The expected reward generated by marketing intervention will be diminished by costs. Marketing costs are action-specific, as well in the sense that sophisticated mailings use to be more costly than simple mailings, while no action causes no costs. Let

$$c_t(a)$$

be the costs of action a applied to a customer which is canvassed at time epoch t .

F. Customer value generated by actions

Application of action a generates the following expected value of customer k which was in state s at time epoch t :

$$CV_{t,k}(s|a) = \sum_{s'} p_{t,k}(s, s'|a, X_{t,k,s,a}) R_{t,k}(s, s'|a) - c_t(a)$$

G. Objective function

Let $\alpha = (a_{t,k})_k$, for each decision epoch t , define a mapping from customers to actions. Then, the intermediate-term expected value of customer k which is in state s_t at time epoch t , given a policy α and a finite horizon of length T , is defined as

$$CV_{t,k}^\alpha(s_t) = \sum_{t'=t}^{T-1} CV_{t',k}(s'_t|a_{t',k}). \quad (2)$$

The optimal policy α is defined as the policy maximizing the intermediate-term expected value (2) for each customer.

III. OPTIMIZATION STRATEGY

A. Estimation of the MDP

Using past customer data concerning transition behavior and covariates, we use multinomial logistic regression models to estimate transition probabilities from one state to another in a Markov Decision Process.

B. Constraint optimization

Let marketing costs be given for each action and customer, and rewards in case of success. The intermediate-term values of customers, given a policy α will be optimized under the following constraints:

- The total marketing costs at a given time epoch t must not exceed a given budget C :

$$\sum_k c_t(a_{t,k}) \leq C$$

- Initial condition on customers: The number of customers in state s at time epoch 0 equals:

$$\sum_a \mathcal{N}_0(s|a),$$

where $\mathcal{N}_t(s|a)$ denotes the number of customers in state s at time epoch t experiencing action a .

- The total number of customers across all states is constant over time, i.e. the total number of customers moving to state s' generated by some action a at time

epoch t equals the number of customers in state s' at time epoch $t + 1$:

$$\sum_{a,s} \sum_{k \in \mathcal{N}_t(s|a)} p_{t,k}(s, s'|a) = \sum_a \mathcal{N}_{t+1}(s'|a).$$

IV. A NUMERICAL ILLUSTRATION

To illustrate our model, we give a fairly simple but realistic example and compare our approach to a standard approach, which neglects some of the information available. We assume that each customer is in one of three categories (states) at a given time. The state-dependent rewards (i.e. the revenue the customer contributes during one period) are given by

$$R(1) = 10, \quad R(2) = 20, \quad R(3) = 30.$$

Customers now in state s will be in state s' in the next period with a transition probability which depends on a customer-specific covariate x_k , which can be thought of as summarizing customer information, for example the score of a principal component analysis.

The transition probabilities for a customer k with covariate x_k are assumed to satisfy

$$\ln \frac{p_k(s, s'|a, x_k)}{p_k(s, 1|a, x_k)} = \beta_{0ss'} + \beta_{1ss'}x_k + \beta_{2ss'}a \quad (3)$$

for $s = 1, 2, 3$ and $s' = 2, 3$. (This is a multinomial logistic regression model with state 1 as baseline category.) Here, a indicates the action:

$$a = \begin{cases} 1 & \text{if customer } k \text{ is selected for mailing,} \\ 0 & \text{otherwise.} \end{cases}$$

(For simplicity, we only allow a dichotomous action in this illustration.) This specification leads to probabilities as given in equation (1). For a numerical illustration, the parameter values are:

$$\begin{aligned} \beta_{012} &= -2, & \beta_{112} &= 0.6, & \beta_{212} &= 1.5, \\ \beta_{013} &= -3, & \beta_{113} &= 1.0, & \beta_{213} &= 1.5, \\ \beta_{022} &= 1, & \beta_{122} &= -0.2, & \beta_{222} &= 2.0, \\ \beta_{023} &= -1, & \beta_{123} &= 0.8, & \beta_{223} &= 2.0, \\ \beta_{032} &= 0, & \beta_{132} &= 0.5, & \beta_{232} &= 1.5, \\ \beta_{033} &= 0, & \beta_{133} &= 2.4, & \beta_{233} &= 1.5. \end{aligned}$$

(All these parameters can be estimated from customer-specific data in real-world applications.) It is desirable to make customers move to state 3, where they contribute the highest amount to total revenue. The transition probability to a higher category can be increased by setting $a = 1$, i.e. by applying a marketing measure (mailing). This will incur a cost of $c = 3$ for each customer selected for mailing. The goal is to select customers for mailing such that the next periods's expected total revenue (from all customers combined) is maximized.

Our example assumes that there are initially 10000 customers in each category. Their covariates were taken as simulated values from a standard normal distribution. The idea to solve this optimization problem is as follows. To begin with, assume that no marketing measure is applied at all. The 30000 customers will then create a certain expected

total revenue. The first customer among the 30000 to be selected for mailing will be the one for whom the expected benefit of the action, that is: expected revenue with mailing, minus the sum of expected revenue without mailing and the cost of mailing, is the largest. This selection procedure can then be continued until further mailing is not meaningful anymore (because the marginal benefit for another mail is negative) or the marketing budget is exhausted.

How does the model outlined in this paper compare to a standard approach of action selection in this area? A typical model in customer value optimization practice uses fixed transition probabilities and separate customer scoring. For a comparison with our model, we first calculated from our dataset the matrix of average transition probabilities $p(s, s')$ between states, as well as expected customer values $CV(s)$ with respect to initial state s , generated without action, which define the baseline setting for estimating the benefit of mailing actions by customer scoring:

$$\begin{pmatrix} p(11) & p(12) & p(13) \\ p(21) & p(22) & p(23) \\ p(31) & p(32) & p(33) \end{pmatrix} = \begin{pmatrix} 0.819 & 0.123 & 0.058 \\ 0.235 & 0.644 & 0.121 \\ 0.338 & 0.265 & 0.398 \end{pmatrix}$$

$$\begin{pmatrix} CV(1) \\ CV(2) \\ CV(3) \end{pmatrix} = \begin{pmatrix} 12.38 \\ 18.86 \\ 20.60 \end{pmatrix}$$

We fitted a logistic regression model to mailing simulation data of next states. The target variable was defined to be 1 if a customer had switched from state 1 to 2, from 2 to 3, or had stayed with state 3, and it was set to 0 in all other cases. The values of the covariate were the same as those used in our model, but we added the initial state level information. As usual in customer scoring, the score values are interpreted as probabilities of switching to the target state.

For each customer k , the expected customer value conditional on selection for mailing $CV_k(s|a)$ was defined as score probability times reward from the target state plus converse probability times weighted reward from other states, less mailing costs. Customers are selected according to magnitude of benefit, which was calculated as $CV_k(s|a)$ compared to baseline expected customer value $CV(s)$.

Figure 1 shows the total expected revenues dependent on the number of mails — for the typical model (dotted line) in comparison with our model (solid line). Both lines reach a peak at about the same number of mails, that is, both models agree with respect to the optimal number of mails. However, our model outperforms the standard approach, which does not use state-specific transition probabilities based on a Markov Decision Process.

V. DISCUSSION

The transition probabilities in our model depend on covariates. Including covariates permits us to retain a simple state-based structure of the model while incorporating a potentially vast amount of information on each customer. This permits customer-specific action selection to optimize overall expected customer value. A numerical illustration

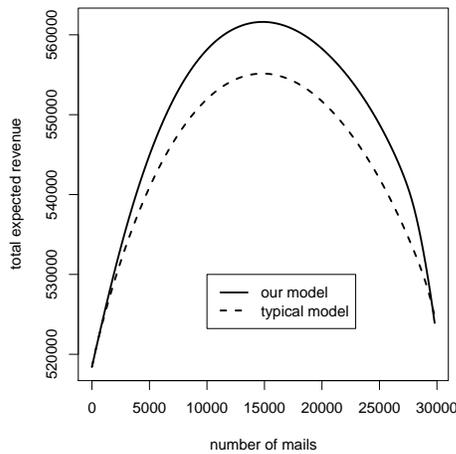


Fig. 1. Total expected revenue by number of mails: two models

with a realistic structure shows that this model can lead to higher expected revenues than the traditional approach, which separates the analysis of customer transition behavior and customer scoring.

REFERENCES

- [1] W.-K. Ching, M. K. Ng, *Markov Chains. Models Algorithms and Applications*, New York: Springer, 2006.
- [2] S. Gupta, D. Hanssens, B. Hardie, W. Kahn, V. Kumar, N. Lin, N. Ravishanker, S. Sriram, "Modeling customer lifetime value," *IBM J. Res. & Dev.*, vol. 51, pp. 421–431, May/July 2007.
- [3] A. Labbi, C. Berrospi, "Optimizing marketing planning and budgeting using Markov decision processes: An airline case study," *Journal of Interactive Marketing*, vol. 14, pp. 43–55, Spring 2000.
- [4] D. Peppers, M. Rogers, *The One-to-One Future Revisited*, Norwalk, CT: Peppers & Rogers Group, March 2004.
- [5] P. E. Pfeifer, R. L. Carraway, "Modeling customer relationships as Markov Chains," *Journal of Interactive Marketing*, vol. 14, pp. 43–55, Spring 2000.
- [6] R Development Core team, *R: A language and environment for statistical computing*, R foundation for statistical computing, Vienna: 2009. URL <http://www.R-project.org>.
- [7] R. T. Rust, P. C. Verhoef, "Optimizing the marketing interventions mix in intermediate-term CRM," *Marketing Science*, vol. 24, pp. 477–489, Summer 2005.
- [8] G. Tirenni, A. Labbi, A. Elisseeff, C. Berrospi, "Efficient allocation of marketing resources using dynamic programming," *Proc. SIAM International Conference on Data Mining*, Newport Beach, CA, 2005, pp. 581–585.

A Language Modeling Approach for the Classification of Music Pieces

Gonçalo Marques¹ and Thibault Langlois²

¹Instituto Superior de Engenharia de Lisboa, Portugal (email: gmarques@isел.pt)

²Universidade de Lisboa, Faculdade de Ciências, Departamento de Informática, Portugal (email: tl@di.fc.ul.pt)

Abstract—*The purpose of this paper is to present a method for the classification of musical pieces based on a language modeling approach. The method does not require any metadata and is used with raw audio format. It consists in 1) transforming music data into a sequence of symbols 2) building a model for each category by estimating n-grams from the sequences of symbols derived from the training set. The results obtained on three audio datasets show that, providing the amount of data is sufficient for estimating the transitions probabilities of the model, the approach performs very well. The performance achieved with the dataset used at the Musical Genre classification challenge of the International Society for Music Information Retrieval (ISMIR 2004) is comparable to the best published results on the same dataset.*

Keywords: Music Information Retrieval, Data Mining, Language Modeling

1. Introduction

The task of automatic genre classification, based solely on the audio contents of music signals, is a challenging one. Genre classification is not by any means consensual, even when performed by human experts. This is partly due to the complexity of music signals: a given song can be a mix of several genres. Therefore, it is not possible to achieve 100% accuracy in a classification system. Additionally, audio signals are not suited to be directly fed into a classification system, therefore some alternate, more compact representation is needed. Typically, some audio characteristics are extracted, such as timbre, chroma, chords, rhythm, melody or chorus. Nevertheless, it is difficult to combine the resulting features, since they often have different time scales.

Despite the complexity of the problem, techniques for music genre classification have attracted considerable attention in recent years (see for instance [1], [2] and references therein).

The most common approach to genre classification of audio music signals is to divide the signal in short overlapping frames (generally 10 – 100ms with a 50% overlap), and some features, usually based on the spectral representation of the frame are extracted (eg MFCCs, spectral spread, rolloff, centroid, etc). After this process, each music signal is represented by a sequence of feature vectors that can be thought of as samples from a distribution, which can be modeled by various techniques. Similarly, the distributions of the classes can be estimated by grouping songs of the same genre. For instance, *k*-means [3], or gaussian mixture models (GMM) [2], [4], [5]

can be used to model the class distributions. Once the models are obtained, one can use the “bag of frames” classifiers [6], compare models using a Earth Mover’s distance [3], or use the Kullback-Leibler divergence [2]. The main drawback with this type of approaches is that only the short-time characteristics of the signal are modeled. They do not take into account the ordering of the feature vectors, and therefore, the dynamics are discarded. To overcome this limitation several authors complement the short-time features with other sets of features that model the dynamics of the audio signal. Rhythmic features are a typical example [5], [7], but other long-term descriptors such as the fluctuation patterns in [4] can be used.

Another approach is to aggregate several short-time frames in larger scale windows (usually a few seconds) in order to capture the long-term dynamics. For example [5], [6], [8], [7] model temporal variations by calculating some statistics of the short time features over longer temporal windows. Some authors report a significant improvements in classification accuracy when the long-term windows are used, although the work of Aucouturier and Pachet [9] contradicts this result.

In this work, we use a language model approach to classify music signals in different genres. The most related works include Soltau et al [10], Chen et al. [11], and Li and Sleep [12].

In Soltau et al [10], each music is converted in sequence of 10 distinct music events, and unigram-counts, bigram-counts, and trigram-counts of the events and other statistics such as the event activation are fed as the features to a neural network used for classification. In Chen et al. [11] they propose to use a text categorization technique to perform musical genre classification. They build a HMM from the MFCC coefficients using the whole database. The set of symbols is represented by the states of the HMM. Music symbols are tokenized by computing 1 and 2-grams. The set of tokens is reduced using Latent Semantic Indexing. In Li and Sleep [12], a support vector machine is used as a classifier. The feature are based on n-grams of varying length obtained by a modified version of the Lempel-Ziv algorithm.

The outline of this paper is as follows. In section 2 we describe the language model approach. In section 3 the feature extraction and classification processes for audio files are explained. In the following two sections the results obtained with audio signal databases are evaluated. We close with some final conclusions and future work.

2. A language modeling approach

The idea behind our proposal is to use language modeling techniques[13] for the classification of music in audio format. In order to use this kind of approach we have to:

- 1) build a dictionary of symbols that are used to represent any song;
- 2) define a procedure which transforms a song into a sequence of symbols;
- 3) build a model for each category of music;
- 4) Find a procedure which, from a set of models and a sequence, determines the best model that fits this sequence.

3. Classification of Audio files

3.1 Two-stage clustering

For audio files we use classical twelve Mel Frequency Cepstrum Coefficients (MFCC) as the only feature¹. The first step consists in extracting the most representative frames for each song of the training set. This is done using the k -means clustering algorithm. The same value for the k parameter is used for every piece of music of the training set. We call k_1 the number of clusters per music used in this phase. We obtain $n \times k_1$ vectors where n is the number of songs of the training set. Let us call \mathcal{F}_1 this set.

The second step consists in finding a set of representative frames in \mathcal{F}_1 . Again, we use the k -means clustering algorithm. Let's call \mathcal{F}_2 the set of k_2 centroids obtained from the clustering. A symbol is assigned to each centroid. The dictionary \mathcal{D} is therefore composed of k_2 symbols.

The procedure used to transform a song into a sequence of symbols is as follows: 1. Compute the MFCC. 2. For each frame compute the 1st nearest neighbor in \mathcal{F}_2 and assign the corresponding symbol of the dictionary.

Thanks to this two stage approach, our algorithm is very scalable. We can process the whole music database and use the sets of k_1 centroids as a compact representation of musics. Several music genre models can be build based on this representation. This aspect contrasts from the approach proposed by Chen [11] where the whole set of MFCC frames are used to build a HMM for each genre.

3.2 Language model estimation

The following phase is the estimation of a language model for each category into which we want to classify the songs.

For each music category, the probability of each bi-gram is computed by processing every sequence of symbols and counting the occurrences of the symbols transitions. The result is a transition probability matrix that contains, for each pair of symbols (s_i, s_j) , the probability $P(s_j|s_i)$ of symbol s_i to be followed by the symbol s_j . In the context of a genre classification task, a model, represented by a transition

probability matrix is estimated for each genre by processing the n -grams of the files that belong to each genre.

After this estimation, the probability of many transitions is zero which is not desirable. Indeed the training sets used to estimate the models are finite and small. Without modification, if a single transition that has not been seen before in the training set is observed in a test sequence, the probability that the sequence belongs to the model would automatically be zero. In order to avoid this zero-frequency problem, the model is smoothed by adding a small constant $\epsilon = 1.0e - 5$ to each transition that has not been observed in the data set.

3.3 Classification of music files

The classification of a music file is done by transforming the music into a sequence of symbols and computing the probability that each model would generate this sequence. Given a model M , the probability that it would generate the sequence $S = s_1, s_2, \dots, s_n$ is:

$$P_M(s_{i=1..n}) = P_M(s_1) \prod_{i=2}^n P_M(s_i|s_{i-1}) \quad (1)$$

which is better calculated as:

$$S_M(s_{i=1..n}) = \log(P_M(s_{i=1..n})) \quad (2)$$

$$S_M(s_{i=1..n}) = \log(P_M(s_1)) + \sum_{i=2}^n \log(P_M(s_i|s_{i-1})) \quad (3)$$

This score is computed for each model M_j and the class corresponding to the model that maximize the score values is assigned to the sequence of symbols.

The approach described in this paper can be seen as a set of Vector Quantization-based Markov Models built for each category to be classified. It is, to our knowledge, the first time such approach is used for Musical Genre classification. The following sections describe some results obtained with this technique on various datasets.

4. Results

4.1 ISMIR 2004 Genre Classification

We used two different datasets to evaluate our method. The first one is the ISMIR 2004 genre classification dataset which is composed of six musical genres with a total of 729 songs for training and 729 songs for test². This data set was used for the Genre Classification contest organized in the context of the International Symposium on Music Information Retrieval - ISMIR 2004 (<http://ismir2004.ismir.net>). All the songs were courtesy from the Magnatune website where they are available under a Creative Commons (see creativecommons.org) license for non-commercial use.

¹All audio files were sampled at 22050 Hz, mono and frame duration of 93ms.

²The distribution of songs along the six genres is: classical: 319; electronic: 115 jazzblues: 26; metalpunk: 45; rockpop: 101; world: 123 for the training and the test set.

Table 1

PERCENTAGE OF CORRECTLY CLASSIFIED SONGS ON THE TEST SET, FOR VARIOUS k_1 AND k_2 PARAMETER VALUES.

k_1	k_2	% correct
10	100	79.29
20	150	79.15
30	150	80.52
20	200	80.11
30	200	80.52
40	200	80.11
20	300	79.84
30	300	79.97
20	400	80.25
40	400	79.42

Table 2

CONFUSION MATRIX OBTAINED WITH THE BEST RESULT OF TABLE 1. THE LAST COLUMN CORRESPOND TO THE PERCENTAGE OF CORRECTLY CLASSIFIED SONG FOR EACH GENRE.

CLASSICAL	300	1	0	0	0	19	93.75%
ELECTRONIC	2	95	1	1	7	8	83.33%
JAZZBLUES	0	3	14	0	6	3	53.85%
METALPUNK	0	0	0	20	23	2	44.44%
ROCKPOP	2	15	0	4	76	5	74.51%
WORLD	12	16	0	0	12	82	67.21%

Table 1 shows the percentage of correctly classified songs in the test set for various k_1 and k_2 parameter values. The best result (80.52%) is detailed in Table 2 where the confusion matrix is shown. This result must be compared to the results obtained by the participants of the ISMIR 2004 Genre classification Challenge³ and the results published thereafter. Pampalk et al. [4] obtained 84.07% and Annesi et al. [14] obtained 82.10%. If we weight the percentages with the prior probability of each class Pampalk obtains a 78.78% and we obtain 80.53%. Even if we do not obtain the best results for every evaluation metric, the results are interesting especially if we take into account that only simple spectral-based features are used. However, as noted by Aucouturier, we may be reaching a “glass ceiling” in this case.

4.2 Our dataset

The second dataset was made by us. It is composed of 7 genres: Jazz, Rock'n'Roll, Bossanova, Punk, Fado, Oriental, and Classical. We chose artists/albums that belong to each genre without ambiguity:

- Jazz: Dave Brubeck, Duke Ellington, John Coltrane, Miles Davis, Thelonious Monk and Louis Armstrong (110 songs).
- Rock'n'Roll: Bill Haley, Chuck Berry, Jerry Lee Lewis, Little Richard and The Shadows (167 songs).
- Bossa Nova: António Carlos Jobim, Dori Caymmi and João Gilberto (110 songs).
- Punk: Bad Religion, Buzzcocks, Down by Law, No Fun at All and Sham 69 (158 songs).

³See http://ismir2004.ismir.net/genre_contest/results.htm.

Table 3

PERCENTAGE OF CORRECTLY CLASSIFIED SONGS FOR VARIOUS k_1 AND k_2 VALUES.

k_1	k_2	% correct
10	25	74.88
10	50	81.03
10	100	86.21
20	100	84.98
20	200	85.71
20	300	86.70
20	400	86.95

Table 4

THE CONFUSION MATRIX OBTAINED WHEN USING $k_1 = 20$ AND $k_2 = 200$. THE LAST COLUMN SHOWS THE SUCCESS RATE FOR EACH CLASS.

JAZZ	44	6	2	0	0	4	0	78.57%
ROCKNROLL	2	76	1	4	0	1	0	90.48%
BOSSANOVA	3	2	47	0	2	1	0	85.45%
PUNK	1	20	0	58	0	0	0	73.42%
FADO	0	0	0	0	54	0	0	100.00%
ORIENTAL	4	3	0	0	0	37	0	84.09%
CLASSICAL	2	0	0	0	0	0	32	94.12%

- Fado: Ana Moura, Camané, Carlos do Carmo, Mafalda Arnauth and Mariza (109 songs).
- Oriental: Anouar Brahem, Rabih Abou-Khalil and Ravi Shankar (88 songs featuring traditional oriental string instruments such as esraj, sarangi and percussions).
- Classical: Several Piano Concertos, from the Romantic Period, by: Moscheles, Pierné, Parry, Stanford, Mendelssohn, Vianna da Motta, Balakirev, Rimsky-Korsakov, Alkan, Henselt and Kalkbrenner (69 songs).

Although the albums were chosen for being homogeneous in their musical style, there are exceptions, for example blues songs in a Rock'n'Roll album. These exceptions were **not** removed from the dataset. In the first set of experiments, we split every album in two, keeping the first half of the songs for the training set and the second half for test. The Table 3 shows the percentage of correctly classified songs on the test set. One can see the (little) sensibility of the algorithm with respect to a wide range of the parameters k_1 and k_2 . A typical confusion matrix is shown in Table 4.

We made a second set of experiments where 50% of the whole dataset was randomly selected for training. When repeating ten times this experiment (using $k_1 = 20$ and $k_2 = 200$) we obtain a mean success rate of 87.52% with a standard deviation equal to 1.87.

One of the reasons to constitute our own dataset was to be able to study the influence of various aspects. One of this aspects is whether the classifier is doing artist identification instead of genre classification. Pampalk [4] recommends using Artist Filtering⁴ (AF) in order to avoid this problem. We did a set of experiments with AF by selecting an artist for

⁴Artist Filtering consist building the datasets such that no artists appear in both training and test sets.

the test set of each genre while keeping the other artists for the training set. Repeating eight times we get an average success rate of 65% with a standard deviation of 4.37. These results confirms those described in [4]. The success rate is significantly lower *on average* than without AF, but if we look at the best performance, 71% of the songs are correctly classified. While doing these experiments, we learned a few lessons:

Our approach consists in building a model based on a representation of timbres (and probability transitions between these timbres). The approach is not immune to over-fitting but we think that failures are due mainly to the absence, in the training set, of a kind of timbre that is relevant to the musical genre we want to model. This is only partially related to the artist.

Certain conditions adversely affect our method. For example, when leaving the Bossa Nova artist João Gilberto in the test set, one of his albums was completely misclassified. It was a live recording with significant sequences of applause and speech. We believe this is the reason why it was not correctly classified. To be correctly classified we would need other live recordings in the training set.

Our method needs a large amount of data because it needs to collect representative samples of timbre that characterize a genre and estimate the probability transitions as closely as possible.

For illustrations purpose we show a picture of the transition probability matrix obtained for the Jazz-Blues genre of the ISMIR 2004 dataset on Figure 1. Each pixel represents $\log(P(s_j|s_i))$ (the quantity that is summed in equation 3). The diagonal with white pixels represents the transition probability to the same symbol (which is high), gray horizontal lines correspond to symbols that are very rarely found in that style and the other gray pixels show the contribution of the corresponding transitions towards the identification of the genre.

5. Conclusion and Future Work

We proposed a genre classification framework for music files, based on a language modeling approach. Experiments on audio music signals show the potential of the method. Our system performs well, especially if we take into account the simplicity of the features used. Also, it is worth noting that the classifier accuracy is not significantly affected by the values of k_1 and k_2 . In this work, due the size limitations of the datasets, we only estimated the probability of bi-grams, but we intend to build larger datasets to be able to estimate the transitions probabilities of three or more consecutive elements of the feature sequences. In the future, we also intend to experiment the use of vector quantization-based and continuous-density HMMs to model music genres.

Acknowledgements

This work was partially supported by FCT, through the Multi-annual Funding Programme.

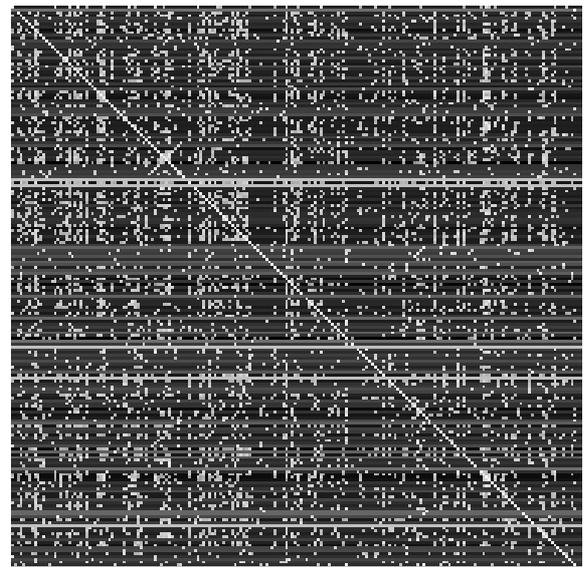


Fig. 1

THE TRANSITIONS PROBABILITIES MATRIX OBTAINED FOR THE JAZZBLUES GENRE OF THE ISMIR 2004 DATASET.

References

- [1] Berenzweig, A., Logan, B., Ellis, D., Whitman, B.: A large-scale evaluation of acoustic and subjective music similarity measures. *Computer Music Journal* **28** (2004) 63–76
- [2] Aucouturier, J.J., Pachet, F.: Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences* **1** (2004)
- [3] Logan, B., Salomon, A.: A music similarity function based on signal analysis. In: ICME. (2001)
- [4] Pampalk, E., Flexer, A., Widmer, G.: Improvements of audio-based music similarity and genre classification. In: ISMIR. (2005)
- [5] Tzanetakis, G., Cook, P.: Musical genre classification of audio singals. *IEEE Trans. on Speech and Audio Processing* **10** (2002) 293–302
- [6] West, K., Cox, S.: Finding an optimal segmentation for audio genre classification. In: ISMIR. (2005)
- [7] Lidy, T., Rauber, A.: Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In: ISMIR. (2005) 34–41
- [8] Bergstra, J., Casagrande, N., Erhan, D., Eck, D., Kégl, B.: Aggregate features and AdaBoost for music classification. *Machine Learning* **65** (2006) 473–484
- [9] Aucouturier, J.J., Pachet, F.: Improving timbre similarity: How high is the sky? *Pattern Recognition Letters* **28** (2007) 654–661
- [10] Soltau, H., Schultz, T., Westphal, M., Waibel, A.: Recognition of music types. In: ICASSP. (1998)
- [11] Chen, K., Gao, S., Zhu, Y., Sun, Q.: Music genres classification using text categorization method. In: MMSP. (2006) 221–224
- [12] Li, M., Sleep, R.: A robust approach to sequence classification. In: ICTAI. (2005)
- [13] Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: *Research and Development in Information Retrieval*. (1998) 275–281
- [14] Annesi, P., Basili, R., Gitto, R., Moschitti, A., Petitti, R.: Audio feature engineering for automatic music genre classification. In: RIAO, Pittsburgh (2007)

SESSION

DATA MINING FOR TIME SERIES DATA - FORECASTING, CLASSIFICATION AND CLUSTERING

Chair(s)

Dr. Sven F. Crone
Dr. Nikolaos Kourentzes

Dynamic Data Mining: A Novel Data Mining Process Model

Xiong Deng¹, Yike Guo¹, and Moustafa M. Ghanem¹

¹Data Mining Group, William Penny Lab, Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2AZ, UK

Abstract - This paper introduces a novel data mining process model for dynamic data mining. The process model is especially suitable for non-stationary continuous data environments. It offers three main improvements over traditional process models. First, it allows concurrent tasks and iterations to be included in the data mining steps. Second, it allows knowledge from previous modeling steps to be incrementally updated or promptly reused in mining new data which naturally fits existing incremental and online data mining approaches. Third, it introduces an explicit data pre-analysis step which enables traditional non-incremental data mining models to adapt to continuously changing knowledge. We present the details of the process model and provide an illustrative example of its use for handling concept drift.

Keywords: dynamic data mining, data mining process, data pre-analysis, model updating, model selection, model evaluation.

1 Introduction

The major reason that data mining has attracted a great deal of attention in the past decade is the wide availability of huge volumes of data and the imminent need for turning such data into useful knowledge. The traditional data mining process involves numerous iterative steps, normally including *data selection*, *data cleaning and preprocessing*, *data transformation and reduction*, *data mining task and algorithm selection*, and finally *post-processing and interpretation of discovered knowledge*. This process model was originally proposed by Fayyad *et al.* [1] in 1996, as shown in figure 1. Since then, several different data mining process models have been developed in both academia and industry [2], the most famous of which is Cross Industry Standard Process for Data Mining (CRISP-DM) [3]. CRISP-DM is the industry standard methodology for data mining and predictive analytics, consisting of six phases, *i.e.* *business understanding*, *data understanding*, *data preparation*, *modeling*, *evaluation* and *employment*, as shown in figure 2.

In general, there are the three main essentials inside these process models. Firstly, they are dominantly data-centric. Most of these process models are structured as sequences of steps that focus on performing manipulation and analysis of data and knowledge surrounding the data.

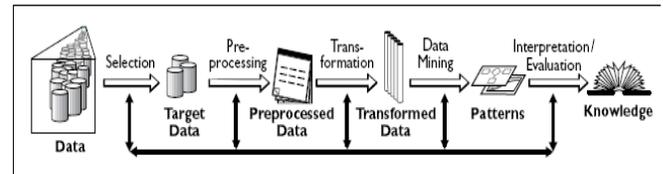


Fig. 1. Overview of the steps of traditional data mining process

Secondly, in the life cycle of the data mining processes, *data preparation*, *modeling*, and *knowledge interpretation and display* (with *evaluation*) are the three key sequential steps, bolstering the frame of data mining. Simply stated, during the process, data are first pre-processed within *data preparation* in order to adapt to the chosen data mining task or algorithms (data mining models). Then data mining models within *modeling* are applied to search the insights into these well-prepared data. Satisfied knowledge resulting from the analysis is determined by *evaluation* according to certain pre-defined requirement metrics and presented to users by *knowledge interpretation and display*.

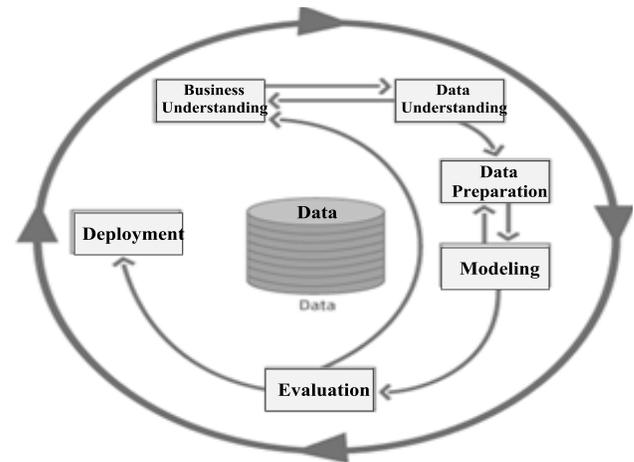


Fig. 2. Phases of the CRISP-DM process model

Thirdly, through a practical view, although iterations are involved in traditional data mining and lessons learned during a previous iteration can trigger new, often more focused iteration, most of the key steps themselves are still static and isolated from previous iterations. For example, the *modeling* step often involves repeated iterative application of particular data mining models. However, in each iterative step of the

process, the data mining models are simply re-trained, from the scratch, on all the existing pre-processed data and some new parameters. Reusability of previous data mining models is often out of consideration.

Nowadays, the modern organizations, such as large retailers, telecommunications providers, and scientific research projects, produce data at unprecedented rates per week, per day, even per minute, but there are not enough processing time or resources to recollect and rescan all the previous data efficiently prior to the analysis of the current data. In addition, the concepts inside these data may change over time leading to what is known as *concept drift* [4]. A large body of data mining algorithms have been suggested to address such issues, including work on stream and time series data mining as well as work on algorithms for online and incremental learning [5-7] [7-9]. Nevertheless, no new data mining process models have been developed to address the relevant issues. Traditional data mining approaches to collections, pre-processing and analyses are much less competent in this continuously fast-changing situation because they lack considerations of the continuity, non-stability and complexity of the data.

To address this challenge, we propose a novel data-centric data mining process model based on *dynamic data mining*. Through this process, data mining performance can be greatly improved. The process model includes three main improvements over traditional ones. First, the *modeling* step is parallelized. Second, through the process, existing valuable data mining models from previous iterative *modeling* step can be promptly reused for upcoming data analysis with a good performance. Third, the step of *data pre-analysis* is introduced into the process. Thereby, not only conventional non-incremental data mining models can be applied in the framework, but also novel mining algorithms such as *incremental mining* and *online mining* models, which can perform dynamical model updating, instead of retraining models, according to new data, are well-supported to response the changing knowledge and preferred through the process.

2 Dynamic Data Mining

2.1 Definition

Dynamic data mining [10-12] is an improvement of conventional data mining in the dimension of time. Due to the computational time and resource constraints of this kind of data analysis, the main principle of *dynamic data mining* is to support concurrent operations of dynamic knowledge updating and dynamic knowledge selection. On one side of the concurrency, current state of the knowledge base is applied to analyze the upcoming examples in an online manner; on the other side, the discrepancy between the mining results and the real facts would smoothly trigger automatic modifications to the knowledge base in either online or offline way.

Precisely, *dynamic (data) mining* can be defined as a data mining process in which knowledge obtained from previous mining process can be continually updated based on new arriving data without rescanning the previous data in progress. With the *dynamic mining* process, costs of computational time can be cut to some extent.

Moreover, in our *dynamic mining*, there are another two improvements to reduce the computational costs. First, the traditional execution dependence between the tasks of *training* and *classification* has been relieved. That is, under the condition of knowledge evolving, we do not have to train new data mining models before classifying new examples; Second, traditional non-incremental data mining models can be also utilized in the process to adapt to the changing knowledge. As a result, most of the traditional data mining achievements can be efficiently reused in the process. Generally, the proposed *dynamic mining* has the following six benefits.

- It performs *classification* and *training* in parallel.
- It correctly performs incremental update of the obtained knowledge;
- It can detect and adapt to *concept drift* automatically without being explicitly informed;
- It may deal with real-time data besides historic data, tolerating not more than a constant time delay;
- It may give continuous feedbacks and yield results anytime during mining;
- It may allow users to control the process in progress.

2.2 Process Overview

Figure 3 provides a general overview of the proposed process model for dynamic data mining. The process model has the similar backbone sequence to the conventional one in terms of *data preparation*, *modeling* (including *training* and *classification*), to *knowledge interpretation and display* (with *evaluation*). However, based on the special mining requirements, there are distinguished improvements and modifications in the process. Firstly, traditional *data preparation* is kept but often weakened. Secondly, *data pre-analysis*, *model updating*, and *model selection* are introduced and traditional *evaluation* is enhanced. These four steps can be concluded as the traditional *modeling*. Thirdly, fast and friend *knowledge interpretation and visualization* is also necessary. Besides, *dynamic data mining* is iterative. Here we roughly outline the process.

1. The classical *data preparation* is becoming simplified in *dynamic mining*. The urgent data analytics cannot afford the large costs of computational resources of elaborately pre-processing the large volumes of continuous and online data. Instead, it would rather sacrifice such data quality as completion, consistence and cleanness to an acceptable extent for a shorter time delay. Therefore basic and simple strategies for operations of *data preparation* are preferred in *dynamic data preparation*.

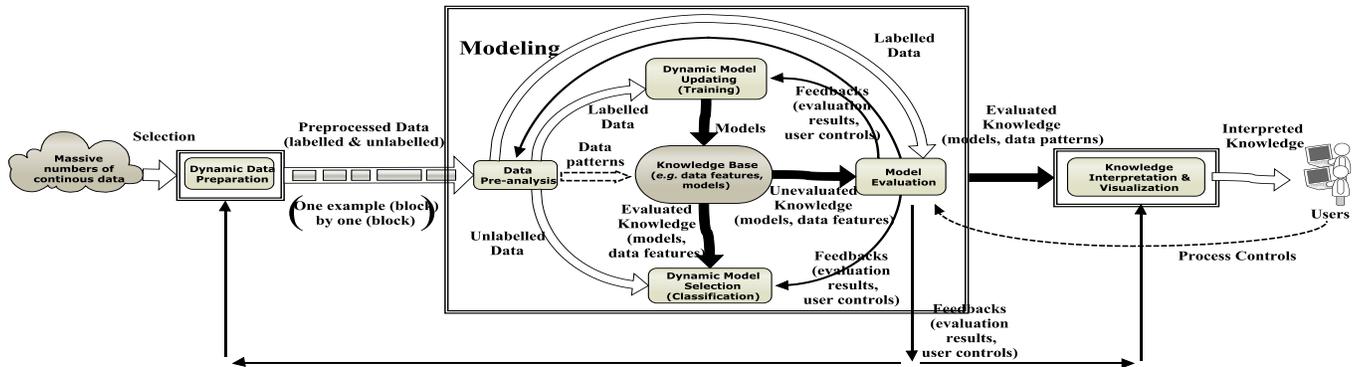


Fig. 3. Overview of the steps constituting the concurrent dynamic data mining process

2. In contrast, the proposed *data pre-analysis*, which takes the pre-processed data (blocks) as input, is emphasized in *dynamic mining*. *Data pre-analysis* is different from traditional *data preparation*. Firstly, *data pre-analysis* is conducted to discover useful data patterns (e.g. entropy of the data [13]), which are used to assist knowledge updating or selection in next steps. Secondly, *data pre-analysis* is performed to reduce computational costs of *model updating and selection*. The operations include effectively selecting data and appropriately organizing data. For example, *sliding window* [8] and *data weighting* (e.g. the *decay coefficients*) [14] are two common *data pre-analysis* techniques, both of which are introduced to facilitate further information extraction and knowledge discovery. *Advanced data sampling*, which is beyond traditional *data preparation*, can be also included into this step. *Advanced data sampling* of *dynamic mining* is more effective than classical *naïve sampling* such as *random* or *stratified sampling* in traditional *data mining*. It is normally based on evaluation of data examples, selecting data with the consideration of removing examples but not losing mining performance. One instance of *advanced data sampling* is Cheng's algorithm [15], in which non-informative training examples are eliminated incrementally based on their weights.
3. After *data pre-analysis*, two parallel processes are conducted. One process is *model selection*. *Model selection* is conducted to choose relatively good knowledge from the *knowledge base* for upcoming analyses. It is majorly used for online classification, choosing valuable models to classify real-time unlabelled data examples through certain schemes.
4. The other process is *model evaluation and updating*. *Model evaluation*, which is used to evaluate mining performance and detect *concept*, plays important role in *dynamic mining*, because *concept drift* and degradation of model performance occur frequently and commonly in *dynamic mining*. It normally starts after unlabelled examples become labelled; *Model updating* incrementally updates knowledge in the *knowledge base* on continuous data, including creating, updating and

removing of the knowledge. Normally, *model updating* is invoked after *model evaluation*, since the evaluated results indicate what and how to update the knowledge.

5. The evaluated results from *model evaluation*, together with user feedbacks, act as continuous adjustments to activate *environment* tuning, and then the changed *environment* again stimulates tuning of the *dynamic mining* process

The state-of-the-art knowledge display techniques can be used to display and explain the mining results in the step of *knowledge interpretation and visualization*. The more interactive, visualized, user-friendly and timely the knowledge-display is the more beneficial the *dynamic mining*.

3 Process Refinement of Dynamic Data Mining

3.1 Data Pre-analysis

Data pre-analysis is the step between the traditional *data preparation* and *modeling*. In this process, a series of effective analyses are performed on the pre-processed data. These data include blocks of labeled and unlabelled examples. In the step, some *data patterns* may be discovered to represent the processing data. *Data patterns* are summarized information to feature the data, e.g. the data's *entropy* [13] and *self-similarity* measured by *fractal dimension* [16]. It is useful for assist *model updating and selection*. For example, for each non-incremental traditional mining model, a data pattern is attached, which are used to match the incoming data for *model selection*. A data pattern that has not been matched for long may indicate the corresponding non-incremental models are outdated. Thereafter, the models are removed. Why we introduce this step into *dynamic mining* is that the pre-analysis techniques can greatly reduce data analysis costs, improve mining performance and effectively aid model updating. The goals are achieved by using the following primary techniques.

- *Sliding window* is a kind of data organization methods. Many incremental mining algorithms take *sliding window* into account, including *data based window* [8;17], in which the data are collected in certain numbers, and *time based window* [14;18], in which the data are collected

based on certain time intervals.

- *Advanced data selection and reduction* techniques offer data samples for *model updating* or *selection*, aiming to reduce unnecessary examples without losing mining performance. They include the approaches to calculating the number of examples required to train the models and then fetch this number of examples without taking into account characteristics of each single example [19;20], and evaluating examples through certain characteristics and then filtering the examples which potentially cannot improve the model performance [15;21].
- *Data weighting* measures the importance of examples (or example features [22;23]) for *model updating* or *selection* by assigning them weights. Data (or data features) with greater weights play more important role in updating or selecting the models. Normally, there are two weighting schemes: (1) *static weighting*; (2) *dynamic weighting* [14;24]. *Static weighting* allocates stable weights to examples, while the weights are changeable in *dynamic weighting*. For example, α -ISVM [24] introduces the changing distribution knowledge as the weights of examples. In *dynamic mining*, the latter is more suitable due to *concept drift* in the data.
- *Data structuring* provides data separation and organization mechanisms for effective *model updating* or *selection*, rather than simply learn the data sequentially or orderly. These techniques either group training or testing data based on certain principles in order to effectively offer more valuable data to model learning [24;25] or splitting data into portions, which are learned by different models later, in order to effectively adapt to different concepts in the data [15;16]. For example, in α -ISVM [24], the dataset is organized into the three sub-sets: working set, caching set, and backup set. The examples can be moved from one to another based on their changing distribution knowledge during the incremental training process.
- *Data summarizing* techniques generalize data, and summarize their patterns for *model updating* and *selection*. These techniques include not only the classical statistical measures and tests such as means, variance and F-test, but also the advanced ones such as *wavelet analysis*, *principal component analysis* (PCA), *Fourier analysis*, and even *clustering*. For example, Lim *et al.* [26] propose to cluster the input data into different recognition categories and then train probabilistic neural networks (PNNs) on these categories, which results in a significant reduction of the pattern nodes in PNNs. In AWSOM [27], data are first represented by *wavelet*, and then *autoregression* is used to model the wavelet-represented data¹.

3.2 Model Updating and Selection

Model updating and selection, taking blocks of labeled and unlabeled examples from data pre-analysis as inputs are preformed to achieve the traditional tasks of model training and classification. *Model selection* is the step of choosing and combining appropriate knowledge for classification. There are the three kinds of model selection methods: *selecting single model*, *combining multi-models* and *combining predictions of multi-models*.

- *Selecting single model*: The simplest way to conduct classification based on *model selection* is to select a single model for a block of examples.
- *Combining multi-model* refers to that multi-models are chosen, and then combined into next level's models. *Stacking* [28] and *meta-learning* [29] are in this group.
- *Combining predictions of multi-models* indicates the methods of combining the prediction results of selected models (*e.g. voting*).

Through *model updating*, knowledge can be created, removed, and updated. The knowledge, here, is represented in *incremental models*, which can acquire new valuable knowledge while forgetting old invaluable knowledge, the data patterns created from *data pre-analysis*, and certain *novel knowledge representations* helping conventional models, which are not incremental, to update knowledge incrementally. The following techniques are popularly employed to update models.

- *Structural adaption* is widely-used in structure-based models such as *decision trees* and *neural networks* for incremental knowledge updating. In these models, knowledge is mainly represented in the form of model structures. Updating of knowledge can therefore be conducted by modifying the structures.
- *Model weighting* is popularly used in updating of single or multiple models. In the case of updating a single model, different knowledge inside a model is assigned different weights. Through *model weighting*, the knowledge which becomes less (or more) accurate in analyzing past data is gradually attached a smaller (or bigger) weight and therefore plays less (or more) important role in analyzing current and future data. For example, most structure-based models can perform *dynamic mining* with the assistance of *model weighting* [14;30]. That is knowledge updating in these models can be conducted by assigning different importance to different parts of the model structures. The existing structures that contain the knowledge turning to be not precise are ignored with less and less weights, and vice versa. In the case of updating multi-models, weights are assigned to models instead of portions of each single model. For example, in *ensemble*, the multi-model weighting is widely used.
- *Ensemble* is an overproduce and choose method [31], carried to construct a set of diverse classifiers (*i.e.*

¹ *I.e.*, the *autoregression* is performed to model the wavelet coefficients at each wavelet level.

models)² and then classify unlabelled examples by taking a (weighted or unweighted) vote of their predictions. The technique handles the issues of how to discount old data in favour of new data in a distributed manner. Multiple models are flexible to adapt to *concept drift* in *dynamic mining*, and ensembles are reported to be much more accurate than individual classifiers that make them up.

- *Hybrid learning* combines two or more incremental models into one integrated model or employs diverse aforementioned dynamic model updating techniques together. For example, Potts [32;33] has introduced a binary *decision tree* model with a linear functional model in each leaf to mine data incrementally.
- *Novel knowledge representations and effective data structures*: In order to assist in dynamic knowledge updating of conventional data mining models which are not incremental, *novel knowledge representations* and *effective data structures* are introduced. *Novel knowledge representations* symbolizing knowledge to support dynamic model updating are proposed on the basis of certain uncommon designing considerations. Take BIRTH [34] as an example, although traditional *clustering* is not incremental, the *clustering features* (CF) in BIRCH is proposed to store summarized information of a cluster, through which merging of two clusters can be performed through a linear addition of the CFs of the two clusters instead of re-learning all the examples. *Effective data structures* are maintained to effectively store and locate the models, through which the computing complexity of updating models can be greatly reduced. These structures are mainly *index-based*, *tree-based* or a hybrid of them. They are often utilized in *frequent-itemset-based* mining algorithms [35-37], though they can theoretically be utilized to help performance improvement in any kind of models. For instance, in IMCS [37], a compact data structure called *CSTree* is designed to assist in maintaining the incrementally-changing closed sequential patterns.

3.3 Model Evaluation

Model evaluation evaluates model performance and detects *concept drift* in *dynamic mining*. It is normally performed in the way of taking existing models and new data examples as input, conducting evaluation internally, and sending feedbacks as output to other steps. The feedbacks are important incentives for tuning parameters of the mining process. For example, the feedbacks may arouse *structural adaption* in model updating, activate changes of sizes of *sliding windows*, and stimulate modifications of data weights in *data pre-analysis*. There are some points which are of relevance to *model evaluation*.

- *Choosing metrics*: During *dynamic mining*, traditional

model performance metrics are still available. There are two reasons for this. Firstly, performance metrics, such as *accuracy*, are often pre-defined based on requirements of a mining project before the data analysis. Secondly, most traditional metrics take no or little effect of the dynamic knowledge updating, since most knowledge updating schemes are performed without reference to the metrics.

- *Detecting concept drift* is a frequently emerging operation in *dynamic mining*. Drifting concept makes models built on old data inconsistent with new data. The nature of *concept drift* is diverse and abundant. An obvious cause of *concept drift* can be changes in characteristic properties in the observed example [38]. Moreover, it is believed that the hidden contexts in mining domains can influence the behavior of the *concept drift* nature [39]. For example, in finance, a successful stock buying strategy can change dramatically in response to interest rate changes, world events, or with the seasons. Hidden contexts may change over time. As a result, concepts learned at one time can become inaccurate. Therefore, the main *concept drift detecting* schemes can be abstracted as: (1) monitoring the changes of data features, *e.g.* distribution of examples and data's *entropy*; (2) measuring model performance drops; (3) comparing model relevance, *e.g.* the complexity of the current rules [40] and properties of classifiers in *model ensemble*. Whenever a change in the underlying concept generating data occurs, the data patterns change, at least in some regions of the example space. However, it is possible to observe changes in the data patterns without *concept drift* [38]. This is referred as *virtual drift* [41]. Therefore, measurement of model performance and comparison of model properties (especially in the context of model ensemble) can be performed to post-detect the *concept drift*. That is a big performance drop³ and obvious differences (of model properties) between newly-trained and old models can be indications of *concept drift*. Normally, the data patterns are summarized in the *data pre-analysis* stage, while model performance and model relevance can be computed from the *dynamic model updating* stage.

4 Handling Concept Drift: An Illustrative Example

In order to practically explain the aforementioned stages and techniques in *dynamic mining*, in this section, we dwell on a specific application of *dynamic mining* as an example: handling *concept drift*. In order to effectively handle *concept drift*, *dynamic mining* continuously updates the mining models in the manner of integrating the techniques discussed

² Two classifiers are diverse if they make different errors on new examples.

³ A *concept drift* is possible if a drop in performance is detected, but the drift is determined if the drop is persistent. That is termed *true concept drift*. Otherwise, it is *concept noise*.

above. Due to the space limitation of the paper, we present only the three main steps of *dynamic mining* dealing with *concept drift*.

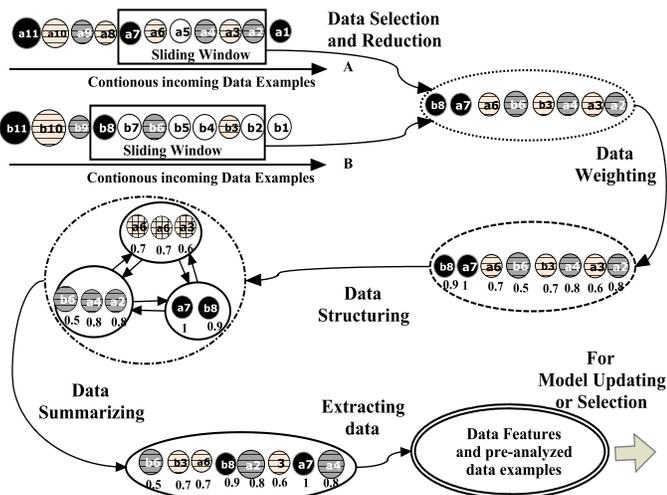


Fig. 4. Illustrative example of pre-analyzing examples

In *data pre-analysis*, an illustrative combination of the techniques is demonstrated, as shown in figure 4. The circles with different filling patterns represent the continuous labeled or unlabeled data examples with different potential patterns. The indices of each example represent time stamps. The examples from multi-sources are first sampled and combined through the multiple *windows*. The *Hoeffding-bounds*-based data selection scheme used in VFDT is employed [19]. Then *data weighting* is performed to evaluate the importance of each examples (e.g. based on their distributions [24]), which are then grouped according to the data structuring schemes as α -*ISVM* [24]. After that, the examples are extracted from the structure without having to be in their original time order. Finally, *data summarizing* is performed on these examples using *wavelet* as [27]. The final output of this step is the summarized data patterns and the pre-analyzed examples.

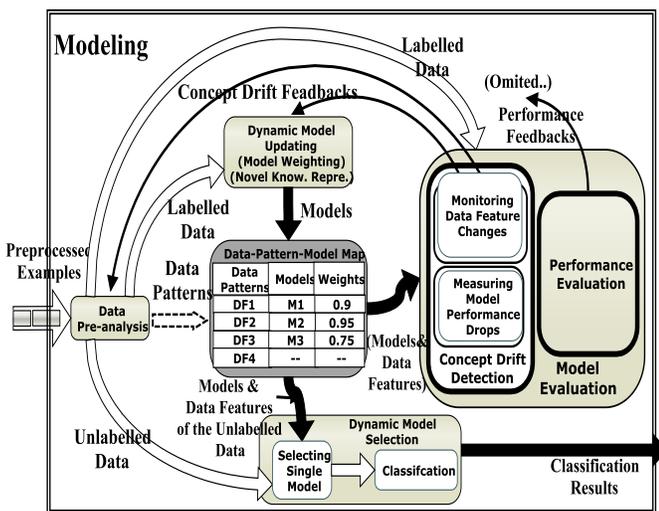


Fig. 5. Illustrative example of model updating, selection and evaluation

In *model updating* and *selection*, traditional *decision trees* are selected as the models, which are trained on the

examples which are pre-analyzed. Mappings (*i.e. a novel knowledge representation*) between the data patterns and trained models (with their changing weights⁴) are then built up. During selection, a single model is promptly chosen through matching the data pattern of an unlabeled example block with those in the map. When updating, the data-pattern-model entries in the maps are updated (including the weights), removed or added to reflect the changing concept. In that case, a non-incremental model can be identified as the outdated by a low weight, therefore is removed and new model is learned with its data pattern. The maps are updatable under a passive model updating scheme with the support of *model evaluation*. With this scheme, whenever a *concept drift* is detected, *model updating* is stimulated to update the maps by the spurs sent from *concept drift detection* in *model evaluation*. The *concept drift* can be detected through prior-detection (*i.e. monitoring data pattern changes* [42]) and post-detection (*i.e. measuring model performance drops* [41]).

5 Relationship to Related Work

The idea of conducting automatic adaption to non-stationary knowledge has been proposed for years. Many adaptive data mining algorithms such as incremental [5-7], online [7-9] and dynamic mining [10-12] algorithms have been designed. However, to the best of our knowledge, there has not been a unified process model to support most of the techniques used in them before our work. In addition, there is no other work that focuses on improving data mining process in eliminating task dependence.

Although there have not been universal agreements on the recognitions of incremental mining and online mining, they can be broadly included into dynamic data mining, which the proposed dynamic mining also belongs to.

In comparison with batch learning [43], incremental mining approaches⁵ [5-7] are addressed to the situations where training datasets are not fully available at the beginning of a mining process or there are so many data that the mining models are not able to be trained on all of them at one time. Incremental mining is generally performed through firstly processing subsets of the data and then incrementally expanding the subsets. Compared with offline learning [45], online mining approaches [7-9] oppose the challenges in which data arrive straightly after they are produced without being stored. Online mining has to guarantee the analysis results are produced up-to-date and with small time delay.

Incremental and online mining are performed under either limited memory or time, much less multiple passes over the data, and often are anytime controllable and stoppable with an acceptably deterministic and accurate result. All of these are the basic features of the proposed dynamic mining. Moreover, many of the techniques used in

4 The weights can be used to measure the frequency the model is selected and the accuracy of the model.

5 Zhou *et al.* [44] further dissect *incremental mining* into example-incremental learning (E-IL), class-incremental learning (C-IL), and attribute-incremental learning (A-IL).

these algorithms fit in the process of dynamic mining (These techniques have been summarized in section 3).

Hundreds of these incremental and online data mining algorithms have been published. Here, the three typical algorithms are presented: the FLORA family [41], the VFDT series [19;42;46-48] and the OLIN series [8;13;49]. In these algorithms, aforementioned dynamic mining techniques are widely employed. The family of FLORA (1995) [41] is among the first dynamic mining algorithms. Distinguished from traditional mining algorithms, the algorithms are based on the proposed principle that knowledge forgetting should permit fast recovery after a context change by getting rid of outdated and contradictory information incrementally. FLORA introduces a knowledge forgetting operator, controlled by a dynamic *data based window* over input stream. Besides, a *novel knowledge representation* is represented in the form of three sets. Through the dynamic transfer of the items from one set to another, the algorithms is able to react to *concept drift* and takes advantage of the situations where a context reappears. The serial algorithms of VFDT deal directly with the issue of mining high-speed online data streams. VFDT (2000) [19] incrementally builds decision trees on selected examples according to the *Hoeffding bounds (advanced data selection)*. CVFDT (2001) [47] is then designed as an extension to VFDT to deal with *concept drift* under one-pass mining. It works through keeping its model consistent with an adjustable *window* of examples and growing alternative subtrees (*structural adaptation*) immediately after *concept drift*. VFDTc (2003) [48] extends VFDT by introducing the Naïve Bayes classifiers at the tree leaves (*hybrid learning*). The Naïve Bayes takes into account not only the prior distribution of the classes, but also the conditional probabilities of the attribute-values given the class. Thereby there is a much better exploitation of the available information at each leaf. Gama *et al.* (2006) [42] then extend VFDTc to deal with *concept drift* by continuously monitoring differences between the two class-distribution (*data summarizing*) of the examples: the distribution when a decision tree node was built and the distribution in a time *window* of the most recent examples. fVFDT (2007) [46] improves VFDT and VFDTc by handling continuous attributes.

The OLIN series are addressed to the problem of mining continuous non-stationary data stream. In OLIN (2002) [8], the knowledge is represented in the form of an info-fuzzy network (IFN), a treelike classification model (*novel knowledge representations*). The system is repeatedly constructing a new IFN from a *sliding window* of latest examples. The latest model classifies the examples that arrive before the subsequent network re-construction. *Concept drift* is detected with an unexpected rise in the classification error rate. With the detection of a concept drift, the system recalculates the size of the training *window* by using the principles of information theory and statistics. IOLIN (2004) [49] is an incremental version of OLIN. The basic intuition behind the approach is to update, instead of regenerate, an OLIN model in the current training *window*. The Multiple-Model IOLIN (2005) [13] is an enhanced algorithm for

IOLIN. It searches for the best model representing the current data from all the old ones, if *concept drift* is detected. The Pure Multiple-Model IOLIN (2005) [13] is a variation of the Multiple-Model IOLIN. When the concept is stable, it will use an old model, instead of update, the current model. With this approach, it is expected to save additional running time because an appropriate model will probably be found in the adjacent former *windows*.

6 Conclusion and Future Work

In this paper, we have introduced a novel data mining process model for *dynamic data mining*, which is particularly suitable for fast-changing continuous data environment. Detailed techniques involved in the process have been summarized; illustrative application study of this process has been presented. The proposed data mining process model can be considered a unifying framework, in which most existing dynamic knowledge-adapting techniques such as *incremental* and *online data mining* can be concurrently running, to conduct parallel adaption to non-stationary knowledge. In the framework,

Performance evaluation of the process model through real world applications is being conducted. Based on the model, we will design more efficient *dynamic mining* algorithms than existing ones in the near future.

7 References

- [1] F. Usama, P. Gregory, and S. Padhraic, "The KDD process for extracting useful knowledge from volumes of data," *Commun. ACM*, vol. 39, no. 11, pp. 27-34, 1996.
- [2] L. A. KURGAN, *et al.*, "A survey of Knowledge Discovery and Data Mining process models," *The Knowledge Engineering Review*, vol. 21, no. 01, pp. 1-24, 2006.
- [3] The CRISP-DM consortium, "CRISP-DM," 2000.
- [4] A. Tsymbal, "The problem of concept drift: Definitions and related work," Department of Computer Science, Trinity College, Dublin, Technical Report TCD-CS-2004-15, 2004.
- [5] G. Fung, *et al.*, "Incremental support vector machine classification," Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, 2001.
- [6] R. Polikar, *et al.*, "Learn++: a classifier independent incremental learning algorithm for supervised neural networks," 2 ed. J. Byorick, Ed. 2002, pp. 1742-1747.
- [7] W. G. Aref, *et al.*, "Incremental, online, and merge mining of partial periodic patterns in time-series databases," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 16, no. 3, pp. 332-342, 2004.
- [8] M. Last, "Online classification of nonstationary data streams," *Intelligent Data Analysis*, vol. 6, no. 2, pp. 129-147, 2002.
- [9] Q. Zhang, "Online Least Squares Support Vector Machines Based on Wavelet and Its Applications," *Lecture notes in computer science*, vol. 4493, p. 416, 2007.
- [10] Seokkyung Chung and Dennis McLeod, "Dynamic Pattern Mining: An Incremental Data Clustering Approach," *Journal on Data Semantics II*, pp. 85-112, 2005.

- [11] Vijay Raghavan and Alaaeldin Hafez, "Dynamic Data Mining," in *Intelligent Problem Solving. Methodologies and Approaches* Springer Berlin / Heidelberg, 2000, pp. 207-238.
- [12] F. Crespo, "A methodology for dynamic data mining based on fuzzy clustering," *Fuzzy Sets and Systems*, vol. 150, no. 2, p. 267, 2005.
- [13] L.Cohen, G.Avrahami, M.Last, A.Kandel, and O.Kipersztok, "Incremental Classification of Nonstationary Data Streams," Porto, Portugal.: 2005, pp. 117-124.
- [14] A. Tveit, M. Hetland, and H. Engum, "Incremental and Decremental Proximal Support Vector Classification using Decay Coefficients," 2003, pp. 422-423.
- [15] S. Cheng, *et al.*, "An improved incremental training algorithm for support vector machines using active query," *Pattern Recognition*, vol. 40, no. 3, pp. 964-971, Mar.2007.
- [16] G. Folino, *et al.*, "An Adaptive Distributed Ensemble Approach to Mine Concept-Drifting Data Streams," 2 ed. C. Pizzuti, Ed. 2007, pp. 183-188.
- [17] C. Yun, W. Haixun, P. S. Yu, and R. R. Muntz, "Moment: maintaining closed frequent itemsets over a stream sliding window," W. Haixun, Ed. 2004, pp. 59-66.
- [18] J. Cheng, Y. Ke, and W. Ng, "Maintaining frequent closed itemsets over a sliding window," *Journal of Intelligent Information Systems*, 2007.
- [19] D. Pedro, *et al.*, "Mining high-speed data streams," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* Boston, Massachusetts, United States: ACM, 2000, pp. 71-80.
- [20] Joao Gama, *et al.*, "Forest trees for on-line data," in *Proceedings of the 2004 ACM symposium on Applied computing* Nicosia, Cyprus: ACM, 2004, pp. 632-636.
- [21] R. Liva, d. Florence, and -Buc, "Incremental Support Vector Machine Learning: A Local Approach," in *Proceedings of the International Conference on Artificial Neural Networks* Springer-Verlag, 2001, pp. 322-330.
- [22] J. Mendez, *et al.*, "Tracking Concept Drift at Feature Selection Stage in SpamHunting: An Anti-spam Instance-Based Reasoning System," 2006, pp. 504-518.
- [23] R. C. Vitor and W. C. William, "Single-pass online learning: performance, voting schemes and online feature selection," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* Philadelphia, PA, USA: ACM, 2006, pp. 548-553.
- [24] X. Rong, W. Jicheng, and Z. Fayan, "An approach to incremental SVM learning algorithm," 2000, pp. 268-273.
- [25] Gert Cauwenberghs and Tomaso Poggio, "Incremental and decremental support vector machine learning," in *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000* Denver, CO, USA: MIT Press, 2000, pp. 409-415.
- [26] C. P. Lim, "An Incremental Adaptive Network for On-line Supervised Learning and Probability Estimation," *Neural Networks*, vol. 10, no. 5, p. 925, 1997.
- [27] S. Papadimitriou, *et al.*, "Adaptive, unsupervised stream mining," *Vldb Journal*, vol. 13, no. 3, pp. 222-239, 2004.
- [28] H. W. David, "Stacked generalization," *Neural Netw.*, vol. 5, no. 2, pp. 241-259, 1992.
- [29] R. Vilalta and Y. Drissi, "A Perspective View and Survey of Meta-Learning," *Artificial Intelligence Review*, vol. 18, no. 2, pp. 77-95, 2002.
- [30] H. Wu, "AttributeNets: An Incremental Learning Method for Interpretable Classification," *Lecture notes in computer science*, vol. 4426, p. 940, 2007.
- [31] F. Roli, G. Giacinto, and G. Vernazza, "Methods for Designing Multiple Classifier Systems," 2001, pp. 78-87.
- [32] D. Potts, "Fast Incremental Learning of Linear Model Trees," in *PRICAI 2004: Trends in Artificial Intelligence 2004*, pp. 221-230.
- [33] D. Potts, "Incremental Learning of Linear Model Trees," *Machine Learning*, vol. 61, no. 1, p. 5, 2005.
- [34] Z. Tian, R. Raghu, and L. Miron, "BIRCH: an efficient data clustering method for very large databases," *SIGMOD Rec.*, vol. 25, no. 2, pp. 103-114, 1996.
- [35] Ke Wang, *et al.*, "Incremental discovery of sequential patterns," Montreal, Canada: ACM, 1996, pp. 95-102.
- [36] Y. Chen, *et al.*, "Incremental Mining of Sequential Patterns Using Prefix Tree," 2007, pp. 433-440.
- [37] L. Chang, *et al.*, "IMCS: Incremental Mining of Closed Sequential Patterns," 2007, pp. 50-61.
- [38] J. Gama, "Learning with Local Drift Detection," *Lecture notes in computer science*, vol. 4093, p. 42, 2006.
- [39] M. B. Harries, "Extracting Hidden Context," *Machine Learning*, vol. 32, no. 2, p. 101, 1998.
- [40] R. Klinkenberg and I. Renz, "Adaptive information filtering: learning in the presence of concept drift," Madison, WI.: AAAI Press, 1998, pp. 33-40.
- [41] G. Widmer, "Learning in the presence of concept drift and hidden contexts," *Machine Learning*, vol. 23, no. 1, p. 69, 1996.
- [42] Joao Gama, *et al.*, "Decision trees for mining data streams," *Intell. Data Anal.*, vol. 10, no. 1, pp. 23-45, 2006.
- [43] A. Sharma, "A note on batch and incremental learnability," *Journal of Computer and System Sciences*, vol. 56, no. 3, p. 272, 1998.
- [44] Z. H. Zhou, *et al.*, "Hybrid decision tree," *Knowledge-Based Systems*, vol. 15, no. 8, pp. 515-528, Nov.2002.
- [45] S. Ben-David, "Online learning versus offline learning," *Machine Learning*, vol. 29, no. 1, p. 45, 1997.
- [46] T. Wang, "An Incremental Fuzzy Decision Tree Classification Method for Mining Data Streams," *Lecture notes in computer science*, vol. 4571, p. 91, 2007.
- [47] H. Geoff, *et al.*, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* San Francisco, California: ACM, 2001, pp. 97-106.
- [48] Joao Gama, R. Ricardo, and M. Pedro, "Accurate decision trees for mining high-speed data streams," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* Washington, D.C.: ACM, 2003, pp. 523-528.
- [49] L.Cohen, G.Avrahami, and M.Last, "Incremental Info-Fuzzy Algorithm for Real Time Data Mining of Non-Stationary Data Streams," Brighton UK: 2004.

ATM's Daily Cash Money Demand Forecasting with Recurrent Neural Networks

M. Abou-Nasr¹

¹Ford Motor Company, Dearborn, Michigan, U.S.A.

Abstract - A simple technique is presented in this paper for forecasting ATM's daily cash money demand with recurrent neural networks. Eleven time series from the NN5 competition were used for evaluating the proposed technique. No special measures were taken to deal with seasonality or trend in the time series. The forecasting performance of this technique is compared to the performances of the contestants in the NN5 competition. The average SMAPE on the out-of-sample data for the 11 series in this work was 24.69 which would rank it at the 12th place in comparison with the CI submissions. The results are encouraging given the simplicity of the technique and they motivate future research for further enhancements.

Keywords: Recurrent, Neural Networks, Time Series, SMAPE.

1 Introduction

Literature survey reveals mixed findings and contradictory conclusions regarding the effectiveness of neural networks as models for time series forecasting. While some of the published research indicates that neural networks forecasting models are generally better than the simpler linear models [1]-[4], several studies showed that neural networks have performed about the same as or even worse than simpler linear models [5].

Competitions like the M3, NN3 and the NN5 provide rich platforms for researchers to evaluate and compare the effectiveness of their methods. The M3 competition for example provides diverse time series of different periodicities (yearly, quarterly, monthly and others) and different types (industry, finance, demographic and others). The NN3 provides 111 time series representing monthly empirical business time series, while the NN5 provides 111 time series representing daily cash money demand at automatic teller machines at different locations in England.

The aim of this paper is to examine the effectiveness of a technique based on recurrent neural networks that was used in the NN3 competition for forecasting monthly time series,

in forecasting the daily time series representing cash money demand at England ATM's provided in NN5 competition. In the NN3 competition an ensemble of 10 recurrent neural networks trained on all the in-sample data, was used to forecast the out-of-sample data. Despite its simplicity, its forecasting results were the best submitted for the IJCNN07 and seventh among all submissions world-wide. After obtaining the complete NN3 time series including the withheld test data, an evaluation of the decisions that were made during the completion was conducted. The evaluation has resulted in a modification of the technique. Instead of using a an ensemble of 10 networks trained on all the in-sample data, I form ensembles of different sizes (5-20) networks, each of which are trained on a subset of the in-sample-data after withholding 10% for testing. The different ensembles are then tested on the withheld data and the best performer as measured by the average SMAPE is selected to forecast the out-of-sample data. The majority of the forecasts obtained by the modified techniques were better than the forecasts submitted to the NN3 competition.

In this work, the technique is simplified to use only one recurrent network out of three that are multi-stream trained [6], based on its average SMAPE performance on the last 56 points of the in-sample data which were withheld for this purpose. A comparison between network selections based on the average SMAPE vs. the average RMS performance is also made. The experiments are performed on the subset of the time series that are provided in the NN5 competition, named "Reduced Set", including series 101-111.

The rest of this paper is organized as follows. In section 2, the research methodology for this work is described. Results are reported in section 3. Summary and conclusions are provided in section 4.

2 Research Methodology

In order to evaluate the simplified recurrent neural network technique mentioned above, I use the reduced set of the NN5 competition including the test data that was withheld during the competition. I was not one of the participants in that competition, but I compare the average

SMAPE obtained by the simplified technique to the average SMAPE obtained by the contestants as published in the NN5 competition web site.

2.1 NN5 Data

The data consists of two years of daily cash money demand at various automatic teller machines (ATMs, or cash machines) at different locations in England. The data may contain a number of time series patterns, including multiple overlying seasonality including day of the week effects, week or month in the year effects, calendar effects that are fixed to a specific date (e.g. Christmas), calendar effects that are linked to a public holiday that may move around the year (e.g. Easter) etc. They contain both observations of "0" withdrawals, which indicate that no money was withdrawn at that day, and missing observations which are blank " " without any values, which indicate days in which no value was recorded for various reasons. No missing values exist in the test set, but 0 values may occur. All time series start on March 18, 1996 and run until March 22, 1998, providing two years of daily data. The objective is to predict the next 56 days (i.e. 8 weeks) into the future [7].

In the technique I am evaluating in this paper, no special measures are taken to deal with seasonality or trends in the time series. Their effects on the technique's forecasting accuracy will be studied in future work. The only data transformation performed is to scaling the data to be within (-1, 1) to facilitate network training. The data are rescaled back during evaluating the network forecasting performance. The missing data were replaced with zeros. This decision may not have been the best as the missing data may represent data that were not recorded for unknown reasons.

2.2 Neural Networks

I use three-layer recurrent neural network [6] in this work. Biases are used at all the network nodes. A sigmoid transfer function is employed at each node of the two hidden layers of the network and a linear transfer function is employed at the single output node of each network. There is only one input to the network and with some empirical experimentation the number of hidden recurrent nodes was chosen to be five. In this particular recurrent network, the output of each node in the first hidden layer is fed back to the inputs of all the other nodes in this layer and to its own input with a fixed delay time which was one in this configuration. The second hidden layer has three nodes; each has a sigmoid transfer function and no recurrent connections.

2.3 Training Approach

The implicit assumption behind the NN training approach for this work is that the essence of each of the time series can be described by a generative model, i.e., that each point in the

series can be predicted to some extent from past values of the series. As a generative model, a recurrent network is employed that has a single input, a single output and internal states to afford it the possibility of richer dynamics. The input value is the current value of the series and the output of the network is a prediction of the next value of the series. Training is organized into trajectories, making use of the series points that are known. A trajectory is conceptually similar to a moving window across the data. The training software randomly chooses the beginning point of each trajectory during a training epoch. In this training approach, actual points from the series in the beginning of the trajectory are used for training. The number of these points is L . For the rest of the points in the trajectory M , we feed the output of the network to its input where L and M are integers. Figures 1 and 2, depict the training during the L and M parts of the trajectory respectively.

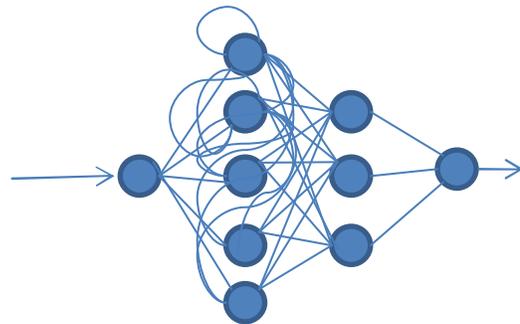


Fig.1 Recurrent Neural Network during the L part of the training trajectory. Input is from the in-sample data.

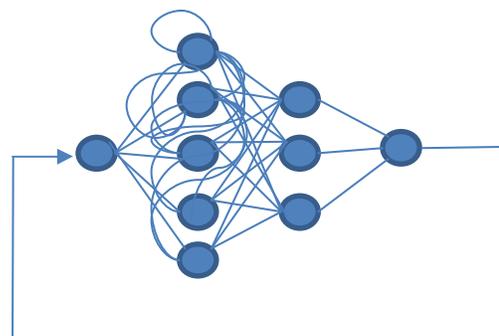


Fig.2 Recurrent Neural Network during the M part of the training trajectory. Input $i(t)$ is previous network output $o(t-1)$.

The idea is to train the network to forecast consecutive outputs when there are no more in-sample points in the series. The choice of L and M is a trade-off between how

much of previous history is needed for the network to produce reliable forecasts, and how much variety of patterns are needed to effectively train the network. The M or the portion of the trajectory when the network is fed with its previous output at its input is chosen such that it is greater than or equal to the forecasting horizon in a particular forecasting task. To produce the 56 points in the forecast horizon for the competition, we set L to be the number of the in-sample data in the series, and we set M to be 56, thus feeding the outputs of the network to its input immediately after the end of the in-sample data. During the training, the network weights are adjusted using multi-stream extended Kalman Filter (EKF) training as described in [6] to minimize the RMS error between the provided series target value and the network output. The EKF training parameters can be found in [6]. They roughly affect the network learning rate and can be adjusted for each training task to achieve specific training goals, e.g. $RMS < target\ RMS$. Figure 3 outlines a pseudo-code of the training algorithm.

```

Start
  Select Architecture:
    No of layers, No of nodes, Type of each node
    (linear/nonlinear)
    Delay values for recurrent nodes
    Connectivity pattern
  Randomly initialize network weights
  Scale data values between -1, 1
  Select the no of trajectories for this training session

For this training session
  Select trajectory length = L + M
  Select the number of initialization points: L
  Select the number of forecast points: M
  Select the EKF training parameters

For n epochs:
  Minimize the RMS error between the target and
  the network output (forecast) over the trajectory

  If (RMS < target RMS)
    Save network representation
    Exit training
  End If
End For
End
    
```

Fig.3. Recurrent Network training Algorithm

3 Evaluation Results

For each time series of the reduced set of the NN5, I trained three networks on a subset of the set that does not include the last 56 points of the in-sample data. Each of the networks was trained using a trajectory that has $L = 112$ points, but the M part of the trajectory was different for the three networks. For the first network, $M = 112$ points, for the second, $M = 136$ and for the third $M = 56$ points. In essence, the experiments will reveal whether the generative network can sustain itself with its own forecasting for a short, medium or a long time. After testing on the withheld in-sample data, one of the networks was chosen based on its SMAPE performance, to forecast the withheld competition data. The results for different series are shown in the following tables. The best testing performances on the withheld in-sample and out-of-sample data are denoted by an asterisk in the respective columns of each table.

TABLE I
Results for NN5_101 Time Series

L	M	TESTING	TESTING	TESTING
		RMS	SMAPE	SMAPE
		(WITHHELD	(WITHHELD	(WITHHELD
		56 POINTS	56 POINTS	56 POINTS
		IN-SAMPLE	IN-SAMPLE	COMPETITION
		DATA)	DATA)	DATA)
112	112	0.326	22.0	19.3
112	136	0.407	30.2	19.01
112	56	0.30*	19.6*	18.4*

TABLE II
Results for NN5_102 Time Series

L	M	TESTING	TESTING	TESTING
		RMS	SMAPE	SMAPE
		(WITHHELD	(WITHHELD	(WITHHELD
		56 POINTS	56 POINTS	56 POINTS
		IN-SAMPLE	IN-SAMPLE	COMPETITION
		DATA)	DATA)	DATA)
112	112	0.211	35.32	32.7
112	136	0.186*	28.5*	26.5*
112	56	0.241	40	37.6

TABLE III
Results for NN5_103 Time Series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.145	20.09	27.94
112	136	0.127*	18.02*	0.265*
112	56	0.142	20.33	0.376

TABLE VI
Results for NN5_106 Time Series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.347*	50.8	20.2
112	136	0.356	49.6*	19.5*
112	56	0.406	59.1	22.12

TABLE IV
Results for NN5_104 Time series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.258	37.03	24.94
112	136	0.277	39.81	29.3
112	56	0.253*	34.7*	22.05*

TABLE VII
Results for NN5_107 Time series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.148*	25.5*	27.83
112	136	0.303	47.01	29.32
112	56	0.385	68.89	27.74*

TABLE V
Results for NN5_105 Time series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.222	27.74	33.04
112	136	0.222	27.49	24.6*
112	56	0.220*	27.13*	24.7

TABLE VIII
Results for NN5_108 Time series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.170*	27.1*	20.4
112	136	0.171	27.4	20.34
112	56	0.277	37.4	20.2*

TABLE IX
Results for NN5_109 Time series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.262	35.98	14.0
112	136	0.100*	16.25*	13.4*
112	56	0.274	38.27	14.6

TABLE X
Results for NN5_110 Time Series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.230	30.5	45.1
112	136	0.198	30.5	43.3*
112	56	0.189*	26.7*	51.3

TABLE XI
Results for NN5_111 Time Series

L	M	TESTING RMS	TESTING SMAPE	TESTING SMAPE
		(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS IN-SAMPLE DATA)	(WITHHELD 56 POINTS COMPETITION DATA)
112	112	0.107	20.3	21.9
112	136	0.104	20.02	19.8*
112	56	0.102*	19.13*	23.4

The network SMAPE forecasting performance on the withheld 56 in-sample points coincides with the RMS forecasting performance in 10 out of the 11 series. So, at least in these experiments either the SMAPE or the RMS performance measures could be used to evaluate the network forecasting performance.

The SMAPE performance on the withheld 56 in-sample points was a good predictor of the performance on the 56 out-of-sample withheld competition points in 6 out of the 11 series. In three out of the remaining five series, the network selected by the lowest SMAPE performance on the withheld in-sample data did not differ by much from the network of best performance on the out-of-sample data. In the other two series, NN5-110 and NN5-111 the networks selected by the SMAPE performance on the withheld in-sample data differed from the best performing network on the out-of-sample data by 8% and 3.6% respectively.

The average SMAPE on the out-of-sample data for the 11 series in this work was 24.69 which would rank it at the 12th place in comparison with the CI submissions. This would suggest that the technique is a contender but needs some enhancements. An area that may lead to performance improvement is the handling of seasonality and trend in the time series. Another is averaging the forecasts from different forecasting origins to avoid the cases where the forecasting origin is an outlier [8]. Conducting more training experiments with different values of L and M of the training trajectory will provide a richer pool to select from based on the networks forecasting performance on the in-sample withheld data. It would be also interesting to compare the forecasting performance of a single selected network to the performance of an ensemble of the best performing networks which will be the subject of future work.

In Figures 4-7, the best and the worst forecasting performances are shown. Figure 4 shows the NN5_109 time series and Figure 5 shows the comparison between the network forecasts and the actual out-of-sample data that was not provided during the competition. This is the best performance obtained by this technique based on the SMAPE performance measure. Figure 6 shows the NN5_110 time series, while Figure 7 shows the comparison between the network forecasts and the actual out-of-sample data. According to the SMAPE performance measure in table X, this is the worst forecasting performance obtained by this technique.

4 Conclusions

Reflecting on the results in table I-XI and denoting the best performance in each of the last three columns in the tables by an asterisk, I make the following observations.

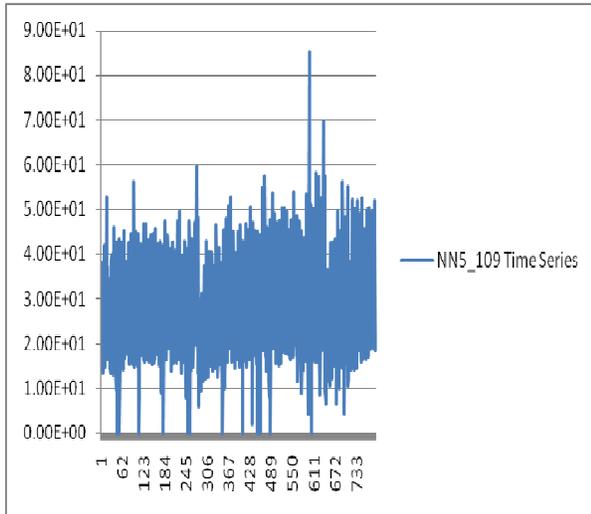


Fig.4 NN5_109 Time Series

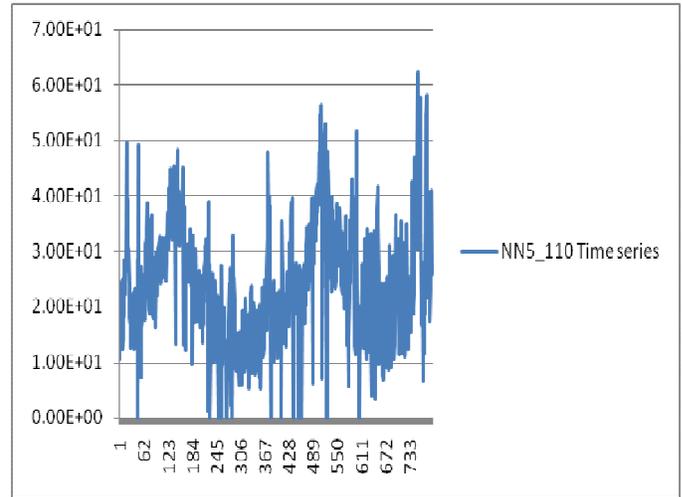


Fig.6 NN5_110 Time Series

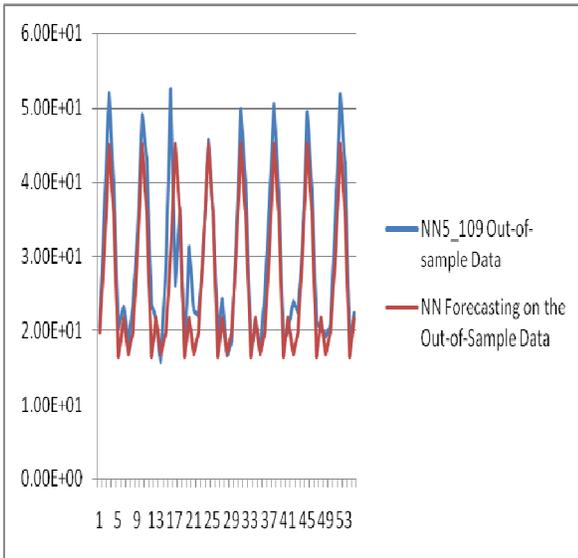


Fig.5 Comparison of the Neural Network Forecasting of the Actual Out-of-Sample Data for Time Series NN5_109

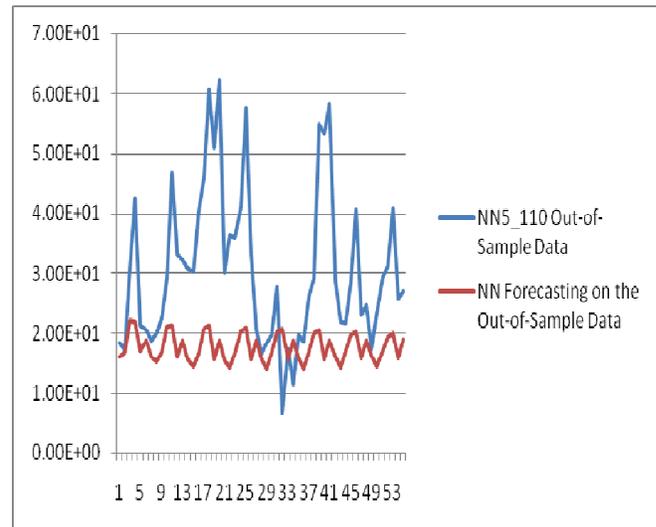


Fig.7 Comparison of the Neural Network Forecasting to the Actual Out-of-Sample Data for Time Series NN5_110

5 References

- [1] T. Hill, M.O'Connor, and W. Remus, "Neural network models for time series forecasts," *Manag. Sci.*, vol. 42, pp. 1082–1092, 1996.
- [2] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks," *Int. J. Forecasting*, vol. 14 (8), pp. 35–62, 1998.
- [3] S. D. Balkin and J. K. Ord, "Automatic neural network modeling for univariate time series," *Int. J. Forecasting*, vol. 16 (4), pp. 509–15, 2000.
- [4] G. Q.Zhang, B. E. Patuwo, and M. Y Hu, "Forecasting with artificial neural networks: The state of the art," *Int. J. Of Forecasting*, vol. 14, pp. 35-62, 1998.
- [5] C. Ntungo, and M. Boyd, "Commodity futures trading performance using neural network models versus ARIMA models," *The Journal of Futures Markets*, vol. 18, pp. 965-983, 1998
- [6] S.F Crone, NN5 Forecasting Competition for Artificial neural networks& computational intelligence, (2008) <http://www.neural-forecasting-competition.com/>
- [7] L.J. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International Journal of Forecasting*, vol. 16, pp. 437-450, 2000.

Incorporating Conditional Probability Functions in Time-Dependent Bayesian Networks

Dung N. Lam and Cheryl E. Martin

Applied Research Laboratories, The University of Texas at Austin, USA

Abstract – A Time-dependent Bayesian Network (TDBN) model is presented for explicitly modeling dependencies on time as trend-sensitive temporal nodes in a Bayesian network. Existing Bayesian network learning algorithms are used to model correlational patterns in a dataset, and time series analysis techniques are then used to learn time-dependent conditional and prior probabilities. This approach enables extrapolations of Bayesian probabilities to future time periods. The TDBN model captures time-dependent patterns and trends, and this paper demonstrates improved prediction accuracy over previous models in forecasting events.

Keywords: Bayesian networks, time series forecasting

1 Introduction

THIS research addresses the need to adapt Bayesian network (BN) models for use in event forecasting domains that are inherently dependent on time and temporal trends. Although data mining using BN learning algorithms is a powerful technique for modeling patterns in a dataset [1], a BN cannot extrapolate temporal trends, nor can it model non-stationary (or dynamically changing) temporal dependencies. Our research addresses these two critical gaps with an innovation that combines time-series analysis (TSA) with BN learning to create a Time-dependent Bayesian Network (TDBN) model from a set of event records.

In learning a BN, as in most other data-mining methods, data instances are assumed to be independent. Thus, their relative temporal order and the temporal distances over which they occur are not considered unless time is discretized and represented as one or more attributes. In those cases, each derived temporal attribute is treated as any other categorical or numeric feature. The approach presented in this paper recognizes the additional information content of temporal attributes, which can help in modeling both cyclic patterns and non-cyclic trends (e.g., growth or decay trends).

The TDBN model leverages probability distribution functions (PDFs) and extends a BN model's ability to represent conditional probabilities by introducing *conditional probability functions* (CPFs). CPFs address the need to represent temporal trends in conditional probabilities. This approach leads to improved predictions of correlational strengths at future time periods (i.e., improved forecasting).

2 Related Work

There have been numerous previous efforts to incorporate temporal information into BN models [2-5]. Traditional Bayesian networks [6] can capture cyclic patterns (such as seasonal or weekly cycles) by introducing temporal attributes (such as “month” or “day-of-week”), which are treated as typical nodes in a Bayesian network. Non-cyclic trends (such as growth and decay) can also be modeled, but are limited in scope to the range of time values considered when building the model (e.g., those contained in a training dataset). If an estimate outside this temporal range is desired, traditional Bayesian networks cannot extrapolate trends in the probability distribution. The TDBN model addresses the need to extrapolate non-cyclic trends.

Dynamic Bayesian Networks (DBNs) [5] evolved from numerous efforts to model time-dependent Bayesian systems. DBNs account for temporal dependencies through the use of hidden nodes, which affect nodes in future time slices. A DBN model consists of (1) an initial BN model representing the probabilities at time 0, and (2) a transition BN model representing all successive time slices. Predictions for future times can be made by “unrolling” the DBN model for each intermediary time slice until the target time is reached [7].

DBNs are constrained by the Markov assumption. Hence, information from time slices preceding the previous time slice cannot be used, and trends over several time slices cannot be leveraged. In addition, the effects of smoothing are absent in DBNs. For instance, if the state of the previous time slice is incorrect (e.g., due to noise), then predictions based on the previous state may be erroneous. In contrast, the TDBN model uses time-dependent functions to model temporal correlations. The functions are fitted to and smoothed over the entire temporal scope of the training dataset and can be readily extrapolated to any future time.

To address another limitation of DBNs—the reliance on using the appropriate granularity for each discrete time slice—Nodelman, Shelton, and Koller [8] developed Continuous Time Bayesian Networks (CTBNs), which enabled time slices to have variable duration. However, like the DBN model, CTBN is based on a homogeneous Markov process. Hence, the CTBN model does not use trend information from before the previous time slice. Furthermore, time is modeled relative to the last state change, whereas in the TDBN model, absolute time can be used to model trends over the entire time period of the training data.

DBN models and their derivatives differ significantly in their target usage from TDBN models. DBNs are more appropriate for modeling sequences of states, while TDBNs are more appropriate for modeling temporal patterns or trends. Instead of outputting discrete states (e.g., event counts are in an “increasing” or “decreasing” state) like DBN models, TDBN models can be used to forecast numeric counts of events with specified characteristics occurring within a specified time period.

The TDBN approach is similar to Tawfik and Neufeld’s work [4] that proposed *probability transfer functions* to describe instantaneous event occurrences and the decay of conditional probabilities over time (i.e., survival functions), thereby allowing dependence on time to be modeled totally within a Bayesian network node. However, this work differs from the TDBN approach in that (1) the survival functions’ dependence on time must be relative rather than absolute, and (2) a subject matter expert is relied upon to analyze the data and create the necessary generalized time-dependent functions rather than learning the functions from training data.

A major contribution of the TDBN approach is the use of time series analysis (TSA) techniques to automate the learning of probability functions from a training dataset. TSA techniques include data smoothing, identifying auto- and cross-correlations, seasonal modeling, and trend estimation. TSA has been previously applied to countless applications where time is a critical factor [9].

3 TDBN Concepts and Definitions

A TDBN model contains temporal nodes representing time (e.g., “year”, “month”, or “day-of-week”). It also contains time-dependent probability functions, which are parameterized by temporal attribute(s). Where applicable, these functions replace a subset of the discrete probabilities traditionally found in conditional probability tables (CPTs) of Bayesian network nodes. In addition to enabling extrapolation to future times, probability functions can offer a more intuitive and concise understanding of the behavior of probabilities than a tabular CPT representation.

In the TDBN approach, time-dependent probability functions are learned by performing temporal analysis (e.g., smoothing, regression, cross-correlation analysis, or other TSA techniques) on the probability values over time. Probability values in CPTs are viewed from two aspects: all probabilities in a row of a CPT can be represented by a *probability distribution function* (PDF), and a subset of probabilities within a column can be represented by a *conditional probability function* (CPF). These two aspects are described in the subsections below.

3.1 Probability Distribution Functions

In a learned BN model, each node represents a dataset attribute. Within a CPT for a node, all probabilities in a row constitute a probability distribution across all known possible values of the attribute, given specific values for all parent nodes. A PDF can be fitted to the data and used to represent all the probability values in a CPT row. A PDF for a temporal attribute T is defined as follows:

$$\begin{aligned}
 \text{PDF}(T=t) &\equiv P(t | pa(T)) \\
 &\text{where } pa(T) \text{ are the parents of } T \\
 &\text{and } \int \text{PDF}(t) dt = 1.0
 \end{aligned}
 \tag{1}$$

As an illustration, Table 1 provides an event count of the number of times any person was observed going out to lunch on a given day. To simplify the example, the *day* attribute is represented as relative to a particular origin time point. The increasing trend is hypothetical; it could be a result of some exogenous variable, such as a popular new restaurant. Table 2 shows CPT values derived from this dataset (only the relevant CPT row is shown; the BN topology and conditional dependencies on any parent nodes could vary). Fig. 1 plots the probabilities for each day from Table 2 shows that a simple line fits the values (without overfitting the data). A PDF describing the fitted line can replace all the probabilities in the CPT row, as shown in Table 3.

To estimate or forecast an event count under specified conditions, the joint probability for those conditions is multiplied by a scaling factor called a count coefficient, C , denoted as C_{day} in Table 3. The count coefficient is the total number of events from which the probabilities in the row were calculated. Using count coefficients and extrapolating probability functions enables more accurate predictions of event counts for future times (forecasting).

Table 1: Example event count data for each day

	Day									
	1	2	3	4	5	6	7	8	9	
Event Count	6	10	13	16	17	20	24	26	20	152 total

Table 2: Probability table for the *day* attribute

	Day									
	1	2	3	4	5	6	7	8	9	
Probability	0.04	0.07	0.09	0.11	0.11	0.13	0.16	0.17	0.13	1.0 total

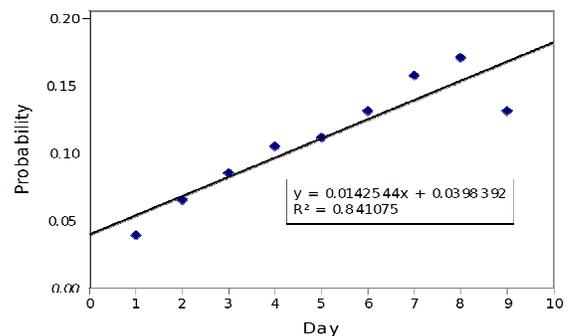


Fig. 1: Plot of probability values from Table 1 with linear trend line

Table 3: Probability table for the *day* attribute using a time-dependent probability function

Day								
d=1	d=2	d=3	d=4	d=5	d=6	d=7	d=8	d=9
PDF(d)=0.0143 * d + 0.0398; C _{day} =152								

3.2 Conditional Probability Functions

Probabilities in columns of CPTs represent the probability of an attribute having a certain value across various possible combinations of parent attribute values. A CPF is applied when a functional relationship exists across the values of one (or more) of the parents, given values for the remaining parent attributes. The TDBN model focuses on creating CPFs using temporal parameters (e.g., *hourOfDay*); however, CPFs can be applied with any parameter type. A CPF for attribute A when $A=a$ is defined as follows:

$$\text{CPF}_{A=a,pa(A)-T}(T=t) \equiv P(t | pa(A)-T) \quad (2)$$

where $pa(A)-T$ are the parents of A excluding T

Suppose each event record in a training dataset describes the day and hour of the day a person goes out to lunch and whether the person ate breakfast that day. Fig. 2 shows a Bayesian network topology learned from this dataset.

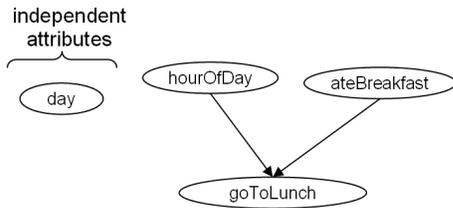


Fig. 2: BN topology for the *goToLunch* example

A TDBN model includes all attributes specified in the training dataset. In this example, the *day* attribute is independent of any other attributes, and the *goToLunch* attribute is dependent on the *hourOfDay* and *ateBreakfast* parent attributes. Table 4 shows the CPT for the binary *goToLunch* attribute. The left two columns enumerate the possible combinations of attribute values for the parent attributes, and the right two columns contain conditional probability values for each possible value of the *goToLunch* attribute (in this case, “Yes” and “No”).

There are functional relationships within the CPT columns in Table 4 that can be modeled as conditional probability functions parameterized by *hourOfDay*. Specifically, a functional relationship exists for column $goToLunch=Yes$ over the *hourOfDay* attribute values, given $ateBreakfast=Yes$. As shown in Table 5, the probability of a person’s going out to lunch given that s(he) ate breakfast is a normal (Gaussian) distribution centered around $hourOfDay=12$:

$$\begin{aligned} \text{CPF}_{goToLunch=Yes,ateBreakfast=yes}(hourOfDay=h) \\ &\equiv P(h | ateBreakfast=yes) \\ &= N(h-12,1) \end{aligned} \quad (3)$$

where N is a Gaussian distribution function

Note that each CPF may account for only a subset of the probabilities in a column, unlike a PDF, which must account for all probabilities in a row. Extrapolations of relevant CPFs can be performed to get the probability at $hourOfDay=15$. If desired, interpolations can also be performed using the CPFs to get the probability at $hourOfDay=11.5$ (or 11:30 AM).

Table 4: CPT for the *goToLunch* BN node

<i>ateBreakfast</i>	<i>hourOfDay</i>	$P(goToLunch ateBreakfast, hourOfDay)$	
		Yes	No
yes	9	0.1	0.9
yes	10	0.2	0.8
yes	11	0.4	0.6
yes	12	0.7	0.3
yes	13	0.4	0.6
yes	14	0.2	0.8
no	9	0.2	0.8
no	10	0.4	0.6
no	11	0.7	0.3
no	12	0.4	0.6
no	13	0.2	0.8
no	14	0.1	0.9

Table 5: CPT with derived CPFs, where $N(\text{mean}, \text{variance})$ is the normal distribution function

<i>ateBreakfast</i>	$P(goToLunch ateBreakfast, hourOfDay)$	
	Yes	No
yes	$N(hourOfDay-12,1)$	$1 - N(hourOfDay-12,1)$
no	$N(hourOfDay-11,1)$	$1 - N(hourOfDay-11,1)$

3.3 Estimating Event Counts

To forecast the probability of an event with a certain set of attribute values $\mathbf{E} = \{e_1, e_2, e_3, \dots\}$ at a given time T , Bayesian inferencing is performed to construct a formula that calculates the joint probability that an event with those characteristics occurs. This joint probability formula is written in terms of prior and conditional probabilities, whose values can be found in the CPTs of the Bayesian network model. When a particular probability is needed and a probability function exists for that CPT entry, then the probability function (which may be a PDF or CPF) is evaluated for the given attribute values \mathbf{E} . To forecast a count of events for the given set of characteristics \mathbf{E} , the joint probability is multiplied by the count coefficient, C , corresponding to the temporal scope of \mathbf{E} .

4 TDBN Application to Synthetic Dataset

This section illustrates the proposed approach for building a TDBN model based on event records.

4.1 Generative Model for Synthetic Dataset

A synthetic dataset was created based on a generative model consisting of a known BN model and injected temporal patterns and trends. The generative BN model represents causal dependencies in the Computer Network Defense (CND) domain (as interpreted at an abstract level by a domain expert) and is shown in Fig. 3. The generated dataset consists of time-stamped records, each representing an attack event on a target (network or computer) caused by a particular hacker and resulting in some amount of damage.

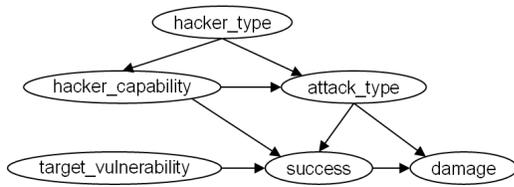


Fig. 3: Ground-truth Bayesian network for the CND domain

There is an increasing growth trend in event counts over the years 2001 to 2009, as shown in the histogram in Fig. 4. The histograms show different hacker capabilities in corresponding shades of gray.

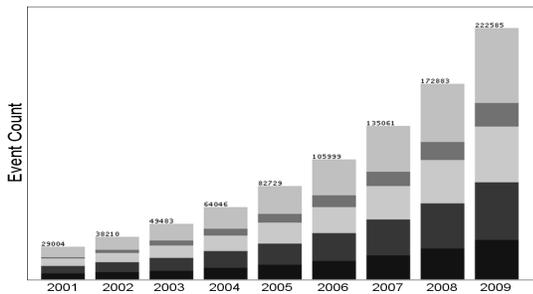


Fig. 4: Yearly growth trend shaded by hacker_capability

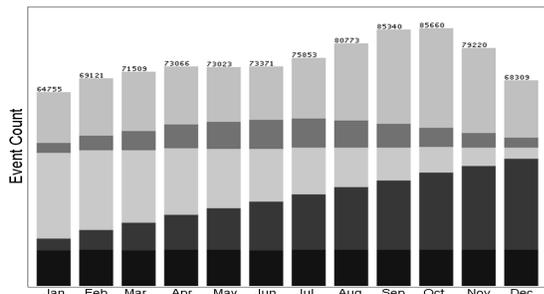


Fig. 5: Monthly patterns shaded by hacker_capability

The monthly distribution pattern of the events for a given hacker capability for all years is shown in Fig. 5. These generative distributions were selected arbitrarily: for hacker_capability=1, 2, 3, 4, and 5, respectively, the monthly distribution is uniform, linearly increasing, linearly decreasing, Gaussian, and multi-Gaussian.

A generative model is used only to generate a dataset. No a priori model is considered when building a TDBN model from a generated dataset.

4.2 Building the TDBN Model

The dataset used to build and evaluate this TDBN model contained a total of 300,000 event records generated from the generative model. Data for the first 8 years (2001 through 2008) was used as the training dataset, whereas data for the last year (2009) was used to test forecasting accuracy. Given the training dataset, the following steps were performed to create a TDBN model:

1. Identify correlations among temporal and non-temporal attributes in the dataset.
2. Determine an appropriate Bayesian network topology based on identified correlations.
3. Apply probability functions where applicable.

We have developed tools to automate much of the process of building a TDBN model from training data. As with many data-mining techniques, some manual review is required to ensure that the resulting model makes sense in the particular domain application.

Step 1. Identify correlations among attributes

The initial step in building a TDBN model is to identify correlations among attributes in the dataset. Our tool applies multiple Bayesian network learning algorithms from the Weka data mining toolkit [10] to discover correlations. There are many reasons to test several learning algorithms. Each algorithm has a bias towards a certain type of network topology; many have parameters (e.g., maximum number of parents) that further bias the learned topology; and some are dependent on the class attribute (the attribute being estimated or forecast). A user may not know which algorithm or algorithm parameters are appropriate for the application domain or dataset. Additionally, there may not be a single class attribute of interest for prediction.

Our tool applied a total of 108 combinations of BN learning algorithms and parameter choices. Table 6 summarizes how often each correlation was found as a percentage of learned networks that contain that correlation. Correlations found in more than 50% of the learned networks (shaded Table 6 cells) are automatically identified as likely correlations. From this set of identified correlations, a user can verify which correlations make sense in the particular domain.

The presence or absence of arcs (dependencies) among nodes is critical information in a Bayesian network because independence simplifies joint probability calculations [6]. The direction of arcs must also be considered to ensure that the network is acyclic and that the causal dependencies implied by the network are generally consistent with causal directions in the domain. Russell and Norvig advise that Bayesian networks based on causal direction lead to more concise and comprehensible models [11]. However, in many cases, arc direction is not critical. This can be observed in Table 6, where dependencies in both directions were found between hacker_capability and hacker_type. The more frequent dependency (i.e., 61%) from hacker_capability to hacker_type

Table 6: Frequency of correlations found – without temporal attributes

Dependency to:	Dependency from:					
	hacker_type	target_vulnerability	hacker_capability	attack_type	success	damage
hacker_type	0 %	0 %	61 %	13 %	0 %	0 %
target_vulnerability	0 %	0 %	15 %	21 %	33 %	0 %
hacker_capability	39 %	10 %	0 %	15 %	12 %	0 %
attack_type	76 %	6 %	85 %	0 %	9 %	5 %
success	8 %	67 %	65 %	83 %	0 %	20 %
damage	7 %	28 %	17 %	89 %	80 %	0 %

was used for the remainder of the steps, but using the reverse dependency produces similar results.

Correlation information is generated in two passes. First, to ensure that important domain-specific causal correlations are not masked by temporal relationships, correlations among non-temporal attributes are identified by applying the tool to the training dataset while ignoring timestamps. The results from this pass are shown above in Table 6.

In a second pass, temporal correlations are identified by applying our tool to the data with the timestamps explicitly represented as temporal attributes. A major challenge for this step is modeling time as temporal attributes suitable for the domain. To help identify potential temporal correlations in the dataset, the tool can translate timestamps into typical temporal attributes such as *year*, *month*, and *day*. Other representations of time that are specific to the domain application (e.g., *dayOfWeek*, *semester*, or *season*) should also be added.

From the dataset now annotated with the typical temporal attributes, temporal correlations were identified and are shown in Table 7. Among the non-temporal attributes, the correlations with frequencies above the threshold of 50% (shaded) are the same as in Table 6, although with different frequencies throughout. Therefore, for this dataset, temporal correlations did not mask any non-temporal correlations enough to exclude them from the model.

In Table 7, the attribute *month* is correlated with *hacker_capability*, which is consistent with the generative model (in Fig. 3) used to create the synthetic dataset. However, direction of the dependency must be reversed from the findings in Table 6. Time is not a dependent variable in a causal sense. Therefore, whenever possible, identified correlations between temporal attributes and non-temporal attributes should be modeled with an arc direction indicating the non-temporal attribute's dependence on the temporal attribute. In general, this makes the resulting networks more semantically correct as long as it does not violate the requirement that the Bayesian network remain acyclic.

Note also that the month attribute is possibly correlated (41%) to the day attribute. Correlations between temporal attributes can be ignored if they are artifacts of the calendar system. For example, investigation of the CPT reveals that month is correlated with day because the only probable values of month for day=31 are the months with 31 days.

Correlations with frequencies between 40% and 50% should not be overlooked in general. For instance, there is a possible correlation between *hacker_capability* and *year*. (In this case, indirect dependencies that existed in the generative model may have been interpreted by the BN learning algorithms as a correlation.) If a correlation is close to the 50% threshold, a user can decide to include it for further analysis and possible future elimination. This approach is illustrated in the remaining steps.

Step 2. Determine an appropriate network topology

Based on the identified correlations from the previous step, a topology for the TDBN is selected by testing candidate topologies and/or by using domain knowledge. We developed a tool to build and test candidate topologies. A user can select among the topologies automatically ranked by this tool or generalize the correlations manually.

With respect to the given dataset, network topologies based on the definite and possible correlations in Table 6 and Table 7 were considered. Specifically, the possible correlation between *year* and *hacker_capability* was explored by testing three network topologies: one without any correlation to *year*, one where *hacker_capability* is dependent on *year*, and one where *year* is dependent on *hacker_capability*. The remaining correlations identified in Step 1 were consistent across all topologies.

For each class attribute, the tool trains each candidate topology on the training dataset and tests the learned model to measure how well it fits the training dataset. Accuracy results for all three network topologies were equivalent in this case, so the most parsimonious network was selected (i.e., the topology without correlations to *year*).

Fig. 6 shows the final TDBN topology. Modifications from the automatic findings in Step 1 include reversing the arc to

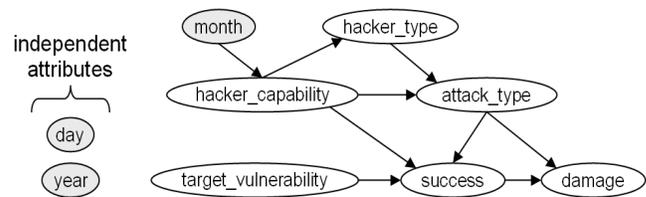


Fig. 6: Revised Bayesian network topology

Table 7: Frequency of correlations found after including temporal attributes

Dependency to:	Dependency from:								
	year	month	day	hacker_type	target_vulnerability	hacker_capability	attack_type	success	damage
year	0 %	22 %	0 %	0 %	0 %	30 %	0 %	0 %	0 %
month	16 %	0 %	41 %	3 %	3 %	77 %	3 %	0 %	0 %
day	5 %	34 %	0 %	5 %	5 %	0 %	0 %	0 %	0 %
hacker_type	0 %	0 %	0 %	0 %	0 %	56 %	20 %	0 %	0 %
target_vulnerability	0 %	0 %	0 %	0 %	0 %	15 %	16 %	33 %	0 %
hacker_capability	45 %	42 %	5 %	44 %	0 %	0 %	40 %	25 %	6 %
attack_type	0 %	3 %	5 %	66 %	1 %	60 %	0 %	3 %	0 %
success	0 %	0 %	3 %	8 %	67 %	75 %	88 %	0 %	19 %
damage	0 %	3 %	5 %	6 %	25 %	6 %	89 %	81 %	0 %

month and changing year to an independent attribute. Using this topology, CPT probabilities were learned from the training dataset for the next step.

Step 3. Insert probability functions

Given the network topology and CPTs determined in the previous step, the next step is to create time-dependent probability functions for attributes that represent time or attributes that are dependent on time. In this case, the year, month, day, and hacker_capability attributes are candidates for translating probability values to time-dependent probability functions.

To help determine applicability of probability functions and model them within CPTs, we developed a tool to apply regression analysis and identify functions that accurately represent the probability values. In general, a user should review the probability functions to ensure that they make sense for the domain.

For the example dataset, our tool found PDFs for the year and month attributes. The tool provided several candidate functions, including polynomial functions and Gaussian functions, that fit the data well. A PDF was not created to represent the probability distribution for the day attribute, which was more appropriately handled as probability values in the CPT because the cardinality of the day attribute is not consistent for all months; i.e., not all months have the same number of days.

Conditional probabilities learned from the training dataset for the hacker_capability attribute were used to create CPFs. Our tool performed regression on the time series of conditional

probabilities from the CPT and identified polynomial functions (parameterized by month) that best fit the conditional probabilities for each hacker_capability, as shown in Table 8 and illustrated in Fig. 7.

The trends from the generative model (shown in Fig. 5) are apparent in the fitted polynomial functions in Fig. 7. Polynomial functions are robust enough to fit many distributions, but caution should be taken when extrapolating far beyond the scope of the training dataset. Domain expertise can be very useful in determining an appropriate probability function type.

5 Model Comparison

As a proof-of-concept, accuracy was evaluated with respect to the generative model, which was taken as ground truth. The accuracy of count predictions produced from the TDBN model was compared to that of other models for a set of joint probability queries. These test queries were generated by randomly selecting attribute values for some randomly selected subset of attributes. Under the intended use of the TDBN model, the selected attribute values would characterize a situation of interest for event prediction.

The comparative results show differences among three Bayesian network models:

- **BN** : Bayesian network; no temporal nodes
- **tBN**: Bayesian network with temporal nodes
- **TDBN**: Bayesian network with temporal nodes and probability trend functions

The network topologies for tBN and TDBN are the same; the only difference is that probability values are represented as functions in the TDBN model and CPTs with discrete entries in the tBN model. The BN topology is similar except that there are no temporal nodes. Note that, since no temporal information is included in the BN model, results using this model reflect averages across all times observed in the training data.

Two types of comparisons were made. The estimation evaluation measures how accurately the learned model reflects event counts within the temporal scope of the training data, and the forecasting evaluation measures how well the learned model can predict event counts in the future.

The models were tested using 4000 joint probability test queries for estimation and 4000 for forecasting. For each query, an estimated count was produced from each model. The scaled deviation from the actual count in the generated test set is calculated as “(estimated-actual)/log(actual+1)”; it is scaled so that estimation differences at large actual counts are comparable to those at smaller actual counts.

Fig. 8 presents results for the estimation evaluation as a combined scatter-plot (to view individual test points) and boxplot (to view concentration of data points in quartile segments) for each of the three models. Fig. 9 presents results for the forecasting evaluation.

Table 8: CPT for P(hacker_capability | month) with CPFs, where m is the month

hacker_capability (hc)				
1	2	3	4	5
CPF _{hc=1} (m)	CPF _{hc=2} (m)	CPF _{hc=3} (m)	CPF _{hc=4} (m)	CPF _{hc=5} (m)

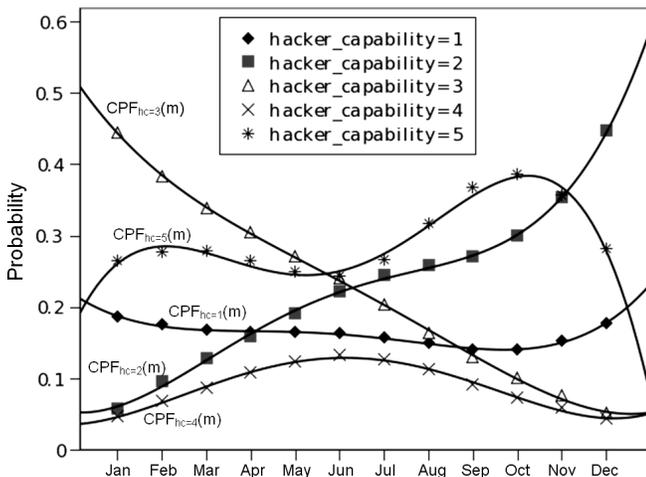


Fig. 7: Monthly distribution for each hacker_capability

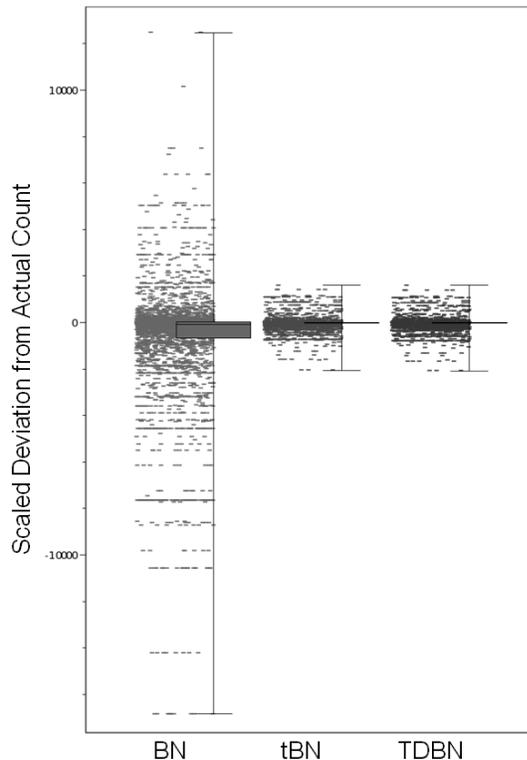


Fig. 8: Estimation error comparison

The results show that by accounting for time in the tBN and TDBN models, both estimation and forecasting accuracy are significantly improved relative to the BN model. The performance of the tBN and TDBN models is comparable for the estimation evaluation; however, the TDBN model shows a significant improvement over tBN for forecasting. For the tBN model, the most temporally proximate probabilities that were available (those associated with year 2008) were used to forecast counts for year 2009. As a result, the tBN model tends to underestimate the event counts compared to the TDBN model, which can extrapolate the temporal trend.

6 Summary

This paper provides a proof-of-concept for Time-dependent Bayesian Networks (TDBNs), a novel approach for incorporating temporal information into BN models. The steps involved in building a TDBN are described, including learning time-dependent probability functions and determining an appropriate network topology.

Incorporating time into any model is a non-trivial task which poses challenging problems, such as extending what is learned in one time period to another time period. In previous Bayesian network models, probabilities are learned for the temporal range of the training dataset and are used without leveraging some of the inherent information about time-dependent patterns and trends. The TDBN approach proposes using CPFs, in addition to leveraging PDFs, to integrate trend information and improve extrapolations of probabilities to future time periods.

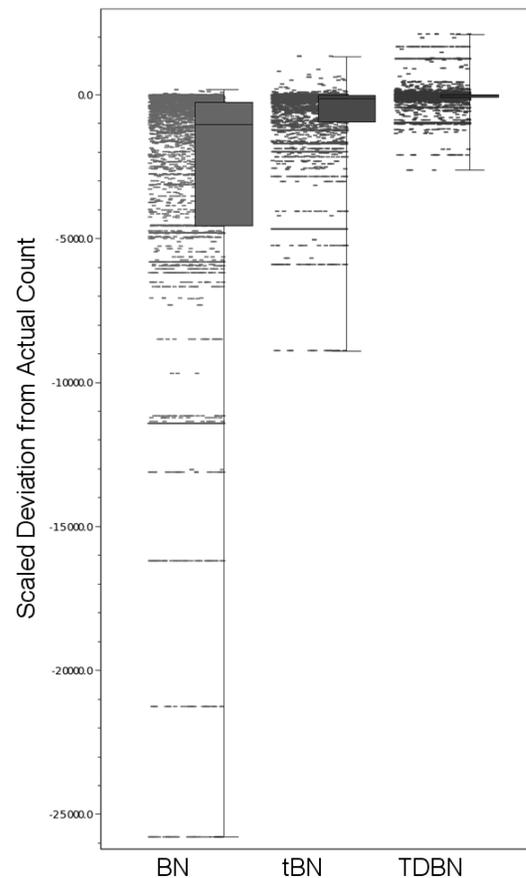


Fig. 9: Forecasting error comparison

References

- [1] D. Heckerman, A Tutorial on Learning with Bayesian Networks, in *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA: MIT Press, 1999.
- [2] V. Mihajlovic and M. Petkovic, Dynamic Bayesian networks: a state of the art. Technical Report TR-CTIT-34, Centre for Telematics and Information Technology, University of Twente, Enschede, The Netherlands, 2001.
- [3] C. Berzuini, Representing time in causal probabilistic networks, presented at the Fifth Annual Conference on Uncertainty in Artificial Intelligence, Amsterdam, The Netherlands, 1990.
- [4] A. Y. Tawfik and E. Neufeld, Temporal Bayesian networks, presented at the International Workshop on Temporal Representation and Reasoning (TIME-94), 1994.
- [5] N. Friedman, K. Murphy, and S. Russell, Learning the structure of dynamic probabilistic networks, presented at the 14th Annual Conference on Uncertainty in AI, Madison, WI, 1998.
- [6] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco: Morgan Kaufman, 1988.
- [7] J. Forbes, T. Huang, K. Kanazawa, and S. Russell, The BAT mobile: towards a Bayesian automated taxi, presented at the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI), Montreal, Quebec, Canada, 1995.
- [8] U. Nodelman, C. R. Shelton, and D. Koller, Learning continuous time Bayesian networks, presented at the Nineteenth Conference on Uncertainty in Artificial Intelligence, Acapulco, Mexico, 2003.
- [9] StatSoft Inc., *Time series analysis, in Statistics: Methods and Applications*. Tulsa, OK: StatSoft Inc., N.D.
- [10] Weka 3: Data Mining Software in Java, Accessed 28 Oct. 2008 at <http://www.cs.waikato.ac.nz/ml/weka/>.
- [11] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, N.J.: Prentice Hall, 1995.

Mining Sequential Episodes from Segmentation of Multivariate Time Series*

Li Wan^{1,2}, Jianxin Liao^{1,2}, and Xiaomin Zhu^{1,2}

¹ State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China 100876

² EBUPT Information Technology Co., Ltd, Beijing, China 100083
 {wanli, liaojianxin, zhuxiaomin}@ebupt.com

Abstract: We argue that frequent patterns in different distributions represent different events and present a two-step algorithm SEM (Sequential Episode Mining) to discover frequent patterns and their distributions in a given multidimensional time series. By segmenting time series in the first step, SEM reduces the dimensions of multivariate time series and the result of segmentation will directly impact the features of the found frequent patterns. We proposed a lattice based prune strategy in the second step, which help SEM to find out maximal frequent patterns efficiently. Experiments on synthetic and real-world data demonstrate our algorithm outperforms existing algorithms. The practical application of SEM shows that it's effective in detecting different types of events.

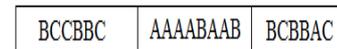
Index Terms: Time series analysis, Sequential Pattern, Episode

I. INTRODUCTION

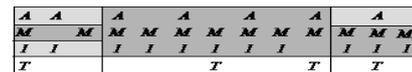
Temporal data sets are widely generated by many real world systems in business, engineering and scientific scenarios. In recent decades, many data mining techniques for analyzing such data streams have been proposed [1,2,3,4,5]. Frequent pattern discovery is one of the challenge problems in temporal data mining [14]. A pattern is a local structure that describes relationships among several variables or data points. In temporal data, there are mainly three categories of frequent temporal patterns, i.e. sequential pattern, episode and partial order pattern. Discovering frequent patterns in a set of time series and a single long time series are two bunch of problems have attracted a lot of attention in temporal data mining [1,15,16]. The central idea of these approaches is to propose fast algorithm to find expressive patterns.

There are many existing literatures work on event prediction in time series based on frequent episode in a time series but they didn't consider the distribution of episodes along the time series [22,23,24,25]. We argue that all the distribution of frequent sequences along a time series is very important for prediction. Because in real world, both "periodical event" and "event burst" occurs. That means some events occurs periodically but some may occur frequently in a special period of time but not frequently all over the time series. Take the time series in Fig.1(a) as an example, if we set minimum

support as 3, sub-sequence A, B, C are all surely frequent episodes. But it's clear that event A doesn't occur frequently in all the scope of the time series. If we split the time series into three pieces as follows, we could say that event A bursts in the second time piece (A is called burst events). In this case, episode-based prediction gets troubles. It will get high probability that event A occurs after A. But this rule is only strong in the second time piece not all over the time series.



(a)



(b)

Fig.1 (a) Event "A" burst in the second piece of time. (b) A multivariate time series is segmented into segments, every segment is subdivided into groups (covered by different colors)

Keeping discovering frequent patterns and their time distribution in mind, we presented an algorithm SEM, which firstly segment time series into segments, then mines out frequent patterns in every segment and the set of segments. In this way, SEM discriminates discovered patterns into sequential, episode and sequential-episode categories (See definition in Section 2). Patterns in different categories have different time distributions. In time series, sequential pattern represents the event appears in a number of the segments, but not frequently occurs in every segment. On opposite, episode pattern represents event occurs frequently in at least one segment, but not appears in so many segments. The event represented by a sequential-episode pattern is the combination of the above two, it frequently occurs in many segments in the segment set. Event which doesn't belong to any frequent pattern is considered as noise.

The contributions of this paper are as follows:

1. We argue that frequent patterns occur in different time distribution along time series represent different events hidden behind temporal data. It's necessary to discriminate different types of frequent patterns in different applications.
2. We presented a two-step algorithm SEM (Sequential Episode Mining) to discovery sequential-episode patterns in a multidimensional time series. In first step, SEM segments time series into segments based on MDL (Minimum Description Length), every segment contains

*This work was jointly supported by: (1) National Science Fund for Distinguished Young Scholars (No. 60525110); (2) National 973 Program (No. 2007CB307100, 2007CB307103); (3) Program for New Century Excellent Talents in University (No. NCET-04-0111); (4) Development Fund Project for Electronic and Information Industry (Mobile Service and Application System Based on 3G).

events occurring in similar periodicity. In second step, SEM utilizes prefix-span framework to find out different kinds of frequent patterns. As we are only interested maximal frequent pattern (See Def. 3) in event prediction, a proposed prune strategy is utilized in the second step.

3. Experimental studies on synthetic and real world data sets demonstrate that SEM is both efficient and scalable and outperforms existing algorithms.

The rest of the paper is organized as follows: Section 2 provides some preliminaries. Section 3 describes the SEM algorithm. Section 4 gives the experiment results. Section 5 gives related works and we conclude in section 6.

II. PRELIMINARIES

Definition 1. (Sequential Pattern) Given a set of time segments and a minimal support γ_S . A sequence occurs once or more than once in a segment contributes 1 to its support. If a sequence's support is larger than γ_S , it's a sequential pattern.

Definition 2. (Episode) Given a time series and a minimal support γ_E . A sequence occurs once in this time series contributes 1 to its support. If a sequence's support is larger than γ_E , it's an episode.

Definition 3. (Maximal Frequent Pattern) A sequence is a frequent pattern (either sequential pattern or episode). If it's not included by any other frequent pattern, it's called maximal frequent pattern.

III. SEM ALGORITHM

A. Time Series Segmentation

A proper segmentation of time series could not only reduce the dimensions of multivariate time series but also help discovering meaningful frequent pattern. Based on MDL (Minimum Description Length) principle, the proposed encoding scheme compresses the events occurring with similar periodicity along time series into the same segments. In our scheme, we encoded the segmentation of time series in Huffman code, the segmentation could be encoded by least bits is the result we want. We suppose that the input time series is represented as a $m \times n$ matrix. The encoding cost of a time series T is calculated by equation (5), where k is the number of segments, n is the length of T . In equation (1), M_i is the model of segment S_i , X_j is the group of events in segment S_i , $p(X_j)$ is the probability of an event occurring in group X_j , $p_{\tau(E)}(X_j)$ is the probability of a length $\tau(E)$ time interval between two successive events whose event type is E occurs in group X_j , $n^{(E, S_i)}$ represents the number of length a time interval could be in group X_j . In equation (3), l and m is the number of group and event types in segment S_i

respectively. Take the time series in Fig.1 (b) as an example, a group of several types of events in the segmentation is considered as one dimension in the remaining steps of SEM algorithm.

$$P(S_i | M_i) = \prod_{j=1}^l \prod_{E \in X_j} \prod_{\tau=1}^{n^{(E, S_i)}} (p(X_j) p_{\tau(E)}(X_j)) \quad (1)$$

$$LD(S_i | M_i) = -\log P(S_i | M_i)^1 \quad (2)$$

$$LM(M_i) = 2l \log m + m \log m \quad (3)$$

$$LS(S_i, M_i) = LD(S_i | M_i) + LM(S_i | M_i) \quad (4)$$

$$TS(S, M) = k \log n + \sum_{i=1}^k LS(S_i, M_i) \quad (5)$$

We utilize a greedy algorithm to segment the time series in a bottom-up fashion. The algorithm starts with segmentation S^1 , in which each data point is a segment of its own. At the l -th iteration of the algorithm, we remove the boundary b in M^l whose removal cause the maximum decrease of $TL(S, M^l)$, after the removal we get M^{l+1} . If there is no removal of the boundary cause the decrease of encoding cost, we output M^l . In each iteration step, we maintain a hash table to store the boundaries and their corresponding contributions to the encoding cost of segmentation after their removals as $H = \langle \langle b_1, \Delta(b_1) \rangle, \langle b_2, \Delta(b_2) \rangle, \dots, \langle b_n, \Delta(b_n) \rangle \rangle$.

$$\Delta(b_j) = L^*(S[b_{j-1}, b_{j+1} - 1]) + \log n - L^*(S[b_{j-1}, b_j - 1]) - \log n - L^*(S[b_j, b_{j+1} - 1]) - \log n \quad (6)$$

$\Delta(b_j) > 0$ means the removal of b_j increases the encoding cost of the segmentation, vice versa. After removing b_j , we only need to update the contributions of b_{j-1} and b_{j+1} . For segmenting a time series with n time points, since there are at most $n-1$ boundary candidates for removal, there are at most $(n-1)$ iterations in the algorithm.

Algorithm: GreedySegment

Input: a single long time series T

Output: a segmentation of T corresponding to minimum encoding cost

1. For every time points b_i with occurrence of events
 2. Calculate the encoding cost of segments $S[b_{i-1}, b_i]$, $S[b_i, b_{i+1}]$, $S[b_{i-1}, b_{i+1}]$ according to equation (4)
 3. Calculate $\Delta(b_i)$ according to equation (6)
 4. Insert $\langle b_i, \Delta(b_i) \rangle$ into H
 5. While H is not empty
 6. Find the minimum $\Delta(b_i)$
 7. If $\Delta(b_i) < 0$
-

¹ According to information theory, if an event occurs in probability p , it could be described by $-\log p$ bits.

8. Remove b_i and update $\Delta(b_{i-1}), \Delta(b_{i+1})$
9. Else if $\Delta(b_i) \geq 0$
10. Break while loop
11. Output H

the prefix of current frequent sequence P_l
 the count of iterates l

- 1: count the support of all events in $S = \{S_1, S_2, \dots, S_n\}$ and get $spt(e_i) = \{S_1(e_i), S_2(e_i), \dots, S_n(e_i)\}$
- 2: if $|spt(e_i)| > \min Spt_S$ or $\exists S_j(e_i) > \min Spt_E$ input $P_l = P_l + e_i$ into lattice L, update L according to proposed prune strategy.

B. Discovering Frequent Patterns from Time Series Segmentations

As we are only interest in maximal frequent patterns (See Def.3: 3), we prune the candidate sub-sequence included by any other frequent patterns. For pruning search space during the mining, we store the found sub-sequences in a lattice, as shown in Fig. 2. Each node represents a sub-sequence in the search space attached its supports in every segment. Each layer consists of: frequent sequence with length l . In iterate l , SEM input: l -length frequent sequences into lattice L, and update the support of $(l-1)$ -length frequent sequences. If any sequence in lattice L is not frequent after the updating, SEM removes it away. For example, as shown in Fig.2, in iterate 1, $\{A\}, \{M\}, \{I\}$ are found out as the frequent sub-sequences. In iterate 2, we first found out all the length 2 frequent sub-sequences whose prefixes are $\{A\}$ (e.g. $\{A,I\}, \{A,M\}$). Then we update the supports of $\{M\}$ and $\{I\}$ just by subtracting the supports of $\{A,M\}$ and $\{A,I\}$ from $\{A\}$'s, $\{M\}$'s and $\{I\}$'s original supports respectively. After that, if any of the three length 1 sequences' support is less than minimum support, we need not to visit any sub-sequence starts with $\{M\}$ or $\{I\}$ in the following iterates and remove the non-frequent sequences.

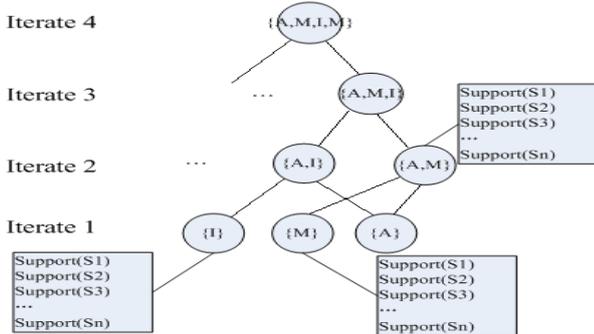


Fig. 2 lattice stores the maximal sequences.

Algorithm SEM is described as follow:

Algorithm SEM
 Input: Original time series T
 Minimum episode support $\min Spt_E$
 Minimum sequential support $\min Spt_S$
 Output: A close set of maximal frequent patterns

- 1: initiate lattice $L = \{\phi\}$, the prefix of current frequent sequence $P_l = \phi$, the count of iterates $l = 1$
- 2: call S=GreedySegment(T)
- 3: call SEM_Lattice(S, P_l, l)
- 4: output all the maximal frequent patterns in lattice L

SEM_Lattice
 Input: the set S of time segments in time series T

For every P_l which are input into L
 if (P_l is in L and $|spt(p_l)| > \min Spt_S$ or $\exists S_j(p_l) > \min Spt_E$)
 $S = S | e_i$ (e_i is the last event in), $l = l + 1$
 call SEM_Lattice(S, P_l, l).

$S_j(e_i) (S_j(p_l))$ represents the time event e_i (sequence p_l) appears in time segment S_j , $|spt(e_i)| (|spt(p_l)|)$ is the number of non-zero element in $spt(e_i) (spt(p_l))$. $\min Spt_S$ and $\min Spt_E$ are minimum frequent sequential support and minimum episode support respectively. $S | e_i$ represents the projection of e_i on time segment S , i.e. $\forall S_j \in S, pos(e_i, S_j)$ is the position in which e_i appears in S_j , then $\exists S'_j \in S | e_i$, $S'_j = \{subSegment(k, S_{j_m}) | k \in pos(e_i, S_{j_m}), S_{j_m} \in S_j\}$, where $subSegment(k, S_{j_m})$ is the sub-segment of S_{j_m} starts from position k to the end. For example $S = \{S_1, S_2\}$ and $S_1 = \{a, b, b, a, c, d\}, S_2 = \{b, c, d, a, c\}$, then $S | a = \{S'_1, S'_2\}$ where $S'_1 = \{\{_, b, b, a, c, d\}, \{_, c, d\}\}, S'_2 = \{\{_, c\}\}$.

As listed in table 1, SEM could discovery three kinds of frequent patterns and noise in a time series by minimum sequential support and minimum episode support.

Table 1 Noise and three types of pattern categorized by \min_spt_s and \min_spt_e

	$> \min_spt_e$	$< \min_spt_e$
$> \min_spt_s$	sequential-episode pattern	sequential pattern
$< \min_spt_s$	Episode	noise

IV. RELATED WORK

Ref. [11] proposes an approach to find annotated sequential pattern in time series. This kind of pattern not only contains the sequential information among events, but also the temporal information between successive events. The authors of [11] also proposed an efficient algorithm to find out the frequent

annotated sequential pattern in a set of itemset sequences. But they don't consider the time distribution of the frequent patterns.

Ref.[12] proposes a complete algorithm (i.e. WinMiner) to find frequent episode pattern in a single long sequence. WinMiner only constrains the maximum time interval between every two successive events in episodes, so the length of sequential pattern is unbound. But WinMiner does not consider the time interval pattern among events in the sequential pattern. There is only order pattern among events in the sequential pattern found by WinMiner.

Ref.[13] propose a greedy algorithm to segment multivariate symbolic time series. It based on MDL. The authors proposed a scheme to describe each segment S_i by a local model M_i and encode each segment based its local model. The optimal function is the bits needed to encode the whole time series segment (i.e. the sum of bits utilized to encode each segment S_i and its local model M_i). But the encoding scheme in [13] loses the time interval information in and among segments.

V. EXPERIMENTS

A. Evaluation Criteria

In this section we report our experiments on real-world data sets and evaluate our algorithm from two aspects. Firstly, we evaluate the quality of time series S 's segmentation produced by an algorithm A , by reporting compression ratio $CR(A)$.

$$CR(A) = \frac{TS(S, M_A)}{TS(S, M_o)}$$

Where M_A is the segmentation produced by algorithm A , $TS(S, M_A)$ is the cost of encoding segmentation M_A calculated by equation (5). M_o is the segmentation which segments every two successive events into a time segment. By definition, compression ratio takes values in $[0,1]$, the smaller the value of $CR(A)$ the better segmentation achieved by algorithm A . We also evaluate the time efficiency of GreedySegment and SEM algorithms.

We used relative support in experiments. For example, a segmentation of given time series is $S = \{S_1, S_2, \dots, S_n\}$, minimum support is $(0.2, 0.3)$, then $\min Spt_S = 0.2 \times n$, $\min Spt_E(S_j) = 0.3 \times |S_j|$, where $|S_j|$ is the length of segment S_j .

We compared our segmentation algorithm GreedySegment with Greedy-Greedy [13] and compare SEM with PrefixSpan + WinMiner (P+W), P+W utilizes PrefixSpan [7] to discover sequential patterns and WinMiner [8] to discover episode patterns.

B. Experiment on Synthetic Data

The datasets: We generate synthetic datasets as follows: we first fix n , the length of the time series, m , the number of different event types that appear in the sequence and k , the number of segments that we artificially "plant" in the generated event sequence. In addition to $\{0\}$ and $\{n + 1\}$ we

select $k-1$ other unique segment boundaries at random from points $\{2, \dots, n\}$. These boundaries define the k segments. Within each segment $S_i = [b_i, b_{i+1})$ we randomly pick the number of groups to be formed. Each such group X_{ij} , is characterized by parameter $p(X_j)$ and $p_{\tau(E_m)}(X_j)$, that corresponds to the probability of occurrence of each event type and the time span between every two successive events in X_{ij} in segment S_i . The values of $p(X_j)$ and $p_{\tau(E_m)}(X_j)$ are normally distributed in $[0, 1]$. Parameter V is used to control the noise level of the generated event sequence. When $V = 0$, for every segment S_i and every X_{ij} in S_i , events of any type $E \in X_{ij}$ are generated with probability sampled from the normal distribution $\mathbb{N}(p(X_j), V)$, the time spans between every successive events are generated with probability sampled from the normal distribution $\mathbb{N}(p_{\tau(E_m)}(X_j), V)$.

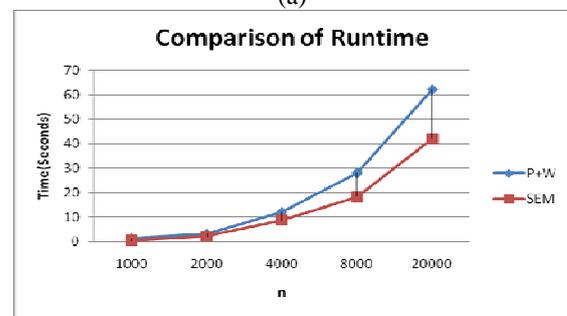
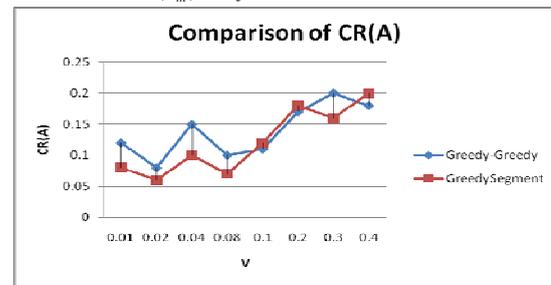


Fig. 3 (a) shows the comparison result of compression ratio, Synthetic dataset: $n=5000, m=10, k=20$, $V=\{0.01, 0.02, 0.04, 0.08, 0.1, 0.2, 0.3, 0.4\}$; (b) shows the comparison result of runtime between SEM and P+W, Synthetic dataset: $n=\{1000, 2000, 4000, 8000, 20000\}, m=10, k=20, V=0.02$

As shown in Fig. 3 (a), with the increment of noise level, either of the two algorithms' compression ratio goes high. GreedySegment outperforms Greedy-Greedy in different noise levels. As shown in Fig. 3 (b), SEM is efficient than P+W, especially when the time series is longer, SEM outperforms P+W, this is because when the time series is longer, there are more frequent but not maximal patterns need to be pruned.

C. Experiments on Sensor Network Data

We experiment our algorithm on data sets generated by sensor networks in smart buildings and demonstrate its utility. The real-world data set is gathered from a sensor network in a smart building by Bosch. There are six sensors in the network, and could throw out events simultaneously. The event types include temperature, motion, illumination, acoustic, humidity and carbon dioxide. The finest time granularity is minute. As listed in table 2, we split five month's data into three data sets.

Table 2. Data sets generated by a sensor networks in smart building

Data sets	Number of records	Month
Dataset1	43359	6
Dataset2	89125	7-8
Dataset3	116993	9-11

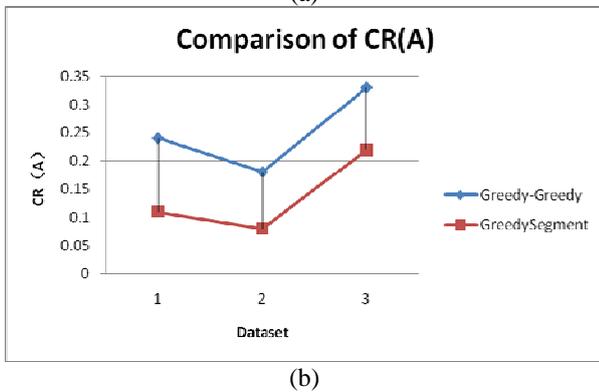
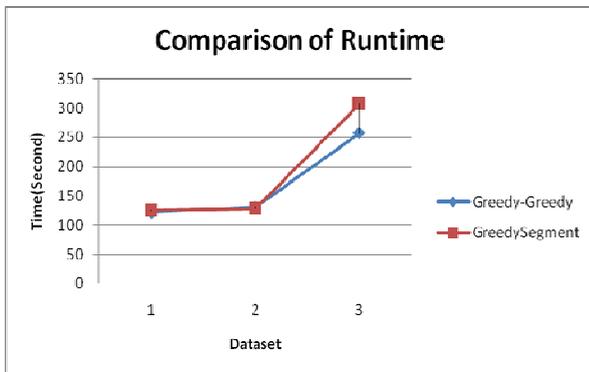


Fig. 4 (a) Comparison of runtime, GreedySegment vs.. Greedy-Greedy[13]. (b) Comparison of CR(A), GreedySegment vs.. Greedy-Greedy[13]

Fig. 4 shows the comparison of runtime and compression ratio between the two segmentation algorithms. Although our GreedySegment cost a little more time to segment given time series, but it achieved much better segmentation quality. The time series in the three datasets are segmented into 232, 306, 447 segments respectively. Fig. 5(a) shows the time efficiency of SEM, it outperforms P+W under every minimum supports setting. Especially when the minimum supports are small, SEM performs much better than P+W, this is because the minimum supports are smaller the more frequent but not maximal pattern should be pruned. We also implemented a prototype system to analysis sensor network generated

temporal data. Fig. 5(b) shows the visualization of patterns and their distributions in dataset1 implemented in our prototype system.

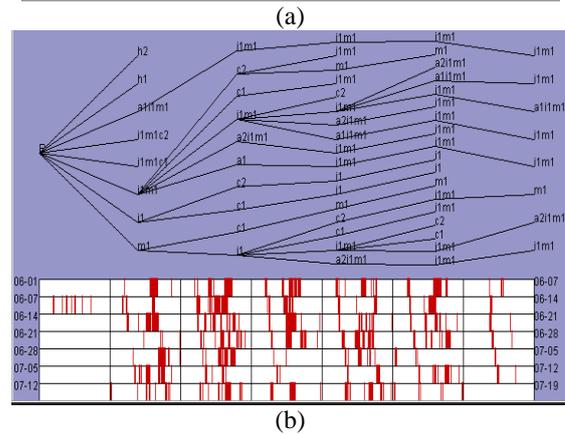
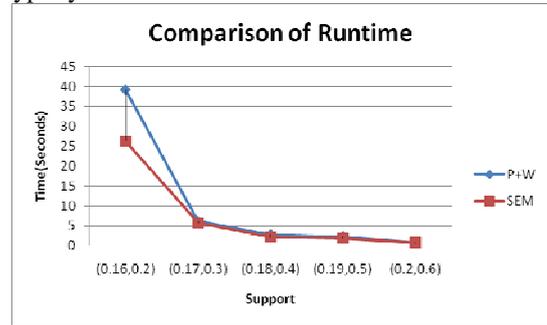


Fig. 5 (a) comparison time efficiency on dataset 2 : SEM vs. P+W, (b) visualization of patterns and their distributions on time series in dataset 1.

In table 3, we give a case study of the analysis on dataset 2 by utilizing SEM. As we can see that when the two minimum supports is high to (0.2,0.6), there still exist two episodes. By looking into the original time series, we find out these two episodes are an acoustic sensor fires event 1 with duration 62 and 21 minutes. By checking with engineers from Bosch, we knew that the acoustic sensor is set artificially at these two special time periods. This case study shows that SEM is effective in detecting burst events.

Table 3. The numbers of different kind of patterns found in dataset 2 by SEM.

Minimum supports	(0.16,0.2)	(0.17,0.3)	(0.18,0.4)	(0.19,0.5)	(0.2,0.6)
Sequential-Episode patterns Num.	32	30	23	11	7
Sequential patterns Num.	18	10	6	2	0
Episodes Num.	122	80	53	12	2

We also evaluate our algorithm on a public dataset, namely "Intel Lab Data" [26]. It's a dataset includes data collected from 54 sensors (labeled from No. 1 to No.54) from February 28th to April 5th, 2004. Fig. 6 shows the location of the 54 sensors. We select the sensor from No. 1 to No. 10 to form a 40 dimensional time series. According to [26], every record includes 4 values fired by a sensor. We just padded 0 values to the timestamps when there is no value fired out by sensors. By

		(0.15,0.2)	(0.18,0.3)	(0.21,0.4)	(0.23,0.5)	(0.5,0.6)
ILData 1	Sequential-Episode patterns Num.	230	103	78	31	4
	Sequential patterns Num.	17	10	9	0	0
	Episodes Num.	442	357	235	72	0
ILData 2	Sequential-Episode patterns Num.	67	43	58	24	4
	Sequential patterns Num.	9	10	6	3	1
	Episodes Num.	42	57	53	30	1

discretizing the time series by algorithm proposed in Ref.[27], we got a 3110400 long 40 dimensional discrete time series and labeled as ILData1. Following the same we pick sensor No.24, 25 and 26 to form a 3110400 long 12 dimensional discrete time series and labeled as ILData2.

Table 4. runtime and CR(A) comparison on datasets ILData1 and ILData2.

	Runtime (minute)		CR(A)	
	GreedySegment	Greedy-Greedy	GreedySegment	Greedy-Greedy
ILData 1	362	307	0.28	0.47 r
ILData 2	48	23	0.22	0.38

As listed in table 4, Greedy-Greedy runs faster than GreedySegment, but GreedySegment beats Greedy-Greedy on CR(A).

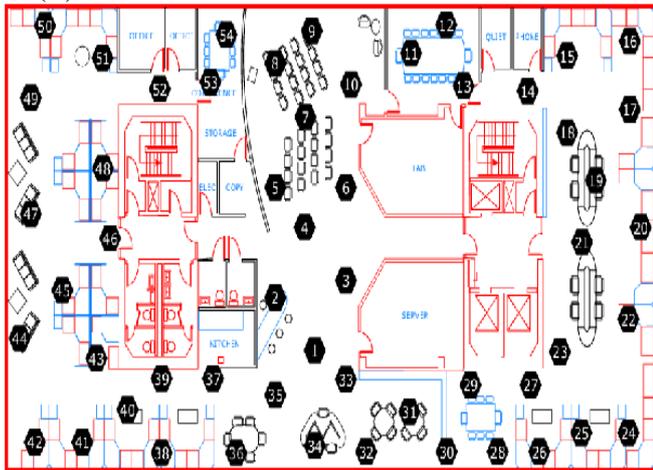


Fig. 6 The location of sensors in the “Intel Lab Data”.

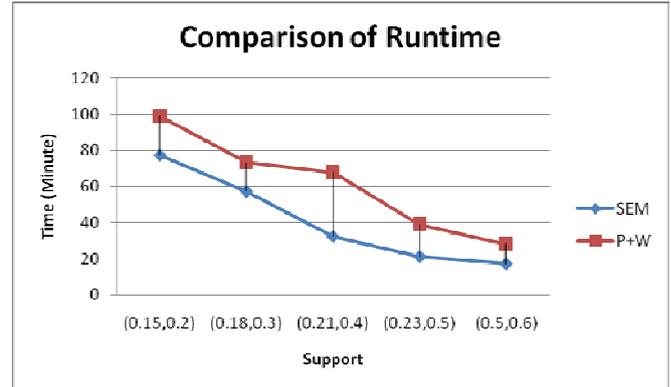


Fig. 7 the comparison of runtimes between SEM and P+W on ILData1

Table 5. The numbers of different kind of patterns found in the “Intel Lab Data” time series segmented by GreedySegment.

Table 6. The numbers of different kind of patterns found in the “Intel Lab Data” time series segmented by Greedy-Greedy.

		(0.15,0.2)	(0.18,0.3)	(0.21,0.4)	(0.23,0.5)	(0.5,0.6)
ILData 1	Sequential-Episode patterns Num.	330	198	118	71	32
	Sequential patterns Num.	107	100	93	56	3
	Episodes Num.	352	251	355	112	10
ILData 2	Sequential-Episode patterns Num.	37	49	76	14	4
	Sequential patterns Num.	12	13	26	10	7
	Episodes Num.	48	42	63	11	9

Because we got similar results on the two datasets generated from “Intel Lab Data”, we only listed the runtime comparison result on ILData1 in Fig. 7. As shown in Fig. 7, SEM is more efficient than P+W and scalable to large datasets. In Table 5, we can find that the numbers of sequential-episode patterns are less than those of episode patterns in ILData1. But the situation is inverted in ILData2, the numbers of sequential-episode patterns are more than or very closed to those of episode patterns. As shown in Fig.6, our selected sensors in ILData1 (sensors from No.1 to No.10) are located in a public area but the selected sensors in ILData2 (sensor No.24 to 26) are located in a comparatively private area. Remind that the sequential-episode patterns represent events happened periodically and episode patterns represent burst events (see section 1), what we found in Table 5 mentioned above make sense that in public area there are less regular events happen which occur periodically and burst events emerge from public behaviors (such as a crowd gather together and has a meeting or just chat separately). But there are more regular events happen in a private area, because one person or several persons’ habits are comparatively easier to

be recorded by sensors. In table 6, the patterns are also found by the frequent pattern discovery sub-algorithm in SEM but the time series segmentations are generated by Greedy-Greedy, but we couldn't see the discrimination of events trends in public and private environments as we did based on the data in Table 5. So we conclude that discriminating frequent patterns into different types is necessary and it's important to properly segment time series in order to get meaningful patterns which represent different events.

VI. CONCLUSION

In this paper, we argue that frequent patterns with different distribution in time series represent different features of the events included in these patterns. We discriminated frequent patterns into three different categories, i.e. sequential-episode, sequential and episode patterns. Different categories of frequent patterns have different distribution in a time series. We proposed SEM algorithm to discover all these three kinds of patterns simultaneously. The segmentation phase is important in SEM. It reduces the dimension of given time series and is the base of mining sequential patterns. We evaluate SEM on synthetic and real-world data. The results on both kinds of data demonstrate that SEM is efficient and scalable to large scale time series. We applied SEM in analyzing time series generated by sensor network in a smart building and SEM is effective to detect burst events. SEM is also applied in analyzing time series in "Intel Lab Data" and find out different events trends between public and private areas which could not be found out by existing algorithms. Future work will focus on temporal-spatial sequence mining.

REFERENCES

- [1] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. S. P. Chen, editors, Proceedings of the 11th International Conference on Data Engineering (ICDE'95), pages 3–14. IEEE Press, 1995.
- [2] Y.L. Chen, S.S. Chen, and P.Y. Hsu, "Mining Hybrid Sequential Patterns and Sequential Rules," *Information Systems*, vol. 27, no. 5, pp. 345-362, 2002.
- [3] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu, "FreeSpan: Requent Pattern-Projected Sequential Pattern Mining," *Proc. 2000 Int'l Conf. Knowledge Discovery and Data Mining (KDD '00)*, pp. 355-359, Aug. 2000.
- [4] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu, "Mining Access Pattern Efficiently from Web Logs," *Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining*, pp. 396-407, 2000.
- [5] J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," *Proc. 17th Int'l Conf. Data Eng.*, pp. 215-224, 2001.
- [6] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal, "MultiDimensional Sequential Pattern Mining," *Proc. 2001 Int'l Conf. Information and Knowledge Management (CIKM '01)*, Nov. 2001.
- [7] R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," *Proc. Fifth Int'l Conf. Extending Database Technology (EDBT '96)*, pp. 3-17, Mar. 1996.
- [8] C.C. Yu, "Algorithms for Mining Sequential Patterns from Multi-Dimensional Sequence Data," MSc thesis, Information Management, Nat'l Central Univ., Taiwan, 2002.
- [9] M.J. Zaki, "SPADE: An Efficient Algorithm for Mining Frequent Sequences," *Proc. Machine Learning J.*, special issue on unsupervised learning, vol. 42, pp. 31-60, Jan./Feb. 2001.
- [10] C.-C. Yu and Y.-L. Chen. Mining sequential patterns from multidimensional sequence data. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):136–140, 2005.
- [11] Fosca Giannotti Mirco Nanni Dino Pedreschi Efficient Mining of Temporally Annotated Sequences. *Proceeding of SDM 2006*
- [12] N. M'eger and C. Rigotti. Constraint-based mining of episode rules and optimal window sizes. In J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, editors, Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'04), pages 313–324. Springer, 2004.
- [13] Jerry Kiernan, Evimaria Terzi Constructing Comprehensive Summaries of Large Event Sequences *Proceeding of KDD 2008*
- [14] Han J, Kamber M 2001 *Data mining: Concepts and techniques* (San Francisco, CA: Morgan Kauffman)
- [15] H. Mannila and H. Toivonen. Discovering generalized episodes using minimal occurrences. In E. Simoudis, J. Han, and U. M. Fayyad, editors, Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96), pages 146–151. AAAI Press, 1996.
- [16] H. Mannila, H. Toivonen, and I. Verkamo. Discovery of frequent episodes in event sequences. In U. M. Fayyad and R. Uthurusamy, editors, Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining (KDD'96), pages 210–215. AAAI Press, 1995.
- [17] J.S. Liu, A.F. Neuwald, and C.E. Lawrence, "Markovian Structures in Biological Sequence Alignments," *J. Am. Statistics Assoc.*, vol. 94, pp. 1-15, 1999.
- [18] D. Chudova and P. Smyth, "Pattern Discovery in Sequences under a Markovian Assumption," *Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, July 2002.
- [19] D.L. Wang and B. Yuwono, "Anticipation-Based Temporal Pattern Generation," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 25, no. 4, pp. 615-628, 1995.
- [20] F. Korkmazskiy, B.H. Juang, and F. Soong, "Generalized Mixture of HMMs for Continuous Speech Recognition," *Proc. 1997 IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP-97)*, vol. 2, pp. 1443-1446, Apr. 1997.
- [21] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic, "Discovering Clusters in Motion Time Series Data," *Proc. 2003 IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. I-375-I-381, June 2003.
- [22] S. Laxman, P. S. Sastry, and K. P. Unnikrishnan. Discovering frequent episodes and learning Hidden Markov Models: A formal connection. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1505–1517, Nov. 2005.
- [23] R. Vilalta and S. Ma. Predicting rare events in temporal domains. In Proceedings of the 2002 IEEE International Conference on Data Mining, ICDM 2002, pages 474–481, Maebashi City, Japan, Dec. 9–12 2002.
- [24] G. M. Weiss and H. Hirsh. Learning to predict rare events in event sequences. In Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD 98), pages 359–363, New York City, NY, USA, Aug. 27–31 1998.
- [25] S. Laxman, Stream Prediction Using A Generative Model Based On Frequent Episodes In Event Sequences *Proceeding of KDD 2008*
- [26] db.csail.mit.edu/labdata/labdata.html.
- [27] Jessica Lin, Eamonn Keogh, Stefano Lonardi, Bill Chiu. A Symbolic Representation of Time Series, with Implications for Streaming Algorithms, In *Proceeding of DMKD*, 2003

An Algorithm for Efficiently Clustering High-dimensional Data streams

Guojun Mao, Xialing Yang

Computer College, Beijing University of Technology, Beijing, P. R. China

Abstract—Clustering high-dimensional data streams is an important problem in the field of mining data streams. Based on analyzing *Cell-Tree*, a popular algorithm to cluster high-dimensional data streams, this paper presents a new data structure called *List-Tree* in main memory which tries to avoid problems existing in *Cell-Tree* and stores summary information from data streams. On the base of *List-Tree*, with the techniques of damped window, a mining algorithm called *LTC* is given in this paper to efficiently cluster high-dimensional data streams. Experimental results show that the proposed method has better performances than *Cell-Tree* in terms of time and space efficiency.

Keywords: data mining; high-dimensional data streams; clustering.

I. INTRODUCTION

With data streams drawing more and more attentions, knowledge discovery in data streams become important study subject. How to cluster a data stream has been discussed widely [1-4]. To code with the potential high dimension of data, researchers proposed subspace clustering algorithms [5-7]. The structure of *Cell-Tree* is proposed for clustering high-dimensional data streams, and typical subspace clustering algorithms based on *Cell-Tree* are designed [8]. Using a tree-shape memory model, each new element from data streams is related with some grid cells, and these cells can be managed by a *Cell-Tree* layer by layer. Therefore, *Cell-Tree* provided a good framework for processing multi-dimensional data streams. However, some potential problems in *Cell-Tree* would be analyzed as follows:

(1) A large number of empty grid cells could exist, which will result in increasing merging-cell frequency and decreasing processing efficiency.

(2) With increasing the number of dimensions in data streams, update time in grid cells will obviously increase, and so how to depress these costs will be a noteworthy problem.

This paper gives a new data structure called *List-Tree* and designs a new algorithm called *LTC* (List-Tree Clustering) for clustering high-dimensional data streams, and their characters or performance are discussed through a series of definitions and experiments.

II. LIST-TREE : A NEW DATA STRUCTURE FOR PROCESSING HIGH-DIMENSIONAL DATA STREAMS

Differing from traditional static data, data streams are dynamic, that is, every element of a data stream is related with a timestamp. So List-Trees will relate these timestamps to the multi-dimension data in a streaming form.

Definition 1 (n -dimension data stream). Given a n -dimension data space $N = N_1 \times \dots \times N_n$, a n -dimension data stream D on N is denoted as follows:

$$D = \{e_{i_1}, \dots, e_{i_j}, \dots\} \quad (1 \leq i < j < +\infty, 0 \leq t_i < t_j < +\infty)$$

Where $e_{i_j} \in N$ is an element of D on time position t_{i_j} , and has n dimension value, denoted as:

$$e_{i_j} = \langle e_{i_j}^1, e_{i_j}^2, \dots, e_{i_j}^k, \dots, e_{i_j}^n \rangle \quad (1 \leq k \leq n).$$

For an n -dimension data stream D , a current time t_c will be related to a definite data window, called current window D_{t_c} . If e_{t_x} ($t_x \leq t_c$) is the newest element coming from D , then the current window D_{t_c} is $D_{t_c} = \{e_{t_1}, e_{t_2}, \dots, e_{t_x}\} \quad (1 \leq i < j \leq x, t_i < t_j)$.

Definition 2 (List-Tree). List-Tree is a super-tree and satisfies the following conditions:

- (1) Root node is empty and have only one child node.
- (2) Every non-root node is a link list called *Cell-Array-List* related with a certain dimension, and is composed of a series of *Cell-Arrays*. A Cell-Array is an array of *Cells*. A Cell is a place to record the related information of an element from data streams.

(3) The relation between different dimensions of cells is linked through the pointer from a Cell on a dimension to a Cell-Array-List on the next dimension.

Fig. 1 gives an example of List-Tree structure sketch.

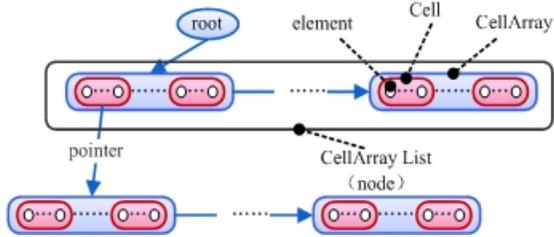


Fig. 1 An example of List-Tree structure

Definition 3 (Cell). Given a current window D_{t_c} of n -dimension data streams on $N = N_1 \times \dots \times N_n$, Cell is not empty and is denoted as $C(t, d, I, c, cp)$, where t is the generation time of the newest element in Cell, d is the dimension of Cell, $I = [l, u]$ represents value range related with this merge on dimension d , c is the quantity of elements related to this Cell while $C.c / |D_{t_c}|$ is the support of C , cp denotes a pointer which is empty or else points at a Cell-Array-List called the child node of Cell.

A Cell is special if it satisfies certain parameter conditions. Special Cells are shown in Fig. 2.

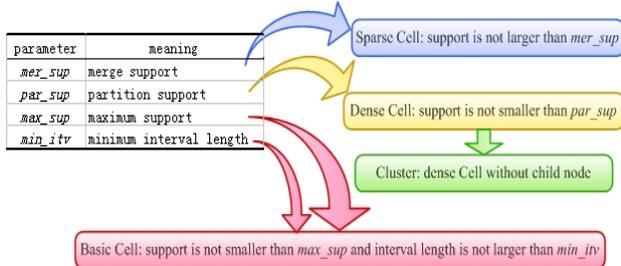


Fig. 2 Important parameters and special Cells

Definition 4 (Cell-Array). A Cell-Array consists of an array and a pointer and is denoted as follows:

$$CA(A[1..m], rp),$$

Therein, array element $A[i](i = 1, 2, \dots, m)$ represents the statistics of a Cell (suppose there are at most m Cells in a Cell-Array); pointer rp is empty or else point at another Cell-Array. When $1 \leq i < j \leq m$ is true, it must be guaranteed that if $A[j]$ is nonempty then $A[i]$ is nonempty, as well as the following expressions are true:

$$A[i] \neq A[j], A[i].d = A[j].d, A[i].I.u < A[j].I.l$$

Definition 5 (Cell-Array-List). A Cell-Array-List is a single linked list consisting of Cell-Arrays and contains at least one Cell-Array. If there are two Cell-Arrays in a Cell-Array-List, and let the former be CA_p and the latter be CA_q , then the following conditions must be satisfied:

- (1) $CA_p.A[i].d = CA_q.A[j].d (i, j = 1, 2, \dots, m)$.
- (2) $CA_p.A[i].I.u < CA_q.A[j].I.l$.
- (3) $CA_p.rp$ points at CA_q .

III. LTC ALGORITHM

Based on structure *List-Tree*, a clustering model is constructed and maintained in algorithm *LTC* to process continuous arriving high-dimensional data from data streams. Owing to clustering belongs to an unsupervised learn, the construction of the memory model comes with the arrival of elements from data streams. The pseudocode of *LTC* is shown in Fig. 3.

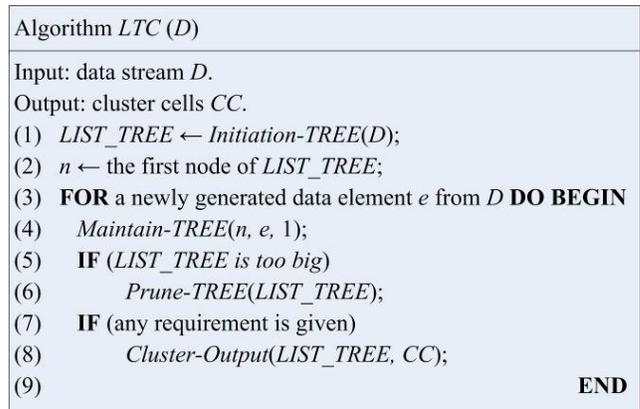


Fig. 3 Algorithm LTC

There are three main phases in *LTC*:

(1) Model initialization phase. Shown as (1)-(2) in Fig. 3, *Initiation-TREE* mainly constructs the initial state of model by setting the child node of root.

(2) Model maintenance phase. This phase corresponds to (4)-(6) in Fig. 3 with a kernel process *Maintain-TREE*. It updates List-Tree by disposing continuous arriving elements from data streams. In addition, the process of *Prune-TREE* is for clearing away out-of-time Cell.

(3) Clustering output phase. Shown as (7)-(8) in Fig. 3, this phase is called periodically or at the request of users because it is impossible to output the clustering results on the arrival of every element.

Maintain-TREE is a recursive process which is described in Fig. 4.

Aiming at a layer of model, this process is called to cope with data streams elements on this layer. There are three phases in this process on each layer:

```

Procedure Maintain-TREE( $n, e, k$ )
(1)  $curCell \leftarrow Fix-CurrentCell(n, e)$ ;
(2) add  $e$  into  $curCell$ ;
(3)  $Decay(D, curCell)$ ;
(4) IF ( $curCell.c/|D| \geq max\_sup$  and  $curCell.l \leq min\_itv$ )
(5)   IF ( $curCell$  has no child and  $k < dim\_cnt$ ) BEGIN
(6)     generate a new node as the child of  $curCell$ ;
(7)   RETURN;
(8)   END
(9) IF ( $curCell.c/|D| \geq par\_sup$  and  $curCell$  has no child) BEGIN
(10)   $Partition-Cell(curCell)$ ;
(11) RETURN;
(12)  END
(13) IF ( $curCell.c/|D| \leq mer\_sup$ ) BEGIN
(14)   $Merge-Cell(curCell)$ ;
(15) RETURN;
(16)  END
(17) IF ( $curCell$  has a child) BEGIN
(18)   $n \leftarrow$  the child of  $curCell$ ;
(19)   $Maintain-TREE(n, e, k+1)$ ;
(20) END

```

Fig. 4 The maintenance of List-Tree

(1) Placement phase. It is shown as (1)-(2) in Fig. 4. On the current layer k , make sure a Cell best fitting for current element e . Supposing this Cell should be $curCell$, add e into $curCell$. The main process of finding $curCell$ will be traversing all Cells in the current node n through Cell-Array-List to get the most appropriate one. The appropriate Cell means that the value of e on the dimension correlative to n is closest to this Cell (expressed as C), that is $|e^{C.d} - C.l|$ or $|C.l.u - e^{C.d}|$ gets the minimum. If the current Cell is a “basic Cell” and its interval length becomes larger than min_itv after the addition of e , then it is necessary to construct a new Cell to restore the value of e on the corresponding dimension.

(2) Decaying phase. It is shown as (3) in Fig. 4. Based on current system time, update $|D|$, $curCell.c$ and $curCell.t$. Given current time t_u and the latest time of updating t_v , setting the decay rate $1 - \tau$ and the data quantity of $curCell$ before updating $curCell.c^*$, the update method is as follows:

$$\begin{cases} |D_{t_u}| = |D_{t_v}| \times \tau^{t_u - t_v} + 1 \\ curCell.c = curCell.c^* \times \tau^{t_u - curCell.t} + 1 \\ curCell.t = t_u \end{cases}$$

(3) Adjustment phase. This phase is correspond to (4)-(16) in Fig. 4. After adding e into $curCell$, there are three process methods to adjust the structure: (a) Generate a child node, as (4)-(8); (b) Partition dense Cells, as (9)-(12); (c) Merge sparse Cells, as (13)-(16). What method is chosen rests with

the type of $curCell$ as shown in Fig. 2.

During the phase of adjustment, there are three adjusting methods corresponding to three kinds of Cells respectively:

(1) Generate a child node. To the basic Cell without child nodes, calculate the standard deviation of its elements on each dimension except that one correlative to this Cell and the ancestry of its node. According to the dimension with the minimum standard deviation, construct a new Cell expressed as C' . Based on C' , construct a Cell-Array denoted as CA' , that is $CA'.A[1] = C'$. According to CA' , construct a Cell-Array-List CAL' as the child node of $curCell$.

(2) Partition dense Cell. To the dense Cell without child nodes, based on partitioning parameter par_cnt and interval length of the dense Cell, partition the Cell into par_cnt Cells. Delete the Cell without elements in it.

(3) Merge sparse Cell. To the sparse Cell, try to merge its adjoining sparse Cells into it. If the Cell after merger is not dense, then delete the offspring nodes of the Cells involved in merger to finish merging. Otherwise cancel the merger.

If adjusting is no long necessary, then process the current element on the next layer, as (17)-(20) in Fig. 4.

IV. EXPERIMENTAL COMPARE AND ANALYSIS

In the following experiments, the same data stream is adopted for algorithm *LTC* and *Cell-Tree*. Elements from the data stream are generated at random. Supposing there are at most 20 Cells in each Cell-Array. Set values of parameters as follows: $\tau = 0.999999$, $max_sup = 0.01$, $par_sup = 0.008$, $mer_sup = 0.001$, $min_itv = 4$, $par_cnt = 4$. Software and hardware environments are as follows: Java, Eclipse 5.0, Microsoft XP 2003, 5.6GHz CPU and 1G memory.

Experiment 1. Compare of the average processing time of *LTC* and *Cell-Tree* on 3000 elements from the data stream when the data dimension ranges from 20 to 50. The result of

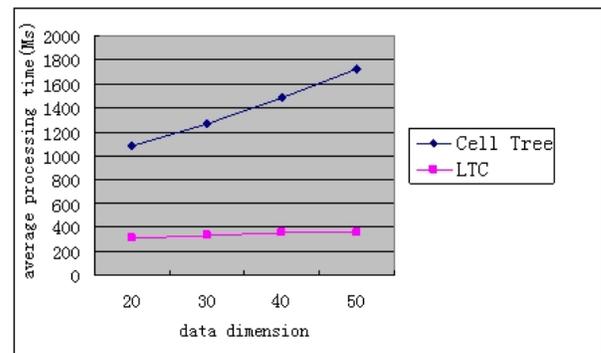


Fig. 5 The average processing time the compare is shown in Fig. 5.

Seen from Fig. 5, no matter which data dimension is chosen, the average processing time of *LTC* is shorter than that of *Cell-Tree* with the same data size. In addition, with the increase of data dimension, the processing time of *LTC* takes on slight increase.

Experiment 2. Compare the average memory usage of *LTC* and *Cell-Tree* on 3000 elements from the data stream when the data dimension ranges from 20 to 50. The result of

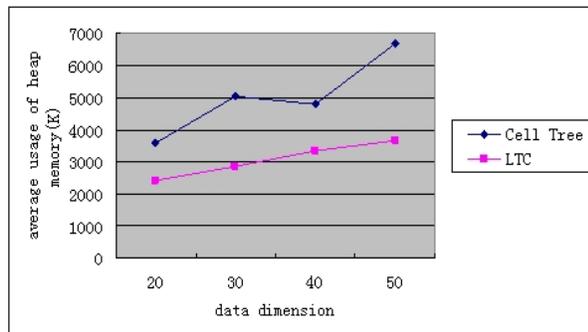


Fig. 6 The average usage of heap memory the experiment is shown in Fig. 6.

It can be found in Fig. 6, on each data dimension, the occupied heap memory size of *LTC* is smaller than that of *Cell-Tree*. When the data dimension varies, the fluctuation of memory usage of *LTC* is gentle.

Experiment 3. Compare the count of clusters generated from *LTC* and *Cell-Tree* on 3000 elements from the data stream when the data dimension ranges from 20 to 50. The result of the experiment is shown in Fig. 7.

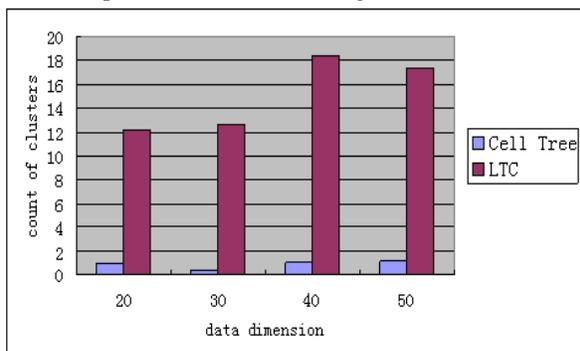


Fig. 7 The number of clusters

It is known from Fig. 7 that no matter which dimension is considered the clusters from *LTC* are more than that from *Cell-Tree*. It can be inferred that *LTC* can discover more information of data difference.

V. CONCLUSION

List-Tree is a better memory structure for mining high-dimension data streams, and can effectively avoid potential empty Cells, so that the space efficiency of memory is improved. Based on this data structure, *LTC* adopts absolute time to measure the decayed size of data streams and prunes the memory model, which decreases the calculation time of decaying data streams and efficiently controls the size of memory model. The experiments show that on the same condition *LTC* processes data streams more efficiently than *Cell-Tree*.

REFERENCES

- [1] B. Babcock, S. Babu, M. Datar, R. Motawani. "Models and issues in data stream systems," in *Proc. 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. Madison, ACM Press, 2002: 1-16.
- [2] S. Muthukrishnan. "Data streams: algorithms and applications," in *Society for Industrial and Applied Mathematics*, Philadelphia, 2003.
- [3] C. Q. Jin, W. N. Qian. "Analysis and management of streaming data: a survey," *Journal of Software*. 2004, 15(8): 1172-1181.
- [4] M. M. Gaber, A. Zaslavsky. "Mining data streams: a review," *ACM SIGMOD Record*, 2005, 34(2): 18-26.
- [5] A. Heburg. "Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering," in *Proc. of the 25th International Conference on Very Large Data Bases*, Morgan Kaufmann, 1999: 506-517.
- [6] C. C. Aggarwal. "Fast algorithms for projected clustering," *ACM SIGMOD Record*, Volume 28, Issue 2, 1999: 61-72.
- [7] C. Baumgartner, C. Plant. "Subspace selection for clustering high-dimensional data," in *Proc. of the 4th IEEE International Conference on Data Mining*, Washington, DC, USA, 2004: 11-18.
- [8] N. H. Park. "Cell-trees: an adaptive synopsis structure for clustering multi-dimensional on-line data streams," *Data & Knowledge Engineering*, 2007, 63(2): 528-549.

Forecasting Seasonal Time Series with Multilayer Perceptrons – an empirical Evaluation of Input Vector Specifications for deterministic Seasonality

Sven F. Crone, Nikolaos Kourentzes

Abstract - Research in forecasting with Neural Networks (NN) has provided contradictory evidence on their ability to model seasonal time series. Several empirical studies have concluded that time series should be deseasonalised prior to modelling, with contradictory evidence pointing to adequate selection of input vectors. However, the nature of seasonality itself has not been considered: econometric theory suggests that deterministic seasonality and stochastic seasonality need to be modelled differently, with only the latter requiring deseasonalisation. As prior research has failed to take the conditions of the underlying seasonality into consideration, this study explores how deterministic seasonality should be best modelled with NN to achieve accurate and robust forecasts. We consider different forms of modelling seasonality as autoregressive lags, through different encodings of explanatory variables and deseasonalisation. We evaluate the results regarding empirical accuracy and the parsimony of the input vector in order to limit the degrees of freedom, develop robust models and simplify the training of NN. Our findings are consistent with econometric literature, i.e. that no deseasonalisation is required for deterministic seasonality, and contributes to current research in NNs by identifying a more parsimonious coding of seasonality based on seasonal indices.

I. INTRODUCTION

NEURAL networks (NN) are recognised as a potent forecasting tool with research and applications in a wide variety of disciplines [1, 2]. Given their theoretical properties, NNs are universal approximators, permitting them to approximate any measurable function to any desired degree of accuracy (and to generalise on unseen data as well) [3], a desirable property in forecasting. NN have proven their capabilities to forecast linear as well as nonlinear time series of synthetic and empirical data [4-6], and across a wide range of time series frequencies including monthly, weekly, daily data with their distinct data properties [7]. However, they are prone to criticism due to the lack of a sound methodology to determine valid and reliable models in empirical evaluations [8-10]. For instance, no consensus exists on how to select a relevant set of input variables and lags for NNs given the different time series components and properties of the dataset [1]; a recent literature survey identified 72% (out of 95) published papers model NNs based purely on invalid and unreliable trial and error approaches. This has a significant impact on the

consistency of NN performance and also hinders our understanding of how to model them [11]. Consequently it remains an important task to empirically evaluate competing methodologies for NN modelling with scientific rigor, in order to derive insight on best practices.

One open research questions currently under discussion [1, 12] is to determine the most efficient and effective modelling approach for NNs on seasonal time series, with the related set of questions whether seasonality should be modelled directly or should be removed by deseasonalising the time series first. Hill et al. [6] showed that NNs using deseasonalised time series from the M1-competition outperformed statistical benchmark models, promising significant improvements in NN performance. Nelson et al. [13] (from the same research group) verified the effect of deseasonalisation using M1-data. They repeated the experiments of Hill et al. without deseasonalising, and provided further evidence that the forecasting performance of NN significantly deteriorate on seasonal data, therefore concluding that deseasonalisation is a necessary step in NN modelling. They postulate that deseasonalising time series allows NNs to focus on learning the trend and the cyclical components; to learn seasonality at the same time would require larger networks, resulting in a larger input vector, which may cause detrimental overfitting. Zhang and Qi [14] reach the same conclusion that deseasonalising helps. However, their research suggest that deseasonalising time series reduces long and dynamic autocorrelation structures that would make the choice of the input vector for NNs more difficult, thus leading to smaller, more parsimonious and hence robust models. Curry [12] examines the ability of NN to model seasonality from a theoretical perspective using data with different properties. His research suggests that NN require adequately long input vectors in order to capture the seasonal effects; ill selected and underspecified input vectors on the other hand can impair the ability of a NN to capture and forecast seasonality, implying that the results by Zhang and Qi can potentially hide input misspecification errors.

However, none of the preceding research studies on NNs distinguishes between the different forms of seasonality, and hence the conditions under which one approach outperforms another. Reflecting upon the statistical literature, both deterministic seasonality and stochastic seasonality (seasonal unit roots) theoretically require different modelling approaches [15-17], which has been ignored in NN literature and the debate on how to model seasonality to date.

Sven F. Crone and Nikolaos Kourentzes (corresponding author) are with the department of Management Science at Lancaster University Management School, Lancaster, LA1 4YX, UK. (phone: +44.1524.5-92991; fax: +44.1524.844885; e-mail: {n.kourentzes; s.crone} @lancaster.ac.uk).

In this analysis we will argue that an adequate distinction implies a different modelling procedure – both from a theoretical perspective but also supported by the empirical evidence. Modelling deterministic seasonality is impaired by prior deseasonalisation of the time series, so that different modelling practises should be considered. To provide evidence we conduct an empirical evaluation of competing approaches to model seasonality using univariate (autoregressive) time series approaches, the use of external variables and differencing on simulated time series. We find that using a set of dummy variables can significantly improve forecasting accuracy over the current NN modelling ‘best practise’ suggested by prior research, where seasonality is either removed or modelled using seasonal lags of the time series. Finally, we propose a parsimonious coding based on seasonal indices, which outperforms other candidate models in accuracy while keeping the degrees of freedom to a minimum in order to construct parsimonious NN architectures.

The paper is organised as follows: section II discusses the different types of seasonality from a theoretical perspective. Section III introduces the methods that will be used to model deterministic seasonality, followed by information on the time series and the experimental design in section IV. Section V discusses the results comparing the different approaches to modelling deterministic seasonality, followed by conclusions and further research objectives in section VI.

II. FORECASTING SEASONAL TIME SERIES

A. Deterministic and Stochastic Seasonality

Seasonality is generally defined as a reoccurring time series pattern within the calendar year, where its structure is defined by the frequency of the times series. A time series is said to have deterministic seasonality when its unconditional mean varies with the season and can be represented using seasonal dummy variables,

$$y_t = \mu + \sum_{s=1}^S m_s \delta_{st} + z_t, \quad (1)$$

where y_t is the value of the time series at time t , μ is the level of the time series, m_s is the seasonal level shift due to the deterministic seasonality for season s , δ_{st} is the seasonal dummy variable for season s at time t , z_t is a weak stationary stochastic process with zero mean and S is the length of the seasonality. Furthermore, the level of the time series μ can be generalised to include trend.

In contrast to deterministic seasonality, seasonality can also be the result of an integrated autoregressive moving average (ARIMA) process,

$$\phi(L)\Delta_S y_t = \gamma + \theta(L)\epsilon_t, \quad (3)$$

resulting in a stochastic seasonality (or seasonal unit root), where L is the lag operator, Δ_S is the seasonal difference operator, ϕ and θ are the coefficients of the autoregressive and moving average process respectively, γ is a drift, and

$\epsilon_t \sim$ i.i.d. with zero mean and standard deviation σ^2 . Of course this can be expressed as a SARIMA model. The variance of y_t under the case of deterministic seasonality is constant over t and the seasonal period s , which is not true here. This stochastic seasonal process can be viewed as a seasonal unit root process, i.e. for each s there is a unit root, which in turn requires seasonal differencing. More details about the seasonal unit root process can be found in [15-17].

B. Modelling NNs for Deterministic Seasonality

Seasonal information can be included in NNs in a variety of ways, including various forms of explanatory variables. Note that deterministic seasonality as in (1) is defined as a series of seasonal level shifts m_s , which describe the seasonal profile and are constant across time, i.e. $m_s = m_{st}$. Also note that as deterministic seasonality is defined in (1) the $\sum m_s = 0$ over a full season. This implies that with the appropriate transformations of μ and m_s a set of $S-1$ or S binary seasonal dummy variables can be used to code the seasonality. Furthermore, due to z_t each value of the time series deviates over its respective seasonal mean with a constant variance over both s and t , which means that the deterministic seasonal process forces the observations to remain close to their underlying mean [17]. Modelling (1) with S seasonal dummies and $\mu \neq 0$ using a linear model (e.g linear regression or a linear NN) introduces multicollinearity, hence $S-1$ dummies must be used [18]. For nonlinear methods such as NNs with only linear transfer functions and $H > 1$ multicollinearity can exist even for $S-1$ dummies, since the dummies provide input into several hidden nodes. This hinders inference from a NN, but does not necessarily harm its predictive power [1, 18]. As this also holds for the case of nonlinear transfer functions, both modelling using $S-1$ or S seasonal dummies may be adequate for NN models.

An alternative way to code deterministic seasonality as in (1) is through its trigonometric representation:

$$y_t = \mu + \sum_{k=1}^{S/2} \left[\alpha_k \cos\left(\frac{2\pi kt}{S}\right) + \beta_k \sin\left(\frac{2\pi kt}{S}\right) \right] + z_t, \quad (2)$$

where α_k and β_k create linear combinations of $S/2$ sines and cosines of different frequencies following the ideas of spectral analysis of seasonality. Equations (1) and (2) have μ and z_t expressed as separate components in both cases, allowing separate modelling of seasonality and the remaining time series components [17]. Note that if less than $S/2$ linear combinations of sines and cosines are used the representation of seasonality is imperfect and it is approximated with some error, the size of which is related to the number of combinations used. Deterministic seasonality as expressed in (2) can equally be modelled in NNs using explanatory variables. Note that an alternative is to approximate (2) using less frequencies by increasing the number of hidden nodes H in a network [3].

Following the same procedure, based on the increase of H , NN can approximate seasonal patterns by combining seasonal dummies in a single integer dummy defined as $\delta = [1, 2, \dots, S]$ [19], which is neither permissible nor explored in linear models. Alternatively the set of m_s can be combined to form a series of interval scaled seasonal indices that can

be used as an explanatory variable for the NN, with the requirement to predetermine the m_s beforehand.

C. Misspecifying NNs for Deterministic Seasonality

For our analysis it is interesting to examine what occurs if deterministic seasonality is misspecified as a seasonal unit root process. Considering seasonal differences (1) becomes

$$\Delta_S y_t = \Delta_S z_t. \quad (4)$$

Essentially in (4) seasonality has been removed, i.e. a deseasonalised form of y_t is modelled. Comparing (1) and (4) we can deduce that it is now impossible to estimate m_s and furthermore $\Delta_S z_t$ is overdifferenced [17]. Therefore, it is preferable to keep deterministic seasonality and model it appropriately.

However, neglecting the properties of the deterministic seasonality NNs are frequently modelled as a misspecified stochastic seasonal unit root process, with the problems implied above. One alternative is to use seasonal integration to remove seasonality and another alternative would be to use an adequate AR structure to model the seasonality as discussed in [12]. Note that much of the debate in literature regarding the deseasonalisation of time series (see section I) falls in the latter two alternatives, which in theory are not advisable for deterministic seasonality. However, for practical applications with small samples it can be shown that it is difficult to distinguish between deterministic and stochastic seasonality [17], therefore these alternatives are still viable options that warrant experimental evaluation.

III. EXPERIMENTAL DESIGN

A. Time Series Data

We employ eight synthetic time series to evaluate the different approaches to model deterministic seasonality using NN. The time series are constructed using as a data generating process the dummy variable representation of deterministic seasonality (1). All time series have $S = 12$, i.e. simulate monthly data, and are 480 observations long. Two different sets of m_s are modelled, reflecting two different seasonal patterns (A & B) derived from retail sales with $\mu = 240$ and z_t is $\varepsilon_t \sim \text{i.i.d. with } (0, \sigma_\varepsilon^2)$. These are superimposed with four levels of increasing random noise σ_ε^2 : for time series without noise $\sigma = 0$ is used (i.e. zero error for all t). For low, medium and high noise we use $\sigma = \{1, 5, 10\}$ respectively. Note that the synthetic time series are constructed in a stricter way than required by (1) in order to create time series in which only the effect of the deterministic seasonal pattern requires modelling, simplifying the specification of the input vector of the NN to focus solely on the effects of the different seasonal coding schemes. For the empirical evaluation all series are split in three equal subsets of $1/3^{\text{rd}}$ training, $1/3^{\text{rd}}$ validation and $1/3^{\text{rd}}$ test data of 160 observations. The first 72 observations of both time series patterns and 4 noise levels are plotted in fig. 1:

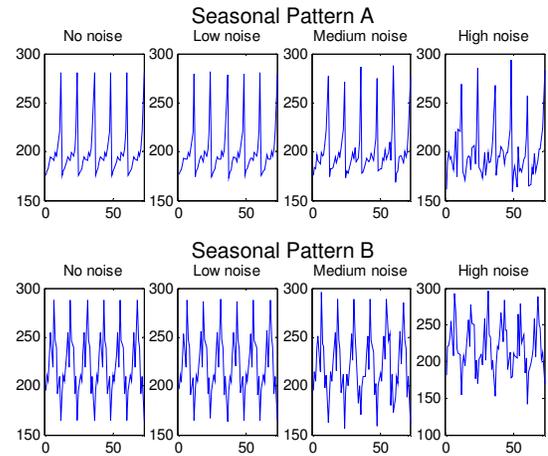


Fig. 1. Plot of the first 72 observations of each synthetic time series.

B. Evaluating Empirical Accuracy

We conduct a fixed horizon, rolling origin evaluation in order to assess the error of forecasting $t+h = 1, \dots, 12$ months in the future across multiple time origins. This evaluation scheme is preferred because it provides a reliable estimation of the out of sample error [20]. Two established and robust error measures are used: the mean absolute error (MAE) allows a direct comparison of the predictive accuracy and the known noise level by evaluating the absolute deviation of actuals X_t and the forecast F_t for all periods t in the test set. In addition the symmetric mean absolute percent error (sMAPE) is used as a scale independent accuracy measure to facilitate comparisons across time series. The accuracy of the competing NN models is evaluated for statistically significant differences using the nonparametric Friedman test and the Nemenyi test, to facilitate an evaluation of nonparametric models without the need to relax assumptions of ANOVA or similar parametric tests [21]. To compare the NN models against the benchmark the best NN initialisation in the validation set is used.

C. Neural Networks for Time Series Prediction

We limit our evaluation to the multilayer perceptron (MLP), the most widely employed NN architecture [1]. The advantage of MLP is that they are well researched regarding their properties and their proven abilities in time series prediction to approximate and generalise any linear or nonlinear functional relationship to any degree of accuracy [22] without any prior assumptions about the underlying data generating process [23], providing a powerful forecasting method for linear or non-linear, non-parametric, data driven modelling. In univariate forecasting MLP is used similarly to a regression model, capable of using as inputs a set of lagged observations of the time series to predict its next value [7]. Data are presented to the network as a sliding window over the time series history. The NN tries to learn the underlying data generation process during training so that valid forecasts are made when new input values are provided [24]. In this analysis single hidden layer NN are used, based on the proof of universal approximation [22]. The general function of these networks is

$$f(X, w) = \beta_0 + \sum_{h=1}^H \beta_h g \left(\gamma_0 + \sum_{i=0}^I \gamma_{hi} x_i \right). \quad (5)$$

$X = [x_0, x_1, \dots, x_n]$ is the vector of the lagged observations (inputs) of the time series and $w = (\beta, \gamma)$ are the network weights. I and H are the number of input and hidden units in the network and $g(\cdot)$ is a non-linear transfer function [25]. In this analysis the hyperbolic tangent (tanh) transfer function is used, which is frequently used for NNs [26].

D. Seasonal Encodings for Neural Networks

We develop and compare MLPs that code the deterministic seasonality of the time series using the seven alternative encodings described in section II.B and C.

The common coding of deterministic seasonality through monthly seasonal dummy variables is implemented in models *Bin11* and *Bin12* which use $S=11$ and $S=12$ seasonal binary dummy variables respectively. The integer dummy variable representation uses only a single integer explanatory variable that repeats values from 1 to 12 (named *Int*). The trigonometric representation is modelled through the use of two dummy variables, one for $\sin(2\pi t/12)$ and one for $\cos(2\pi t/12)$ and is named *SinCos*. Finally, seasonal indices for the time series are identified by calculating the average value for each period of the season in the training set. This is an adequate estimation since the time series exhibit no trend or irregularities. The seasonal indices are repeated to create an explanatory variable which is then used as the only input to the MLP model *SIndex*. No time series lags are used for these models.

To model deterministic seasonality as stochastic process a univariate MLP model is used that employs lag $t-1$ and $t-12$ (named *AR*). To model seasonality as a seasonal unit root process the time series is used after seasonal integration, replicating the common approach of removing seasonality prior to inputting the time series to the MLP. No lags are used and the correct level is estimated by the MLP by assigning the correct weights to the bias terms in the different nodes (named *SRoot*). An overview of the inputs for each model is provided in table I.

TABLE I
Summary of MLP Inputs

Model	Lags*	Explanatory variables**	No of inputs
AR	1, 12	-	2
Bin11	-	11 Seasonal Dummies	11
Bin12	-	12 Seasonal Dummies	12
Int	-	Integer Dummy [1,2...12]	1
SinCos	-	$\sin(2\pi t/12), \cos(2\pi t/12)$	2
Sindex	-	Seasonal Indices	1
SRoot	-***	-	0

* The Lags specify the time lagged realisations $t-n$ used as inputs; ** For all explanatory variables only the contemporary lag is used; *** Time series is modelled after seasonal integration, i.e. $\Delta_S Y_t$.

All the other parameters of the MLPs are held constant for all models. This allows attributing any differences in the performance of the models solely to the differences in modelling seasonality. All MLP use a single hidden layer with six hidden nodes and are trained using the Levenberg-

Marquardt algorithm. This algorithm requires setting the μ_{LM} and its increase and decrease steps. Here $\mu_{LM}=10^{-3}$, with an increase step of $\mu_{inc}=10$ and a decrease step of $\mu_{dec}=10^{-1}$. For a detailed description of the algorithm and the parameters see [27]. The maximum training epochs are set to 1000. The training can stop earlier if μ_{LM} becomes equal or greater than $\mu_{max}=10^{10}$ or the validation error increases for more than 50 epochs. This is done to avoid over-fitting. When the training is stopped the network weights that give the lowest validation error are used. Each MLP is initialised 50 times to account for randomised starting weights and to provide an adequate sample to estimate the distribution of the forecast errors in order to conduct the statistical tests. The MLP initialisation with the lowest error for each time series on the validation dataset is selected to predict all values of the test set. Lastly, the time series and all explanatory variables that are not binary are linearly scaled between $[-0.5, 0.5]$.

E. Statistical Benchmark Methods

Any empirical evaluation of time series methods requires the comparison against established statistical benchmark methods, in order to assess the increase in accuracy and its contribution to forecasting research (a fact often overlooked in NN experiments [11]). In this analysis we use seasonal exponential smoothing models (*EXSM*) with appropriately selected additive seasonality as a benchmark. The smoothing parameters are identified by optimising the one step ahead in-sample mean squared error. This model is selected as a benchmark due to its proven track record in univariate time series forecasting [8]. For more details on exponential smoothing models and the guidelines that were used to implement them in this analysis see [28].

IV. RESULTS

A. Nonparametric MLP Comparisons

All competing MLPs are tested for statistically significant differences using the Friedman and the post-hoc Nemenyi tests based on the mean rank of the errors. As MAE and sMAPE provided the same ranking, and both tests provided consistent results regardless of error measure, the results using a single measure are provided in table II.

The Friedman test indicates that across all time series, across different noise levels and for all time series separately there are statistically significant differences among the MLP models. Inspecting the results of the Nemenyi tests in table II we get a more detailed view on the ranking of each individual model and among which models there are statistically significant differences. It can be observed that across all different noise levels and across all time series the *SIndex* outperforms all other models with a statistically significant difference from the second best model. In all cases the *Bin11* and *Bin12* perform equally with no statistically significant differences both ranking second after *SIndex*. When all time series or only the no and low noise time series are considered, the *SinCos* has no statistically significant differences with the seasonal binary dummies *Bin11* and *Bin12* models. For the case of medium and high noise time series the *SinCos* ranks third after the *SIndex* and

seasonal binary dummy variables models. This demonstrates that although the *SinCos* model is not equivalent to the trigonometrical representation of deterministic seasonality as expressed in (2) it is able to approximate it and in many cases with no statistically significant differences from the equivalent seasonal dummy coding. Furthermore, this representation is $S/4$ times more economical in inputs compared to (2) and $S-2$ and $S-1$ inputs more economical from (1) or *Bin11* and *Bin12* respectively. For the low, medium and high noise the *Int* model follows fifth in ranking. Although this model performs worse than the previous seasonality encodings it still outperforms the misspecified seasonal models *AR* and *SRoot*. This is not true for the no noise time series, which also affects the overall ranking across time series as well. The *AR* model follows. This demonstrates that it is better to code the deterministic seasonality through explanatory dummy variables, than as an autoregressive process, as it would be fitting for stochastic seasonality. Furthermore, in agreement to the discussion in II.C, removing the seasonality through seasonal integration, as in *SRoot*, performs poorly and ranks last in most cases. The reason for this is that the NN are not able to estimate directly the m_s and Δ_{sy_t} is overdifferenced. Note that in the case of no noise all models with the exception of *Int* are able to capture the seasonality perfectly with no error.

TABLE II
Summary of MLP nonparametric comparisons

Time series	All	No noise	Low noise	Medium noise	High noise
Friedman p-value	0.000	0.000	0.000	0.000	0.000
Nemenyi Mean Model Rank* \blacktriangle					
AR	1521.7	330.0	520.3	522.6	555.0
Bin11	1300.5	330.0	251.1	234.9	239.3
Bin12	1302.3	330.0	247.5	238.5	246.3
Int	1548.9	473.5	461.3	422.2	346.0
SinCos	1310.0	330.0	254.8	250.7	257.2
Sindex	1241.1	330.0	143.6	162.3	182.5
SRoot	1579.0	330.0	575.0	622.3	627.3
Nemenyi Ranking at 5% significance level*					
AR	5	<u>1</u>	6	6	6
Bin11	<u>2</u>	<u>1</u>	<u>2</u>	<u>2</u>	<u>2</u>
Bin12	<u>2</u>	<u>1</u>	<u>2</u>	<u>2</u>	<u>2</u>
Int	6	7	5	5	5
SinCos	<u>2</u>	<u>1</u>	<u>2</u>	4	4
Sindex	1	<u>1</u>	1	1	1
SRoot	7	<u>1</u>	7	7	7

* In each column MLP with no statistically significant differences under the Nemenyi test at 5% significance are **underlined**; \blacktriangle the critical distance for the Nemenyi test at 1% significance level is 10.5, at 5% significance level is 9.0 and at 10% significance level is 8.2.

It is apparent that the best method to model the deterministic seasonality is to use the seasonal indices as an explanatory variable for the MLP. Not only this method performs best with statistically significant difference from the second best model, but also it is very parsimonious, requiring a single input to model the deterministic seasonality, as it can be seen in table I. This is an important finding, since this coding of seasonality is not widely used, although it is a direct extension of (1).

B. Comparisons against Benchmarks and Noise Level

Taking advantage of the synthetic nature of the time series we can compare directly the error of each forecasting model with the artificially introduced error level and derive how close is each model to an ideal accuracy. The ideal accuracy is when the model's error is exactly equal to the noise, since that would mean that the model has captured perfectly the data generating process and ignores completely the randomness. On the other hand, a lower error than the noise level would imply possible overfitting to randomness. The comparison is done in MAE for each time series individually. The results are presented in fig. 2. Moreover the benchmark accuracy in MAE for each time series is provided in the same figure.

In fig. 2 it is clear that when there is no noise, for both seasonal patterns, all MLP models and the benchmark forecast the time series perfectly with zero error. Comparing the MLP models to the benchmark the misspecified *AR* and *SRoot* models perform worse than *EXSM*, with the *SRoot* model performing consistently last. This demonstrates that for the case of deterministic seasonality deseasonalising the time series, here through seasonal integration, hinders the NN to forecast the time series accurately. For both seasonal patterns for the low noise time series 2 and 6 all MLP perform worse than the benchmark. The opposite is true for the higher noise level time series. This implies that NN perform better than the statistical benchmark in high noise time series, being able to capture the true data generating process better.

Comparing the models accuracy with the known error due to noise it is interesting that all the MLP models, with the exception of the misspecified *AR* and *SRoot*, for all time series are very close to the ideal accuracy, i.e. having error only due to randomness. Note that for the validation set, on which the best performing initialisation for each of the NN models was chosen, their error is practically only due to noise. The benchmark error consistently increases as the noise level increases. For the case of low noise time series *EXSM* manages to forecast the time series with the error being solely due to randomness, implying a very good fit to the data generating process. The results are consistent across both seasonal patterns.

TABLE III
Summary sMAPE across all time series

Model	Training subset	Validation subset	Test subset
AR	<u>1.90%</u>	<u>1.94%</u>	<u>1.72%</u>
Bin11	1.60%	1.59%	1.45%
Bin12	1.58%	1.58%	1.46%
Int	1.62%	1.61%	1.49%
SinCos	1.59%	1.59%	1.47%
Sindex	1.60%	1.58%	1.44%
SRoot	<u>2.36%</u>	<u>2.21%</u>	<u>1.91%</u>
EXSM	1.86%	1.68%	1.52%

The best performing model in each set is marked with bold numbers. The models that are outperformed by the EXSM benchmark are underlined

Evaluating the performance of all models across the three training, validation and test subsets the models perform consistently and there are no clues of overfitting to the

training set and all models are able to generalise well on the test set. Using MAE it is impossible to aggregate the results across different time series. Instead, the scale independent sMAPE is used. Summary accuracy sMAPE figures for all time series are provided in table III.

The results are consistent with fig. 2. The *AR* and *SRoot* models are outperformed by the benchmark, which is turn is outperformed by all other MLP models. In agreement with the results in table II the *SIndex* model is overall the most accurate, followed by the *Bin12* and *Bin11*. Note that the small sMAPE figures imply that all the models managed to capture the seasonal profile in all the time series and a visual inspection of the forecasts would reveal very small if no differences at all. Finally, the overall error level seems to be different between the three subsets. This is due to the random noise. Although each set contains 160 observations, which simulates in total 40 years of data, it was not enough to ensure equal noise distribution across all subsets.

III. CONCLUSIONS

We have evaluated different methodologies to model time series with deterministic seasonality. By exploring the theoretical properties of deterministic seasonality we show that the current debate in literature on how to model seasonality with NN does not address the problem correctly, omitting the properties of the type of seasonality. Seven competing approaches to model deterministic seasonality are evaluated and compared against a seasonal exponential smoothing model. *SIndex*, which uses the estimated seasonal indices as input to a NN, outperforms all competing MLP models and the benchmarks with statistically significant differences. Moreover, this approach to model seasonality is very parsimonious, requiring only one additional input. This may have particularly significant implications for high frequency data with long seasonal periods, often resulting in a large dimensionality of the input vector that can cause problems in training NN models.

This study does not thoroughly address the issue of how to best estimate the seasonal indices. In this study it proved relatively easy to obtain a good estimation of the seasonal indices for all time series, while this may become more problematic in the presence of irregularities, trends and multiple seasonalities. For future evaluations it is important to evaluate the robustness of the findings with different approaches to estimate the seasonal indices. Similarly, this study should be extended to more time series patterns and empirical time series that exhibit deterministic and stochastic seasonality, which will allow us to validate the findings and provide a reliable solution for practical applications.

REFERENCES

- [1] G. Q. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *International Journal of Forecasting*, vol. 14, no. 1, pp. 35-62, Mar, 1998.
- [2] H. S. Hippert, D. W. Bunn, and R. C. Souza, "Large neural networks for electricity load forecasting: Are they overfitted?," *International Journal of Forecasting*, vol. 21, no. 3, pp. 425-434, Jul-Sep, 2005.
- [3] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359-366, 1989.
- [4] G. P. Zhang, "An investigation of neural networks for linear time-series forecasting," *Computers & Operations Research*, vol. 28, no. 12, pp. 1183-1202, Oct, 2001.
- [5] G. P. Zhang, B. E. Patuwo, and M. Y. Hu, "A simulation study of artificial neural networks for nonlinear time-series forecasting," *Computers & Operations Research*, vol. 28, no. 4, pp. 381-396, Apr, 2001.
- [6] T. Hill, M. O'Connor, and W. Remus, "Neural network models for time series forecasts," *Management Science*, vol. 42, no. 7, pp. 1082-1092, Jul, 1996.
- [7] N. Kourentzes, and S. F. Crone, "Automatic modelling of neural networks for time series prediction - in search of a uniform methodology across varying time frequencies."
- [8] S. Makridakis, and M. Hibon, "The M3-Competition: results, conclusions and implications," *International Journal of Forecasting*, vol. 16, no. 4, pp. 451-476, Oct-Dec, 2000.
- [9] J. S. Armstrong, "Findings from evidence-based forecasting: Methods for reducing forecast error," *International Journal of Forecasting*, vol. 22, no. 3, pp. 583-598, 2006.
- [10] J. L. Callen, C. C. Y. Kwan, P. C. Y. Yip *et al.*, "Neural network forecasting of quarterly accounting earnings," *International Journal of Forecasting*, vol. 12, no. 4, pp. 475-482, Dec, 1996.
- [11] M. Adya, and F. Collopy, "How effective are neural networks at forecasting and prediction? A review and evaluation," *Journal of Forecasting*, vol. 17, no. 5-6, pp. 481-495, Sep-Nov, 1998.
- [12] B. Curry, "Neural networks and seasonality: Some technical considerations," *European Journal of Operational Research*, vol. 179, no. 1, pp. 267-274, May, 2007.
- [13] M. Nelson, T. Hill, W. Remus *et al.*, "Time series forecasting using neural networks: Should the data be deseasonalized first?," *Journal of Forecasting*, vol. 18, no. 5, pp. 359-367, Sep, 1999.
- [14] G. P. Zhang, and M. Qi, "Neural network forecasting for seasonal and trend time series," *European Journal of Operational Research*, vol. 160, no. 2, pp. 501-514, Jan, 2005.
- [15] D. R. Osborn, S. Heravi, and C. R. Birchenhall, "Seasonal unit roots and forecasts of two-digit European industrial production," *International Journal of Forecasting*, vol. 15, no. 1, pp. 27-47, 1999.
- [16] A. Matas-Mir, and D. R. Osborn, "Does seasonality change over the business cycle? An investigation using monthly industrial production series," *European Economic Review*, vol. 48, no. 6, pp. 1309-1332, 2004.
- [17] E. Ghysels, and D. R. Osborn, *The Econometric Analysis of Seasonal Time Series*, Cambridge: Cambridge University Press, 2001.
- [18] A. H. Kvanli, R. J. Pavur, and K. Keeling, *Introduction to Business Statistics*, 6th ed., Mason, Ohio: Thomson/South-Western, 2002.
- [19] S. F. Crone, and N. Kourentzes, "Input variable selection for time series prediction with neural networks-an evaluation of visual, autocorrelation and spectral analysis for varying seasonality." pp. 195-205.
- [20] L. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International Journal of Forecasting*, vol. 16, pp. 437-450, 2000.
- [21] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1-30, 2006.
- [22] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251-257, 1991.
- [23] M. Qi, and G. P. Zhang, "An investigation of model selection criteria for neural network time series forecasting," *European Journal of Operational Research*, vol. 132, no. 3, pp. 666-680, Aug, 2001.
- [24] G. Lachtermacher, and J. D. Fuller, "Backpropagation in Time-Series Forecasting," *Journal of Forecasting*, vol. 14, no. 4, pp. 381-393, Jul, 1995.
- [25] U. Anders, O. Korn, and C. Schmitt, "Improving the pricing of options: A neural network approach," *Journal of Forecasting*, vol. 17, no. 5-6, pp. 369-388, Sep-Nov, 1998.
- [26] T. P. Vogl, J. K. Mangis, A. K. Rigler *et al.*, "Accelerating the convergence of the backpropagation method," *Biological Cybernetics*, vol. 59, pp. 257-263, 1988.
- [27] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, Boston: PWS Publishing, 1996.
- [28] E. J. Gardner, "Exponential smoothing: The state of the art--Part II," *International Journal of Forecasting*, vol. 22, no. 4, pp. 637-666, 2006.

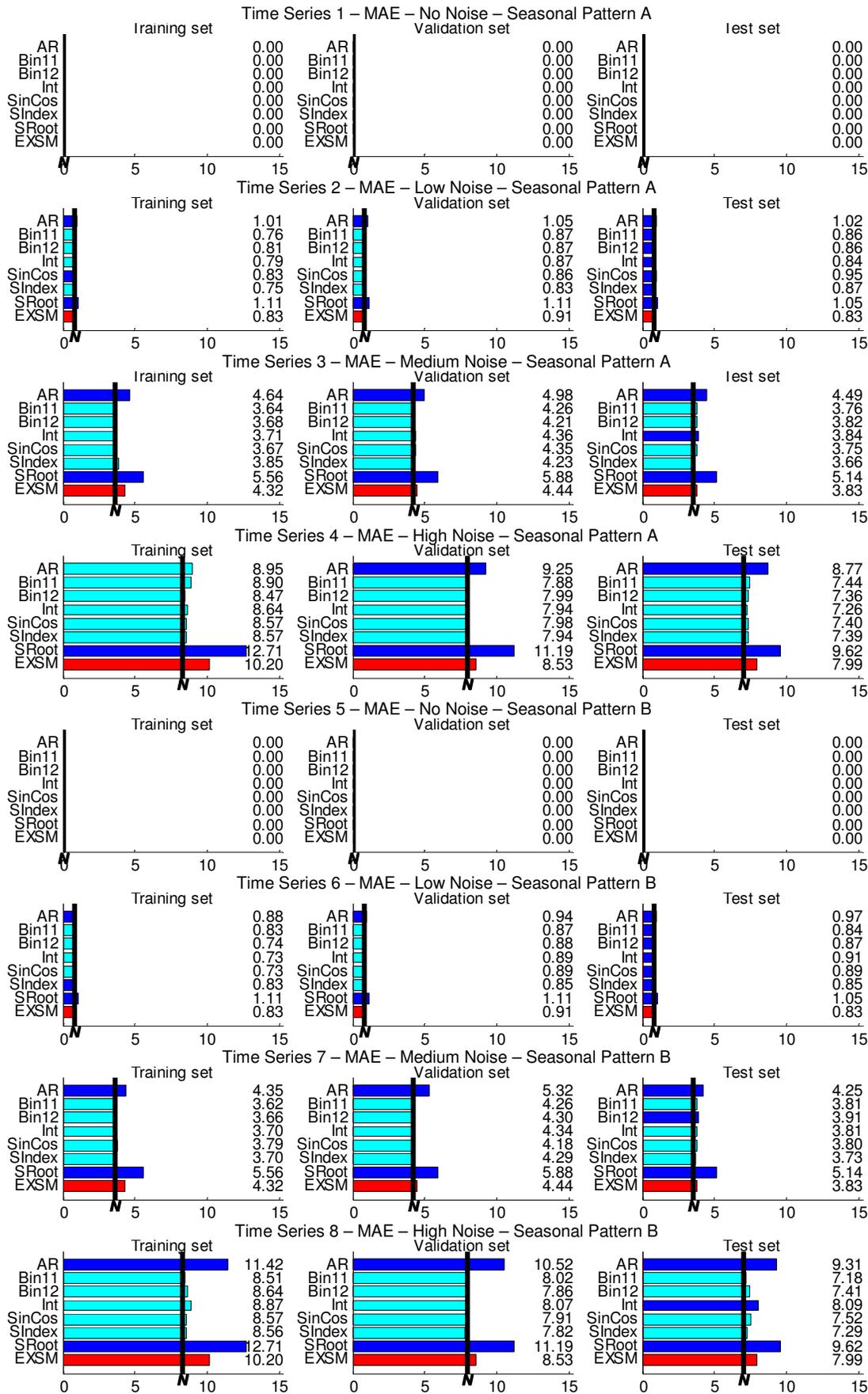


Fig. 2. MAE for each time series for each subset for all models. The noise level is marked by a thick black vertical line. Light coloured bars are models which are better than the benchmark (EXSM). The value of each error is provided at the right side of each row.

A Fuzzy Index Structure for Multi-Dimensional Data

Yong Shi

Building 11, Room 3060

Department of Computer Science and Information Systems

Kennesaw State University

Kennesaw, GA 30144

yshi5@kennesaw.edu

Abstract—*Nowadays large volumes of data with high dimensionality are being generated in many fields. Most existing indexing techniques degrade rapidly when dimensionality goes higher. A large amount of data sets are time related, and the existence of the obsolete data in the data sets may seriously degrade the data processing. In our previous work[7], we proposed ClusterTree⁺, a new indexing approach representing clusters generated by any existing clustering approach. It is a hierarchy of clusters and subclusters which incorporates the cluster representation into the index structure to achieve effective and efficient retrieval. It also has features from the time perspective. In this paper, we apply the Fuzzy concept to improve the performance of the dynamic insertion approach, keeping the index structure always in the most updated status which can further promote the efficiency and effectiveness of data query, data update, etc. This approach is highly adaptive to any kind of clusters.*

1. Introduction

Recently large volumes of data with high dimensionality are being generated in many fields. Many approaches have been proposed to index multi-dimensional data sets for efficient querying. Although most of them can efficiently support nearest neighbor search for low dimensional data sets, they degrade rapidly when dimensionality goes higher. Also the dynamic insertion of new data can cause original structures no longer handle the data sets efficiently since it may greatly increase the amount of data accessed for a query. We also observe that a large proportion of the data sets are time related, and the existence of the obsolete data in the data sets may seriously degrade the data processing. However, few approaches are proposed to implement the maintenance of the whole data sets to get rid of the obsolete data and keep the data sets always in the most updated status for the convenience and effectiveness of data processing such as query and insertion.

In our previous work[7], we proposed ClusterTree⁺, an efficient index structure from clustering for multi-dimensional data sets. Clustering is an analysis technique for discovering interesting data distributions and patterns in the underlying

data sets. Given a set of n data points in a d -dimensional space, a clustering approach assigns the data points to k groups ($k \ll n$) based on the calculation of the degree of similarity between data points such that the data points within a group are more similar to each other than the data points in different groups. Each group is a cluster. The ClusterTree⁺ approach builds an index structure on the cluster structures to facilitate efficient queries.

In this paper, we apply the Fuzzy concept to improve the performance of the dynamic insertion approach, keeping the indexing structure always in the most updated status which can further promote the efficiency and effectiveness of data query, data update, etc. This approach is highly adaptive to any kind of clusters. The ClusterTree⁺ index structure has features from the time perspective. Each new data item is added to the ClusterTree⁺ with the time information which can be used later in the data update process for the acquisition of the new cluster structure. This approach guarantees that the ClusterTree⁺ is always in the most updated status which can further promote the efficiency and effectiveness of data query and update. Our approach is highly adaptive to any kind of clusters.

The rest of the paper is organized as follows. Section 2 summarizes the related work on index structure design. Section 3 presents the dynamic insertion approach of ClusterTree⁺. Section 4 gives the conclusion.

2. Related Work

Existing multi-dimensional tree-like indexing approaches can be further classified into two categories: data partitioning and space partitioning.

A data partitioning approach partitions a data set and builds a hierarchy consisting of bounding regions. Popularly used index structures include R-Tree, R*-tree, SS-Tree, SS⁺-Tree and SR-Tree. An R-tree [4] is a height-balanced tree with index records in its nodes. The R*-tree [2] is an R-tree variant which incorporates a combined optimization of area, margin and overlap of each enclosing rectangle in the tree nodes. In contrast to the R-Tree, SS-Tree [1] uses hyperspheres as region units. SR-Tree [5] is an index structure which combines the bounding spheres and rectangles for the shapes of node regions to reduce the blank area.

Space partitioning approaches recursively divide a data space into disjoint subspaces. K-D-B Tree and Pyramid-Tree are this type. The K-D-B Tree [6] partitions a d -dimensional data space into disjoint subspaces by $(d - 1)$ -dimensional hyper-planes which are alternately perpendicular to one of the dimension axes. The Pyramid-Tree [3]'s main idea is to divide the data space first into $2d$ pyramids, each sharing the center point as its peak (the tip point of a pyramid).

ClusterTree⁺ is an efficient index structure built from clustering for multi-dimensional time-related data sets. The ClusterTree⁺ has the advantage of supporting efficient data query for multi-dimensional data sets. It has two separate structures, one is ClusterTree*, the other is a modified B⁺ tree – B^t tree.

The ClusterTree* is a hierarchical representation of the clusters of a data set. It has two kinds of nodes: internal and leaf nodes. The internal nodes are defined as:

$$\begin{aligned} \text{Node} : & [Node_id, \gamma, ot, nt, (Entry_1, Entry_2, \dots, Entry_\gamma) \\ & (m_{node} \leq \gamma \leq M_{node})], \\ \text{Entry}_i : & (SC_i, BS_i, SN_i), \end{aligned}$$

where $Node_id$ is the node identifier, γ is the number of the entries in the node, ot is the oldest time when data were inserted into that node or its descendants, nt is the newest time when data were inserted into that node or its descendants, and m_{node} and M_{node} define the minimum and maximum numbers of entries in the node. An entry is created for each subcluster of the cluster for which the current non-leaf node represents. In entry $Entry_i$, SC_i is a pointer to the i -th subcluster, BS_i is the bounding sphere for the subcluster and SN_i is the number of data points in the i -th subcluster.

The leaf nodes are defined as:

$$\begin{aligned} \text{Leaf} : & [Leaf_id, \gamma, ot, nt, (Entry_1, Entry_2, \dots, Entry_\gamma) \\ & (m_{leaf} \leq \gamma \leq M_{leaf})], \\ \text{Entry}_i : & (ADR_i, T_i, L_i), \end{aligned}$$

where γ is the number of data points contained in the leaf node, and m_{leaf} and M_{leaf} are the minimum and maximum numbers of entries. $Entry_i$ contains the address of the data point residing at the secondary storage (ADR_i), the time information when the data point is inserted into the structure (T_i), and the link to the time data point in the B^t tree (L_i).

The B^t tree indexes on the time data which corresponds to the times the data were inserted into the structure. It originates from the B⁺ tree with some modifications as follows:

- There is no minimum number requirement of entries in internal and leaf nodes so that there will be no cases of underflow. This corresponds to the nature of the B^t tree that the time data in it will be deleted in batches for the user-specified deletion requirement.
- In the leaf nodes, each entry has an extra field which is a link to the data point it is associated with in the ClusterTree*, thus we can traverse from the B^t tree back to the ClusterTree* efficiently.

2.1 Fuzzy concept

Some data in the real world are not naturally well organized. Clusters in the data may overlap each other. Fuzzy concept can be applied to further smooth the shrinking movement of the "boundary" data points.

The concept of fuzzy sets was first introduced by Zadeh (1965) to represent vagueness. The use of fuzzy set theory is becoming popular because it produces not only crisp decision when necessary but also corresponding degree of membership. Usually, membership functions are defined based on a distance function, such that membership degrees express proximities of entities to cluster centers. In conventional clustering, sample is either assigned to or not assigned to a group. Assigning each data point to exactly one cluster often causes problems, because in real world problems a crisp separation of clusters is rarely possible due to overlapping of classes. Also there are exceptions which cannot be suitably assigned to any cluster. Fuzzy sets extend to clustering in that object of the data set may be fractionally assigned to multiple clusters, that is, each point of data set belongs to groups by a membership function. This allows for ambiguity in the data and yields detailed information about the structure of the data, and the algorithms adapt to noisy data and classes that are not well separated. Most fuzzy cluster analysis methods optimize a subjective function that evaluates a given fuzzy assignment of data to clusters.

One of the classic fuzzy clustering approach is the Fuzzy C-means Method designed by Dunn, J.C., Bezdek, J. C. In brief, for a data set X with size of n and cluster number of c , it extends the classical within groups sum of squared error objective function to a fuzzy version by minimizing the objective function with weighting exponent m , $1 \leq m < \infty$:

$$J_m(U, V) = \sum_{k=1}^n \sum_{i=1}^c (u_{ik})^m d^2(x_k, v_i), \quad (1)$$

where U is a partition of X in c part, $V = v = (v_1, v_2, \dots, v_c)$ are the cluster centers in R^p , and A is any $(p \times p)$ symmetric positive definite matrix defined as the following:

$$d(x_k, v_i) = \sqrt{(x_k - v_i)^\top (x_k - v_i)}, \quad (2)$$

where $d(x_k, v_i)$ is an inner product induced norm on R^p , u_{ik} is referred to as the grade of membership of x_k to the cluster i .

The fuzzy C-Means (FCM) uses an iterative optimization of the objective function, based on the weighted similarity measure between x_k and the cluster center v_i . During each iteration, it calculates the c cluster centers $\{v_{i,t}\}, i = 1, \dots, c$

$$v_{i,t} = \frac{\sum_{k=1}^n u_{ik,t-1}^m x_k}{\sum_{k=1}^n u_{ik,t-1}^m}, \quad (3)$$

for those data points not of any current cluster center, it calculate the following

$$u_{ik,t} = \frac{1}{\sum_{j=1}^c \left(\frac{d_{ik,t}}{d_{jk,t}} \right)^{\frac{2}{m-1}}}. \quad (4)$$

When a predefined termination condition is satisfied, the algorithm is terminated.

3. Fuzzy Based Dynamic Insertion to ClusterTree⁺

Dynamic insertion of high dimensional data becomes a critical issue in real high dimensional databases. Here we present a dynamic insertion approach for multi-dimensional data using a modified ClusterTree⁺, keeping the structure always in optimal status.

A new coming data point in ClusterTree⁺ can be classified into one of three categories:

- *Cluster points*: are either the duplicates of or very close to some data points in a cluster within a given threshold.
- *Close-by points*: are the data points which are neighbors to some points in the clusters within a given threshold.
- *Random points*: are the data points which are either far away from all of the clusters and can not be bounded by any bounding sphere of the ClusterTree⁺, or might be included in the bounding spheres of the clusters at each level, but they do not have any neighboring cluster points within a given threshold.

Different insertion strategies are designed for each type of new data points. The neighbor information of the new data point is available by applying the query process of the new data point on the ClusterTree⁺ so that the new data point can be classified into one of the three categories mentioned above. We can directly insert the *cluster points* into a certain leaf node of the structure without any modification of the tree structure. As for the *close-by points*, they will be inserted into a certain leaf node of the structure which contains its nearest neighbors, but the radius and centroid of the certain node should be slightly adjusted. If a new coming data point is a *random point*, there are two cases: if absolutely no nodes contain it, it will be collected for creating another ClusterTree⁺ later; if it can be bounded by bounding spheres of some clusters in the ClusterTree⁺, but they don't have any neighboring cluster points in it, it should be treated differently.

For the latter case of the random points, we use *maximum inclusion depth* to represent the lowest level of the subclusters whose bounding sphere contains the random point. We here propose a *delayed insertion* approach which stores a single data point without changing the radius and centroid of any node in the ClusterTree⁺, and optimizes the new inserted data points when the amount of random points reaches a certain threshold.

However, there are two major problems in the random points processing mentioned above:

- the threshold of the amount of random points to optimize the current cluster is hard to decide for the complicated cases of the clusters. In a cluster, if the new coming random points are all scattered sparsely in the bounding sphere of the cluster, the original subclusters still can represent the correct status of this cluster efficiently and effectively even though the amount of the new coming random points is very large. On the other hand, if the new coming random points are close to each other, the status of the current cluster should be updated even though the total amount of the new random points is not very large. Thus the threshold could not be simply defined as a certain percent of the whole data points in the current cluster.
- the cluster reorganization process is quite time-consuming when more nodes and data points are involved, and the reorganization could be recursively propagated upward to the top node of the whole tree structure. Thus we should avoid the reorganization cases as much as possible.

According to these concerns, we focus on the dynamic insertion of the second case of the *random points*. The random points mentioned below are of this kind.

There are two ways to implement the dynamic insertion issue in our structure:

- based on the partition of the bounding sphere of the current cluster. It is possible that we separate the "space" of the cluster into several parts which have different possibility of welcoming new coming random points. Of course, the "welcome possibility" of those areas occupied by some subclusters is 0 since those new data points in such areas are either cluster points or close-by points. Based on the possibility level of different areas and some presumptions, we can predict the positions of the new coming data points and further determine if current inserted new data point is an outlier or one which should belong to a cluster. The problem of this approach is that the bounding spaces of those clusters containing high dimensional data points are normally very sparse, so the efficiency and effectiveness can not be guaranteed.
- based on the new coming random points themselves to improve the performance of dynamic insertion. This is a more reasonable and practicable one. Our design is based on this solution.

Definition (pseudo-cluster): a group of the random points in a certain cluster which are close to each other. This term is helpful for the further analysis and processing of the dynamic insertion.

In order to design our dynamic insertion approach, we modify the structure of ClusterTree*. It has two kinds of nodes: internal and leaf nodes. The internal nodes are defined

as:

$$\begin{aligned} \text{Node} &: [Node_id, \gamma, (Entry_1, Entry_2, \dots, Entry_\gamma), \\ &\quad RPS, CM(m_{node} \leq \gamma \leq M_{node})], \\ \text{Entry}_i &: (SC_i, BS_i, SN_i), \end{aligned}$$

where $Node_id$ is the node identifier, γ is the number of the entries in the node, m_{node} and M_{node} define the minimum and maximum numbers of entries in the node. An entry is created for each subcluster of the cluster for which the current non-leaf node represents. In entry $Entry_i$, SC_i is a pointer to the i -th subcluster, BS_i is the bounding sphere for the subcluster and SN_i is the number of data points in the i -th subcluster.

The dynamic insertion in the internal nodes introduces RPS and CM. RPS is the structure containing the random points information:

$$\begin{aligned} RPS &: [RP_1, RP_2, \dots], \\ RP_i &: (ADR_i, PC_i), \end{aligned}$$

where ADR_i is the address of the random point residing on the secondary storage, and PC_i is the pseudo-cluster number it is in. We will explain it later.

The CM is the correlation matrix of subclusters and pseudo-clusters of the current cluster. It stores the amount of common neighbor data points of each (subcluster, pseudo-cluster) pair and each (pseudo-cluster, pseudo-cluster) pair. Also it stores the amount of random points each pseudo-cluster contains and the amount of "original" data points each subcluster contains.

The leaf nodes have the same structure defined in Section 2. If a random point is in a leaf node, it can be inserted into it directly without delay since the leaf node does not contain subclusters which need to be readjusted.

In the dynamic insertion process, we should avoid reorganization as much as possible since it is very time-consuming. When a new random point comes to the suitable subcluster with its maximum inclusion depth, it is firstly inserted into this subcluster node's RPS structure with the pseudo-cluster number of 0. If it is close enough to a previously random points, a new pseudo-cluster is created including both of them; if it is close enough to several random points, it can be a common neighbor point of more than one pseudo-clusters. We inspect the correlations of the (subcluster, pseudo-cluster) pairs and those of the (pseudo-cluster, pseudo-cluster) pairs. If the amount of the common neighbors of two sides exceeds a certain threshold, we regard these two structures as being ready for mergence.

4. Conclusion

In this paper, we propose a fuzzy based dynamic insertion approach for multi-dimensional data using a modified ClusterTree⁺ which can efficiently support the time-related queries and user-specified deletions. The ClusterTree⁺ can keep the data sets always in the most updated status to

promote the efficiency and effectiveness of data query and update. Further experiments will be conducted to demonstrate the efficiency and effectiveness of dynamic insertion using ClusterTree⁺.

This approach can be helpful in the fields of data fusion where the data evolve dynamically and regular approaches often fail to solve the problem of keeping a certain system always containing the most updated data. It can also dynamically supervise the data status of the system and efficiently get rid of obsolete data, and at the same time, reorganize the structure of the data sets when necessary.

References

- [1] White D.A. and Jain R. Similarity Indexing with the SS-tree. In *Proceedings of the 12th Intl. Conf. on Data Engineering*, pages 516–523, New Orleans, Louisiana, February 1996.
- [2] Beckmann N. and Kriegel H.P. and Schneider R. and Seeger B. The R*-tree: an Efficient and Robust Access Method for Points and Rectangles. In *Proceedings of ACM-SIGMOD International Conference on Management of Data*, pages 322–331, Atlantic City, NJ, May 1990.
- [3] B. C. Berchtold S. and K. H. The Pyramid-Technique: Towards Breaking the Curse of Dimensionality. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 142–153, Seattle, Washington, 1998.
- [4] Guttman A. R-Trees: A Dynamic Index for Geometric Data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 47–57, 1984.
- [5] N. Katayama and S. Satoh. The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, pages 369–380, Tucson, Arizona, 1997.
- [6] J. Robinson. The K-D-B-Tree: A Search Structure for Large Multi-dimensional Dynamic Indexes. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 10–18, Ann Arbor, MI, Apr. 1981.
- [7] Yong Shi and Aidong Zhang. Dynamic clustering and indexing of multi-dimensional datasets. In *4th International Conference on Information Fusion*, 2001.

Network-wide Analysis Using Spatio-temporal Association Rules Mining

Weisong He, and Guangmin Hu

Abstract—In this paper, a new definition of spatio-temporal association rules mining algorithm, designed to effectively analyze multi-featured network traffic from multiple Points of Presence(PoPs) are proposed. Using proposed algorithm, we analyze multiple POPs within temporal and spatial domain and report spatio-temporal collaborative behavior pattern. Due to frequent pattern produced by our method, we can monitor network situation more effectively. Experiment with Abilene Netflow data verifies the effect of our solutions.

I. INTRODUCTION

Network-wide traffic analysis is important for network behavior analysis. But, large-scale network traffic is highly dynamic, which across exhibits multi-timescale properties within temporal and spatial domain. Network-wide traffic anomalies have the feature of erupting suddenly without known signs, which can bring great damage to network equipments or computers of network in a short time. Therefore, one of the prepositions to ensure trustworthy networks is to analyze network traffic behavior fast and accurately, determine the reasons that cause abnormal behavior and make reasonable response to them in time.

This paper describes a new defined spatio-temporal association rules mining method for analyzing multi-featured network traffic from multiple sources. Examples include processing all multivariate time series in a network simultaneously, and features from all PoPs simultaneously. The features involved may include fields contained in packet headers as well as traditional volume measures. Network-wide analysis of traffic is a challenging problem involving multivariate statistics. Even a moderate-sized network may contain hundreds of flows, and each traffic stream presents many features that may be of interest. Thus the central problem one confronts in network-wide traffic analysis is the so-called “curse of dimensionality”. To attack this problem we adopt the general strategy of seeking low-dimensional approximations that preserve important traffic properties. Our starting point, and the first contribution of this paper, is to demonstrate that accurate low-dimensional approximations of network traffic often exist. We show that network-wide traffic measurements that exhibit as many as hundreds of dimensions can be approximated well using a much smaller

Weisong He is with the School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, SC 610054, P.R.China (phone: 86-28-83204921; email: weisonghe@uestc.edu.cn).

Guangmin Hu is with the School of Communication and Information Engineering, University of Electronic Science and Technology of China, SC 610054, P.R.China. Email: hgm@uestc.edu.cn

TABLE I
THE FLOW RECORD CONTENTS

Basic information	Related to routing
srcaddr,dstaddr,srcport,dstport	nexthop
dpkts, doctets	src_as, dst_as
first,last	src_mask,dst_mask
tos, tcp_flags	

set of dimensions. This observation of low effective dimensionality is key, and provides leverage on a number of problems related to network operations. In particular, low effective dimensionality leads us to make use of subspace methods. These methods systematically exploit the low dimensionality of multi-feature traffic flows, to capture network-wide normal behavior, and to expose anomalous events that span a network. Our second contribution is to show that spatio-temporal association rules mining method is proposed to effectively analyze multi-featured network traffic from multiple PoPs simultaneously.

The rest of the paper is organized as follows. In Section II, we introduce Netflow and subspace method. In Section III, we illustrate spatiotemporal association rules mining algorithm. In Section IV, we provide experiment and in Section V a conclusion.

II. DATA PREPROCESSING

A. Netflow

The Netflow which is passive monitoring tool includes: statistics about groups of related packets (e.g., same TCP/IP headers and close in time); records header information, counts, and time. As can be seen from Table I, the flow record contents can be divided into two parts: one is the basic information about the flow; the other one is information related to routing. The basic information about the flow are as follows: the first information is source and destination, IP address and port; the second one is packet and byte counts; the third one is start and end times; the fourth one is ToS and TCP flags. The information related to routing are listed as follows: the first Next-hop IP address; source and destination AS; Input and output.

B. Entropy

Let X denote a random variable representing the distribution of values a particular traffic feature (e.g., the source address or destination port of a flow) can take. Let $x_1 \cdots x_N$ denote the range of values that X can take, and for each x_i

let $p(x_i)$ represent the probability that the random variable X takes the value x_i , i.e., $p(x_i) = Pr[X = x_i]$. The entropy [10] of the random variable X is defined as

$$H(X) = - \sum_{i=1}^N p(x_i) \log p(x_i) \quad (1)$$

C. Subspace and Independent Component Analysis

Subspace method is based on a separation of the space of traffic measurement into normal and abnormal subspaces by means Independent Component Analysis(ICA). Effectively diagnosis of anomalies requires the ability to separate them from normal network-wide behavior. Let m denote the number of metrics in single PoPs and t denote the number of successive time intervals of interest. We let Y be the $t \times m$ measurement matrix, which denotes the time series of all metrics within single PoPs. Thus, each column i denotes the time series of the i^{th} metric and each row j represents an instances of all the metrics within single PoPs at time j . Note that Y thus defined has rank at most m . While Y denotes the set of measurements of all metrics over time, we will also frequently work just with y , a vector of measurement of all metrics from single timestep. Thus y is an arbitrary row of Y , transposed to a column vector. We refer to individual columns of a matrix using a single subscript, so the time series of measurements of metric i is denoted Y_i . The subspace method can identify typical variation in a set of correlated metrics, and detect unusual conditions based on deviation from that typical variation. We will apply ICA on our data matrix Y , treating each row of Y as a point in \mathbb{R}^m , to compute a set of m independent components, $\{V_i\}_{i=1}^m$. However, before we can apply ICA, it is necessary to adjust Y so that that its columns have zero mean.

For n independent components, $s_1(t), s_2(t), \dots, s_n(t)$, assume that we observe m linear mixtures $x_1(t), x_2(t), \dots, x_m(t)$ of n independent components.

$$x_j(t) = a_{j1}s_1(t) + a_{j2}s_2(t) + \dots + a_{jn}(t)s_n(t), \quad j = 1, 2, \dots, m. \quad (2)$$

ICA [1, 2, 3] is a statistical technique that represents a multidimensional random vector as a linear combination of non-Gaussian random variables that are as independent as possible. ICA is a nongaussian version of factor analysis, and somewhat similar to PCA. However, PCA only gives components that are uncorrelated. As abnormal network-wide traffic behavior signals are of nongaussian, if mixed signals include many abnormal signals and only one normal signal, we can separate mixed traffic signals into abnormal and normal independent signals using ICA.

III. SPATIO-TEMPORAL ASSOCIATION RULES MINING

A. Apriori Algorithm

$I = \{i_1, i_2, \dots, i_k\}$ is a set of items. X is an *itemset* if it is a subset of I . $T = \{t_i, t_{i+1}, \dots, t_n\}$ is a set of transaction. A transaction t contains *itemset* X iff, for all items, where

$i \in X$, i is a t -*itemset*. All itemset X in a transaction database has a *support*, denoted $Sup(X)$ [4, 5], see as in Eq.(3).

$$Sup(X) = \frac{\sigma(X)}{|T|} \quad (3)$$

$\sigma(X) = |\{t_i | X \subseteq t_i, t_i \in T\}|$. Given an *itemset* X , we find all the rules $X \rightarrow Y$ with minimum *support* and *confidence*. *Support* is a probability that a transaction contains, denoted as $Sup(X \cup Y)$. *Confidence* is a conditional probability that a transaction contains X as well as Y , denoted as $Sup(X \cup Y)/Sup(X)$. *Frequent Itemsets* is used to generate all frequent itemsets in a given database T .

Generally, an association rules mining algorithm contains the following steps: the set of candidate k -*itemsets* is generated by 1-extensions of the large $(k-1)$ -*itemsets* generated in the previous iteration. Supports for the candidate k -*itemsets* are generated by a pass over the database. *Itemsets* that do not have the minimum support are discarded and the remaining itemsets are called large k -*itemsets*. This process is repeated until no more large *itemsets* are found. Apriori is more efficient during the candidate generation process [4]. Apriori uses pruning techniques to avoid measuring certain *itemsets*, while guaranteeing completeness.

B. Symbolic Representation of Time Series

1) *Piecewise Aggregate Approximation*: The basic idea of Piecewise Aggregate Approximation(PAA) [6, 7] is that it represents the time series as a sequence of rectangle basis functions as in Eq.(4):

$$\bar{p}_i = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} p_j \quad (4)$$

n is the length of sequence, N is the number of PAA segments. p_j is the value of the j^{th} element of p . \bar{p}_i is the average value of the i^{th} segment of p . Simply stated, to reduce the time series from n dimensions to N dimensions, the data is divided into N equal sized "frames". The mean value of the data falling within a frame is calculated and a vector of these values becomes the data-reduced representation.

2) *Symbolic Aggregate Approximation*: The basic idea of Symbolic Aggregate approximation(SAX) [6, 7] is that it converts the time series into a discrete symbolic sequence. Having transformed a time series data into the PAA, we can apply SAX to obtain a discrete symbolic representation. Different from other symbolic representations of time series, SAX allows dimensionality/numerosity reduction, and it also allows distance measures to be defined on the symbolic approach that lower bound corresponding distance measures are defined on the original series. The latter feature is particularly interesting because it allows one to run certain data mining algorithms on the efficiently manipulated symbolic representation, while producing same results to the algorithms that process the original data [8, 9].

C. Spatio-temporal Association Rules Mining

Spatio-temporal Association Rules Mining here refers to the extraction of implicit knowledge, spatial and temporal frequent patterns are not explicitly stored in spatiotemporal network data sets. Firstly, in this paper, spatial and temporal relationships exist among spatial network data sets at three levels: packet level, flow level and traffic level. The spatial relations, both metric (such as distance) and non-metric (such as topology, directions, shape, etc.), and temporal relations (such as before or after) may be explicit or implicit in the spatial network data sets. In both cases, such relationships are information bearing and therefore need to be considered in the mining techniques. Secondly, spatial and temporal dependency and heterogeneity are intrinsic characteristic of spatiotemporal data sets. Thirdly, scale effect in space and time is a challenging research issue in geographic analysis. Scale, in terms of spatial resolution or temporal granularity, can have direct impacts on the kinds and/or strength of relationships that can be identified in the datasets. All in all, the unique characteristics of spatiotemporal databases requires significant modification of many data mining techniques that are otherwise only good for non-geographic databases.

In this paper, spatio-temporal association rules mining (STARM) algorithm is proposed for analyzing multi-featured network traffic from multiple PoPs simultaneously. The STARM algorithm is shown in Algorithm 1. In step 2 we compute entropy of each flow-header features because that entropy-based approach provides more fine-grained insights than traffic volume analysis and entropy is compressed representation of huge amounts of data. Multiple flow-header features entropy time series are high dimensional data, making complex data analysis impractical. In step 3 we utilize independent component analysis to convert the multivariate time series into bivariate time series. That is to say, we project multiple dimensional data into 2-dimensional space, which includes normal space and abnormal space. In step 4, we use PAA and SAX to divide the values of each entropy time series at every time interval into alphabets. All symbolic sequences make up multivariate time series \mathcal{S} . Each column of \mathcal{S} denotes one PoPs, which represents a geography location. In step 5, we obtain spatio-temporal patterns of \mathcal{S} using STARM algorithm.

IV. EXPERIMENTAL RESULTS

A. Data Sets

The sampled flow data are collected from the backbone networks: Abilene [11]. Abilene is the Internet2 backbone network, connecting over 200 US universities and peering with research networks in Europe and Asia. It consists of 11 PoPs, spanning the continental US, which is shown in Fig. 1. Sampling is periodic (5 minutes), at a rate of 1 out of 100 packets. For privacy reasons, Abilene anonymizes the last 11 bits of IP address in flow records.

B. Experimental Results

The Netflow data of six PoPs were collected from 08:05 on December 17, 2006 to 08:00 on December 18,

```

Input :  $D$ , multivariate time series;
           $min\_sup$ , the minimum support count
          threshold.
Output :  $L$ , frequent itemsets in  $D$ .
Method:
1 for  $t = 1$  to  $N$  do
2   EntropyTS  $\leftarrow$  ComputeEntropy ( $D$ );
3   AbnormalTS  $\leftarrow$  FastICA (EntropyTS);
4   SymbolicTS  $\leftarrow$  SymRepresent
   (AbnormalTS);
5   AssociationRules  $\leftarrow$  Apriori (SymbolicTS)
6 end

```

Algorithm 1: STARM Algorithm

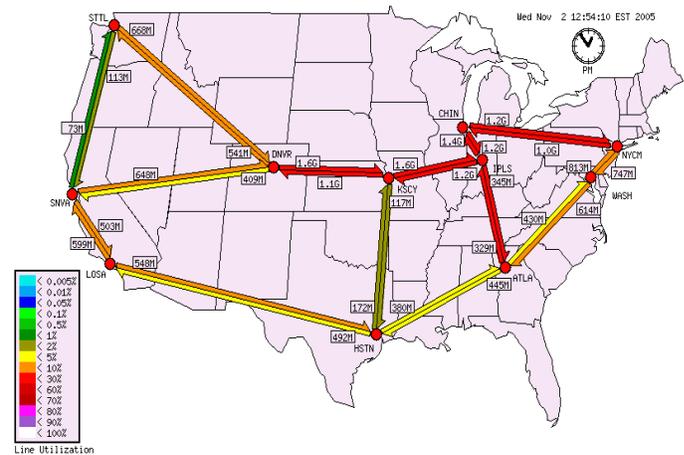


Fig. 1

INTERNET2 BACKBONE NETWORK

2006. Firstly, we compute entropy value of each PoP within each time bin and obtain six entropy time series: $\{R(DNVRng), R(IPLSng), R(KSCYng), R(LOSAng), R(SNVAng), R(STTLng)\}$. Secondly, we use PAA and SAX to divide the values of each entropy time series at every time interval into eight parts using eight alphabets $\{A, B, C, D, E, F, G, H\}$ which denote the eight parts of entropy value.

In order to diagnose network based on Netflow information, we should obtain the association rules of the existing anomalies. Firstly, we apply PAA and SAX to the entropy time series to obtain symbolic series. Then, we extract an anomalous symbolic time series segment

Let $R(DNVRng)=1, R(IPLSng)=2, R(KSCYng)=3, R(LOSAng)=4, R(SNVAng)=5, R(STTLng)=6$, from small to large, 'A' to 'H' denote the value. Then we apply association rules mining to alphabet sequence to get association rules of the anomaly pattern of multiple PoPs.

$1H, 2H, 4H, 5H, 6F \rightarrow 3H(Sup = 10, Conf = 100)$
 $1H, 2H, 3H, 4H, 6D \rightarrow 5H(Sup = 9, Conf = 100)$
 $1C, 3C, 4C, 5C, 6E \rightarrow 2C(Sup = 5, Conf = 100)$

TABLE II
THE TIME DISTRIBUTION OF FIVE RULES

Rule No.	Time Stamps
1	7,23,42,93,131,132,181,190,191,277
2	77,94,95,110,151,152,167,215,233
3	16,88,115,136,200
4	46,66,68,104,129
5	40,53,64

$1B, 2B, 4C, 5B, 6G \rightarrow 3B(Sup = 5, Conf = 100)$

$1A, 2A, 4C, 5C, 6A \rightarrow 3A(Sup = 3, Conf = 100)$

The rules mentioned above can be used to locate their time stamps. The time distribution of five rules are shown in Table II.

V. CONCLUSIONS

In this paper, firstly, we separate origin space into normal subspace and abnormal subspace using ICA method; secondly, we analyze collaborative behavior of multiple POPs using spatio-temporal association rules mining algorithm. From experimental results, we can arrive at the conclusion that our method can be used to analyze large-scale communication network traffic behavior. However we should be aware that there are still some problems demanding prompt solution in the field such as the efficiency and precision problem.

ACKNOWLEDGMENT

The authors would like to acknowledge the editors and reviewers for their suggestions and comments to improve the quality of the paper. This work was supported in part by the Natural Science Foundation of China under grant no. 60872033 and Major National Basic Research Development Program of China (973 Program, no. 2007CB307100).

REFERENCES

- [1] A. Hyvarinen, J. Karhunen, E. Oja. *Independent Component Analysis*, John and Sons Inc., 2001.
- [2] A. Cichocki, S. Amari. *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*, Wiley, 2003.
- [3] P. Comon. "Independent component analysis-a new concept?," *Signal Processing*, vol. 36, pp. 287-314, 1994.
- [4] R.Agrawal;T.Imielinski;A.Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, Jun. 1993.
- [5] C.Zhang; S.Zhang, *Association rule mining:models and algorithms*, Heidelberg: Springer-Verlag Berlin,, 2002.
- [6] Lin, J.; Keogh, E.; Lonardi, S.; Chiu., B.: Locally adaptive dimensionality reduction for indexing large time series databases. *Proceedings of the 8th ACM SIGMOD*

workshop on Research issues in data mining and knowledge discovery, San Diego, California, Jun. 2003.

- [7] Lin, J.; Keogh, E.; Patel, P.; S.Lonardi.: Finding motifs in time series. *Proceedings of the 2nd Workshop on Temporal Data Mining, at the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July. 2002.
- [8] J. Lin, E. Keogh, L. Wei, S. Lonardi: Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, Vol. 15, No. 2., pp. 107-144, 18 October 2007.
- [9] J. Lin, E. Keogh, S. Lonardi, and B. Chiu: A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. *Workshop on Research Issues in Data Mining and Knowledge Discovery, the 8th ACM SIGMOD*. San Diego, CA, 2003.
- [10] C. Shannon:A mathematical theory of communication. *Bell System Technical Journal*, 27:379-423, July 1948.
- [11] Abilene Netflow Data, <http://abilene.internet2.edu>.

Abnormal Process State Detection by Cluster Center Point Monitoring in BWR Nuclear Power Plant

J. Talonen, and M. Sirola

Department of Information and Computer Science, Helsinki University of Technology, Finland

Abstract—This paper proposes a new method to detect abnormal process state. The method is based on cluster center point monitoring in time and is demonstrated in its application to data from Olkiluoto nuclear power plant. Typical statistical features are extracted, mapped to n -dimensional space, and clustered online for every time step. The process signals in the constant time window are classified into two clusters by the K-means method. By monitoring features of the process signals, in addition to signal trends and alarm lists, the operator gains a tool that helps in early detection of the pre-stages of a process fault. By using cluster center point time series monitoring, faults in the process can be seen by at first glance or automatically by notification in the alarm list. This provides a definite advantage to any operating personnel and ultimately improves safety at the nuclear power plant.

Keywords: nuclear industry, abnormal process state detection, high dimensional data, feature extraction, classification.

1. Introduction

The goal of the process state detection method presented in this paper is to detect abnormalities in Olkiluoto boiling water reactor (BWR) type nuclear power plant (NPP) in Finland. At Olkiluoto, thousands of signals are measured and monitored. Because of the high dimensionality of the system, manual selection becomes arduous. When a large numbers of process signals exist, subsets of relevant variables are automatically selected for modeling [1], [2]. This paper, however, does not focus on the variable selection phase. It is assumed that variables can be selected from certain area of the plant or from all around the NPP. In other words, variable selection depends on the need to improve given monitoring in a certain area. The main emphasis of this paper is placed on the cluster center point movement monitoring of those variables.

An earlier study was conducted to investigate classification by principal component analysis (PCA) [2]. The sizes of the groups are same, and each object can be assigned to many groups. In this paper, signals are classified in two categories: *slow* (steady, inactive) and *fast* (quick, noisy). Therefore data is classified by the K-means method into two clusters for every time step of a constant frame size. The sizes of the groups are different, and each object is assigned only to one group. During a normal operation state,

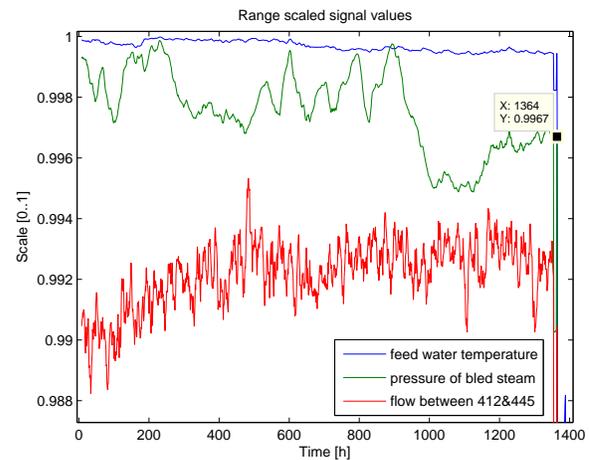


Fig. 1: The time series for three preprocessed process signals. The zero value is the global minimum (stored minimum value of the signal in the database) for the signal and one is the maximum. Most of the variables are near the global maximum value in the normal operation state but this naturally depends on the type of the variable.

most of the signals are classified as *slow*. In an abnormal process state process signals vary and are classified as *fast*. By monitoring features of the process signals the pre-stages of the process fault can be detected.

Within the Olkiluoto control room, there is an overload of alarms and notification which make it difficult for the operator to make discerning decision. Some sort of alarm sanitation is required [3]. The need for alarm handling is reduced if there are meaningful and clear statistics derived from process data [4], [5], [6]. For example: Hotelling's T^2 statistics can be used to detect faults for multivariate process data. This method is actually compared to our method in the section *simulation results*. Other monitoring systems based on noise analysis also already exist [7]. If a fault or its pre-stage is detected, large-scale systems like the Olkiluoto NPP can be improved significantly. In our newest research and in this paper, real data from reactor unit 2 of Olkiluoto NPP is used [8]. In 2007, more than 300 averaged signals were stored, every hour, over a two months period. During this time, an abnormal process state in the turbine section of the NPP was captured in the recorded data.

2. Description of used methods

The traditional way in industrial plants to monitor time series is with Shewhart charts and limit value checking [6]. Monitored signals have lower control limit (LCL) and upper control limit (UCL). With this method it is difficult to find the correct target value and limit values for each signal. One reason for this is that the industrial process generates many different types of signals. For example, Bergquist introduced 14 different signal classes [9]. These classes are periodic, slowly varying, multiple steady state, and containing outliers. Three different signals from Olkiluoto NPP are shown in the Figure 1.

Features from range scaled signals are measured for each time step. Scaling of the signals is important because without it, signal values or features cannot be compared between other signals [2], [10]. The results cannot be reliably clustered without preprocessing the signals, so the effect of white noise is eliminated by moving average (MA)

$$\bar{x}_t = \frac{1}{N_m} \sum_{k=0}^{N_m-1} x_{t-k}, \quad (1)$$

where x_t is a scaled measurement value and N_m is a *frame size* of the moving average. First, the $N_m - 1$ data points are removed from the beginning. The undesirable start effect is erased, which is acceptable because the data is stored continuously and in actuality no data is wasted.

The first feature in this application is the *absolute difference*, which is used to measure the rate of change

$$d_t = \frac{|\bar{x}_t - \bar{x}_{t-N_d}|}{N_d}, \quad (2)$$

where x_t is a preprocessed (scaled and averaged) measurement value and N_d is a *frame size* of the difference. Difference is high pass filter and it extracts changes in the signal. The Second feature is the moving standard deviation (MSDV). It is a common measure of statistical dispersion and it measures how widely the values are spread in time. If the data points are far from the mean, then the MSDV values are large. If all the data values are equal, then the MSDV value at the current time is zero. MSDV is derived from the *sample variance*

$$MSDV_t = \sqrt{\frac{1}{N_s - 1} \sum_{k=0}^{N_s-1} (x_{t-k} - \bar{x}_t)^2}, \quad (3)$$

where N_s is the *frame size* for MSDV, x_t is a scaled sample value and \bar{x}_t is the moving average [11].

In this paper only two statistical features are introduced, but there could be more such as skewness (measure of symmetry of a distribution) and kurtosis (measure of the peakedness or flatness of a distribution when compared with a normal distribution). Selected features depend on the goal of the classification.

K-means method is an unsupervised learning algorithm, which classifies a given data set through a certain number of k clusters [12]. The initial placement of the centroids are randomly defined for every time step, one for each cluster. Each object is assigned to the group that is closest to the centroid. When all objects have been assigned, the positions of the k centroids are recalculated. These steps are repeated until the centroids no longer move. The optimal solution is to minimize the cost-function

$$J = \sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - \mu_i\|^2, \quad (4)$$

where there are k clusters S_i , $i = 1, 2, \dots, k$ and μ_i is the centroid of all the points $x_j \in S_i$. With these features it is clear that the center point near the axes origin relates to the *slow* signals. The absolute value of the rate of change and the MSDV are small compared to the *fast* signals.

The operators in the control room are already overloaded with monitoring work, so a simple index, *unsteadiness*, is introduced. The idea is to monitor the cluster center point coordinates of the *slow* signals. It is more important to concentrate on the *slow* signals because there are remarkably large changes in some measurements in their normal operating state. These signals temporarily have high MSDV and difference values. These rapid but normal changes in process signal values increase the center point coordinates of *fast* signals. Examples of such events in NPP are: control flow from one pipeline to another, watering, and rapid changes in flow in a feed filter. The *unsteadiness* limit can be adjusted to produce an automatic alarm. The MA of the cluster center point of the *fast* signals is measured and the current alarm limit its minimum value.

Our method is compared to the Hotelling's T^2 statistics which is a measure of the variation within the PCA model.

$$T^2 = (\mathbf{H} - \bar{\mathbf{H}})^T \mathbf{S}^{-1} (\mathbf{H} - \bar{\mathbf{H}}), \quad (5)$$

where \mathbf{H} is the score matrix, $\bar{\mathbf{H}}$ and \mathbf{S} are the common estimators for the mean vector and covariance matrix obtained from the scores [13]. The scores \mathbf{H} are the preprocessed data mapped into the new coordinate system defined by the principal components.

Process abnormality is detected with the help of Hotelling's T^2 , which defines the normal operating area corresponding to 95% confidence. The upper control limit (UCL) of the multivariate Hotelling's T^2 statistics can be defined

$$T_{UCL}^2 = \frac{(n-1)(n+1)k}{n(n-k)} F_{\alpha}(k, n-k), \quad (6)$$

where $F_{\alpha}(k, n-k)$ is the upper critical point of the F -distribution with k and $n-k$ degrees of freedom. In practice k is the amount of selected variables and n is the number of measurements. [6]

3. Simulation Results

All variables are range scaled by the database minimum and maximum values because it provides better classification results. In our research the database was constructed offline. It was aggregated by 40 design based and abnormal stored data sets such as watering and scram situations. These data sets are provided by Teollisuuden Voima Oy. The frame size used for the MA, MSDV, and *absolute value of the rate of change* was eight hours.

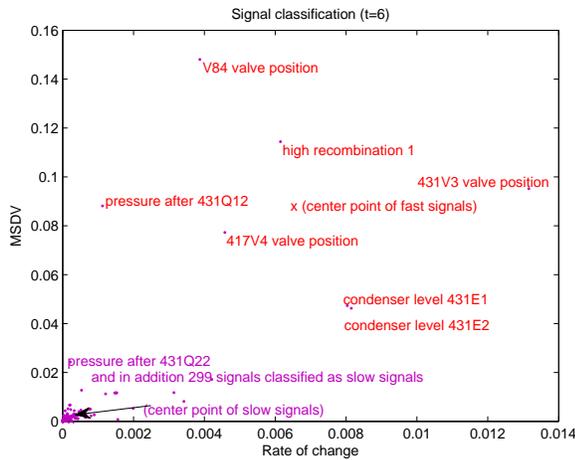


Fig. 2: An example of the clustering result at $t = 6$. Variables located near the axes origin are classified as *slow* signals. Variables near the other cluster center point (X) are classified as *fast* signals.

Signal measurements were selected all around the NPP, because it was decided that common safety improvements were a priority. Features are measured for 307 variables at each time step. The *unsteadiness* in this case monitors the general process state in the NPP. Feature values of each variable, the classification result, and the center points can be illustrated and updated in selected time period (every hour or minute), see Figure 2. This can be useful for expert users but not for operators in the control room.

In Figure 3 the deviation (the center point of MSDVs) for the *slow* and the *fast* signals is shown. The automatic alarm is based on these values. Deviation of the *slow* signals increased and NPP is not stable after $t = 1361$. Because of the limit was exceeded, the *unsteadiness* notification is shown in the display of alarms at the control room. In this visualization, high MSDV values for the *fast* signal can be seen at $t = 961$. It is a normal operation state and the high measurement values are caused by rapid changes in feed filters flows and temperatures. Few variables change the MSDVs and they only have a minor effect to the center point of *slow* signals.

Figures 4 and 5 show time series which can be displayed in the control room. These are actually not mandatory

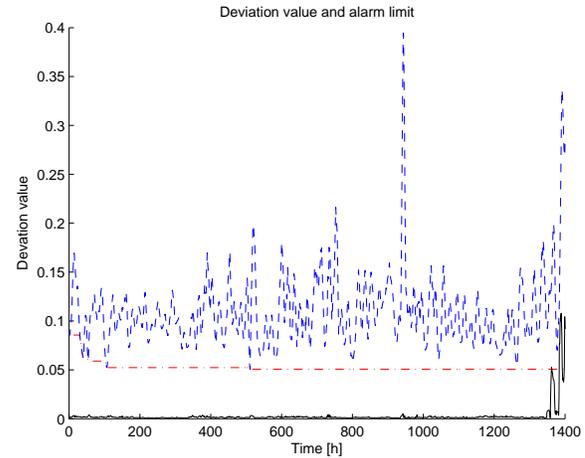


Fig. 3: Deviation values for the *slow* signals (solid line) and the alarm limit (dash dotted line). It is the minimum value of the moving average of deviation for the *fast* signals (dashed line) until current time.

because a notification system using the *unsteadiness* index works without operator monitoring. Of course important information is aggregated to the limit line as in the case of the center point coordinates of the *fast* signals. After the alarm notification, the limit can be re-settled. In this case the notification of *unsteadiness* is given at $t = 1359$ because of the high absolute difference values and at $t = 1361$ because of the high deviation values.

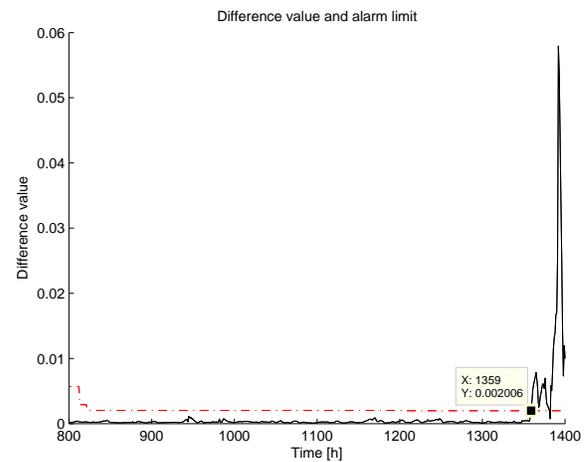


Fig. 4: The absolute difference value of the *slow* signals (solid line) and the minimum value of the *fast* signals cluster center points (dash dotted line).

The simulation results are satisfying, because most of the signals get normal values until $t = 1364$, see Figure 1. The method presented here detects a process fault three hours

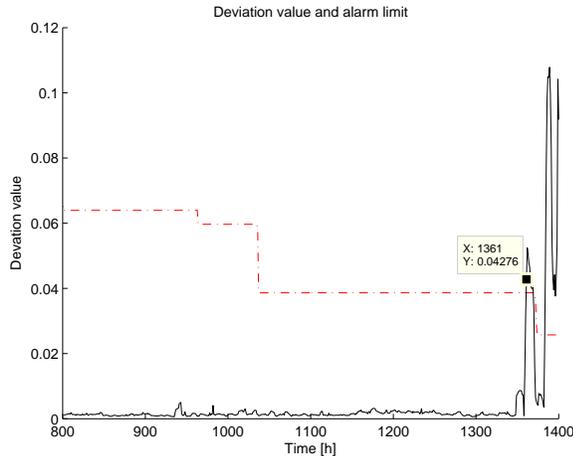


Fig. 5: MSDV of *slow* signals (solid line) and the minimum value of the *fast* signals cluster center points (dash dotted line).

earlier than Hotelling's T^2 statistics, see Figure 6. There are some false alarms: at the time of rapid changes in feed filters flows and temperatures, $t = 961$, and when a gas flow was controlled from one pipeline to another at $t = 1175$.

PCA was performed offline all time series data with same preprocessing parameters. Complicated matrix inversions are usually a problem in multivariate methods. Therefore an online implementation of PCA would be a problem especially when the size of moving time window is small. Because of the feature extraction and data mapping into two dimensional space, these problems do not exist in method presented in this paper.

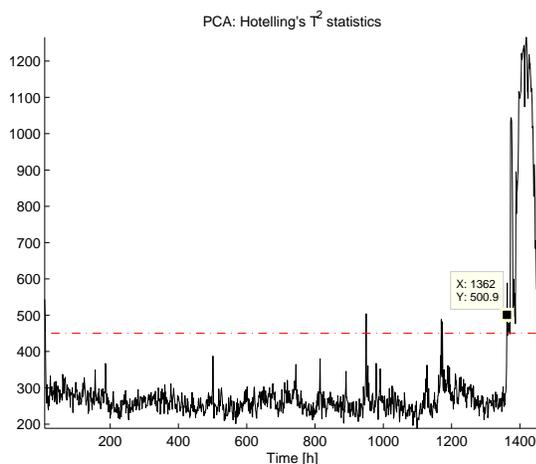


Fig. 6: Hotelling's T^2 statistics (solid line) and the upper control limit (UCL) as an alarm limit (dash dotted line).

4. Conclusion

The method presented in this paper can be used for pre-stage detection of process faults through the introduction of an unsteadiness index. The implementation of this method requires little investment in terms of increased process complexity while offering a solid advantage in terms of pre-stage fault detection capability. Through early detection, NPP will ultimately enjoy reduced maintenance costs, decreased downtimes, and increased safety.

One possible limitation for the method presented here is that the NPP has rather strict rules for the automation system changes. Also computation time can be a problem if data is analyzed too frequently. Future research will concentrate on developing methods for maintenance personnel. There is also a demand for the development of calibration monitoring technology because more than 90 percent of current calibration efforts are unnecessary. Calibration can also be harmful to reactor equipment and even to plant safety [14].

References

- [1] S. Laine. *Using visualization, variable selection and feature extraction to learn from industrial data*. PhD thesis, Helsinki University of Technology, 2003.
- [2] J. Talonen, M. Sirola, and J. Parviainen. Leakage Detection by Adaptive Process Modeling. In R. Stahlbock, S.F. Crone, and S. Lessmann, editors, *Proceedings of The 2008 International Conference on Data Mining (DMIN 2008)*, volume I, II, July 2008.
- [3] J.E. Larsson. Simple Methods for Alarm Sanitation. In *Proceedings of the IFAC Symposium on Artificial Intelligence in Real-Time Control, Budapest, 2000*.
- [4] J. Talonen. Fault Detection by Adaptive Process Modeling for Nuclear Power Plant. Master's thesis, Helsinki University of Technology, 2007.
- [5] J. Chen and K.C. Liu. On-line batch process monitoring using dynamic PCA and dynamic PLS models. *Chemical Engineering Science*, 57(1):63–75, 2002.
- [6] L.H. Chiang, E. Russell, and R.D. Braatz. *Fault Detection and Diagnosis in Industrial Systems*. Springer, 2001.
- [7] J. Ortiz-Villafuerte, R. Castillo-Durán, G. Alonso, and G. Calleros-Micheland. BWR online monitoring system based on noise analysis. *Nuclear Engineering and Design*, 236(22):2394–2404, 2006.
- [8] M. Sirola, J. Talonen, and G. Lampi. SOM based methods in early fault detection of nuclear industry. In *Proceedings of the 17th European Symposium On Artificial Neural Networks ESANN'09*, April 2009.
- [9] T. Bergquist, J. Ahnlund, and JE Larsson. Alarm reduction in industrial process control. In *Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA'03. IEEE Conference*, volume 2, 2003.
- [10] D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufmann, 1999.
- [11] J.F. Hair, R.E. Anderson, R.L. Tatham, and W.C. Black. *Multivariate Data Analysis*. Prentice Hall, 5th edition, 1998.
- [12] S. Theodoridis. *Pattern Recognition*. Academic Press, 2003.
- [13] L.I. Tong, C.H. Wang, and C.L. Huang. Monitoring defects in IC fabrication using a Hotelling T^2 control chart. *Semiconductor Manufacturing, IEEE Transactions on*, 18(1):140–147, 2005.
- [14] H.M. Hashemian. History of On-Line Calibration Monitoring Developments in Nuclear Power Plants. 27th-29th of September 2004. Halden.

SESSION
DATA PRE-PROCESSING

Chair(s)

Dr. Gary M. Weiss

Fused Multi-modal Deduplication

Sabra Dinerstein¹, Christophe Giraud-Carrier¹, Jared Dinerstein², and Parris K. Egbert¹

¹Computer Science Department, Brigham Young University, Provo, UT, USA

²Computer Science Department, Utah State University, Logan, UT, USA

Abstract—*Biometric systems are composed of both biometric (e.g., fingerprint, face, iris) and biographic (e.g., name, age, birth date) data. Accurate identification requires that we have data for every identity that we wish to recognize. Thus, an identification data set can be quite large and is often created from disparate data sources. Unfortunately, such data sets often contain inexact duplicate representations of the same person. The existence of duplicate records negatively impacts both identification speed and accuracy. No adequate solution exists for deduplicating records that contain both biometric and biographic data.*

We propose a novel technique for deduplicating biometric databases. We perform deduplication separately on each type of data: applying rule-based matching to biographic text, and generating biometric match scores using traditional biometric match algorithms. We combine these individual deduplication results via information fusion. We show empirically that our fused deduplication technique yields higher average accuracy than either biometric or biographic deduplication alone.

Keywords: record deduplication, multi-modal systems, data fusion, biometric systems

1. Introduction

An ever-growing number of real-world identification systems incorporate both biometric and biographic data: matching is performed on the biometric data and upon finding a match, both the biometric image and the associated biographic data are reported to the user.

Biometric matching typically involves the comparison of image data. Unfortunately, biometric image data is inherently inexact, and therefore duplicates that are often difficult to detect are common in real-world scenarios.

The presence of duplicates (particularly inexact duplicates) can significantly reduce system performance for two key reasons. First, duplicates increase the number of records that must be tested in order to perform accurate identification. Given that even a single comparison can require a nontrivial amount of time, especially when comparing biometric data, this increased number of necessary tests has a significant impact on system speed. Second, identification results can be ambiguous when a match corresponds to a record that contains only partial information.

Data deduplication, also known as *record linkage* or *record matching*, is the process of identifying records that

correspond to the same real-world entity [11] [14] [19] [26]. Unfortunately, while deduplication would be very beneficial for biometric systems, little work has been done to address this problem. Moreover, traditional text-based approaches to deduplication are suboptimal for a biometric database because text-based deduplication ignores the biometric image data that is contained in a record. A better deduplication technique for a biometric database would exploit both the biographic and the biometric data contained in a record.

We propose a novel learning-based fusion approach for deduplicating records that contain both biographic and biometric data. We demonstrate the benefits of our technique by performing deduplication separately on biographic text data (such as surname, given name, gender, and affiliations) generated with Febrl [7], and biometric face images from the FERET data set [12]. We perform match score-level fusion [16] on these individual deduplication results, using an appropriately trained Support Vector Machine (SVM).

In [9], we presented a fusion technique for biographic text deduplication. Our previous deduplication technique replaces manual rule tuning [11] with learning-based fusion, thereby providing accurate rule-based text deduplication without requiring manual tuning. However, the main limitation of our previous fused deduplication technique is that it only provides deduplication for biographic text records, and does not support biometric deduplication. Thus, just as with other text-only deduplication techniques, our previous deduplication technique is suboptimal for biometric systems.

In this paper, we present a fused deduplication technique for multi-modal records—records that contain both biometric and biographic data. By supporting the combination of biometric and biographic data, the technique that we present in this paper provides a more complete deduplication solution for biometric systems. We show empirically that our fused multi-modal deduplication technique yields higher average accuracy than the deduplication of either biometric or biographic data alone.

2. Related Work

The challenge of data deduplication has existed for decades in domains as varied as medical records, census data and genealogical information. Data records often contain no unique (and consistent) identifier. Thus, data deduplication is complicated by variations in the record data itself. Variations in record data can be caused by a number of

factors, including: user entry error, typographical mistakes, intentional misinformation, multiple enrollment, and missing or unknown data. Further, the use of data from disparate sources can introduce significant differences in schema and notation, which compounds the problem of resolving duplicate entries [8] [11].

Several approaches are currently available for matching text data records, as detailed in [11]. Character-based matching techniques act very locally, on single characters or short character sequences [26]; these techniques employ metrics such as edit distance [17] and affine gap [23]. Q-grams extend character-based similarity metrics to short q-length sequences of characters [24]. Larger-scale similarity information can be measured via token-based techniques, which often use *atomic strings* [18] and *tf.idf* calculations [8]. Text-based deduplication is further aided by the use of phonetic similarity measures such as Soundex [21].

Rule-based similarity measurements take advantage of expert domain knowledge [25], and allow the deduplication efforts to be tuned to the current data set. The accuracy of rule-based deduplication is directly dependent on the set of rules that are used, and often requires many iterations of tuning, including the selection of appropriate thresholds [11].

Learning-based techniques have recently been applied to text deduplication. Active learning [22], genetic programming [4], and Expectation-Maximization [3] have each been employed to learn specialized distance functions that yield good text-based deduplication results. Additionally, fusion decision trees [15] and fusion SVMs [2] [3] have been used to combine the individual match results of all text fields contained in a data record.

Learning-based fusion (often implemented via an SVM or a Bayesian network) has been shown to yield high average match accuracy in biometric systems (when combining only biometric data) [10] [16] [20]. This information fusion is typically performed at either the feature-level, the output decision-level, or the intermediate match score-level [20]. Match score-level fusion (i.e., combining real-valued match scores) has been shown to be applicable to systems where the features of the individual modalities cannot be directly combined [16] [20]. Thus, match score-level fusion is particularly suitable for combining such disparate modalities as biometric and biographic data, as we cannot expect the features of these individual modalities to be related.

These previous efforts lend credence to our application of learning-based fusion to biometric deduplication. However, to date, and to the best of our knowledge, no one has utilized learning-based fusion to perform *multi-modal* deduplication of records containing both biometric and biographic data. Previous deduplication efforts have focused on pure data sets: *only* biometric data [10] [16] [20] or *only* biographic text data [2] [3] [9] [11]. Instead, we present a multi-modal deduplication technique that exploits both the biometric and biographic data contained in a typical biometric system.

```

<PersonNameType>
  <GivenName> Robert </GivenName>
  <SurName> Jones </SurName>
  <SoundexText> R163 J520 </SoundexText>
</PersonNameType>
<BiometricType>
  <DescriptionText> Face </DescriptionText>
  <TestMethodText> Local Binary Patterns </TestMethodText>
  <Binary> <! -- binary image data -- > </Binary>
</BiometricType>

```

Fig. 1: *GJXDM XML fragment*. GJXDM is a standard U.S. government schema that contains biographic elements (e.g., surname and given name) and biometric elements (e.g., modality and binary image data).

3. Deduplication Technique

In this paper, we present a novel technique for biometric deduplication, a largely unexplored problem. First, in parallel, we perform (1) expert rule-based text- and phonetic-similarity matching on biographic text, and (2) biometric matching on face images from the FERET data set [12]. Then, we apply match score-level fusion [16] to these individual biographic and biometric match scores.

Our fused deduplication technique can be applied to any multi-modal deduplication problem domain. Here, we demonstrate our fused multi-modal deduplication technique on records that comply with the Global Justice XML Data Model (GJXDM) schema [13] (a sample of which is shown in Figure 1). This schema is a standard for the storage and communication of biographic data (e.g., given name, surname, date of birth, and affiliations) and biometric data (e.g., fingerprint, face, and iris images) between U.S. government agencies.

Government databases (such as state and national census databases) are frequently very large and contain many inexact duplicates [26]. Deduplication of such databases is a significant real-world problem for which there is no adequate, complete solution; thus, our contributions include the creation of an accurate deduplication system for biometric systems that support the GJXDM schema.

We present our fused deduplication technique in more detail in the following sections.

3.1 Biographic Text Deduplication

Our fused multi-modal deduplication technique can be used with any biographic text deduplication system, provided that the text deduplication system outputs a real-valued match score. Here, we demonstrate our fused deduplication technique using our two-stage, rule-based text-only deduplication system, originally presented in [9]. (Note that our previous work [9] differs from the fused deduplication technique presented in this paper because our previous work deduplicates only text records. Our deduplication technique presented here, however, supports multi-modal records, and therefore represents a more flexible and complete deduplication technique.)

Our rule-based text-only deduplication system consists of two main pieces:

- a set of computationally inexpensive *primary rules*, and
- a set of more expensive *secondary rules*.

Figure 2 gives a pseudo-code description of our two-stage, rule-based deduplication technique.

3.1.1 Primary Rules

Our primary deduplication rules perform *blocking* (i.e., *search space reduction*) for our secondary deduplication rules. As described in Figure 2, the primary rules produce an initial set of probable matches, which we denote the *primary candidate set*. The secondary rules are applied only to this primary candidate set.

To deduplicate GJXDM text, we use four primary rules (summarized in Figures 3 - 6). These primary rules are applied in decreasing order of importance, where Primary Rule 1 is the strictest and most important rule. A *strict rule* relies on *strongly identifying* fields, such as passport and visa numbers. A *lenient rule* is comprised of more ambiguous fields, such as surname and given name. Thus, the stricter primary rules tend to produce a smaller primary candidate set than do the lenient rules. In order to maintain a False Reject Rate (FRR) $\leq 1\%$, we employ a distinctly lenient rule, Primary Rule 4 (summarized in Figure 6), which phonetically compares proper names [21].

We apply our deduplication rules in a short-circuited fashion: if a primary rule returns any candidates, the secondary rules are immediately applied. Subsequent primary rules are only applied if the secondary rules do not return any matches.

3.1.2 Secondary Rules

Our secondary rules (shown in Figures 7 - 9), produce the final deduplication results: match scores are calculated, and any duplicate records are flagged.

```

TwoStageDeduplication( p ) { // p is a biographic probe record

    primaryCandidateSet = PrimaryRule1( p )

    if( primaryCandidateSet is not empty )
        matches = SecondaryRules( p, primaryCandidateSet )
        if( matches.bestScore > 0 )
            return matches

    primaryCandidateSet = PrimaryRule2( p )

    if( primaryCandidateSet is not empty )
        matches = SecondaryRules( p, primaryCandidateSet )
        if( matches.bestScore > 0 )
            return matches

    ...
}

```

Fig. 2: *Two-stage, rule-based deduplication*. The primary rules (applied in decreasing order of importance) perform blocking for the secondary rules, which calculate the record match scores.

```

PrimaryRule1( p ) { // p is a biographic probe record
    create empty candidateList

    for each g in the gallery {
        if( p.passportID == g.passportID, first 3 chars )
            if( p.passportCountry == g.passportCountry, first 3 chars )
                if( p.visaID == g.visaID, first 3 chars )
                    if( p.visaType == g.visaType, first 3 chars )
                        add g to candidateList
    }

    return candidateList
}

```

Fig. 3: *Primary Rule 1*. Primary Rule 1 is the strictest of our four primary rules, and relies on strongly identifying fields such as *passportID*.

```

PrimaryRule2( p ) { // p is a biographic probe record
    create empty candidateList

    for each g in the gallery {
        if( p.passportID == g.passportID, first 3 chars )
            if( p.gender == g.gender )
                if( p.nationality == g.nationality, first 3 chars )
                    if( p.dateOfBirth exists and g.dateOfBirth exists )
                        add g to candidateList
    }

    return candidateList
}

```

Fig. 4: *Primary Rule 2*. This rule is applied only if Primary Rule 1 does not return any candidates. More lenient than Primary Rule 1, Primary Rule 2 does not use as many strongly identifying fields.

The secondary rules utilize identifying fields, or *clues*, such as *passportID*. We weight each clue by the discriminatory power of its underlying field(s). We measure clue similarity via the Levenshtein edit distance [17]: if the edit distance is within the specified *allowable edit distance* then the weight of that clue is added to the record match score. Only those record match scores that meet the rule-level threshold are returned as matches. These thresholds are based on the discriminatory power of the clues that are included in the rule: if a rule contains ambiguous fields, we allow a larger edit distance and apply a looser rule-level threshold.

```

PrimaryRule3( p ) { // p is a biographic probe record
    create empty candidateList

    for each g in the gallery {
        if( p.gender == g.gender )
            if( p.nationality == g.nationality, first 3 chars )
                if( p.surname == g.surname, first char )
                    if( p.givenName == g.givenName, first char )
                        if( p.orgName == g.orgName, first char )
                            add g to candidateList
    }

    return candidateList
}

```

Fig. 5: *Primary Rule 3*. This rule is even more lenient than the previous primary rules: for most fields, we compare only the first character.

```

PrimaryRule4( p ) { //p is a biographic probe record
  create empty candidateList

  for each g in the gallery {
    if( Soundex(p.surname, g.surname) ≥ 0.75 ) {
      if( Soundex(p.givenName, g.givenName) ≥ 0.75 )
        add g to candidateList
    }
  }

  return candidateList
}

```

Fig. 6: *Primary Rule 4*. This rule uses Soundex to phonetically compare proper names, which are known to be ambiguous fields.

```

SecondaryRule1( p, c ) { //p is a probe, c is a candidate match
  matchScore = 0

  if( EditDistance(p.passportID, c.passportID) ≤ 1 )
    matchScore += 10 //clue weight
  if( EditDistance(p.passportCountry, c.passportCountry) ≤ 1 )
    matchScore += 5
  if( EditDistance(p.visaID, c.visaID) ≤ 1 )
    matchScore += 8
  if( EditDistance(p.visaType, c.visaType) ≤ 1 )
    matchScore += 2

  if( matchScore ≥ 20 ) //rule-level threshold
    return (matchScore/25)
  else
    return 0
}

```

Fig. 7: *Secondary Rule 1*. Our secondary rules calculate the match score of two records, using clues such as *passportID*. We weight each clue by its discriminatory power, and return only those match scores that meet the rule-level threshold.

3.2 Biometric Deduplication

Our fused multi-modal deduplication technique can be applied to any biometric modality (e.g., face, fingerprint, or iris). Additionally, our deduplication technique can utilize any biometric matcher, as long as that matcher outputs a real-valued match score.

Here, we demonstrate the effectiveness of our technique by implementing a regional face matcher, based on Local Binary Patterns (LBP) [1]. We apply our LBP matcher to face images from the FERET data set [12], which we embed in our example GJXDM records. (See Figure 1 for an example GJXDM record.)

3.2.1 LBP Code Creation

LBP is an image matching technique that encodes greyscale pixel neighborhood information in an *LBP code*—one LBP code per pixel. Each pixel in an image (except for the boundary pixels) is surrounded by 8 neighboring pixels, as shown in Figure 10. In its simplest form, this pixel neighborhood is a 3x3 square. However, LBP can also employ a circular neighborhood, and use interpolation to determine the location and greyscale value of each neighbor. In our experiments, we constructed an LBP code for each (8, 2) circular neighborhood (i.e., 8 neighbors at a radius of

```

SecondaryRule2( p, c ) { //p is a probe, c is a candidate match
  matchScore = 0

  if( EditDistance(p.passportID, c.passportID) ≤ 1 )
    matchScore += 8 //clue weight
  if( p.gender == c.gender )
    matchScore += 3
  if( EditDistance(p.surname, c.surname) ≤ 2 )
    matchScore += 6
  if( EditDistance(p.nationality, c.nationality) ≤ 1 )
    matchScore += 3
  if( EditDistance(p.dateOfBirth, c.dateOfBirth) ≤ 1 )
    matchScore += 3

  if( matchScore ≥ 17 ) //rule-level threshold
    return (matchScore/23)
  else
    return 0
}

```

Fig. 8: *Secondary Rule 2*. The selection (and total number) of clues in each rule is based on the discriminatory power of each clue.

```

SecondaryRule3( p, c ) { //p is a probe, c is a candidate match
  matchScore = 0

  if( p.gender == c.gender )
    matchScore += 3 //clue weight
  if( EditDistance(p.nationality, c.nationality) ≤ 1 )
    matchScore += 3
  if( EditDistance(p.surname, c.surname) ≤ 2 )
    matchScore += 7
  if( EditDistance(p.givenName, c.givenName) ≤ 2 )
    matchScore += 5
  if( EditDistance(p.orgName, c.orgName) ≤ 2 )
    matchScore += 8

  if( matchScore ≥ 20 ) //rule-level threshold
    return (matchScore/26)
  else
    return 0
}

```

Fig. 9: *Secondary Rule 3*. This rule contains ambiguous clues, such as proper names; thus we apply looser clue thresholds.

2 pixels from the center pixel).

After selecting the pixel neighborhood, we construct a per-pixel LBP code by comparing the greyscale value of the current pixel to each of its neighbors:

$$\forall_{n \in N}. LBP_n = \begin{cases} 1 & \text{if } greyscale(n) \geq greyscale(c) \\ 0 & \text{otherwise} \end{cases}$$

Here, c is the current center pixel; n is a pixel in c 's surrounding neighborhood, N ; and $greyscale(n)$ is the greyscale value of the neighbor pixel, n . (In our experiments, each neighborhood, N , includes 8 neighbors at a radius of 2 pixels from c .) An LBP bit value of 1 indicates that the neighbor's greyscale value is greater than or equal to that of the center pixel, while a value of 0 indicates that the neighbor's greyscale value is less than that of the center pixel.

We concatenate these binary values (in a clockwise direction) to form a single 8-bit LBP code for the center pixel. See Figure 10 for an example of the creation of an LBP code.

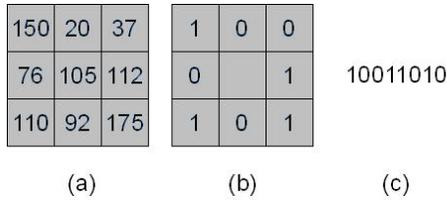


Fig. 10: Example 8-bit LBP code. (a) 3x3 pixel neighborhood. (b) We compare the greyscale value of the center pixel to each of its neighbors: if the neighbor's greyscale value is greater than or equal to that of the center pixel, we assign a value of 1 to that bit; otherwise, we assign a value of 0. (c) We concatenate these binary values to form a single 8-bit LBP code for the center pixel.

3.2.2 Regional Face Matching

Our LBP face matcher compares face images in *regions*, as shown in Figure 11. Regional face matchers have been shown to be less sensitive to slight variations in *image registration* (i.e., the alignment and size of salient image features, such as the eyes) [1]. The size of each image region has a direct impact on the sensitivity to image registration and on the match score: larger blocks are less sensitive to slight differences in image registration, while smaller blocks are useful when comparing the fine details of an image. In our experiments, we used uniform 4x4 pixel regions.

We store the LBP codes for each image region in a histogram that records the incidence count of each LBP code within the current region. Each regional histogram contains 256 slots, representing all possible variations of an 8-bit LBP code.

We calculate the match score of two images by comparing their collocated regional histograms. This histogram comparison can be implemented via any reasonable similarity metric; in our experiments, we use a histogram intersection. The histogram intersection of two regions can be calculated as $\sum_{s=0}^{255} \min(p_s, q_s)$, where s is a histogram slot, and p and q are the two regional histograms to compare—the larger the histogram intersection, the greater the similarity between the two image regions. To calculate the similarity of two entire face images, we sum the histogram intersection of the collocated regions in the two images.

We apply our LBP face matcher to registered, cropped



Fig. 11: Regional face matching. To compare two face images, we create an LBP histogram for each image region, and measure the histogram intersection of the collocated image regions.

face images from the FERET data set [12], such as the face image shown in Figure 11. Employing the standard FERET probe and gallery sets, our LBP face matcher achieved an average match accuracy of 91.7%.

3.3 Fused Multi-modal Deduplication

In the previous sections, we have described the two individual pieces of our multi-modal technique: (1) biographic text deduplication, and (2) biometric image deduplication. Now we describe how our novel multi-modal deduplication technique uses both the biographic and biometric data that is contained in a standard biometric system:

- In parallel, we calculate the *single modality* record match scores:
 - We perform rule-based matching on the biographic text data, using our deduplication rules (described in Section 3.1). Our biographic deduplication rules return a set of biographic matches and the corresponding match scores.
 - We perform biometric matching, using our regional LBP face matcher (described in Section 3.2). Our LBP face matcher returns a set of biometric matches and the corresponding match scores.
- Next, we perform fused *multi-modal* deduplication, using an appropriately trained SVM: our fusion SVM takes the single modality match scores as input and outputs one overall classification decision, indicating *match/no match*.

3.3.1 Training the Fusion SVM

First, we create *single modality* match score examples (representing either biographic or biometric matches), by separately applying our rule-based deduplication to the biographic text data and our LBP face matcher to the biometric image data. We scale these single modality match scores to be in the range $[-1, 1]$, where a value of 1 represents a perfect match [5].

Next, we concatenate the *single modality* match score examples of the same class (i.e., *match/no match*) to create *multi-modal* match score examples of the form:

(Classification of the *multi-modal* example,
Scaled *biographic text* match score,
Scaled *biometric image* match score).

Finally, we train an SVM to fuse the single modality match scores, and output one overall classification decision:

$$SVM: g \times m \rightarrow \{match/no\ match\}$$

where g is the real-valued biographic text match score, and m is the real-valued biometric image match score.

We select the training examples randomly, with replacement, from the set of multi-modal match score examples. Testing examples are selected in the same manner. We explicitly disallow crossover between the training and testing

examples: no example may be used for both training and testing. To account for the variability in example selection, we perform 10 runs of training and testing.

4. Empirical Results

4.1 SVM Implementation and Parameters

We implemented our fusion SVM via LIBSVM [5]. LIBSVM is a library that supports SVMs for both classification and regression, and provides multiple types of kernels.

In our experiments, our fusion SVM employs a Radial Basis Function (RBF) kernel: $e^{-\gamma(|u-v|^2)}$. We choose the appropriate γ -value and constraints-violation cost, C , at runtime by performing k -fold (stratified) cross-validation on the current set of training examples: we keep the γ - and C -values that produce the best cross-validation accuracy (on the current training set).

We perform a grid search to determine the best γ - and C -values for the current training set: we search for γ in the range $[2^{-15}, 2^3]$, using a step size of 4; we search for C in the range of $[1, 2^{10}]$, with a step size of 4.

4.2 Data Sets

Privacy is a serious concern when using biometric data sets. Similar to many real-world problem domains that contain identifying information, complete real-world biometric record data is not publicly available [6]. Thus, for our experiments, we generated data sets that conform to the GJXDM schema [13]: we used Febrl [7] to create biographic text data, and we incorporated face images from the standard FERET data set [12].

The (frontal pose) FERET face data set contains over 3000 images of 1000 different subjects. In our experiments, we used the standard FERET gallery (f_a) and probe (f_b) sets. To create appropriate multi-modal examples, we used Febrl to generate 2000 duplicates of 1000 unique text records. Febrl is a record linkage and data set generation tool [7] that creates both *original* and *duplicate* text records, containing a user-specified set of biographic text fields.

For our experiments, we used Febrl to generate the biographic text portion of our GJXDM records (see Table 1), and then added face images from the FERET data set to those records. To create gallery records, we used the original Febrl text records and added face images from the FERET gallery set, f_a . To create probe records, we used the duplicate Febrl text records and added face images from the FERET probe set, f_b .

4.3 Results

Figure 12 describes the match accuracy of our fused multi-modal deduplication technique. The horizontal axis of Figure 12 indicates the average similarity (between the original and duplicate records) of each Febrl-generated biographic text data set: we calculate the average similarity of each

Modifications per Record	Modifications per Field	Data Set Similarity
1	1	96.58%
3	1	90.07%
6	1	81.68%
10	1	72.16%
50	10	62.72%

Table 1: *Febrl-generated data sets*. For each biographic data set, we specified the maximum number of modifications per field and per record, and calculated the average data set similarity using the average edit distance between each original Febrl record and its corresponding duplicate records.

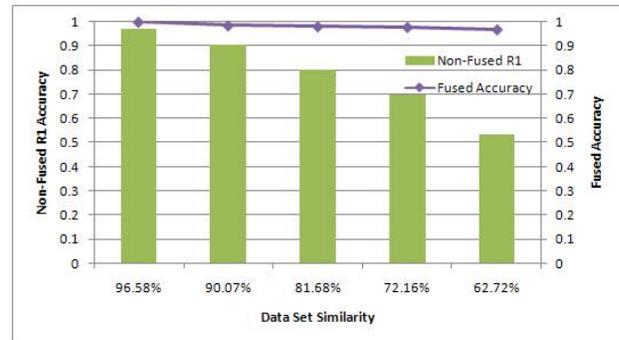


Fig. 12: *Average deduplication accuracy*. Our fused multi-modal deduplication technique maintains high average accuracy, even when there is significant noise in the data set. The fused accuracy shown here was achieved with 200 training examples and is averaged over 10 runs.

biographic data set, using the average edit distance between each original Febrl record and its corresponding duplicate records. (As summarized in Table 1, we generated biographic text data with different amounts of randomness, varying both the number of modifications per field and the total number of modifications per record.)

The fused multi-modal deduplication accuracy shown in Figure 12 was achieved with 200 training examples and 3500 testing examples, and is averaged over 10 runs. We varied the number of fusion training examples from 100 to 3500 and found that even with only 100 training examples, our fused deduplication technique achieves high average accuracy; when using more training examples, the average deduplication accuracy increased only slightly. This behavior is likely affected by the data used in our experiments—other data sets (such as a face data set that contains challenging poses) might require additional training examples. (The fused multi-modal deduplication accuracy values shown in Figure 12 are based on the 91.7% average accuracy of our LBP face matcher on the FERET set.)

As summarized in Figure 12, for each of the data sets in our experiments, our fused multi-modal deduplication technique yields higher average accuracy than either biometric or biographic deduplication alone. Figure 12 also shows the Rank 1 match accuracy of (non-fused) biographic

text deduplication on these data sets, where Rank 1 match accuracy represents the percentage of comparisons where the highest-score match is an actual match. As can be seen in Figure 12, as the similarity of the generated data sets decreases, the accuracy of the (non-fused) biographic deduplication decreases; the accuracy of our fused multi-modal deduplication technique, however, remains high. This implies that our fusion SVM has learned an appropriate decision threshold for the overall system and for the individual modalities.

The (non-fused) biographic text deduplication accuracy shown in Figure 12 reflects the fact that we purposefully refrained from manually tuning the biographic deduplication rules. These results imply that fused multi-modal deduplication reduces the amount of manual tuning that is required for rule-based deduplication [11], which corroborates our previous fused biographic text-only deduplication result [9]. Further, our multi-modal deduplication achieves higher average accuracy (99.60%) than our previous text-only deduplication (96.69%) [9], on the same set of GJXDM records (with an FRR of 0.67%). Note that our individual biographic text deduplication rules share some text fields. Due to this overlap in the original text deduplication rules, we expect the fusion of the text-only rules to yield slightly lower match accuracy than the fusion of independent modalities, such as biographic text data and biometric image data [20].

Additionally, our experiments include a commonly used biometric modality (face) that is known to be inexact. We specifically include face images in our experiments, to demonstrate the efficacy of our fused multi-modal deduplication technique for real-world biometric systems. As shown in Figure 12, our fused multi-modal deduplication technique achieves high average accuracy, even with imperfect individual matchers.

5. Conclusions

We have presented a novel fused multi-modal deduplication technique, and have demonstrated the benefits of our technique on GJXDM records [13] that contain both biographic text and biometric image data. Our fused multi-modal deduplication technique achieves higher average accuracy than either biographic or biometric deduplication alone. Further, our fused multi-modal deduplication technique reduces the need for manual tuning of the biographic deduplication rules. Additionally, our technique produces high average accuracy, even when using non-ideal biometric modalities, such as face images. We have proven in the conducted empirical study, using known data sets, that our fused multi-modal deduplication technique is effective for deduplicating biometric systems.

References

- [1] T. Ahonen, A. Hadid, and M. Pietikainen. Face Description with Local Binary Patterns: Application to Face Recognition. *Pattern*

- Analysis and Machine Intelligence, IEEE Transactions on*, 28:2037–2041, 2006.
- [2] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. *Intelligent Systems, IEEE*, 18:16–23, 2003.
- [3] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of KDD'03*, pages 39–48, New York, NY, USA, 2003. ACM.
- [4] M. G. Carvalho, A. H. F. Laender, M. A. Gonçalves, and A. S. da Silva. Replica identification using genetic programming. In *Proceedings of SAC'08*, pages 1801–1806, New York, NY, USA, 2008. ACM.
- [5] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [6] P. Christen. Probabilistic data generation for deduplication and data linkage. In *IDEALS'05, Springer LNCS 3578*, pages 109–116. Springer LNCS, 2005.
- [7] P. Christen. Febrl: an open source data cleaning, deduplication and record linkage system with a graphical user interface. In *Proceedings of KDD'08*, pages 1065–1068, New York, NY, USA, 2008. ACM.
- [8] W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. *SIGMOD Rec.*, 27(2):201–212, 1998.
- [9] J. Dinerstein, S. Dinerstein, P. K. Egbert, and S. W. Clyde. Learning-based Fusion for Data Deduplication. In *Proceedings of ICMLA'08*, pages 66–71. IEEE Computer Society, 2008.
- [10] S. Dinerstein, J. Dinerstein, and D. Ventura. Robust Multi-Modal Biometric Fusion via Multiple SVMs. In *Proceedings of SMC'07*, pages 1530–1535. IEEE Computer Society, 2007.
- [11] A. Elmagarmid, P. Ipeirotis, and V. Verykios. Duplicate record detection: A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 19:1–16, 2007.
- [12] The FERET Database. <http://www.itl.nist.gov/iad/humanid/feret/>.
- [13] GJXDM. Global Justice XML Data Model. <http://www.it.ojp.gov/jxdm/>.
- [14] K. Goiser and P. Christen. Towards Automated Record Linkage. In *Proceedings of AusDM'06*, volume 61 of *CRPIT*, pages 23–31, Sydney, Australia, 2006. ACS.
- [15] S. Ivie, B. Pixton, and C. Giraud-Carrier. Metric-Based Data Mining Model for Genealogical Record Linkage. In *Proceedings of IRI'07*, pages 538–543. IEEE Systems, Man, and Cybernetics Society, 2007.
- [16] A. Jain and A. Ross. Multibiometric systems. *Commun. ACM*, 47:34–40, 2004.
- [17] V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.
- [18] A. Monge and C. Elkan. The Field Matching Problem: Algorithms and Applications. In *Proceedings of KDD'96*, pages 267–270, 1996.
- [19] H. Newcombe and J. Kennedy. Record linkage: making maximum use of the discriminating power of identifying information. *Commun. ACM*, 5(11):563–566, 1962.
- [20] A. Ross and A. Jain. Information fusion in biometrics. *Pattern Recogn. Lett.*, 24:2115–2125, 2003.
- [21] R. Russell. United states patent: 1261167, Apr. 1918.
- [22] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proceedings of KDD'02*, pages 269–278, New York, NY, USA, 2002. ACM.
- [23] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, March 1981.
- [24] E. Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theor. Comput. Sci.*, 92:191–211, 1992.
- [25] Y. Wang and S. Madnick. The Inter-Database Instance Identification Problem in Integrating Autonomous Systems. In *Proceedings of ICDE'89*, pages 46–55, Washington, DC, USA, 1989. IEEE Computer Society.
- [26] W. Winkler. The state of record linkage and current research problems. Technical Report RR/1999/04, Statistics Research Division, U.S. Bureau of the Census, 1999.

A Combinatorial Fusion Method for Feature Construction

Ye Tian¹, Gary M. Weiss², D. Frank Hsu³, and Qiang Ma⁴

¹Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

^{2,3}Department of Computer and Information Science, Fordham University, Bronx, NY, USA

⁴Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Abstract - *This paper demonstrates how methods borrowed from information fusion can improve the performance of a classifier by constructing (i.e., fusing) new features that are combinations of existing numeric features. The new features are constructed by mapping the numeric values for each feature to a rank and then averaging these ranks. The quality of the fused features is measured with respect to how well they classify minority-class examples, which makes this method especially effective for dealing with data sets that exhibit class imbalance. This paper evaluates our combinatorial feature fusion method on ten data sets, using three learning methods. The results indicate that our method can be quite effective in improving classifier performance.*

Keywords: Feature construction, class imbalance, information fusion

1 Introduction

The performance of a classification algorithm is highly dependent on the descriptions associated with the example. For this reason, good practitioners will choose the features used to describe the data very carefully. However, deciding which information to encode and how to encode it is quite difficult and the best way to do so depends not only on the domain, but on the learning method. For this reason, there have been a variety of attempts over the years to automate part of this process. This work has had a variety of names over the years (although sometimes the emphasis is different) and has been called constructive induction [13], feature engineering [17], feature construction [6] and feature mining [11]. In this paper we discuss how existing numerical features can be combined, without human effort, in order to improve classification performance.

The work described in this paper is notable for several reasons. First, unlike the majority of work in this area, we are specifically concerned with improving the performance of data with substantial class imbalance. Such problems are challenging but quite common and are typical in domains such as medical diagnosis [7], fraud detection [4], and failure prediction [19]. Furthermore, there are reasons to believe that this important class of problems has the most to benefit from feature construction, since some learners may not be able to detect subtle patterns that only become apparent when several features are examined together [18]. Our work also differs from other work in that our feature combination operator does not directly use the values of the component features but

rather their ranks. This allows us to combine numerical features in a meaningful way, without worrying about issues such as scaling. This approach is particularly appropriate given the increased interest in the use of ranking in the data mining [10] and machine learning communities [5]. Our approach also can be viewed as an extension of work from the information fusion community, since techniques similar to the ones we use in this paper have been used to “fuse” information from disparate sources [9]. The work in this paper can be viewed as a specific type of information fusion, which we refer to as feature fusion.

We describe our combinatorial feature-fusion method in detail in Section 2 and then describe our experiments in Section 3. The results from these experiments are described and analyzed in Section 4. Related work is discussed in Section 5. Our main conclusions and areas for future work are then described in Section 6.

2 Combinatorial Feature Fusion

This section describes the basic combinatorial feature-fusion method. We introduce relevant terminology and describe some of the basic steps employed by the feature fusion method. We then describe some general schemes for fusing features and end the section with a detailed description of our combinatorial feature fusion algorithm.

2.1 Terminology and Basic Steps

In this section we will use a simple example to explain the relevant terminology and preliminary steps related to feature fusion. This example will also be used later in this Section to help explain the feature-fusion algorithm. Because our feature-fusion method only works with numeric features, for simplicity we assume all features are numeric. Non-numeric features are not a problem in practice—they simply will be passed, unaltered, to the classifier.

A data set is made up of examples, or records, each of which has a fixed number of features. Consistent with previous work on information fusion [9, 10] we view the value of a feature as a *score*. Typical examples of scores are a person’s salary, a student’s exam score, and a baseball pitcher’s earned run average. In the first two cases a higher score is desirable but in the last case a lower one is preferable.

Table 1 introduces a sample data set with eight examples, labeled A-H, with five numeric features, F1-F5, and a binary class variable. In this example class 1 is the minority class and comprises 3/8 or 37.5% of the examples.

TABLE 1
A SAMPLE DATASET

	F1	F2	F3	F4	F5	Class
A	1	4	3	2	8	1
B	3	3	5	5	4	0
C	5	5	2	6	7	1
D	7	6	15	3	2	0
E	11	13	16	7	14	0
F	15	16	4	13	11	0
G	9	7	14	1	18	1
H	17	15	9	8	3	0

Early in our combinatorial feature-fusion method we replace each score with a rank, where a lower rank is better. We convert each score into a rank using a *rank function*, which adheres to the standard notion of a rank. We sort the score values for each feature in either increasing or decreasing order and then assign the rank based on this ordering. Table 2 shows the values of the features for the sample data set after the scores have been replaced by ranks, where the ranks were assigned after sorting the feature values in increasing order. As a specific example, because the three lowest values for F3 in Table 1 are 2, 3, 4 and these values appear in rows C, A, and F, respectively, the ranks in Table 2 for F3 for these records are 1, 2, and 3, respectively.

TABLE 2
SAMPLE DATASET WITH SCORES REPLACED BY RANKS

	F1	F2	F3	F4	F5
A	1	2	2	2	5
B	2	1	4	4	3
C	3	3	1	5	4
D	4	4	7	3	1
E	6	6	8	6	7
F	7	8	3	8	6
G	5	5	6	1	8
H	8	7	5	7	2

We determine whether the ranks should be assigned based on increasing or decreasing order of the score values by determining the performance of the feature using both ordering schemes and selecting the ordering that yields the best performance (we describe how to compute a feature’s performance shortly). In our method, once the scores are replaced with a rank the scores are never used again. The rank values are used when combining features and are the features values that are passed to the learning algorithm.

Next we show how to compute the “performance” of a feature. This performance metric essentially measures how well the rank of the feature correlates with the minority-class examples. That is, for a feature, do the examples with a good rank tend to belong to the minority class? We explain how to compute this performance metric using feature F2 from the sample data set. First we sort the records in the data set by the rank value of F2. The results are shown in Table 3. The performance of F2 is then computed as the fraction of the records at the “top” of the table that belong to the minority class. The

number of “top” records that we examine is based on the percentage of minority-class examples in the training data. In this case 3 of 8 of the training examples (37.5%) belong to the minority class so we look at the top 3 records. In this example that means that the performance of F2 is 2/3, since two of the three class values for these records is a “1”, which is the minority-class value. Given this scheme, the best performance value that is achievable is 1.0.

TABLE 3
RANKED LIST FOR F2

	F2 Rank	Class
B	1	0
A	2	1
C	3	1
D	4	0
G	5	1
E	6	0
H	7	0
F	8	0

We may similarly compute the performances for all of the individual features. Table 4 shows that for this simple example F1–F4 all have performances of 2/3 and F5 has a performance of 0.

TABLE 4
PERFORMANCE VALUES FOR ORIGINAL FEATURES

Feature	Performance
F1	0.67
F2	0.67
F3	0.67
F4	0.67
F5	0.00

This method is also used to compute the performance of the “fused” features. To do this we need to first determine the rank of a fused feature, so we can sort the examples by this rank. We compute this using a *rank combination function* that averages the ranks of the features to be combined. This is done for each record. As an example, suppose we want to fuse features F1–F5 and create a new feature, F1F2F3F4F5, which we will call F6. Table 5 shows the rank values for F6 for all eight records. The value for F6 for record A is computed as: $(\text{Rank}(F1) + \text{Rank}(F2) + \text{Rank}(F3) + \text{Rank}(F4) + \text{Rank}(F5))/5 = (1+2+2+2+5)/5 = 2.4$. We see that for this new feature, record “A” has the best (lowest) rank. Given these values, one can now compute the performance of the feature F6. Note that even though the values in Table 5 are not integers we can still consider them ranks. In order to compute the performance of F6, we only need to be able to sort by these values.

TABLE 5
RANK VALUES FOR F6 (F1F2F3F4F5)

	F6		F6
A	2.4	E	6.6
B	2.8	F	6.4
C	3.2	G	5.0
D	3.8	H	5.8

2.2 Combinatorial Fusion Strategies

The previous section introduced the terminology and basic steps required by our combinatorial fusion algorithm, but did not discuss how we decide *which* features to fuse. We discuss that topic in this section.

There are many possible strategies for choosing features to “fuse.” In this paper we consider combinatorial strategies that look at all possible combinations or more restrictive variants that look at subsets of these combinations. Let n equal the number of numeric features available for combination. To look at all possible combinations would require that we try each single feature, all pairs of features, all triples, etc. The total number of combinations therefore equals $C(n,1) + C(n,2) + \dots + C(n,n)$, which equals $2^n - 1$. We refer to such a combinatorial fusion strategy as a fully-exhaustive fusion strategy.

We consider more restrictive variants of the fully-exhaustive fusion strategy because, depending on the value of n , this strategy may not be practical. The k -exhaustive fusion strategy will create all possible combinations using k of the n ($k < n$) numeric features. For example, a 6-exhaustive strategy for a data set with 20 numeric features will select 6 features and then fuse them in all possible ways, reducing the number of feature combinations by a factor of 2^{14} . In our algorithm we choose the subset of k features based on the performance values for the features, such as the ones in Table 6. Because it will not be expensive to include all of the original features, we always include the $n - k$ original features. The 6-exhaustive fusion strategy is one of the three strategies analyzed in this paper.

The k -exhaustive fusion strategy trades off a reduced number of features for the ability to fully combine these features. In some cases it may be better to involve more features in the fusion process, even if they cannot be fused in all possible ways. The k -fusion strategy will use all n numeric features, but the length of the fused features is limited to length at most k . Thus if we have a data set with 20 numeric features and employ 2-fusion, all possible combinations of single features and pairs of features will be generated. This would yield $C(20,1) + C(20,2) = 20 + 190 = 210$ features. Similarly, 3-fusion would consider $C(20,1) + C(20,2) + C(20,3)$, or 1140 feature combinations.

Table 6 shows the number of features generated by the different fusion strategies. In all cases, as stated before, all original features are included. Some cells are empty since $k \leq n$. If $k = n$ then the value computed is displayed in bold and corresponds to the fully-exhaustive strategy. Table 6 demonstrates that, given a limit on the number of features we can evaluate, we have a choice of fusion strategies. For example, given ten numeric features, one can use all ten features and generate combinations of length four, which would generate 385 features, or instead select the seven best ones and then fuse those in all possible ways (i.e., up to length 7), which would generate about 127 features (actually 130 when the three original features are included).

TABLE 6
COMBINATORIAL FUSION TABLE

Number Features	k-fusion for values of k shown below									
	1	2	3	4	5	6	7	8	9	10
1	1									
2	2	3								
3	3	6	7							
4	4	10	14	15						
5	5	15	25	30	31					
6	6	21	41	56	62	63				
7	7	28	63	98	119	126	127			
8	8	36	92	162	218	246	254	255		
9	9	45	129	255	381	465	501	510	511	
10	10	55	175	385	637	847	967	1012	1022	1023

2.3 The Combinatorial Fusion Algorithm

We now describe the algorithm for performing the combinatorial fusion. This algorithm is summarized in Table 7. We explain this algorithm by working through an example based on the data set introduced in Table 1.

For this example, we will use the 5-exhaustive strategy, so that we select the five best performing features and then fuse them in all possible ways. On line 1 of the algorithm we pass into the *Comb-Fusion* function the data, the features, a k value of 5 and a value of True for the Exhaustive flag. The next few steps were already described in Section 2.1. We convert the scores to ranks (line 3) and then calculate the performance of the original (unfused) features in the loop from lines 4-6. Then in lines 7-11 we determine which features are available for fusion. Since the Exhaustive flag is set, we restrict ourselves to the k best features (otherwise all features are available although they then may not be fused in all possible ways).

TABLE 7
THE FEATURE-FUSION ALGORITHM

```

1. Function Comb-Fusion (Data, Features, k, Exhaustive)
2. {
3.   ConvertScoresToRanks(Data, Features);
4.   for (f=1, f ≤ length(Features), f++){
5.     Perf[f]=CalculatePerformance(f);
6.   }
7.   if (Exhaustive == TRUE) {
8.     FeaturesForFusion = best k features from Perf[];
9.   } else {
10.    FeaturesForFusion = Features;
11.  }
12.  New = FuseFeatures(FeaturesForFusion, k, Exhaustive);
13.  for (f=1, f ≤ length(New), f++){
14.    CalculateRank(f);
15.    Perf2[f]=CalculatePerformance(f);
16.  }
17.  Sort(Perf2);
18.  Candidates = Perf2.features;
19.  // We now build up the final feature set
20.  Keep = Features; // always use original features
21.  partition(Data, *TrainValid, Test);
22.  for (f in Candidates)
23.  {
24.    for (run=1; run ≤ 10, run++)

```

```

25.  {
26.    partition(TrainValid, *Training, *Validation);
27.    classifier = build-classifier(Training, Keep);
28.    PerfWithout[run] = evaluate(classifier, Validation);
29.    cand = pop(Candidates);
30.    classifier=build-classifier(Training, Keep ∪ cand);
31.    PerfWith[run] = evaluate(classifier, Validation);
32.  }
33.  if ( average(PerfWith[ ]) > average(PerfWithout[ ]) )
34.  {
35.    pval = t-test(PerfWith[], PerfWithout[]);
36.    if (pval ≤ .10) {
37.      Keep = Keep ∪ cand;
38.    }
39.  }
40. } // end for (f in Candidates)
41. final-classifier = build-classifier(Training, Keep);
42. final-performance = evaluate(Test, Keep);
43. } // end Function Comb-Fusion

```

The actual generation of the fused features occurs on line 12. In this case, the five best features in *FeaturesForFusion* will be combined in all possible ways (in this example there are only five features to begin with). Given our decision to always include the original features to the classifier, the original features need not be returned by *FuseFeatures* (they are handled later on line 20).

Next, on lines 13-16 we calculate the rank for each fused feature and then calculate their performance. This is essentially the same steps that were done earlier for the original features. We then sort the features by decreasing performance value (line 17) and extract the features from this sorted list and save them (line 18) in *Candidates*, the ordered list of candidate fused features. The results for the best 14 performing fused features for our simple example are shown in Table 8. In this case *Candidates* equals {F3F4, F1F2, F1F3, ...}.

TABLE 8
PERFORMANCE VALUES FOR 5-EXHAUSTIVE STRATEGY

Priority	Feature	Perf.	Priority	Feature	Perf.
1	F3F4	1	8	F1F2F4	0.67
2	F1F2	0.67	9	F1F3F4	0.67
3	F1F3	0.67	10	F1F3F5	0.67
4	F2F3	0.67	11	F2F3F4	0.67
5	F2F4	0.67	12	F3F4F5	0.67
6	F3F5	0.67	13	F1F2F3F4	0.67
7	F1F2F3	0.67	14	F1F2F3F5	0.67

In the second half of the algorithm, starting at line 19, we decide which of the *Candidate* features to include in the final feature set. We begin by initializing *Keep* to the set of original features. We then partition the data (line 21) into one set to be used for training and validation and another for testing. Beginning on line 22 we iterate over all of the fused features in the *Candidate* set.

A key question is how we determine when to add a feature. Even though a feature has a good performance score, it may not be useful. For example, the information encoded in the feature may be redundant with the features already included

in the feature set. We adopt a pragmatic approach and only add a feature if it improves classifier performance on the validation set and the improvement is statistically significant. To determine this, within this main loop in the second half of the algorithm (lines 22 – 40) we execute ten runs (lines 24 – 32), repeatedly partitioning the training data into a training set and a validation set (line 26). If, averaged over the 10 runs (line 33) the classifier generated with the candidate feature (line 30) outperforms the classifier generated without it (line 28) and the p-value returned by the t-test (line 35) is $\leq .10$ (line 36), then we add the feature to *Keep* (line 37). A p-value $\leq .10$ means that we are 90% confident that the observed improvement reflects a true improvement in performance. In steps 41 and 42 we build the final classifier and evaluate it on the test set.

We should point out a few things. First, the actual implementation is more efficient in that we only need to build one classifier in the main loop, since the classifier from the previous iteration, and its performance, is still available. Similarly, we do not need to rebuild the classifier as indicated on line 41. Also, the performance of the classifier can be measured using either AUC or accuracy, and we use both measures in our experiments.

Table 9 shows the behavior of our simple example as each feature is considered. We only show the performance for the first 3 features. The last column indicates the feature being considered and a “+” indicates that it is added while the lack of this symbol indicates that it is not added because the conditions on lines 33 and 36 are not both satisfied. Each row corresponds to an iteration of the main loop starting at line 22 in the algorithm. The first row is based on the classifier built from the original feature set, containing features F1-F5. Note that the first and third features that are considered are added, because they show an improvement in AUC and the p-value is $\leq .10$. As we add features we also measure the performance of each classifier on the test set, although this is not used in any of the decision making. The AUC for the test set at the end is reported, however. If we stopped the algorithm after the three iterations, we can conclude that the performance improved from an AUC of .682 to .774. It is of course critical not to use the test set results to determine whether to add a feature (and we do not).

TABLE 9
THE EXECUTION OF THE ALGORITHM ON A SIMPLE EXAMPLE

AUC		p-value	Feature (+ means added)
valid	test		
0.670	0.682	--	{F1,F2,F3,F4,F5}
0.766	0.757	0.001	+F3F4
0.731			F1F2
0.771	0.774	0.063	+F1F3

3 Description of Experiments

In this section we describe the datasets employed in our empirical study, the three learning methods that are utilized, and the methodology we use to conduct our experiments.

Table 10 describes the ten data sets used in our study. Note that the data sets are ordered in terms of decreasing class imbalance. The data sets come from several sources. The hepatitis, bands, income and letter-a data sets were obtained from the UCI machine learning repository [14], the crx data set was provided in the Data directory that came with the C4.5 code, the physics and bio data sets are from the 2004 KDD CUP challenge, the stock data set was provided by New York University's Stern School of Business, and the boa1 data set was obtained from researchers at AT&T.

TABLE 10
THE DATA SETS

Dataset Name	% Minority Class	Number Features	Dataset Size
protein+	0.59	14	20,000
letter-a*	3.9	15	20,000
income*+	5.9	12	10,000
stock*+	9.9	27	7,112
hepatitis*	19.8	12	500
physics+	24.9	8	20,000
german*	30.0	19	1,000
crx*+	44.1	5	450
bands*+	42.2	13	538
boa1+	49.8	25	5,000

In order to simplify the presentation and the analysis of our results, data sets with more than two classes were mapped to two-class problems. This was accomplished by designating one of the original classes, typically the least frequently occurring class, as the minority class and then mapping the remaining classes into the majority class. The data sets that originally contained more than two classes are identified with an asterisk (*). The letter-a data set was generated from the letter-recognition data set by making the letter "a" the minority class. Because we are only employing feature fusion for the numeric features, we deleted any non-numeric features from the data sets. While this is not necessary, since our method could just ignore the non-numeric fields, we did this so that we could better determine the impact of the feature fusion method. The data sets that had any non-numeric features are identified with a "+".

All of the learning methods that we use in this paper come from the WEKA data mining software [12]. The three learning methods that we use are Naïve Bayes, decision trees and 1-nearest neighbor. The decision tree algorithm is called J48 in WEKA and is an implementation of the C4.5 algorithm. The 1-nearest neighbor algorithm is referred to as IB1 in WEKA.

The experiments in our study apply a combinatorial feature-fusion strategy to each of the ten data sets listed in Table 10 and then record the performance with and without the fusion strategy. This performance is measured in terms of the area under the ROC curve (AUC), because ROC analysis [3] is a more appropriate performance metric than accuracy when

there is class imbalance. Nonetheless we repeat some of our experiments with accuracy as the performance metric, since doing so is quite straightforward and accuracy is a very commonly used performance metric. The three combinatorial fusion strategies that are evaluated are the 2-fusion, 3-fusion and 6-exhaustive fusion strategies described in Section 2. In this study we utilize the three learning algorithms listed in Section 3 in order to see how the feature-fusion method benefits each algorithm. In the algorithm in Table 7 the data is partitioned such that 50% is used for training, 20% for validation, and 30% for testing.

4 Results

In this section we describe our main results. Because we are interested in improving classifier performance on data sets with class imbalance, and because of the known deficiencies with accuracy as a performance metric [16], we use AUC as our main performance measure. These AUC results are summarized in Table 11. The results are presented for ten data sets using the Naïve Bayes, decision tree, and 1-NN learning methods. Three combinatorial fusion strategies are evaluated: 2-Fusion (2F), 3-fusion (3F) and 6-Exhaustive (6EX). The AUC results are presented first without (w/o) and then with (w) the combinatorial fusion strategy. The "diff" column shows the absolute *improvement* in AUC resulting from the combinatorial fusion strategy, with negative values indicating that combinatorial fusion degraded the performance.

TABLE 11
AUC IMPROVEMENT WITH COMBINATORIAL FUSION

Dataset	Strat.	Bayes			Decision Trees			1-NN		
		w/o	w	Diff	w/o	w	Diff	w/o	w	Diff
bio	2F	.923	.923	-.020	.752	.752	-.256	.663	.663	-.164
	3F	.943	.954	.010	.496	.742	.247	.499	.651	.152
	6EX	.926	.926	-.017	.759	.759	-.264	.663	.663	-.164
letter-a	2F	.963	.963	.000	.943	.943	-.021	.961	.961	-.024
	3F	.963	.960	-.003	.922	.919	-.003	.937	.937	.000
	6EX	.962	.962	-.001	.937	.937	-.014	.961	.961	-.024
income	2F	.901	.901	.000	.736	.736	.241	.612	.612	.020
	3F	.901	.897	-.004	.494	.734	.239	.593	.621	.028
	6EX	.900	.900	-.001	.739	.739	.245	.612	.612	.020
stock	2F	.762	.762	.037	.755	.755	.260	.575	.575	.051
	3F	.725	.767	.043	.496	.751	.255	.524	.578	.054
	6EX	.769	.769	.044	.747	.747	.252	.564	.564	.040
hepatitis	2F	.869	.869	.005	.755	.755	.000	.803	.803	-.016
	3F	.864	.868	.004	.755	.759	.004	.819	.826	.007
	6EX	.864	.864	.000	.760	.760	.005	.821	.821	.002
physics	2F	.498	.498	.000	.499	.499	.000	.504	.504	.000
	3F	.498	.506	.008	.499	.499	.000	.504	.495	-.008
	6EX	.506	.506	.008	.499	.499	.000	.504	.504	.000
german	2F	.751	.751	.011	.609	.609	.118	.607	.607	-.001
	3F	.740	.723	-.017	.492	.606	.115	.609	.593	-.015
	6EX	.736	.736	-.004	.654	.654	.162	.609	.609	.000
crx	2F	.762	.762	.000	.646	.646	.000	.653	.653	.014
	3F	.762	.779	.017	.646	.670	.024	.639	.673	.034
	6EX	.755	.755	-.007	.722	.722	.076	.667	.667	.028
bands	2F	.779	.779	.029	.611	.611	.108	.559	.559	-.096
	3F	.750	.747	-.003	.504	.603	.099	.655	.644	-.012
	6EX	.744	.744	-.006	.580	.580	.076	.655	.655	.000
boa1	2F	.596	.596	.024	.538	.538	.041	.509	.509	-.005
	3F	.571	.602	.031	.497	.548	.050	.515	.509	-.006
	6EX	.589	.589	.018	.553	.553	.056	.509	.509	-.005

The results in Table 11 indicate that the combinatorial feature fusion method is effective and most effective for the decision tree learning method. The overall impact of the methods is shown in Table 12, which summarizes the results for each combinatorial fusion strategy and learning method, over the ten data sets. It displays the average absolute improvement in AUC as well as the win-lose-draw (W-L-D) record over the 10 data sets.

TABLE 12
SUMMARIZED AUC RESULTS FOR TEN DATA SETS

Strategy	Bayes		DT		1-NN	
	AUC	W-L-D	AUC	W-L-D	AUC	W-L-D
2-fusion	0.009	5-1-4	0.105	7-0-3	0.016	5-4-1
3-fusion	0.009	6-4-0	0.103	8-1-1	0.023	5-4-1
6-exhaustive	0.003	3-6-1	0.115	9-0-1	0.027	6-1-3

The results from both tables indicate that decision trees benefit most from combinatorial fusion, with the one-nearest neighbor learning method also showing substantial improvement. We believe that the decision tree algorithm improves the most because without combinatorial fusion it is incapable of learning combinations of numeric features, since decision trees only examine a single feature at a time.

The results do not demonstrate that any of the three combinatorial feature-fusion strategies is a clear winner over the other two. The 6-exhaustive strategy performs best for decision trees and one-nearest neighbor, but performs worst for naïve Bayes. The results for the 2-fusion and 3-fusion strategies are comparable even though the 3-fusion strategy generates more combinations. Our detailed results indicate that with the 3-fusion method some “3-fused” features make it to the final feature set, but apparently these are not essential for good performance. The fact that the 2-fusion strategy performs competitively indicates that most of the benefits that one can achieve with our combination operator can be achieved by combining only two features.

We generated Table 13 to determine if the combinatorial feature fusion method is more effective for the four most skewed data sets, where less than 10% of the data belongs to the minority class. These results, when compared to Table 13, show that the combinatorial fusion method yields substantially greater benefits when evaluated on the most highly unbalanced data sets, when the decision tree and one-nearest neighbor methods are used (the results for Bayes are much less convincing). Because of the limited number of datasets analyzed, these results cannot be considered conclusive, but nonetheless are quite suggestive.

TABLE 13
SUMMARIZED AUC RESULTS FOR FOUR SKEWED DATA SETS

Strategy	Bayes		DT		1-NN	
	AUC	W-L-D	AUC	W-L-D	AUC	W-L-D
2-fusion	0.004	1-1-2	0.195	4-0-0	0.065	4-0-0
3-fusion	0.012	2-2-0	0.185	3-1-0	0.059	3-0-1
6-exhaustive	0.006	1-3-0	0.194	4-0-0	0.062	4-0-0

It makes some sense that our method is most beneficial for highly unbalanced data sets. Given the performance measure described in Section 2, which is based on the correlation between the fused features and the minority-class examples, we expect to generate features that are useful for classifying minority-class examples. Furthermore, it is often quite difficult to identify “rare cases” in data and algorithms that look at multiple features in parallel are more likely to find the subtle classification rules that might otherwise get overlooked [18].

Although our primary interest is in improving classifier performance with respect to the area under the ROC curve, our method can be used to improve accuracy as well. We repeated a subset of our experiments using accuracy instead of AUC when determining whether adding a fused feature improves the performance with the required level of statistical confidence. Table 14 provides these results when using the 2-fusion strategy. We did not repeat these experiments for the other two strategies because AUC is our primary measure of interest and because the three strategies appear to perform similarly.

TABLE 14
SUMMARIZED AUC RESULTS FOR FOUR SKEWED DATA SETS

Dataset	Bayes			Decision Trees			1-NN		
	w/o	w	Diff	w/o	w	Diff	w/o	w	Diff
bio	98.8	98.8	0.0	99.4	99.4	0.0	99.2	99.2	0.0
letter-a	98.4	98.4	0.0	98.6	98.6	0.0	98.9	98.9	0.0
income	92.0	92.0	0.0	94.5	94.5	0.0	92.4	92.4	0.0
stock	80.4	80.4	0.0	90.3	90.3	0.0	86.3	86.3	0.0
hepatitis	84.0	84.0	0.0	86.2	80.7	-5.6	89.3	89.3	0.0
physics	68.9	75.3	6.5	75.1	75.2	0.1	62.6	75.0	12.4
german	73.2	73.2	0.0	69.5	73.0	3.5	68.1	71.6	3.5
crx	70.1	70.1	0.0	60.3	75.1	14.9	60.4	73.6	13.2
bands	67.0	67.0	0.0	61.4	61.4	0.0	65.3	65.3	0.0
boa1	55.0	57.0	2.0	51.0	56.9	6.0	52.6	57.5	5.0

The results in Table 14 indicate that our combinatorial fusion method is also effective for accuracy. While many of the data sets show no improvement, in ten cases there was an increase in accuracy, while in only one case was there a decrease in accuracy. In virtually every case where the accuracy remains the same, the combinatorial fusion strategy did not add any fused features. Similar to what we saw for AUC, the naïve Bayes method shows the least improvement.

5 Related Work

There has been a significant amount of work on feature mining/feature construction and in this section we mention some representative work. We organize the work in this area based on the operator used to combine the features. In our work, for example, numeric features are combined by mapping their feature values to ranks and then averaging the values of these ranks.

One approach is to assume that the features represent Boolean values and then use the standard logical operators to combine the features [2]. Other methods, such as the X-of-N

method [20] differ in some ways but can be used to implement most logical operators. These logic-based methods require that all features first be mapped into Boolean values. This is not necessarily difficult but loses information and can lead to other problems. For example, in a decision tree repeatedly partitioning a numeric feature into binary values can lead to data fragmentation. In contrast our method reduces this problem by combining multiple numeric features.

Other methods are much more ambitious in the operators they implement. Some systems implement multiple mathematical operators, such as $+$, $-$, \times , and \div , and relational operators such as \leq and \geq [1] [15]. Because these systems provide a rich set of operators, it is not feasible for them to try all possible combinations and thus they tend to employ complex heuristics. Thus our method has the advantage of simplicity. Again, a key difference is that our method combines ranks, whereas this other work combines the scores.

Feature selection [8], which involves determining which features are useful and should be kept for learning, is often mentioned in the same context as feature construction. Although we did not discuss feature selection in this paper, the techniques described in this paper have been used to implement feature selection and we hope to investigate this topic in the future.

6 Conclusion

This paper examined how a method from information fusion could be applied to feature construction from numerical features. The method was described in detail and three combinatorial fusion strategies were evaluated on ten data sets and three learning methods. The results were quite positive, especially for the data sets with the greatest class imbalance. When measuring AUC, the methods were of greatest benefit to the decision tree learning method, although it also substantially improved the 1-nearest neighbor method. Our results also indicate that our method can improve accuracy.

The work described in this paper can be extended in many ways. Our analysis would benefit from additional data sets—including several highly imbalanced data sets. It would also be interesting to evaluate additional combinatorial feature-fusion strategies, other than the three we evaluated in this paper. However, we suspect more complex fusion strategies will not yield substantial further improvements, so we do not view this as a critical limitation of our current work.

We also think that the basic algorithm can be extended in several ways. We plan on evaluating heuristic methods that would prune feature combinations that perform poorly. A heuristic method would enable us to evaluate more complex fusion schemes while potentially reducing the computation time. In this same vein, we also wish to consider simplifying the method for deciding whether to add a feature. Currently we use a validation set and only add a feature if the improvement in performance passes a statistical significance test.

While there are benefits to this strategy, it also increases the computational requirements of the algorithm.

7 References

- [1] E. Bloedorn and R. S. Michalski, "Data-driven constructive induction in AQ17-PRE: a method and experiments," in *Proc. of the 3rd International Conference on Tools*, 1991.
- [2] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, December 1997, 97(1-2):245-271.
- [3] A. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, 30, 7(July 1997), 1145-1159.
- [4] P. K. Chan and S. J. Stolfo, "Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection," in *Proc. of the Fourth International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 1998, 2001, 164-168.
- [5] W. Cohen, R. Schapire and Y. Singer, "Learning to order things," *Journal of Artificial Intelligence Research*, 10 (1999), 243-270.
- [6] P. Flach and N. Lavrac, "The role of feature construction in inductive rule learning," in *Proc. of the ICML 2000 Workshop on Attribute-Value and Relational Learning: crossing the boundaries*, 2000.
- [7] J. W. Gryzmala-Busse, Z. Zheng, L. K. Goodwin and W. J. Gryzmala-Busse, "An approach to imbalanced data sets based on changing rule strength," in *Learning from Imbalanced Data Sets: Papers from the AAAI Workshop*, AAAI Press, 2000.
- [8] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, 3 (2003), 1157-1182.
- [9] D. F. Hsu, Y. Chung and B. Kristal, "Combinatorial fusion analysis: methods and practices of combining multiple scoring systems," *Advanced Data Mining Technologies in Bioinformatics. Hershey, PA: Idea Group Publishing*; 2006, 32–62.
- [10] D. F. Hsu and I. Taksa, "Comparing rank and score combination methods for data fusion in information retrieval," *Information Retrieval*, 8 (3), 2005, 449-480.
- [11] C. Ma, D. Zhou and Y. Zhou, "Feature mining and integration for improving the prediction accuracy of translation initiation sites in eukaryotic mRNAs", in *5th International Conference on Grid and Cooperative Computing Workshops*, 2006, 349-356.
- [12] Z. Markov and I. Russell, "An introduction to the WEKA data mining system," in *Proc. of the 11th SIGCSE Conference on Innovation and Technology in Computer Science Education*. 2006, 367–368.
- [13] C. J. Matheus and L. A. Rendell, "Constructive induction on decision trees," in *Proc. of the 11th International Joint Conference on Artificial Intelligence*, 1989, 645-650.
- [14] D. J. Newman, S. Hettich, C. L. Blake and C. J. Merz. *UCI repository of machine learning databases* [<http://www.ics.usi.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science. 1998.
- [15] F. Otero, M. Silva, A. Freitas, and J. Nievola, "Genetic programming for attribute construction in data mining," in *Proc. of 6th European Conference*, April 14-16, 2003.
- [16] F. Provost, T. Fawcett and R. Kohavi, "The case against accuracy estimation for comparing classifiers," in *Proc. of the 15th International Conference on Machine Learning*, Moran Kaufmann, 1998, 445-453.
- [17] S. Scott and S. Matwin, "Feature engineering for text classification," in *Proc. of the 16th International Conference on Machine Learning*, 1999, 379-388.
- [18] G. M. Weiss, "Mining with rarity: a unifying framework," *SIGKDD Explorations*, 6, 1 (Dec. 2004), 7-19.
- [19] G. M. Weiss and H. Hirsh, "Learning to predict rare events in event sequences," in *Proc. of the 4th International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 1998, 359-363.
- [20] Z. J. Zheng, "Constructing X-of-N attributes for decision tree learning," *Machine Learning*, 40, 1 (2000), 35-75.

Influence of Noisy and High-Dimensional Data on Semi-Supervised Learning

Tianya Hou and Hyunjung Shin*

Abstract—Semi-supervised learning (SSL) has become a popular tool in machine learning. In SSL, label prediction for an unlabeled data point is determined by the similarities from its adjacent data points, thus how to make the similarity matrix is a critical factor determining the performance of SSL. When the data is noisy and has high dimensionality, however, the similarity matrix is no longer reliable and consequently, it is hard to expect reasonable performance of SSL. To overcome the difficulties, we propose to make the similarity matrix with the extracted features either from principal component analysis (PCA) or nonlinear principal component analysis (NLPCA). The method was validated on one artificial- and five real-world-problems. Thanks to the newly made similarity matrix based on the extracted features, the performance of SSL becomes robust against noise features and high dimensionality.

I. INTRODUCTION

RECENTLY, semi-supervised learning (SSL) has become a popular tool in machine learning, particularly in domains such as text classification and bioinformatics [5]-[6]. Given sets of labeled and unlabeled data points, the task of predicting the missing labels can be aided by the information from unlabeled data points, for example, by using information about the *manifold structure* of the data in input space. Many state-of-the-art methods implement a SSL approach in that they incorporate information from unlabeled data points into the learning paradigm [1]-[4]. Despite their many variations, one thing common to most algorithms is the use of a matrix of values representing the pairwise relationships between data points. The matrix is denoted as the “similarity matrix” and plays a critical role in determining the performance of SSL. The label prediction of an unlabeled data point is made through the propagation of the labels of its adjacent data points, and the strength of influence of each is proportional to the similarity between them. However, the similarity matrix can be easily affected by the noise features and high dimensionality of the raw dataset [7]. If irrelevant or highly correlated features are included in calculating the similarity, and even worse if the data is high-dimensional, the similarity matrix no more well reflects the points’ influence on each due to either “noisy influence” or the “curse of dimensionality.” Consequently, it can probably deteriorate the

generalization ability of SSL.

To circumvent this, a preventative against noisy influence and the curse of dimensionality becomes necessary before making the similarity matrix. One method is to use the feature extraction (dimensionality reduction) techniques in the preprocessing step. Feature extraction refers to the process of finding a mapping that reduces the dimensionality and of removing the noise effect from the dataset. With the newly extracted features, we can exclude the influence of redundant or noisy features from the similarity matrix with little or no loss of information [8], [9], and therefore, we can expect a better performance of SSL. There are various kinds of the method for feature extraction. As a linear method, principal component analysis (PCA) is the representative. By calculating the eigenvectors of the covariance matrix of the original data, PCA linearly transforms a high-dimensional vector of the input features into a low-dimensional one whose components (extracted features) are uncorrelated [10], [11]. On the contrary, there are several types of implementation of nonlinear PCA (NLPCA) [12]. Autoassociative neural network (AANN) is one of the well-known nonlinear transformation methods. In AANN, the network is trained to perform the identity mapping where the values of input features are approximated at the output layer, and the nonlinear principal components can be obtained from the hidden nodes in the bottleneck layer [13]-[16].

The primary purpose of this paper is to diagnose how robust an SSL algorithm against the noisy and high-dimensional input features, and therefrom suggest to use PCA or NLPCA as a preventative. Through feature extraction, the data points in the higher input space are projected onto a lower dimensional space of the new features extracted either from PCA or NLPCA. In the experiment, the performances of SSL with the original features and with the extracted features are compared. Superiority of PCA to NLPCA (or vice versa) depends upon linearity (or nonlinearity) of the intrinsic features of the underlying problem, but in practice it is difficult to know *a priori* that the given problem is linear or nonlinear. Therefore, the performance comparison between the two extraction methods will not of great interest in this paper.

The rest of this paper is organized as follows. In section 2, the theory of SSL is presented. In section 3, PCA and AANN, the feature extraction methods, are described, respectively. Section 4 provides the experimental results, followed by the conclusions in the last section.

II. SEMI-SUPERVISED LEARNING

In (graph based) semi-supervised learning algorithm, a data point $\mathbf{x}_i \in R^m$ ($i = 1, \dots, n$) is represented as a node i in a graph,

This work was supported in part by Post Brain Korea 21 and the research grant from Korean Government (MOEHRD, Basic Research Promotion Fund, KRF-2008-531-D00032).

Tianya Hou (houtianya@ajou.ac.kr) is in the master course of Industrial & Information Systems Engineering, Ajou University, Suwon, 443-749 Korea

* Corresponding author: Hyunjung Shin (shin@ajou.ac.kr) is a professor of the department of Industrial & Information Systems Engineering, Ajou University, Suwon, 443-749, Korea

and the relationship between data points is represented by an edge where the connection strength from each node j to each other node i is encoded as w_{ij} of a weight matrix W . A weight w_{ij} can take a binary value (0 or 1) in the simplest case. Often, a Gaussian function of Euclidean distance between points with length scale σ is used to specify connection strength:

$$w_{ij} = \begin{cases} \exp\left(-\frac{(x_i-x_j)^T(x_i-x_j)}{\sigma^2}\right) & \text{if } i \sim j \\ 0 & \text{otherwise} \end{cases}, \quad (1)$$

The $i \sim j$ stands for node i and j having an edge between them which can be established either by k nearest neighbors or by Euclidean distance within a certain radius r , $\|x_i - x_j\|^2 < r$. The labeled nodes have labels $y_l \in \{-1, 1\}$ ($l = 1, \dots, L$), while the unlabeled nodes have zeros $y_u = 0$ ($u=L+1, \dots, L+U$). The algorithm will output an n -dimensional real-valued vector $f = [f_l^T \ f_u^T]^T = (f_1, \dots, f_L, f_{L+1}, \dots, f_{L+U})^T$ which can be thresholded to make label predictions on f_{L+1}, \dots, f_{L+U} after learning. It is assumed that (a) f_i should be close to the given label y_i in labeled nodes and (b) overall, f_i should not be too different from its adjacent nodes f_j ($i \sim j$). One can obtain f by minimizing the following quadratic functional:

$$\text{Min}_f (f - y)^T (f - y) + \mu f^T L f, \quad (2)$$

where $y = (y_1, \dots, y_L, 0, \dots, 0)^T$, and the matrix L , called the graph Laplacian matrix, is defined as $L = D - W$, where $D = \text{diag}(d_i)$, $d_i = \sum_j \omega_{ij}$. The first term corresponds to the loss function in terms of condition (a), and the second term represents the smoothness of the predicted outputs in terms of condition (b). The parameter μ trades off loss versus smoothness. The solution of this problem is obtained as

$$f = (I + \mu L)^{-1} y, \quad (3)$$

where I is the identity matrix.

III. FEATURE EXTRACTION

A. Principal Component Analysis (PCA)

PCA can be used for dimensionality reduction in a data set by extracting important hidden features that contribute most to its variance. Technically, PCA attempts to find orthonormal axes which maximally decorrelate the original features of the data. Given the input data points $x_i \in R^m$ ($i = 1, \dots, n$ and $\sum_{i=1}^n x_i = 1$, usually $m < n$), PCA linearly transforms each data point x_i into a new one s_i by

$$s_i = \underbrace{U^T}_{m \times m} x_i, \quad i = 1, \dots, n, \quad (4)$$

where U is the $m \times m$ orthogonal matrix whose k th column u_k is the k th eigenvector of the covariance matrix $C = \frac{1}{n} \sum_{i=1}^n x_i x_i^T$. The matrix U can be obtained by solving the eigenvalue problem on C ,

$$\lambda_k u_k = C u_k, \quad k = 1, \dots, m, \quad (5)$$

where λ_k is an eigenvalue of C and u_k is the corresponding eigenvector. The magnitude of an eigenvalue stands for the proportion of variance that can be explained by the corresponding eigenvector. Therefore, when taking the first p eigenvectors $\tilde{U}^T = \{u_1, u_2, \dots, u_p\}$ referring to the descending order of eigenvalues, $\lambda_1 > \lambda_2 > \dots > \lambda_p > \dots > \lambda_m$, we can find “lower” dimensional orthonormal space ($m \rightarrow p$) yet still retaining most important aspects of the data. A projected data point onto the lower dimensional space, \tilde{s}_i , is calculated as the orthogonal transformations of x_i ,

$$\tilde{s}_i = \underbrace{\tilde{U}^T}_{p \times m} x_i, \quad i = 1, \dots, n, \quad (6)$$

PCA is a well-established dimensionality reduction method. However, its applicability is limited by the assumptions on linearity that the data set to be “linear” combinations of certain features. Therefore, if the data set shows non-linear relationship among features, there is no guarantee that the extracted features by PCA will contain important features.

B. Nonlinear Principal Component Analysis: Autoassociative Neural Network (AANN)

Another approach to dimensionality reduction is through the use of an autoassociative neural network (AANN), a special kind of feed-forward neural networks [17]. AANN finds and eliminates nonlinear correlations in the data. Analogous to principal component analysis, it can be used to reduce the dimensionality of data by removing redundant features. General structure of AANN is shown in Fig. 1. It consists of an input layer, an output layer, and multiple hidden layers. Both the number of input nodes and that of output nodes are equally set to m . Among the hidden layers, the mapping layer models the mapping function (F_1) and the demapping layer models the demapping function (F_2). The number of nodes in a particular hidden layer (p), so called “bottleneck layer”, is set to be less than the number of nodes in the input/output layer ($p < m$).

In autoassociative mapping, the target data are set to be identical to the input data. This “identity mapping” creates a global reduction of the data dimensionality while the input data go through the bottleneck layer before appearing at the output layer. Let F denote the autoassociative mapping learnt by the network. If $\{\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_n\}$ is the set of output data produced by the AANN when the input data set $\{x_1, x_2, \dots, x_n\}$ is given, the F can be found while minimizing the mean square error E ,

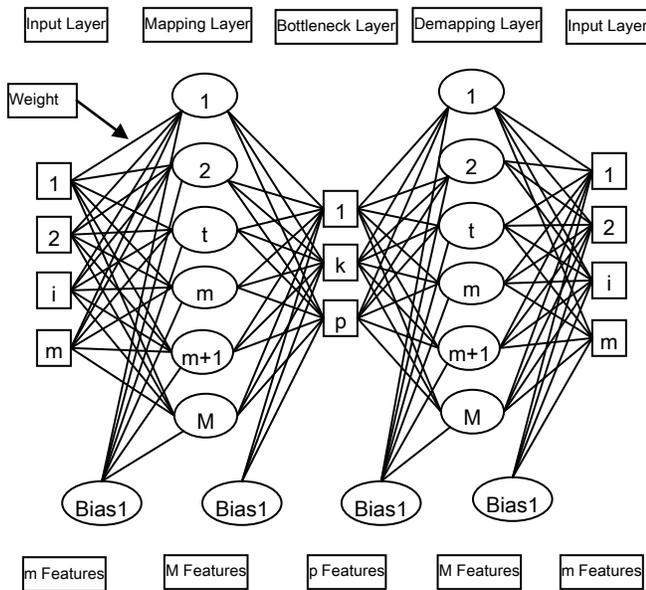


Fig.1 Architecture of AANN

$$E = \sum_{i=1}^n (\mathbf{x}_i - \tilde{\mathbf{x}}_i)^T (\mathbf{x}_i - \tilde{\mathbf{x}}_i) = \sum_{i=1}^n (\mathbf{x}_i - F(\mathbf{x}_i))^T (\mathbf{x}_i - F(\mathbf{x}_i)). \quad (7)$$

The mapping function F can be separated into F_1 and F_2 , so that $F(\cdot) = F_2(F_1(\cdot))$, where F_1 is the transformation in the network from the input layer upto the dimension compressing hidden layer (the bottleneck layer), and F_2 is the transformation from the bottleneck layer upto the output layer. To summarize, the data is first compressed to lower dimensionality and then reconstructed. The mapping from the input layer to the bottleneck layer can be regarded as “nonlinear” projection onto the lower dimensional space ($m \rightarrow p$), and each node in the bottleneck can be considered as an extracted feature retaining significant information of the data. New data $\tilde{\mathbf{x}}_i$ are then calculated as

$$\tilde{\mathbf{x}}_i = F_1(\mathbf{x}_i), \quad (8)$$

In theory, AANN can extract good features from any type of nonlinear relationship occurring in the data if the architecture is well designed. However, there is no definitive method for deciding *a priori* the number of nodes in the bottleneck layer.

IV. EXPERIMENT RESULTS

We applied the proposed method to various kinds of problems: an artificial problem and five real-world problems. The original dimensionality of each dataset was reduced by using PCA and NLPCA respectively, and the similarity matrices from the original features and the two types of new

extracted features were calculated. Hereafter, we denote those matrices as $\mathbf{W}_{\text{original}}$, \mathbf{W}_{PCA} and $\mathbf{W}_{\text{NLPCA}}$. For performance comparison of SSL among the three different similarity matrices, the area under the ROC curve (AUC) were measured. This experimental setting will show how influential the similarity matrix is to the performance of SSL, particularly when the original data is noisy and the dimensionality is high.

A. Artificial problem

The proposed method was evaluated on Two-moon data as shown in Fig. 2. A total of 500 two-dimensional data points were generated from two underlying classes and each class has 250 data points. We added different numbers of distractor features, called ‘noises’, which have nothing to do with classification accuracy. The number of the noise features varies depending on the degree of noise, 13 and 52, respectively, as shown in Fig. 3.

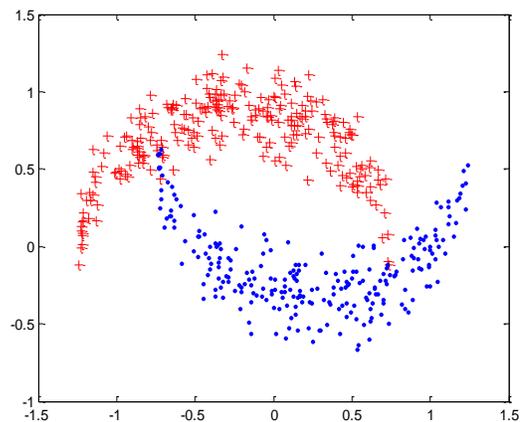


Fig. 2 Two-moon data

According to the experimental setting, several types of similarity matrix were calculated from the original two features (two-dimension), the 13 noise-added features (15-dimension), the 52 noise-added features (54-dimension), the two sets of the PCA extracted two features from 13 (or 52) noise-added feature set (two-dimension) and the two sets of NLPCA extracted two features from 13 (or 52) noise-added feature set (two-dimension), respectively. Then the AUCs of seven similarity matrices were measured under various combinations of parameters $\{k, \mu, N\} \in \{3, 5, 10\} \times \{0.1, 1, 10\} \times \{2, 5, 10, 20\}$ where k is the number of k -nearest neighbors in Eq.(1), μ is the loss-smoothness tradeoff parameter in Eq.(2) and N is the percentage (%) of the labeled data points in the data set.

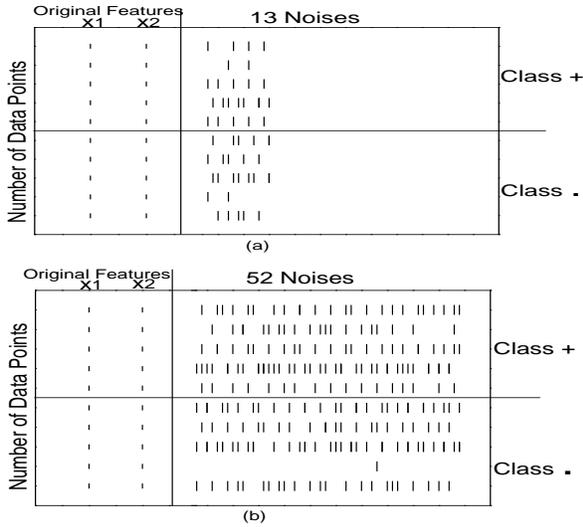


Fig. 3 Experimental setting with two different degrees of noise added to the two original features, (a) 13 noise features and (b) 52 noise features, respectively.

The results are shown in Fig. 4. For simplicity, the similarity matrix of the original two features is denoted as ' $W_{original}$ ', that of the noise-added features as ' $W_{w/noise}$ ' and those of the extracted features through PCA and NLPCA as ' W_{PCA} ' and ' W_{NLPCA} ', respectively. Fig. 4(a) shows the results for the case of the 13 noise-added features. The average AUC of $W_{original}$ is 0.952, while the average AUC of $W_{w/noise}$ is 0.734. And so the AUC decreases 22.0% after noise feature addition. This implies that the existence of noise features degrades the original performance of SSL.

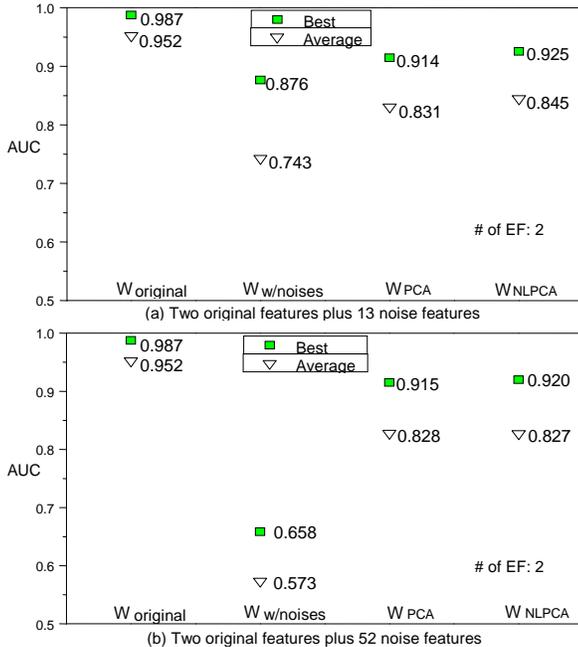


Fig. 4 The comparison of AUC for different similarity matrices. The EF stands for the extracted features through PCA or NLPCA. The square indicates the best AUC after repetition of experiments over every combination of parameters, and the reverse triangle indicates the average AUC.

The best AUCs of $W_{w/noise}$ is 0.876, while those of W_{PCA} and W_{NLPCA} are 0.914 and 0.925, respectively. Both methods improve the accuracy. In Fig. 4(b), when the number of the added noise features is 52, the avg. AUC for $W_{w/noise}$ becomes considerably lower than that of $W_{original}$ (0.952 vs. 0.573). The amount of the decrease is about 39.8% of the original performance. However, no matter which feature extraction method is used, the large amount of original performance is regained: 0.828 for W_{PCA} and 0.827 for W_{NLPCA} . The best AUC of $W_{w/noise}$ is 0.658 while those of W_{PCA} and W_{NLPCA} are 0.915 and 0.920, respectively. Through feature extraction, the AUC can be increased by 39.6% in the best case.

Comparing the two feature extraction methods, NLPCA performed slightly better than PCA. The reason can be explained by that the underlying features of Two moon problem are nonlinear. And NLPCA is better at discovering nonlinear features than PCA which is based on linear transformation. However, it is hard to know in advance whether the problem in hand is linear or nonlinear.

When comparing the two sets of experiments, the 13 noise added features and the 52 noise-added features, we can conclude that the more noise features incur the more serious degradation in performance of SSL. To recover the loss, feature extraction can be employed. Either PCA or NLPCA recovers most of the original performance. In our experiment, we used only "two" extracted features (EF), but we can expect better performance with more extracted features. And, it is interesting to see that the performance of feature extraction method is not sensitive to the changes of degree of noise from 13 to 52: 0.831 to 0.828 for PCA and 0.845 to 0.827 for NLPCA. This means the two feature extraction methods successfully found the intrinsic dimensions of the problem (two dimensions in our problem) in both experimental settings. Therefore, we can expect more stabilized performance of SSL through feature extraction regardless of the degrees of noise or high dimensionality.

B. Real-world problems

a. Data

Five real-world data sets were used for benchmarking. Table 1 summarizes the data sets from diverse fields: Pima Indians Diabetes, SPECTF and WDBC are available at [18] and Digit1 and USPS are available at [19].

TABLE I
FIVE REAL-WORLD DATA SETS

Data set	Classes	Original features	Data Points	New Features
Pima Indians Diabetes	2	8	768	3
SPECTF	2	44	267	4
WDBC	2	32	569	7
Digit1	2	241	1500	20
USPS	2	241	1500	20

In order to increase readability for the results, the AUCs are shown at a fixed set of the values of hyperparameters, $\{k, \mu, N\}$ at (10, 1, 10%).

b. The number of extracted features

It is difficult to determine the number of extracted features when the intrinsic dimension is unknown as in usual real-world problems. One of the rule-of-thumbs for PCA is to draw a scree plot and find the elbow point to determine the appropriate number of features to be extracted. In a scree plot, the proportion of all eigenvalues is drawn in their decreasing order. The plot looks like the side of a mountain, and “scree” refers to the debris falling from a mountain and lying at its base. So it proposes to determine the number of extracted features at the point the mountain ends and debris begins [20]. For AANN, however, it is hardly known how to determine the number of hidden nodes in the bottleneck layer. In our experiments, the elbow points for PCA were also used to determine the number of hidden nodes for NLPCA.

Fig. 5 shows the resulting scree plots for the five data sets. As shown in the plots, the elbows for Pima, SPECTF, WDBC, Digit1, and USPS were found at 3, 4, 7, 20, and 20 respectively, and they became the number of the features to be extracted for both PCA and NLPCA.

c. Accuracy

The AUC results for the five real-world data sets are shown in Fig. 6. For Pima data set, both W_{PCA} and W_{NLPCA} achieved a reasonable accuracy in AUC. Similar results can be found at WDBC, Digit1 and USPS. For SPECTF data set, there is a pronouncing improvement in AUC: 0.528 for $W_{original}$ jumped upto 0.68 and 0.71 for W_{PCA} and W_{NLPCA} , respectively. We may have a conjecture that SPECTF data set contains a lot of redundant or noise features, and so the PCA or NLPCA properly works to extract relevant features among them. Across the five data sets, there is a slight competition between W_{PCA} and W_{NLPCA} . For instance, W_{PCA} outperforms W_{NLPCA} in SPECTF (0.68 vs. 0.71). On the contrary, W_{NLPCA} outperforms W_{PCA} in USPS (0.95 vs.0.92). Superiority of one method to the other seems to be dependent upon linearity (or nonlinearity) of the intrinsic features of the underlying problem. A more important fact we found through the experimental results is that W_{PCA} or W_{NLPCA} enables us to obtain a similar or better accuracy with much less number of features. The number of the extracted features ranges from about 8% to 38% of the original number of features: 8% in either Digit1 or USPS and 38% in Pima. This will be studied further in the next section. Table I shows the details. The Wilcoxon signed-ranks test were used to compare W_{PCA} (or W_{NLPCA}) and $W_{original}$ [21]. A smaller p -value stands for W_{PCA} (or W_{NLPCA}) outperforms $W_{original}$ with a greater statistical significance.

d. Efficacy of a feature

The results in Fig.5 and table I showed that a similar or better accuracy can be obtained with much less number of features through W_{PCA} or W_{NLPCA} . In order to study efficacy of a feature in length, we defined “CF” as follows,

$$CF = \frac{AUC}{\text{number of features}}, \quad (9)$$

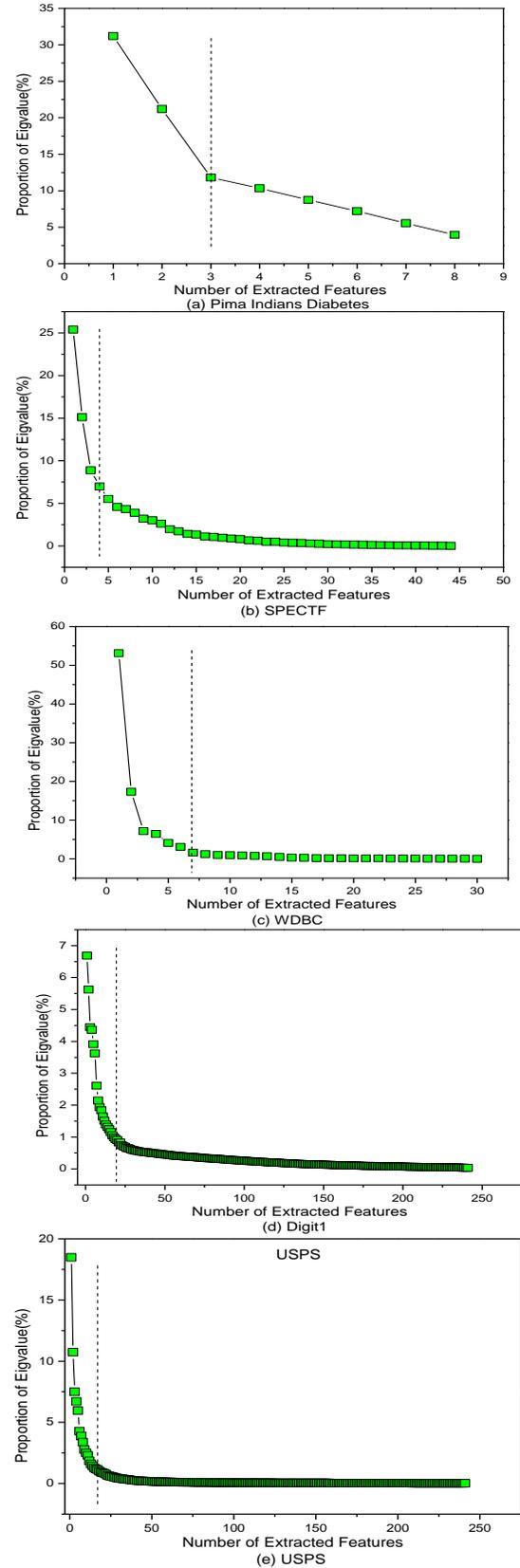


Fig. 5 Scree plots for five data sets: the dotted line indicates the number of the features to be extracted, 3 for Pima Indian Diabetes, 4 for SPECTF, 7 for WDBC, 20 for Digit1, and 20 for USPS.

where it quantifies the amount of contribution of a feature to overall accuracy. A larger value of CF means a higher efficacy of a feature.

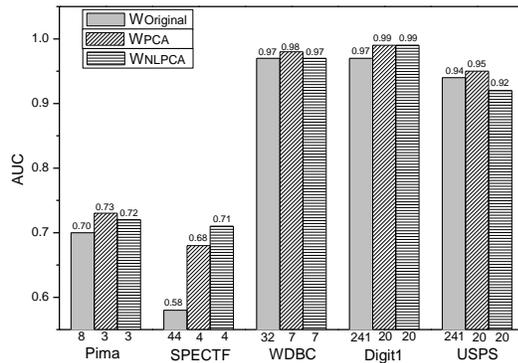


Fig. 6 AUCs for the Five Real-world Data Sets: The number below the individual histogram stands for the number of the features used for making the similarity matrix W .

TABLE I
WILCOXON SIGNED-RANKS TEST

Data set	WOriginal		WPCA		WNLPCA	
	mean \pm std	p-value	mean \pm std	p-value	mean \pm std	p-value
Pima	0.70 \pm 0.037	0.00	0.73 \pm 0.033	0.00	0.72 \pm 0.035	0.06
SPECTF	0.58 \pm 0.100	0.00	0.69 \pm 0.129	0.00	0.72 \pm 0.108	0.00
WDBC	0.97 \pm 0.010	0.09	0.98 \pm 0.011	0.00	0.97 \pm 0.014	0.70
Digit1	0.97 \pm 0.007	0.00	0.99 \pm 0.003	0.00	0.99 \pm 0.004	0.00
USPS	0.94 \pm 0.012	0.00	0.95 \pm 0.007	0.00	0.93 \pm 0.027	0.00

Fig. 7 shows the results. The CF of W_{PCA} or W_{NLPCA} is relatively much higher than that of $W_{Original}$ for each of the five problems. For Pima Indian Diabetes, the CF of W_{PCA} or W_{NLPCA} is higher by three times than that of $W_{Original}$. Similarly, the efficacy of a feature increased for other problems: 12, 5, 12, and 12 times for SPECTF, WDBC, Digit1 and USPS, respectively. This implies that PCA or NLPCA extracts efficacy-enhanced features out of irrelevant (or noise) ones, which consequently enables us to achieve a similar or better performance with a much smaller set of features.

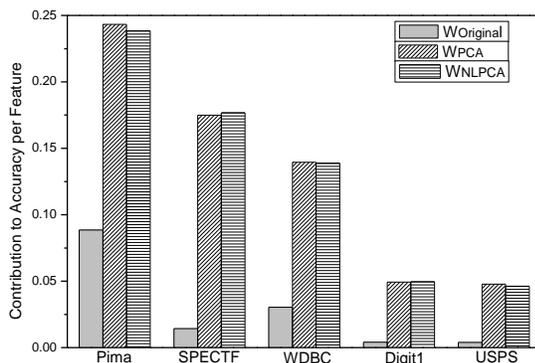


Fig. 7 The CF of a Feature

V. CONCLUSION

In this paper, we addressed an issue on how robust the SSL algorithms against the noisy and high dimensional data. Both factors can incur negative influence on the similarity matrix, and consequently, degrade the generalization ability of SSL. As a method of overcoming the difficulty, we proposed to use the extracted features from PCA or NLPCA when calculating the similarities between data points. In the experiment, we showed how much loss of performance can occur because of the noisy and high dimensional features, and presented how to regain or recover the original performance through PCA or NLPCA.

As a preliminary step, the current work takes very simple conventional feature extraction methods as its basis. However, incorporated into more sophisticated state-of-the-art feature extraction algorithms or newer design of a wrapper algorithm for SSL, our approach has the potential to improve considerably on the original performance of SSL and enhance its robustness as well.

ACKNOWLEDGEMENT

This work was supported in part by Post Brain Korea 21 and the research grant from Korean Government (MOEHRD, Basic Research Promotion Fund, KRF-2008-531-D00032).

REFERENCES

- [1] Chapelle O., Weston J., and Schoelkopf B.. Cluster kernels for semi-supervised learning. In: *Advances in Neural Information Processing Systems*, edited by S. Becker, S. Thrun, and K. Obermayer, 15, pages 585-592, MIT Press, 2003.
- [2] Zhu X.. Semi-supervised learning with graphs. *Ph.D. dissertation*, Carnegie Mellon University, Pennsylvania, USA, 2005.
- [3] Shin H., Hill N.J., and Raetsch G.. Graph-based semi-supervised learning with sharper edges. *Proceeding of the 17th European Conference on Machine Learning*, pages 402-413, 2006.
- [4] Shin H., Lisewski A.M. and Lichtarge O.. Graph sharpening plus graph integration: a synergy that improves protein functional classification. *Bioinformatics*, 23(23), pages 3217-3224, Oxford University Press, 2007.
- [5] Su W., Carpuat M., and Wu D.. Semi-supervised training of a kernel PCA-based model for word sense disambiguation. *Proceedings of the 20th International Conference on Computational Linguistics*, pages 29-35, Switzerland, 2004.
- [6] Shin H., Tsuda K.. Prediction of protein function from networks. In book: *Semi-Supervised Learning*, edited by O. Chapelle, B. Schoelkopf, A. Zien, 20, pages 339-352, MIT press, 2006.
- [7] Guyon I., Elisseeff A.. An introduction to variable and feature selection. *Machine Learning Research*, 3, pages 1157--1182, 2003.
- [8] Scherf M., Brauer W.. Feature selection by means of a feature weighting approach. *Technical report*, Technische Universität München, 1997.
- [9] Setiono R., Liu H.. Neural-network feature selector. *IEEE Transactions on Neural Networks*, 8, pages 654-662, 1997.
- [10] Smith, L.I.: A Tutorial on Principal Components Analysis. 2002
- [11] Cao L.J., Chong W.K.. Feature extraction in support vector machine: a comparison of PCA, KPCA and ICA. *Proceedings of the 9th International Conference on Neural Information Processing*, 2, pages 1001-1005, 2002.
- [12] Saegusa R.. A study of nonlinear principal component analysis using neural networks. Waseda University, 2005.

- [13] Kramer M.A.. Nonlinear principal component analysis using autoassociative neural networks. *AICHE*, 37(2), pages 43-49, 1991.
- [14] Ikbal M.S., Misra H., and Yegnanarayana B.. Analysis of autoassociative mapping neural networks. *Neural Networks*, 5, pages 3025-3029, 1999.
- [15] Pokajac D., Lazarevic A.. Applications of unsupervised neural networks in data mining. *Neural Network Applications in Electrical Engineering*, pages 17-20, 2004.
- [16] Hsieh W.W.. Nonlinear principal component analysis of noisy data. *Neural Networks*, 20(4), pages 434-443, 2007.
- [17] Kamimur R.T., Bicciato S., Shimuzu H., and etc. Mining of multivariate temporal biological data: a framework for the rational design of data-driven models. *BioKDD*, San Francisco, CA, 2001.
- [18] <http://archive.ics.uci.edu/ml/>
- [19] <http://www.kyb.tuebingen.mpg.de/ssl-book/benchmarks.html>
- [20] Zhu M., Ghodsi A.. Automatic dimensionality selection from the scree plot via the use of profile likelihood. *Computational Statistics & Data Analysis*, 51, pages 918-930, 2006
- [21] Demsar, J.. Statistical comparisons of classifiers over multiple data sets. *Machine Learning Research*. 7, pages 1-30, 2006

Efficient Record Linkage using a Double Embedding Scheme

Noha Adly

*Department of Computer and Systems Engineering
Faculty of Engineering, Alexandria University
Alexandria, 21544 Egypt
noha.adly@alex.edu.eg*

Abstract—Record linkage is the problem of identifying similar records across different data sources. The similarity between two records is defined based on domain-specific similarity functions over several attributes. In this paper, a novel approach is proposed that uses a two level matching based on double embedding. First, records are embedded into a metric space of dimension K , then they are embedded into a smaller dimension K' . The first matching phase operates on the K' -vectors, performing a quick-and-dirty comparison, pruning a large number of true negatives while ensuring a high recall. Then a more accurate matching phase is performed on the matching pairs in the K -dimension. Experiments have been conducted on real data sets and results revealed a gain in time performance ranging from 30% to 60% while achieving the same level of recall and accuracy as in previous single embedding schemes.

Keywords- data cleaning; similarity matching; record linkage; embedding schemes

I. INTRODUCTION

The record linkage problem is to find similar records, across different data sources, that refers to the same real world entity, e.g. patient, customer or author. When record linkage is performed within the same source, the problem is referred to as duplicate detection. The record linkage arises in the context of data cleaning that usually precedes data analysis and mining. It is important when integrating two data sources into one and in improving the quality of data by comparing to more accurate sources. The major challenges in record linkage are reducing computational complexity while maintaining high recall and accuracy. Several techniques have been proposed in the literature, see [7, 16] for recent surveys.

A naïve approach for discovering matching records in two sources is to perform a nested-loop comparing each record in one source to all records in the second source. However, $O(N^2)$ comparisons is computationally infeasible, especially for large datasets. Further, performing an approximate matching between two records requires the computation of distance functions among textual attributes, which is an expensive factor in the cost.

Several techniques have been introduced to reduce the quadratic number of comparisons that are based on a two phase approach. Blocking [1, 2] use record attributes, or subsets of attributes as a blocking key (e.g. first 4 characters of the surname) to split the data into blocks, then a detailed comparison is carried out only between records that fall into the same block. Although blocking increase the speed of comparison, it can lead to an increased number of false

negatives due to selection of a blocking step that places entries in the wrong blocks. Multiple runs using different blocking fields are performed to improve effectiveness. Sorted neighborhood [10], or merge/purge approach, relies on sorting records based on a sorting key, then moving a window of fixed size w sequentially where only records within w are paired with each other. This approach relies on the assumption that duplicate records will be close in the sorted list. Also, its effectiveness is dependent upon the comparison key. Similar to blocking, it has shown to be more effective with multi-passes, which causes an increase in the runtime. McCallum et al [18] proposed a cheap comparison metric to group records into overlapping clusters called canopies, then records within the same cluster are accurately compared. A recent work [21] exploited the semantic ambiguity of the data sources, using social network analysis, and applied relaxed matchers to less ambiguous data.

Reducing the complexity of record comparisons has been addressed by techniques such as feature subset selection algorithms [23]. Recent approaches [15, 20] have used embedding textual attributes into Euclidean space while preserving the distances between the record values and performed the comparison in the metric space which is much cheaper than string comparison. In [20] SparseMap was used to map strings in a private environment to Euclidean space then used multidimensional index based on KDTree to perform the comparison. Jin et al [15] used FastMap for converting strings then applied a similarity-join algorithm based on R-tree to compare the metric vectors.

This paper proposes a novel two-level matching scheme that exploits the advances developed in mapping records into a multidimensional Euclidean space while preserving the distances between the record values. It relies on a quick-and-dirty matching process that is employed first, to prune a large number of mismatches and produces a smaller set of pairs of records that are used as input to a more expensive matching process. Datasets are first embedded into a metric space of dimension K , which captures an accurate representation of the data. It is followed by a second embedding that converts vectors from K dimension to vectors of smaller dimension K' . The quick-and-dirty matching phase is performed on the output of the second embedding, with the goal of discarding a large number of true negatives while ensuring that the resulting potential matching pairs includes all true positives and zero false negatives, yet achieving this with a small cost. The returned results are ingested to the second level of matching, which applies a similarity function on the potential pairs, but in their K -dimension representation. The second

level matching is more expensive, but more accurate, and its goal is to refine the results of the first level by excluding false positives and any true negatives that have not been pruned in the first phase. With $K' < K$, the first level matching process is much cheaper than having the similarity function applied on all pairs in their K -representation. The expensive matching is performed only on the pairs detected in the first phase, which are much less, and cause an overall gain in time performance while achieving a high recall and accuracy.

Recently, a similar approach has been proposed in [27] in the context of private record linkage. They assumed the data already embedded in the metric space, which is then represented as a point in the complex plane where a relaxed matching is performed, detecting pairs likely matching. Then a more accurate matching phase is performed on the likely matching pairs only. The second embedding though is bound to the complex plane where $K'=2$, and is not contractive. In this paper, the second embedding is generalized and a proper K' is chosen according to the nature of the data and its size. It has been shown that as the data size increases, higher values of K' results in higher performance gains.

By selecting a proper embedding scheme in each phase, preserving the distances between the record values while guaranteeing contractiveness, it is ensured that results obtained are with a high recall and accuracy. The proposed matching scheme based on double embedding has been implemented and a set of experiments have been conducted on real data sets and compared with a scheme using a single embedding. Results showed that improvement in time performance ranging from 30% to 60% is achieved while maintaining the same level of recall and accuracy.

The remainder of the paper is organized as follows. A formal definition of the problem is presented in Section II. In Section III, we describe a single embedding matching scheme that is similar to previous schemes proposed and will be used for comparison. In section IV, the steps of the matching scheme based on double embedding are presented. In section V, the experiments conducted to evaluate the performance of the protocol are presented and results are discussed. Finally, Section VI concludes the paper.

II. PROBLEM FORMULATION

The process of identifying similar record pairs consists of building a classifier that takes as input a set of thresholds and accurately classifies pairs of records as *match* or *mismatch* according to a predefined matching rule. Without loss of generality, it is assumed that the input datasets R and S are represented as relations, and the schema of the two relations is the same $R(a_1, a_2, \dots, a_n)$ and $S(a_1, a_2, \dots, a_n)$.

Given a distance function $d_i: Dom(R.a_i) \times Dom(S.a_i) \rightarrow \mathbb{R}^+$ defined over domains of corresponding attribute of R and S , and matching thresholds $\theta_i \geq 0$, record linkage can be expressed as a join operator over R and S . A record pair (r, s) where $r \in R$ and $s \in S$, is a matching pair if $d_i(r.a_i, s.a_i) \leq \theta_i$ for all attributes $1 \leq i < n$. Then the join condition can be defined based on the following matching rule that returns *true* for matching record pairs and *false* for mismatching record pairs

$$MR(r, s) = \begin{cases} true & \text{iff } d_i(r.a_i, s.a_i) \leq \theta_i \quad \forall 1 \leq i \leq n \\ false & \text{otherwise} \end{cases}$$

The presented definition for the matching function is used by most of the record linkage approaches. The distance function d_i defines the similarity metric at the attribute level and is domain specific. In the domain of strings, there are a variety of metrics including the Edit distance, Smith-Waterman distance, Jaro distance, q-gram and others (refer to [5, 7] for a survey). In this work, the Edit distance a.k.a. Levenshtein distance, a common measure of textual similarity, is used although any other metric could be used. Formally, given two strings s_1 and s_2 , their edit distance is the minimum number of insertion, deletions and replace operations of single characters that are needed to transform s_1 to s_2 . For instance, the edit distance between Johnson and Jonsan is 2, as Johnson is obtained by adding h and replacing a by o. In the metric domain, the most common metric distance function used is the Minkowski metrics based on the L_p norms, $\|x\|_p = (\sum |x_i|^p)^{1/p}$, with $p \geq 1$. In this work, the Euclidean distance d_E ($p=2$), is used as the distance metric in the embedded space, although any other metric can be used.

III. SINGLE EMBEDDING SCHEME

In this section we describe the Single Embedding Scheme, which consists of two steps. In the first step, strings are mapped to objects in a multidimensional Euclidean space, such that the mapped space preserves the original string distance. In the second step, a multidimensional similarity join is performed in the Euclidean space. This approach is similar to previous work such as [15, 20] and will be used for comparison. Several methods have been proposed to embed a set of objects in a metric space, including FastMap[8], SparseMap[13], MetricMap[25] and others (see [12] for a survey). Among those methods, SparseMap has been chosen because it has proven to be contractive when the original space is strings [12]. Contractiveness ensures that distances in the embedded space are a lower bound for distances in the original space, thus improving the quality of the embedding in terms of recall. In the following, the concept of the SparseMap technique is introduced and a description of how it is used to embed strings into Euclidean space follows. Next, the technique used to perform the similarity join to complete the matching process is described.

A. Embedding Strings to K -dimension Euclidean Space

SparseMap is an embedding method based on a class of embedding known as Lipschitz embedding [3]. Therefore, we first describe Lipschitz embedding followed by the heuristics introduced by SparseMap.

Lipschitz embedding defines a coordinate space where each axis corresponds to a reference set, drawn from the set of objects to be embedded. Given a set of objects O and a distance D in the original space, the embedding is defined in terms of a set S of subsets of O , $S = \{S_1, S_2, \dots, S_k\}$ where S_i is a reference set. Given an object $o \in O$, the mapping F is defined as $F(o) = (D(o, S_1), \dots, D(o, S_k))$, where $D(o, S_i) = \min_{x \in S_i} \{D(o, x)\}$. That is, the coordinate values of object o are the distances from o to the closest element in each set S_i . The method is based on the triangle inequality and exploits the fact that if $|d(o_1, x) - d(o_2, x)| \leq d(o_1, o_2)$, then the property can be extended

to subset S_i and the value $|d(o_1, S_i) - d(o_2, S_i)|$ is a lower bound on $d(o_1, o_2)$. By using a set S of subsets, we increase the likelihood that the distance $D(o_1, o_2)$ is captured adequately by the distance in the embedding space between $F(o_1)$ and $F(o_2)$ i.e. $d(F(o_1), F(o_2))$.

Linial et al [17] have shown that when d , the metric distance function used to compare the embedded object, is one of the Minkowski metrics L_p , a bound can be established on $d(F(o_1), F(o_2))$, provided that $k = \lceil \log_2 N \rceil^2$ and S_i is of size 2^j with $j = \lfloor (i-1)/\log_2 N + 1 \rfloor$. Given $F(o) = (D(o, S_1)/q, \dots, D(o, S_k)/q)$, where $q = k^{1/p}$, it has been proved [17] that the embedding is *contractive* and the *distortion*, that is, the relative amount of deviation of the distance values in the embedding space with respect to the original distance, is guaranteed to be $O(\log N)$.

Lipschitz embedding is rather impractical for two reasons. First, due to the number and sizes of the subsets in S , $O(N^2)$ distance computations is needed to embed an object o , as the distance between o and practically all objects need to be computed, which is exactly what we wish to avoid. Second, the number $= \lceil \log_2 N \rceil^2$ of subsets, which is the number of coordinate values (dimensions) in the embedding is rather large. SparseMap [13] introduces heuristics to overcome the above limitations. The *Distance Approximation* heuristic approximates the distance between object o and subset S_i by computing $\hat{Y}(o, S_i)$, an upper bound on $d(o, S_i)$ by exploiting the partial vector that has been computed for each object.

The algorithm for embedding strings into an Euclidean space of dimension K begins by combining the strings from the two datasets into one set O . It starts by building the reference sets $S = \{S_1, S_2, \dots, S_k\}$. As suggested in [13], each set S_j is composed of any random strings of O of size 2^j where $j = \lfloor (i-1)/\sqrt{K} + 1 \rfloor$. Thus, we get \sqrt{K} reference sets of size 2, \sqrt{K} of size 4, etc, up to size $2^{\lfloor \sqrt{K} \rfloor + 1}$. Then it proceeds by computing the first coordinate for all objects, followed by the second coordinates, etc. The *EmbedString* algorithm is depicted in Figure 1.

```

Algorithm: EmbedString( $O, K$ )
Input:  $O$ : a set of  $N$  strings
       $K$ : dimensionality of Euclidean space
Output:  $SE[1..N][1..K]$  coordinates of the  $N$  strings

// Build  $K$  reference sets  $S = \{S_1, S_2, \dots, S_k\}$ 
for  $i=1$  to  $K$ 
   $S_i \leftarrow 2^{\lfloor (i-1)/\sqrt{K} + 1 \rfloor}$  strings randomly chosen from  $O$ 
for  $i=1$  to  $K$ 
   $\forall o_j \in O$ 
    if ( $i=1$ )
      // 1st coordinate: the distance is the minimum
      // Edit distance between  $o_j$  and every object in  $S_i$ 
       $\hat{Y}(o_j, S_i) = \min_{o \in S_i} \{D(o_j, o)\}$ 
    else {
      // Get Euclidean distance between the ( $i-1$ ) coordinates
      // of  $o_j$  and all objects in  $S_i$ 
      Compute  $d_E(F_{i-1}(o_j), F_{i-1}(o_t)) \forall o_t \in S_i$ 
      Sort  $d_E(F_{i-1}(o_j), F_{i-1}(o_t)) \forall o_t \in S_i$  ascendingly
      Select the first  $\sigma$  objects and place them in set  $\phi$ 
      Compute  $D(o_j, o_t)$  for all  $o_t \in \phi$ 
      Select  $o_r$  s.t.  $D(o_j, o_r) = \min_{o_t \in \phi} \{D(o_j, o_t)\}$ 
       $\hat{Y}(o_j, S_i) = D(o_j, o_r)$ 
    }
   $SE[j, i] = \hat{Y}(o_j, S_i)$ 

```

Figure 1: Pseudo code of *EmbedString* embedding strings to Euclidean space

A drawback of the distance approximation heuristic is that it renders the mapping non-contractive. In this paper, the heuristic proposed by [12] is used in order to make SparseMap contractive. The heuristic suggests that, instead of computing the actual distance $D(o_j, o_t)$ for only a fixed number of objects σ , it does so for a variable number of objects in S_i . In particular, it first computes the approximate distances $d_E(F_{i-1}(o_j), F_{i-1}(o_t))$ for all $o_t \in S_i$ which are lower bounds on the actual distance value $D(o_j, o_t)$. Next, it computes the actual distance value of the object $o_t \in S_i$ in increasing order of their lower bound distances $d_E(F_{i-1}(o_j), F_{i-1}(o_t))$. Let $o_r \in S_i$ be the object whose actual distance value $D(o_j, o_r)$ is the smallest distance value so far. Once $D(o_j, o_r)$ is smaller than all distances $d_E(F_{i-1}(o_j), F_{i-1}(o_t))$ of all remaining elements in S_i , then $d(o_j, S_i) = D(o_j, o_r)$. Although this heuristic increases the number of distance computations, it was decided to adopt it in order to make the embedding contractive.

B. Similarity Join in Euclidean Space

After the two datasets have been mapped into the metric space, it is required to find pairs of objects whose distance in the Euclidean space is within a threshold δ . Many similarity-join algorithms can be applied [11, 14] and usually they employ a form of multidimensional index [9]. In this work, the KDTree index [9] has been used as it is considered one of the most prominent data structure for indexing multidimensional spaces and is designed for efficient nearest neighbor search [22].

KDTree is a binary tree in which every node is a k -dimensional point. Every internal node generates a splitting hyperplane that divides the space into subspaces. Points left to the hyperplane represent the left subtree of that node and the points right to the hyperplane represents the right subtree. The hyperplanes are iso-oriented and their direction alternates among the k possibilities. Building a KDTree is a $O(N \log N)$ operation.

The nearest neighbor algorithm (NN) aims to find the node in the tree which is nearest to a given input vector. This search can be done efficiently ($O(\log N)$) by using the tree properties to quickly eliminate large portions of the search space. The search starts with the root node and moves down the tree recursively until it reaches a leaf and saves that node point as the *nearest*. The algorithm unwinds the recursion of the tree; if the current node is closer than the *nearest*, then it becomes the *nearest*. The algorithm checks whether there could be any points on the other side of the splitting plane that are closer to the search point than the *nearest*. This is done by intersecting the splitting hyperplane with a hypersphere around the search node that has a radius equal to the current nearest distance. If the sphere crosses the plane, there could be nearer points on the other side of the plane, so the algorithm must move down the other branch of the tree from the current node looking for closer points, following the same recursive process as the entire search. If the hypersphere does not intersect the splitting plane, then the algorithm continues walking up the tree, and the entire branch on the other side of that node is eliminated.

A variation of the NN algorithm [19] has been adopted that allows performing range searching. That is, given δ and a vector v , it is required to retrieve all nodes that are within distance δ from v . The variation is mainly that the initial distance is not reduced as closer points are discovered and all discovered points within δ are returned, not just the *nearest*.

The variation of the NN algorithm is applied in order to compare the vectors representing the two datasets SE_1 and SE_2 ; the used distance metric is the Euclidean distance. Specifically, the KDTree for one of the datasets is built, say SE_1 . Then, for every vector in SE_2 the NN range search algorithm is applied to retrieve the nodes in SE_1 that are within distance δ .

IV. DOUBLE EMBEDDING SCHEME

This section introduces the Double Embedding Scheme, which consists of four steps. The first step, combines the strings from the two datasets into one set O and maps them into Euclidean space of dimension K using the *EmbedString* algorithm presented in Section III.A. The second step involves mapping the embedded strings into a more compressed representation in the Euclidean space in dimension K' , where $K' < K$. In this step, the FastMap embedding technique has been selected, because of its simplicity, efficiency and contractiveness. The third step performs the similarity join between the two sets in the K' dimensional space using the KDTree as described in Section III.B. The fourth and last step, takes its input as the potential matched pairs produced from the similarity join and compares the Euclidean distance of the corresponding objects in the K dimension space.

In the following, the technique used in embedding the datasets in K' dimension using FastMap is described, then the overall matching protocol is presented.

A. Embedding K -dimension objects into K' -dimension

FastMap[8] is a general embedding technique that is inspired by dimensionality reduction methods for Euclidean space based on linear transformation. Objects are mapped into points in K' dimensional space, where the coordinate values corresponding to these points are obtained by projecting them on K' mutually orthogonal directions, thereby forming the coordinate axes of the space in which the points are embedded. The projections are computed using the original distance function D . In our case, where the original space is K -dimension Euclidean space $D = d_E^K$. The coordinate axes are constructed one by one, where at each iteration, two objects (referred to as pivot objects) are chosen, a line is drawn between them that serves as the coordinate axis, and the coordinate value along this axis for each object o is determined by projecting o into this line.

For setting the K' -coordinate axis, pivot objects are chosen at each step to anchor the line that form the newly formed axis. To extract more distance information, FastMap attempts to identify a pair of pivot objects that are far away from each other. In order to avoid $O(N^2)$ distance computations to determine the farthest pair of objects, a heuristic is proposed in [8] for computing an approximation of the farthest pair of objects. This heuristic first arbitrary

chooses one of the objects t . Next, it finds the object r which is farthest from t . Finally, it finds the object s which is farthest from r . The last step can be iterated a number of m times in order to obtain a better estimate. Although [8] indicated that setting $m=5$ provides good estimates, [13] has shown that negligible improvements are achieved for $m>2$.

Deriving the first coordinate for the N objects is obtained by projecting each object a on a line between pivots p_1 and p_2 $x_a = [D(p_1, a)^2 + D(p_1, p_2)^2 - D(p_2, a)^2] / 2D(p_1, p_2)$. To derive the i^{th} coordinate, the $(i-1)$ -dimensional hyperplane H , which is perpendicular to the line that forms the previous coordinate axis, is determined and all objects are projected onto H . The projection is performed by defining a new distance d_H that measures the distance between the projections of the objects on H . Let x_0^i be the i^{th} coordinate for object o , $F_i(o) = \{x_0^1, x_0^2, \dots, x_0^i\}$ be the first i coordinate value for $F(o)$, d_i be the distance function used in the i^{th} iteration, and p_1^i and p_2^i be the two pivots chosen at iteration i , then

$$x_0^i = [d_i(p_1^i, o)^2 + d_i(p_1^i, p_2^i)^2 - d_i(p_2^i, o)^2] / 2 d_i(p_1^i, p_2^i)$$

The algorithm of embedding objects from the metric space of dimension K into dimension K' using FastMap is described in Figure 2.

```

Algorithm EmbedNum( $SE, K'$ )
Input:  $SE[1..N][1..K]$ : coordinates of the  $N$  strings
       $K'$ : dimensionality of Euclidean space
Output:  $DE[1..N][1..K']$  coordinates of the  $N$  strings in  $K'$  dimension

for ( $h=1$  to  $K'$ ) {
  ( $p_1, p_2$ ) = ChoosePivot( $h$ );
   $v = \text{FastApproxDist}(p_1, p_2, h)$ ;
  if ( $v=0$ ) // all inter-objects distances are zero
     $DE[i][h] = 0 \ \forall i=1$  to  $N$ 
  else // compute coordinate on this axis  $h$ 
    for ( $i=1$  to  $N$ ) {
       $x = \text{FastApproxDist}(i, p_1, h)$ ;
       $y = \text{FastApproxDist}(i, p_2, h)$ ;
       $DE[i][h] = (x^2 + y^2 - v^2) / 2*v$ ;
    }
}

```

Figure 2(a): Algorithm *EmbedNum* mapping numbers into K'

```

FastApproxDist( $a, b, h$ ) {
   $v = d_E^K(SE[a], SE[b])$ ;
  for ( $i=1$  to  $h-1$ ) {
     $w = DE[a][i] - DE[b][i]$ ;
     $v = |v^2 - w^2|^{1/2}$ ;
  }
  return  $v$ ;
}

ChoosePivot( $h$ ) {
  // choose two pivots from objects represented in  $SE$  on the  $h^{th}$ -dimension
  select an object and set it to be the second pivot  $b$ 
  for ( $i=1$  to  $m$ ) {
    // FastApproxDist() is used to get distance between  $a$  and  $b$ 
    Set  $a =$  farthest object from  $b$ ; Let  $p_a$  be index of  $a$  in  $SE$ 
    Set  $b =$  farthest object from  $a$ ; Let  $p_b$  be index of  $b$  in  $SE$ 
  }
  return( $p_a, p_b$ );
}

```

Figure 2(b): Methods used with *EmbedNum* algorithm

In the method *FastApproxDist*(), since the distance v can be negative, the heuristic developed by [26] has been adopted, namely using the square root of the absolute value of $(v^2 - w^2)$. FastMap has the advantage of being simple and efficient as its cost is linear; it needs $O(2+2m)K'N$ distance

computations. It should be noticed that FastMap has not been chosen as the embedding technique in the first step because it has been proven in [12] that FastMap is not contractive when the original object space is not the Euclidean space.

B. Overall Matching Scheme

In this section, the pseudo-code of the overall matching scheme is illustrated. It starts by combining the strings from the two source datasets into one set O and maps them into Euclidean space of dimension K using the *EmbedString* algorithm, generating a global matrix $SE[N_1+N_2][K]$ containing the K -coordinates of all strings. SE is then split to $SE_1[N_1][K]$ and $SE_2[N_2][K]$ representing each dataset to be matched. This is followed by embedding the K -dimensional vector of each record in a more compressed representation in K' -dimension generating $DE_1[N_1][K']$ and $DE_2[N_2][K']$ for each source. Next, objects in their K' -representation are compared and returns the set P' including those pairs whose Euclidean distance is within a threshold δ' , using the similarity join algorithm described in Section III.B. The set of matching pairs P' has been pruned such that most true negatives have been discarded. Further, it is ensured that all true positives are included with no false negatives, since the embedding used is contractive. Finally, all pairs in P' are compared in K dimension, and those pairs whose Euclidean distance is within threshold δ are extracted. The scheme is presented in Figure 3. It is worth mentioning that one advantage of the presented scheme is that it is open to many embedding schemes, as long as they are contractive, and any multidimensional similarity join algorithms. Also, it does not depend on specific similarity functions, whether in the string domain or the Euclidean space.

The values of the embedding parameters K and K' and the similarity thresholds δ and δ' affects the performance, which will be explored by experimental evaluation in Section V. A heuristic for selecting the parameters based on samples from the datasets is proposed and its application is validated against the experimental results and showed to be very effective. The heuristic and its application will be presented in an extended version of this paper.

```

Algorithm: DoubleEmbedding( $Set_1, Set_2, K, \delta, K', \delta'$ )
Input:  $Set_1, Set_2$ : two datasets of Strings of size  $N_1$  and  $N_2$ 
       $K, K'$ : dimension of first and second embedding
       $\delta, \delta'$ : threshold of first and second embedding
Output:  $P$ : set of matching pairs

Combine  $Set_1$  and  $Set_2$  into one set  $O$ 
Embed  $O$  using EmbedString( $O, K$ ) and get  $SE_1[N_1][K], SE_2[N_2][K]$ 
Embed  $SE$  using EmbedNum( $SE, K'$ ) and get  $DE_1[N_1][K']$  and  $DE_2[N_2][K']$ 
// perform similarity join between  $DE_1$  and  $DE_2$ 
Build the KDtree  $T$  for  $DE_1$ 
Set  $P' = \{\}$ ;
for  $i=1$  to  $N_2$ 
   $P' = P' \cup \text{NNSearch}(DE_2[i], T, \delta')$ 
// compare the pairs in  $P'$  in  $K$  dimension
Set  $P = \{\}$ ;
 $\forall$  pairs  $(p_{ij}) \in P'$ 
  if  $d_E^K(SE_1[i], SE_2[j]) \leq \delta$ 
     $P = P \cup (p_{ij})$ 

```

Figure 3: Pseudo code of **DoubleEmbedding** Algorithm

V. EXPERIMENTS

In order to evaluate the potential benefits of the proposed solution, a set of experiments has been conducted on real datasets with different sizes, with the following goals:

- Tune the parameters of the Single Embedding Scheme (SES) while analyzing the quality of the embedding and evaluating the effectiveness of the resulting matching scheme. Several parameters, namely K and δ , are varied experimentally and the distortion of the embedding and the effectiveness of the matching are measured. The goal is to reach a reasonable selection of K and δ and validate them against previous published results.
- Tune the parameters of the Double Embedding Scheme (DES) while analyzing the efficiency and effectiveness of the matching protocol in comparison with the SES.
- Analyzing the efficiency of the DES while varying the size of the datasets, when compared to the SES. Also, the time performance is compared to record matching performed in the original string space.

In the experiments, a real dataset has been used, representing British Columbia voters' list containing 34,264 records of voters' names and addresses. This data is available at <http://www.rootsweb.com/~canbc/vote1898>. Only the first name and last name fields were used in the experiments. Removing all duplicates from the original set resulted in 29,299 distinct records. From such dataset, two datasets are generated where we controlled and identified the percentage of similar records between each set pair. Three different sizes of datasets pairs were generated, namely with each set containing 4,000, 10,000 and 20,000 records respectively in order to evaluate the scalability of the proposed solution. Throughout the experiments the threshold θ in the string space was set to 2.

Efficiency is measured by the total execution time needed to perform the embedding, indexing and matching. Effectiveness of the scheme is analyzed in terms of:

- Recall: the ratio of the number of matched records pairs generated by the matching protocol to the total number of true matched record pairs. This metric has been sometimes referred by others [4, 6, 15, 24] as pairs completeness.
- Accuracy: the percentage of the correctly classified pairs [6, 24]. It is defined as the number of pairs correctly classified as matches or non-matches to the total number of pairs. This metric has been sometimes referred by others [1, 18, 20] as precision.

The platform used for these experiments was a PC with an Intel dual-Core Duo processor 2.2 GHz and 3GB of memory. The protocol was implemented using Java and tested under Windows XP. In the implementation, the SecondString library [5] has been used for similarity matching and Levenshtein distance has been used. The library is available at <http://secondstring.sourceforge.net/>. For indexing the embedded space, the KDTree implementation available at <http://www.cs.wlu.edu/~levy/kd> has been used. The Java source was modified in order to implement the nearest neighbor using range search as described in Section III.B.

A. Selection of Parameters for First Embedding

The selection of a small dimension K would result in a miss representation of the data, hence distance would not be preserved and similar pairs and dissimilar pairs will not be distinguished. However, setting K to a high value results in high cost, both for embedding and matching, and we risk the curse of dimensionality. For the selection of K , samples of 4000 records from the datasets have been embedded in different dimensions and the quality of the embedding is evaluated with respect to the *stress* [12], measuring the distortion of the embedding defined as

$$stress = \frac{\sum_{o_1, o_2} (\delta(F(o_1), F(o_2)) - d(o_1, o_2))^2}{\sum_{o_1, o_2} d(o_1, o_2)^2}$$

Also, the recall and accuracy were recorded. Results when varying K for three dataset sizes (4K, 10K and 20K) are shown in Figure 4. As expected, increasing K results in lower stress values and higher recall and accuracy. Results revealed that with K set to 25 and higher, very small variation in the stress is obtained and very small improvement in the recall and accuracy are reached. Therefore, in the remaining of the experiments, K is set to 25. These results are similar to the results obtained by [13] and [20].

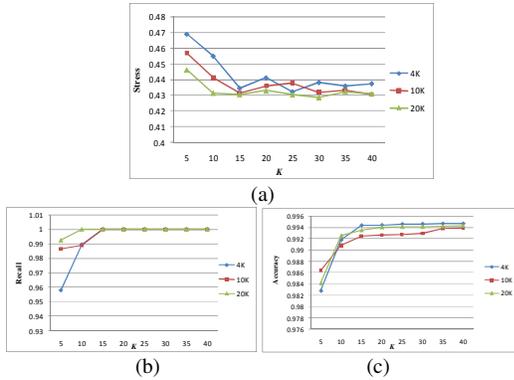


Figure 4: Stress, recall and accuracy of first embedding, varying K

Another important parameter that affects the performance and the effectiveness of the matching protocol is the threshold δ . We ran a set of experiments on the full datasets 4K, 10K and 20K while varying δ from 0.1 to 2. It should be noticed that there is no need to set δ higher than 2 since the mapping used is contractive. Since we knew which records were the true matches, therefore we could compute the recall and accuracy. Results are shown in Figure 5.

As expected, increasing δ results in improving the recall (Fig 5(a)) as large value of δ ensures that all true matches are included in the results returned from the matching protocol, at the cost of an increase in the matching time. The recall reaches good values close to 1 for δ larger than 1.6. At $\delta=1.8$ the recall reaches 100% for all the three data sets. However, increasing δ results in a decrease in the accuracy as larger values of δ results in more false positives returned. However, it is noticed from Figure 5(b) that the decrease in accuracy is very small, ranging from 0.5% to 0.6% at $\delta=1.8$. On the basis of these experiments, the chosen embedding parameters were for K to be set to 25 and for δ to be set at 1.8.

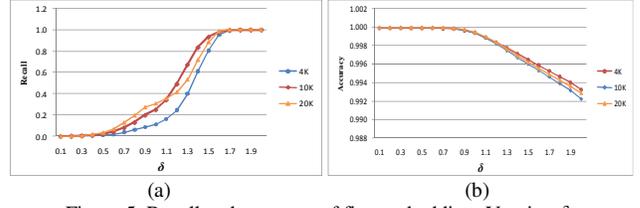


Figure 5: Recall and accuracy of first embedding, Varying δ

B. Selection of Parameters of Second Embedding

In this section, the effectiveness of the proposed Double Embedding Scheme (DES) is evaluated, measured in terms of the recall and accuracy as well as its efficiency measured in terms of the total execution time. The two parameters affecting the performance of the double embedding are K' and δ' . A set of experiments has been conducted varying K' from 2 to 10 and varying δ' from 0.1 to 2 for each K' . Again, δ' does not need to be larger than 2 since the FastMap algorithm used in the second mapping is contractive. The experiments are repeated for the three datasets in order to demonstrate the scalability of the protocol and to observe the effect of variation of the parameters K' and δ' when the data size increases.

Figure 6 shows the recall and accuracy for the 4K dataset, varying δ' from 0.1 to 2 for different values of K' . The results of the Single Embedding Scheme (SES) are also shown for comparison, with $K=25$ and $\delta=1.8$.

As expected, increasing δ' results in an increase in the recall as larger values of δ' increases the number of true values returned from the matching protocol. High recall values are reached for $\delta' > 1.4$ for all K' . The primary reason is that the embedding used is contractive and provides a good distance/similarity preservation. The accuracy on the other hand decreases as δ' increases, since more false positives are returned. However, it reaches the accuracy of SES for $\delta' > 1.4$. The same results were obtained for the 10K and 20K datasets, not shown for space constraint.

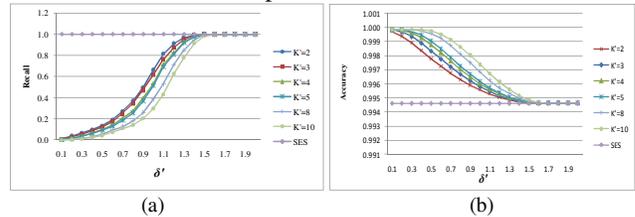


Figure 6: effectiveness of DES for 4K dataset

Figure 7(a) shows the execution time of DES while varying δ' and K' . It is observed that the cost increases as δ' increases. This is expected as while matching in the KDTree, less pruning is done as there are more nodes to be retrieved for larger δ' . Also, this increase is due to the increase in the number of potential matching pairs generated from the first level matching. However, for all δ' and K' , the cost of the DES is substantially lower than SES, resulting in a minimum of 30% improvement for all K' . The effect of the variation of K' on the cost is somehow complex since it consists of three components. The first component is the embedding time, which increases as K' increases. The second component is the cost of indexing, that is building the KDtree for one of

the embedded sets, then applying the nearest neighbor algorithm using range search for the second set. This cost also increases with the increase of K' . The third component, is the final stage of matching, which consists of computing the Euclidean distance between the set of matching pairs resulting from indexing and searching the KDtree. This cost is dependent on the number of matching pairs returned, which decreases as K' increases. This decrease is attributed to a more accurate representation of the embedded records, hence more accurate matching pairs are obtained.

Figure 7(b) shows the total execution time for δ' larger than 1.4. It is plotted separately in order to show the effect of varying K' more closely. δ' is chosen from 1.4 to 2.0 since this is the range δ' will be chosen from to achieve a good recall. It is observed that the cost of DES gives improvement for $\delta' > 1.4$ ranging from 30% to 64% than SES for all values of K' . The lowest cost is achieved with $K'=3$, which achieves the best balance in the cost of the indexing versus the number of potential matching pairs. For $K'=2$, the cost is higher than $K'=3$ because the number of detected pairs in the first matching phase is much higher than for $K'=3$ (2.2m pairs versus 1.7m pairs for $\delta'=1.5$). Hence, the pair matching cost is higher, which increases for larger values of δ' . For $K'=4$ and 5, the cost is quite similar. It is higher than that of $K'=3$ because the increase in the indexing and matching cost (KDTree) is higher than the decrease in the matching pair cost. They are higher than $K'=2$ for δ' smaller than 1.5, then they outperform $K'=2$ since the increase in their indexing and matching cost is lower than its increase in the matching pair cost. When K' is set to 8 and 10 the cost gets higher as the indexing and matching cost increases.

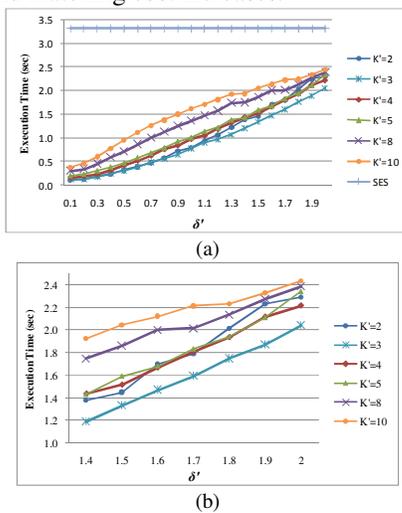


Figure 7: Cost of DES for 4K dataset

Running the same experiments on the 10K dataset revealed that although the improvement in the cost decreases as δ' increases, it ranges from 30% to 60% for all K' for $\delta' > 1.4$. Figure 8(a) shows the cost for the 10K dataset for $\delta' > 1.4$ and Figure 8(b) shows the breakdown of the total cost for $\delta'=1.5$ for all K' . It is noticed that for $\delta'=1.5$, the total cost reaches its minimum at $K'=4$, where the increase in the embedding time and the KDTree is lower than the decrease in the pair matching time. From Figure 8(a), it is observed

that $K'=4$ and 5 yield the best balance between the three components. The cost of $K'=2$ and 3 is higher because the matching pair time is dominant. The cost of $K'=8$ and 10 are higher than all where the indexing and matching (KDTree) increase is more dominant.

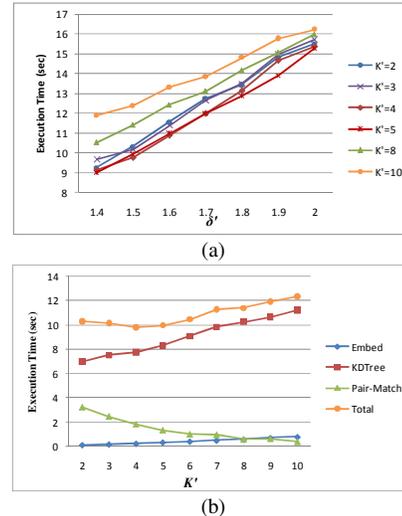


Figure 8: cost of DES for 10K dataset

Figure 9(a) shows the cost for the 20K dataset for $\delta' > 1.4$. Figure 9(b) shows that for $\delta'=1.5$, the total cost reaches its minimum at $K'=6$, where the increase in the embedding time and the KDTree is lower than the decrease in the pair matching time. From Figure 9(a), it is observed that $K'=6$ yields the best balance between the three cost components, followed by $K'=5$, then $K'=8$. The cost of $K'=2$ and 3 experience the highest cost, especially for δ' larger than 1.4 where the matching pair time is dominant. The cost of $K'=10$ is high, specially for small value of δ' , where the indexing and matching (KDTree) increase is more dominant. As δ' increases, K' set to 10 shows lower cost than K' set to 4 or smaller, as the increase in the matching pair time is minimal compared to smaller K' .

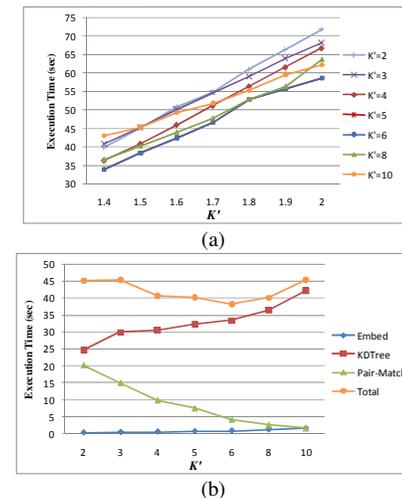


Figure 9: Cost of DES for 20K dataset

The above experiments show that the selection of K' affects the improvement of the cost, and is dependent on the

size of the dataset. As the size of the dataset increases, larger values of K' yields lower cost. However, it is shown that for δ' set to 1.5, the worst selection of K' would result in a 40% improvement, while an optimum selection can lead to improvement ranging from 50% to 60% over SES.

C. Effect of Datasize Variation

To evaluate the scalability of DES, its run time is compared with the run time of matching records in the original space, varying the datasets from 4K to 20K, shown in Figure 10(a). It is obvious that matching strings requires by far more time due to $O(N^2)$ string distance computations and the difference is more dramatic as the data size increases. The scalability of the protocol is studied also in comparison with SES as shown in Figure 10(b). The parameters used for SES were $K=25$, $\delta=1.8$, and for DES $\delta'=1.5$ and $K'=3$ for $N=4K$, $K'=4$ for $N<10K$, $K'=5$ for $N<16K$ and $K'=6$ for $N<20K$. The results show that DES outperforms SES, especially for large datasets, showing improvement ranging from 59% to 64%.

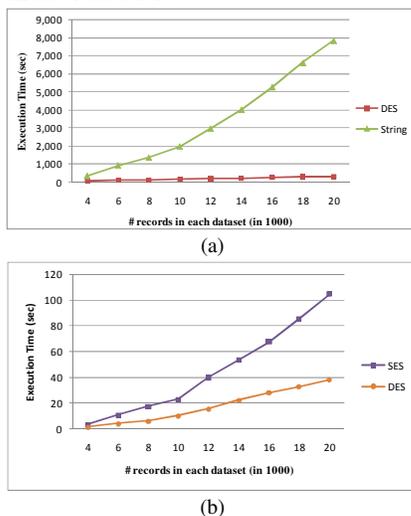


Figure 10: Cost of DES vs string matching and SES

VI. CONCLUSIONS

This paper introduced a novel scheme for record linkage based on double embedding of the data, aiming at improving the efficiency. A two level matching is proposed, with the first level performing a fast and inaccurate matching, ensuring high recall while the second level performs a more expensive matching, on a smaller set of pairs, to improve the accuracy. Experimental evaluation on real datasets revealed that, by using contractive embedding techniques that preserve the distance between records values, the suggested scheme outperforms the single embedding scheme achieving gains in time performance ranging from 30% to 60%, while achieving the same level of recall and accuracy. Future work will address scenarios with more than two parties and different data types such as DNA sequence, etc.

REFERENCES

[1] A. Al-Lawati, D. Lee, P. McDaniel, Blocking-aware Private Record Linkage, *IQIS* 2005.

[2] R. Baxter, P. Christen, A comparison of fast blocking methods for record linkage, *In KDD Workshop on Data Cleaning, Record Linkage and Object Consolidation*, 2003

[3] J. Bourgain, On Lipschitz Embedding of Finite Metric Spaces in Hilbert Space, *Israel Journal of Mathematics*, no. 1-2, 1985

[4] P. Christen, Automatic record linkage using seeded nearest neighbour and support vector machine classification. *In Proc. of 14th ACM SIGKDD Intl Conf. on Knowledge Disc. and Data Mining*, Aug 2008

[5] W. Cohen, P. Ravikumar, S. Fienberg. A comparison of string distance metrics for matching names and records. *In KDD Workshop on Data Cleaning, Record Linkage and Object Consolidation*, 2003

[6] M. Elfeky, V. Verykios, A. Elmagarmid: TAILOR: A Record Linkage Toolbox. *In Proc. of ICDE*, 2002

[7] A. Elmagarmid, G. Panagiotis, S. Verykios, Duplicate Record Detection: A Survey, *IEEE TKDE*, Vol. 19, no. 1, 2007

[8] C. Faloutsos, K. Lin. FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. *SIGMOD Record*, 24(2):163–174, June 1995.

[9] V. Gaede, O. Günther, Multidimensional access methods, *ACM Computing Survey*. 30, 2, June 1998

[10] M. Hernandez, S. Stolfo, Real-world data is dirty: data cleansing and the merge/Purge problem, *Data Mining and Knowledge Discovery* 2(1):9-37, 1998.

[11] G. Hjaltason, H. Samet, Incremental distance join algorithms for spatial databases, *ACM SIGMOD*, 1998

[12] G.R. Hjaltason, H. Samet, Properties of Embedding Methods for Similarity Searching in Metric Spaces, *IEEE TPAMI*, Vol. 25, 2003

[13] G. Hristescu, M. Farach-Colton, Cluster-preserving embedding of proteins, Technical Report, Rutgers Univ., Piscataway, 1999.

[14] E. Jacox, H. Samet, Metric space similarity joins, *ACM Transactions on Database Systems*. 33, 2 Jun. 2008

[15] L. Jin, C. Li, S. Mehrotra, Efficient Record Linkage in Large Data Sets, *DASFAA*, 2003.

[16] N. Koudas, S. Sarawagi, D. Srivastava, Record Linkage: Similarity Measures and Algorithms, *ACM SIGMOD*, 2006.

[17] N. Linial, E. London, Y. Rabinovich. The geometry of graphs and some of its algorithmics application, *Combinatorica*, vol 15, 1995

[18] A. McCallum, K. Nigam, L. Ungar, Efficient clustering of high-dimensional data sets with application to reference matching. *Proc. 6th ACM SIGKDD Intl Conf. on Knowledge Disc. Data Mining*, 2000

[19] A. Moore, "An Introductory Tutorial on Kd-Trees." Extract from Efficient Memory-based Learning for Robot Control (Technical Report 209). Computer Laboratory, University of Cambridge, 1991.

[20] M. Scannapieco, I. Figotin, E. Bertino, A. Elmagarmid, Privacy preserving schema and data matching. *In Proceedings of the 2007 ACM SIGMOD Intl Conference on Management of Data 2007*

[21] W. Shen, P. DeRose, L. Vu, A. Doan, R. Ramakrishnani, Source-aware Entity Matching: A Compositional Approach. *In Proc. of 23rd Intl Conf. on Data Eng., ICDE*, April, 2007

[22] D. Talbert, D. Fisher, An empirical analysis of techniques for constructing and searching k-dimensional trees. *In Proc. of 6th ACM SIGKDD Intl Conf. on Knowledge Disc. and Data Mining*, Aug 2000

[23] V. Verykios, A. Elmagarmid, E. Houstis, Automating the approximate record-matching process. *Info. Sciences*, 126, July 2000

[24] V. Verykios, M. Elfeky, A. Elmagarmid, M. Cochinwala, S. Dalal. On the accuracy and completeness of the record matching process. *In Proc. of the 2000 Conf. on Information Quality*, Oct.2000.

[25] J. Wang, X. Wang, K. Lin, D. Shasha, B. Shapiro and K. Zhang, An Index Structure for Data Mining and Clustering, *Knowledge and Information Systems*, vol. 2, no. 2, May 2000

[26] J. Wang et al, Evaluating a Class of Distance-Mapping Algorithms for Data Mining and Clustering", *Proc. ACM SIGKDD Intl Conf. Knowledge Discovery and Data Mining*, Aug 1999

[27] M. Yakout, M. Atallah, A. Elmagarmid, Efficient Private Record Linkage, *In Proc. of 25th Intl Conf. on Data Eng., ICDE*, April 2009

Feature extraction for graph datasets

Gideon Dror

Abstract—State of the art graph kernels are based on extracting local features. In this work we propose a method for extracting features from unlabeled graphs based on sampling non-isomorphic connected induced subgraphs, also known as graphlets. The method allows systematic expansion in graphlets size, and controls the tradeoff between sampling time on the one hand, and the resolution of the representation on the other hand. We apply the method to artificial and real life graph datasets from biochemistry, bioinformatics and computer science domains, associated with supervised classification or regression tasks. In all cases the merits of the method are evident. We empirically show that at least for the datasets considered, sampling large size graphlets improves the performance of supervised classification and regression.

I. INTRODUCTION

Graph comparison is a fundamental problem in many areas such as software verification, bioinformatics, chemistry, sociology, signal processing and telecommunication. Existing graph comparison algorithms may be classified into three categories: set based, frequent subgraph based and kernel based. Set based approaches represent a graph as a set of edges, as a set of nodes, or both. Graphs are then compared by estimating the similarity between pairs of edges or pairs of nodes between two graphs. These approaches are rather simplistic, as they overlook the structure of the graphs, i.e., their topology. Frequent subgraph algorithms (Yan & Han, 2003) identify subgraphs which are abundant in a given graph dataset. Feature selection is then applied to select the most discriminative subgraphs. Graph kernels represent an attractive middle ground. Various graph kernels that are based on different types of substructures of graphs have been proposed, as random walks (Gärtner et al., 2003; Kashima et al., 2004), shortest paths (Borgwardt & Kriegel, 2005), subtrees (Ramon & Gärtner, 2003), and cycles (Horvath et al., 2004). Several studies have recently shown that these graph kernels can achieve state of the art results on problems from bioinformatics and chemistry (Ralaivola et al., 2005). However, given a specific task, it is not clear which variant would be able to capture the characteristics of the problem that are essential for the execution of the task. Ideally one would like to have a rich representation of a graph that would be generic and flexible for dealing with a wide range of graph distributions and tasks.

A. our contribution

Previous graph kernel methods have been always constrained to specific subgraph classes, e.g. chains, trees or cycles, or to subgraphs of limited size, usually comprising

no more than five vertices. In this work we suggest a method for extracting information about induced subgraphs of essentially unrestricted structure and size. The basic idea is to sample connected induced subgraphs of a given graph, and for each subgraph to find a canonical form. By sampling sufficient number of connected subgraphs, one can estimate the frequency of each canonical form, thereby obtaining a signature of the underlying graph. By sampling subgraphs of various number of vertices one obtains a hierarchical representation: samples of small subgraphs produce robust, dense but low resolution features, whereas samples of large subgraphs produce non-robust, very sparse but high resolution features. As we show in the sequel, such multi-resolution representation can be practically very effective in learning graph data of various domains.

II. SAMPLING NON-ISOMORPHIC CONNECTED SUBGRAPHS

A. Notation and Definitions

A *graph* is an ordered pair $G = (V, E)$ where V is a set of vertices and E the set of edges, which are 2 element subsets of V . The size of a graph is the size of V . To avoid confusion, the graphs we refer to are unweighted, simple graphs.

A graph $G_s = (V_s, E_s)$ is an *induced subgraph* of G if $V_s \subset V$, $E_s \subset E$ and E_s contains all edges of E that connect vertices in V_s .

An *adjacency matrix* of an unweighted simple graph G is a symmetric matrix A of size $|V| \times |V|$. The (i, j) -th entry of A is 1 if there is an edge $e \in E$ whose endpoints are the i -th and j -th elements of V , and zero otherwise.

An *isomorphism* of graphs $G = (V, E)$ and $G' = (V', E')$ is a bijection between the vertex sets of G and G' , $f : V \rightarrow V'$ such that any two vertices u and v of G are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in G' . Graph isomorphism is an equivalence relation on graphs, therefore it partitions the class of all graphs into equivalence classes, called *isomorphism classes*. Two graphs that belong to the same isomorphism class are said to be *isomorphic*.

A *canonical form* of a graph G is a graph $Canon(G)$ that is isomorphic to G , such that $Canon(G) = Canon(G')$ if and only if G and G' are isomorphic.

A *canonical label* $Cl(G)$ of graph G is a unique code, invariant to the ordering of the vertices V . As such, $Cl(G)$ is identical for all graphs isomorphic to G . $Cl(G)$ can be taken as the bit-string obtained by concatenating the elements of the upper triangle of the adjacency matrix, where the columns and rows were symmetrically permuted such that the resulting string be lexicographically minimal (or maximal). In what follows we will loosely use the term *graphlet* to

Gideon Dror is with the Department of Computer Science, The Academic College of Tel-Aviv-Yaffo, Tel Aviv 61083, Israel (Phone +972-3-5444646, Fax: +972-3-6803342; email: gideon@mta.ac.il).

represent both the canonical form of a graph or its canonical label.

B. Graphlet sampling

Researchers have been using graphlets as a features that serve as a basis of characterizing graphs (Kashtan et al., 2004; Borgwardt et al., 2007; Przulj, 2007). More specifically, by counting the number of times each graphlet is embedded in a graph it is possible to get rich representations of graphs that encode both local and global information about the graph. This representation is similar to the popular 'Bag of words' representation in text categorization, or related representations in texture recognition, where small patches of an image serve as local features.

However, in previous works the set of graphlets considered was quite limited, usually to graphlets of small size (e.g. a maximum of 4 or 5 vertices) or having predetermined structure (e.g. trees, paths or cycles). The main reason for this restriction is that the number graphlets is exponential in the number n of vertices

$$N(n) = \frac{2^{n(n-1)/2}}{n!} \left(1 + O\left(\frac{n^2}{2^n}\right) \right), \quad (1)$$

hence it is exhaustive enumeration and counting of all graphlets is prohibitive even for moderate values of n . For example, for $n = 10$ there are already 11716571 distinct connected graphlets.

However, when the graph G is sparse, most graphlets are not induced by the graph, especially for large n . Sampling subgraphs would be the solution to this sparsity, and would enable an estimate of the statistics of induced subgraphs with reasonable time and space resources. This claim would be demonstrated in the experiments that follow.

In this work, sampling induced subgraphs is performed in a method similar to the one used in (Kashtan et al., 2004). Basically, given a graph G we sample a random edge, which, together with its endpoints, may be considered as a subgraph $g(v, e)$. We then randomly select a single vertex w from the set of vertices connected to g . $v' = v \cup \{w\}$ induces a new subgraph g' of G . Further vertices are added to the subgraph in the same manner until the number of vertices reaches some predefined limit, or until e exhausts all edges connected to v . In either case, the next sample starts ab-initio. Notice that unlike (Kashtan et al., 2004), we do not weigh sampled subgraphs for compensate for sampling bias, as this allows considerable speedup of the sampling process which may well compensate for the sampling bias. Notice that since the bias is fixed, then broadly speaking, similar graphs should still have similar subgraph statistics.

One of the merits of this method is that by progressively growing the sampled subgraph, it is possible to collect the statistics not only of the final subgraph, but of all subgraphs produced as intermediate steps. This way one automatically gets a hierarchy of features ranging from small, high-count features to large, low-count features. Such multi-resolution representations are very effective in learning complex objects such as images, text documents and DNA sequences.

Algorithm 1 Graphlet sampling

Input: graph $G = (V, E)$, maximal size m , #samples N

Output: sparse vector X indexed by canonical forms (i.e. graphlets);

repeat

 Sample an edge $e \subset E$. $v \leftarrow$ endpoints of e .

while $|v| < m$ **do**

 select a random vertex $v_i \in V \setminus v$ connected to any vertex in v .

$v \leftarrow v \cup \{v_i\}$;

$g \leftarrow G[v]$ is the subgraph induced by v .

$\tilde{g} \leftarrow Canon(g)$. {using 'nauty'}

end while

until number of samples = N

$X(\tilde{g}) \leftarrow$ #sampled occurrences of \tilde{g}

Since we are dealing here with unlabeled graphs, the representation should be invariant to permutations of vertices. To this end, each induced subgraph sampled is mapped to its canonical form, and count statistics of canonical forms are collected.

The problem of graph isomorphism is neither known to be NP-complete nor polynomial. Still, very effective algorithms exist for graphs of sizes up to several hundred vertices. Nauty (No AUTomorphisms, Yes?) (McKay, 1981), is considered the most efficient implementation for isomorphism testing¹. It can process most graphs of 100 vertices within less than a second. On a 2GHz Pentium 4 machine it could find the canonical isomorphs of a random collection of thousand graphs with nine vertices within 13 milliseconds.

C. Algorithm Convergence

Determining how many samples should be collected is known to be a hard problem in general, especially when the underlying distribution is not known in advance (Chaudhuri et al., 1998). When sampling graphlets of different sizes, large sized graphlets will usually require exponentially more samples than small sized graphlets. So the question of 'how many graphlets to sample' is further aggravated by the question 'what is the size of maximal graphlet size to be considered?'. This, of course, is not known a-priori, as it depends crucially on the task at hand. The experiments conducted in section III demonstrate this issue. In this work we took the simplest method, of collecting as much samples as we could within time constraints. This would lead to reliable estimates for small-sized graphlets and unreliable estimates for the large-sized graphlets. So while possibly many non-informative and noisy features are collected, various machine learning technique should be able, in principle, to find and exploit the informative features buried in the haystack.

¹<http://cs.anu.edu.au/bdm/nauty/>

III. EXPERIMENTS

A. Artificial Graph Dataset

To get a better intuition about the properties of the method, we devised an artificial dataset of graphs. It is based on the following five graphs:

- SQ2D: A square grid, with 8×8 vertices, with periodic boundary conditions (namely, a torus);
- SQ3D: A cubical grid, with $4 \times 4 \times 4$ vertices, with periodic boundary conditions.
- LADDER: ladder graph, with 32×2 vertices, with periodic boundary conditions.
- 4LADDER 16×4 square grid, with periodic boundary conditions (namely, a 16×4 torus).
- HEX: A hexagonal grid with 66 vertices embedded on the surface of a sphere.

We used each basic graph as a prototype for producing 100 examples. Each example is a perturbed version of the prototype, with a vertex set identical to that of the prototype. The edge sets were modified by removing edges randomly with probability 0.05 and introducing new edges between any two disconnected vertices with probability 0.05; This significantly increased the number of edges as compared to the prototype.

Notice that for the first four basic graph types the shortest cycle is a square, and is a triangle for the hexagonal grid. The perturbed examples do not necessarily exhibit this property. Notice further that based on local properties, discriminating between some graphlet classes would be non trivial.

1) *Sampling and Preprocessing*: From each example we sampled a total of 60000 subgraphs sizing between 4 to 9 vertices, namely 10000 examples for each graphlet size. These samples were used to produce a sparse graphlet statistics representation vector. Since the number of unique graphlets is exponential in the number of vertices, the counts of small size graphlets are expected to be orders of magnitude larger than that of large size graphlets. In order to balance the scale of different features, we normalized each feature so it lies in the unit interval

$$X_j^i = \frac{C_j^i}{\max_i(C_j^i)} \quad (2)$$

where C_j^i is the count of graphlet j in example i and X_j^i is the corresponding dimension of the feature vector. We applied common practice and eliminated graphlets that were observed in less than 3 examples, insignificantly reducing the number of features from 32416 to 32354.

2) *Performance*: We assessed the informativeness of the graphlet sampling representation by a stratified 5-fold cross validation of a SVM soft margin linear classifier. For simplicity, we implemented a multi-class classifier using all pairwise binary classifications (Hastie & Tibshirani, 1996). With the full representation, classifiers turned out to be 100% accurate, which was not surprising in hindsight.

3) *Performance as a function of graphlet sizes*: To demonstrate the importance of sampling large size graphlets we calculated the accuracy of pairwise classifiers when graphlets are sampled up to varying values of maximal graphlet size m . Figure 1 depicts the accuracy of pairwise classifications, averaged over 20 repetitions. Accuracies are calculated by stratified 5-fold cross validation. We plotted only the pairwise classifications between the square grid classes (SQ2D, SQ3D, LADDER and 4LADDER) as the HEX class was perfectly discriminated from the other classes even for $m = 4$ graphlets. It can be seen that accuracies monotonically increase as richer graphlets are sampled, until perfect performance is achieved for all cases at $m = 8$.

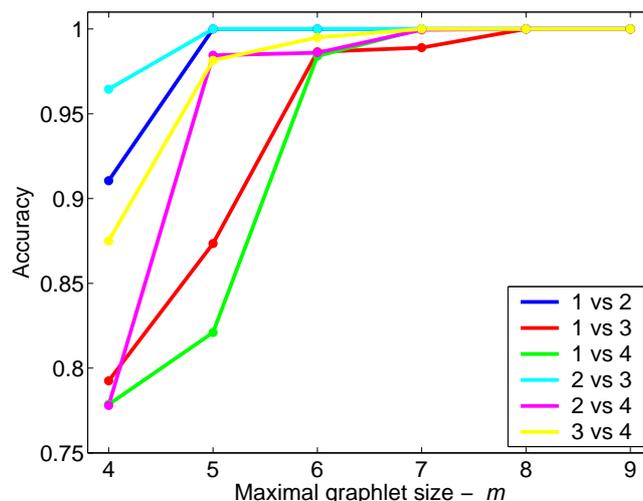


Fig. 1. Average accuracy of pairwise classification between graph classes SQ2D, SQ3D, LADDER and 4LADDER, (encoded as 1, 2, 3 and 4 respectively) as a function of m , the maximal size of graphlets sampled. The monotonicity of all graphs is evident.

4) *Informative Features*: Analyzing the features revealed that the classifiers picked graphlets of various sizes to predict the correct class. As stated, the case of discriminating the hexagonal grid from all square grids turned out to be trivial, as this is the only dataset with examples that are abundant with graphlets containing 3-cycles. Examples of informative features for discriminating between square-grid classes are depicted in Figure 2. The features are simple to interpret, e.g. 2(a) shows a graphlet discriminating between SQ2D and SQ3D. The correlation of this feature with the target variable is 0.929. Evidently, the three dimensional square grid is abundant with vertices having degree 6, whereas the maximal degree for a two dimensional square grid is 4. Hence this feature implicitly capture the simple fact that, in this case, the degree distribution is a very strong discriminator.

B. Bioinformatics and Chemistry

1) *Datasets*: The benchmark datasets we used here contain three real-world datasets: MUTAG (Debnath et al., 1991) and PTC (Toivonen et al., 2003) are molecular compounds datasets, and DD (Dobson & Doig, 2003), a dataset for

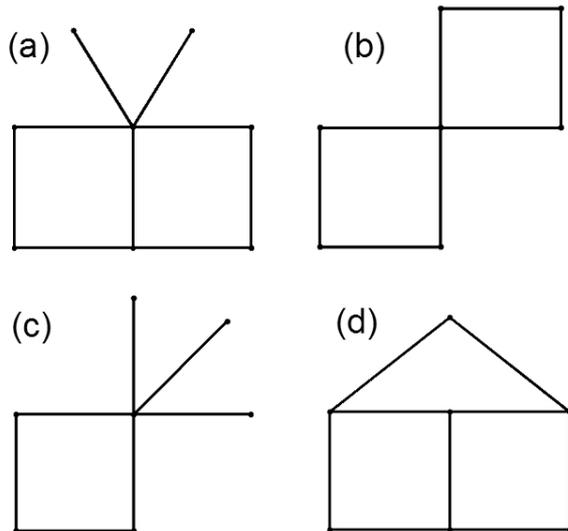


Fig. 2. Examples of graphlets that are highly discriminative in pairwise classifications. The graphlets correspond to the most discriminating features between: a) SQ2D and SQ3D; b) SQ2D and LADDER c) SQ3D and 4LADDER; d) LADDER and 4LADDER.

TABLE I
CHARACTERISTICS OF BIOINFORMATICS AND BIOCHEMISTRY GRAPH DATASETS

Dataset	Examples	% Pos	Vertices	Edges
MUTAG	188	0.335	17.7	38.9
PTC	344	0.442	26.7	50.7
D & D	1178	0.498	284.4	1921.6

protein function prediction task. In this work we used the unlabeled version of these graphs, see e.g. (Borgwardt et al., 2007). The characteristics of the three datasets are detailed in table I.

2) *Sampling and Prediction:* We used a sampling and preprocessing methodology techniques identical to the ones used for the artificial dataset, subsection III-A. We used soft margin linear SVM classifier to test the quality of the sampled graphlet representation, and compared it with the performance of other state of the art graph kernels. Table II compares the classification accuracy on graph benchmark datasets of several graph kernels.

TABLE II
CLASSIFICATION ACCURACY ON GRAPH BENCHMARK DATASETS

Dataset	Random walk	Shortest path	Graphlet enumeration	Graphlet sampling	Graphlet FS
MUTAG	0.719	0.813	0.822	0.855	0.865
PTC	0.554	0.554	0.597	0.606	0.635
DD	>24h	>24h	>24h	0.799	0.841

3) *Performance:* We compare the proposed graphlet sampling kernel with random walk kernel, shortest path kernel, graphlet kernel enumerating all graphlets up to 4 vertices and current graphlet sampling kernel (GS) on the benchmark datasets. For the graphlet sampling kernel we also add a feature selection variant (FS) with moderate feature selection as a preprocessing step. This is an effective method for dealing with very sparse representations (Guyon & Elisseeff, 2003). In this version, half of the features are eliminated using a simple filter, based on the absolute value of the Pearson correlation of each feature with the target variable. Features whose score falls below the median score are eliminated. The elimination of non-informative features improves classification accuracy. From table II, we can see the graphlet sampling Kernel even without feature selection still outperforms the other three kernels in terms of classification accuracy over all three benchmark datasets. Notice that '>24h' means computation did not finish within 24 hours.

C. SAT Problems

SAT is one of the most studied problems in computer science, representing a generic constraint satisfaction problem with binary variables and arbitrary constraints. SAT is known to be a NP-Hard problem, therefore it is not surprising that it has become a primary platform for investigating of average-case and empirical complexity. Randomly-generated SAT instances have been extensively used in research and SAT competitions, since this test-bed offers a range of very easy to very hard instances for any given input size.

An important non trivial subtask in SAT solving is prediction of running time. This is the time required by a specific SAT solver to solve a SAT problem, namely to either find an assignment that satisfies all constraints or to declare it unsatisfiable. Robust estimation of running times is essential in devising effective SAT solvers, such as SATzilla (Nudelman et al., 2004), which are based of SAT solver portfolios; Such solvers exploit the fact that SAT solvers are not correlated in their running times; Accurate prediction of running times of the a wide range of solvers would enable to devise a hard to beat solver.

Although it is not possible to predict SAT solver running time on a general SAT problem, it is still possible to estimate it for restricted classes of problems; SAT problems from a specific domain would have a certain structure distribution, that may be learned, to some degree by machine learning techniques. (Nudelman et al., 2004) used a wide variety of features such as the the number of clauses, the number of variables, and the ratio of the two etc. to build the first successful portfolio SAT solver.

1) *Graph representation of SAT problems:* There are three main ways to encode a SAT problem as a graph: Variable-Clause-Graph is a bipartite graph with a vertex for each variable, a vertex for each clause, and an edge between them when a variable occurs in a clause. Variable-Graph has a vertex for each variable and an edge between variables that occur together in at least one clause. Clause-Graph has vertices representing clauses and an edge between two

clauses whenever they share a negated literal. All of these graph representations correspond to constraint graphs for the associated constraint satisfaction problem; thus, they encode the problems hardness.

2) *Dataset*: We work with a dataset of 1000 artificially created unsatisfiable problems, with a fixed ratio between clauses and variables. Each example is a 3SAT problem with 1704 clauses and 400 variable, so that the clauses to variables ratio is held fixed at 4.26. This fact is crucial as the clauses to variables ratio is an order parameter for SAT, and the value 4.26 is at the critical point (Nudelman et al., 2004). Still the dataset exhibits non trivial variance of running times. Clause-graphs that were calculated had on average 1704 vertices and 16383 edges. The target for each example was taken as the logarithm of the actual running time, averaged over several SAT solvers (KCNFS, OKSOLVER, SATZ, MARCH, ZCHAFF and MINISAT20).

3) *Sampling*: Sampling and preprocessing used the settings employed in III-A and III-B. The resulting graphlet representation consisted of a total of 5973 unique graphlets. To quantify the sparsity of the representation, we notice that 3495 graphlets were observed in less than 5% of the examples, and 513 graphlets were observed in more than 95% of the examples. Such feature distribution are often observed in count data in high dimensional feature spaces, e.g. in Bag of words representation of text documents.

4) *Feature selection and prediction*: We used linear regression to learn the targets and used 5-fold cross validation to measure the generalization performance. Within each fold we selected features by using False Discovery Rate (FDR) method (Benjamini & Hochberg, 1995) based on the p-value of the Pearson correlation between each feature and the target (logarithm of running time). Using FDR guarantees a fixed expected fraction of noisy features, but not a fixed number of features. Parameter r of FDR was taken as $\log_2(r) \in \{-7, -6 \dots -2\}$. The Pearson correlation of the predicted value and the targets are depicted in Figure 3 as a function of $\log_2(r)$.

5) *Performance*: Evidently, the correlations obtained between predicted and actual log running times are inferior to those obtained by more elaborate methods, e.g. (Nudelman et al., 2004). We did not use any global or otherwise engineered features and used a very simple learning and feature selection scheme. Indeed, our aim here was not to beat state-of-the-art SAT predictors. Thus, in spite of the moderate correlations, they are very significant statistically, assessing that the graphlet sampling technique succeeded in extracting important information about the target. To what extent this information can be used to improve the performance of state-of-the-art SAT regressors is an open question.

6) *Feature Analysis*: To identify which features are most informative about the target we scored each feature by its Pearson correlation coefficient with the target and sorted the features by the absolute value of this score. Graphlets that correspond to the four most correlated features are depicted in Figure 4. Interestingly the first 10 features are strongly

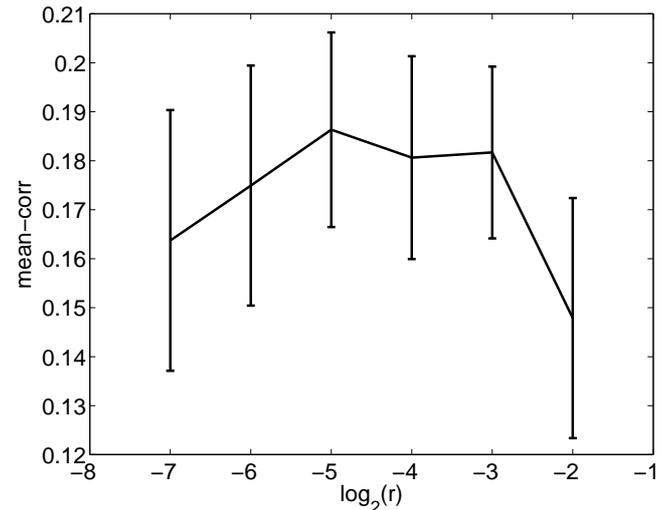


Fig. 3. Performance of SAT prediction: Pearson correlation between log running times predicted by simple linear regression to the actual log running times as a function of r FDR parameter, based on 5-fold cross validation. Lower values of r represent more aggressive feature selection (i.e. fewer, more significant features). The graph depicts the mean values and their standard deviations, obtained by 20 repetitions of the experiment.

dominated by features with negative correlations with the target.

Obviously, we do not claim that these feature are informative in general SAT problems, but that they are for the specific problem type and data distribution. Furthermore, the richness of the graphlet sampling representation allows for identifying informative features for other SAT problem distributions.

IV. DISCUSSION

The graphlet sampling method cannot be used as is for effective portfolio management, since for this application features must be inexpensive to extract. Collecting sufficient number of samples of many graphlets, as in this work, would be clearly be too time consuming. However, it is possible to use the graphlet sampling methodology offline, to identify few informative features, which in runtime will be efficiently extracted (Koyuturk et al., 2004; Borgwardt et al., 2007).

Finally, there are many open question related to the proposed method: For example, whether different sampling method, e.g. MCMC, would be more effective, and would scale to larger size graphlets? What is the effect of non-trivial dependencies among graphlets that are subgraphs of other graphlets? Is it possible to get better representation by sampling even larger size graphlets and compressing the resulting very sparse representation into a 'dense' representation, in approximately lossless manner?

REFERENCES

- Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society - Series B*, 57, 289 – 300.

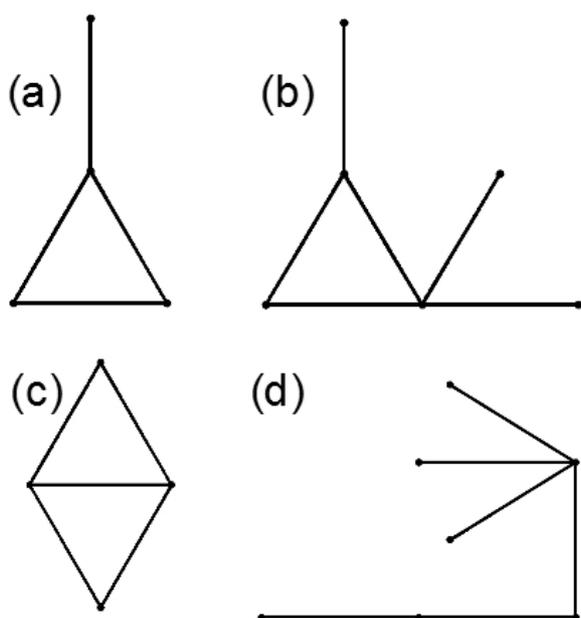


Fig. 4. Example for graphlets that correspond to features that are strongly correlated with the target. Only graphlets that were observed in at least 50 examples were considered. Interestingly, most features strongly correlated with the target correspond to simple graphlets containing one or several 3-cycles, and are negatively correlated with the target. For example, the feature that correspond to graphlets (a)-(c) have correlations -0.181, -0.166 and -0.147 respectively whereas (d) has a correlation of 0.124.

- Borgwardt, K. M., & Kriegel, H.-P. (2005). Shortest-path kernels on graphs. *Proc. Intl. Conf. Data Mining* (pp. 74–81).
- Borgwardt, K. M., Kriegel, H.-P., Vishwanathan, S. V. N., & Schraudolph, N. (2007). Graph kernels for disease outcome prediction from protein-protein interaction networks. *Proceedings of the Pacific Symposium of Biocomputing 2007*. Maui Hawaii: World Scientific.
- Chaudhuri, S., R., M., & Narassaya, V. (1998). Using random sampling for histogram construction: How much is enough? *Proc. ACM SIGMOD Conf.* (pp. 436 – 447).
- Debnath, A. K., Lopez de Compadre, R. L., Debnath, G., Shusterman, A. J., & Hansch, C. (1991). Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *J Med Chem*, 34, 786–797.
- Dobson, P. D., & Doig, A. J. (2003). Distinguishing enzyme structures from non-enzymes without alignments. *J Mol Biol*, 330, 771–783.
- Gärtner, T., Flach, P., & Wrobel, S. (2003). On graph kernels: Hardness results and efficient alternatives. *Proc. Annual Conf. Computational Learning Theory* (pp. 129–143). Springer.
- Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 1157–1182.

- Hastie, T., & Tibshirani, R. (1996). *Classification by pairwise coupling* (Technical Report). Department of Public Health Sciences and Statistics, University of Toronto, Canada.
- Horvath, T., Gärtner, T., & Wrobel, S. (2004). Cyclic pattern kernels for predictive graph mining. *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)* (pp. 158–167).
- Kashima, H., Tsuda, K., & Inokuchi, A. (2004). Kernels on graphs. *Kernels and Bioinformatics* (pp. 155–170). Cambridge, MA: MIT Press.
- Kashtan, N., Itzkovitz, S., Milo, R., & Alon, U. (2004). Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, 20, 1746–1758.
- Koyuturk, M., Grama, A., & Szpankowski, W. (2004). An efficient algorithm for detecting frequent subgraphs in biological networks. *Bioinformatics*, 20 Suppl 1, I200–I207.
- McKay, B. D. (1981). Practical graph isomorphism. *Congressus Numerantium*, 30, 45–87.
- Nudelman, E., Leyton-Brown, K., Devkar, A., Shoham, Y., & Hoos, H. (2004). Understanding random SAT: Beyond the clauses-to-variables ratio. *International Conference on Principles and Practice of Constraint Programming (CP)* (pp. 438–452).
- Przulj, N. (2007). Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23, e177–e183.
- Ralaivola, L., Swamidass, S. J., Saigo, H., & Baldi, P. (2005). Graph kernels for chemical informatics. *Neural Networks*, 18, 1093–1110.
- Ramon, J., & Gärtner, T. (2003). *Expressivity versus efficiency of graph kernels* (Technical Report). First International Workshop on Mining Graphs, Trees and Sequences (held with ECML/PKDD'03).
- Toivonen, H., Srinivasan, A., King, R. D., Kramer, S., & Helma, C. (2003). Statistical evaluation of the predictive toxicology challenge 2000-2001. *Bioinformatics*, 19, 1183–1193.
- Yan, X., & Han, J. (2003). Closegraph: mining closed frequent graph patterns. *KDD* (pp. 286–295).

SESSION

PREDICTIVE MODELLING

Chair(s)

Dr. Wolfram-M. Lippe

K-Fold Cross Validation for Error Rate Estimate in Support Vector Machines

Davide Anguita¹, Alessandro Ghio¹, Sandro Ridella¹, and Dario Sterpi²

¹Dept. of Biphysical and Electronic Engineering, University of Genova, Via Opera Pia 11A, I-16145 Genova, Italy,

{Davide.Anguita, Alessandro.Ghio, Sandro.Ridella}@unige.it

²Smartware & Data Mining s.r.l., Via Gabriele D'Annunzio 2/78, I-16121 Genova, Italy, Dario.Sterpi@smartwaredm.it

Abstract—*In this paper, we review the k -Fold Cross Validation (KCV) technique, applied to the Support Vector Machine (SVM) classification algorithm. We compare several variations on the KCV technique: some of them are often used by practitioners, but without any theoretical justification, while others are less used but more rigorous in finding a correct classifier. The last ones allow to establish an upper-bound of the error rate of the SVM, which represent a way to guarantee, in a statistical sense, the reliability of the classifier and, therefore, turns out to be quite important in many real-world applications. Some experimental results on well-known benchmarking datasets allow to perform the comparison and support our claims.*

Keywords: Model Selection, Support Vector Machine, k -fold Cross Validation

1. Introduction

The *Support Vector Machine (SVM)* is one of the state-of-the-art techniques, when facing classification tasks, which belongs to the field of Artificial Neural Networks (ANNs) [1] but is characterized by the solid foundations of *Statistical Learning Theory (SLT)* [2]. Thanks to its extremely good performance in real-world Data Mining applications, it became quickly part of commercial Data Mining suites [3].

The SVM learning is performed by finding a set of parameters (analogous to the weights of an ANN), found by solving a Convex Constrained Quadratic Programming (CCQP) problem, for which many effective techniques have been developed [4]. This is a large improvement with respect to traditional ANNs, which require the solution of a difficult non-linear optimization problem. The search for optimal parameters, however, does not complete the learning process, as there is a set of additional variables (*hyperparameters*) that must be tuned to reach the optimal classification performance, similarly to ANNs, where the hyperparameter is the number of hidden nodes. This tuning is not trivial and is an open research problem [3], [5], [6], [7].

The process of finding the best hyperparameters is usually called the *model selection* phase in the Machine Learning literature and is strictly linked to the evaluation of the

generalization ability of the SVM or, in other words, the error rate attainable by the SVM on new (unknown) data. In fact, it is common use to select the optimal SVM (i.e. the optimal hyperparameters) by choosing the one with the lowest generalization error. Obviously, the generalization error is impossible to compute, but SLT proposes several methods to obtain a probabilistic upper bound for it: using this bound, it is possible to select the optimal SVM and also to assess the quality of the classification.

The methods for performing the model selection phase can be divided in two categories [8], [5]. Theoretical methods, like the Vapnik-Chervonenkis (VC) bound [2] or the margin bound [7], provide deep insights on the classification algorithms but are often inapplicable, incalculable and too loose for being of any practical use [9]. On the other hand, practitioners have found several procedures [3], [5], which work well in practice but do not offer any theoretical guarantee about the generalization error. Some of them rely on well-known statistical procedures but the underlying hypotheses are not always satisfied or are only asymptotically valid. An example is the very well-known *Bootstrap* resampling technique [10], [11]: we have to assume that the error distribution is Gaussian, which is not always the case and can cause the underestimation of the generalization error [12].

One of the most popular resampling techniques is the *k-Fold Cross Validation (KCV)* procedure [13], which is simple, effective and reliable [14], [15], [16]. We will show in this work that the KCV is also a theoretically sound one and will describe exactly under what circumstances it provides a rigorous upper-bound of the generalization error.

The KCV technique consists in splitting a dataset in k independent subsets: in turn, all but one of these subsets are used to train a classifier, while the remaining one is used to evaluate the generalization error. After the training, it is possible to compute an upper bound of the generalization error, for each one of the trained classifiers, but, as we have now k different models, some questions arise:

- 1) How these k models have to be used and/or combined for classifying the new data?
- 2) Each model is trained on a subset of the original dataset, so it does not use all the available information.

Can they be used for retraining a classifier on the entire dataset?

- 3) If we retrain a classifier on the entire dataset, should we modify the hyperparameters accordingly and how?

Our target, in this work, is to find answers to the previous questions by analyzing and comparing techniques and heuristics for the KCV applied to the model selection task.

The paper is organized as follows: Sections 2 briefly describe the SVM algorithm. Section 3 details the KCV procedure and the rigorous generalization error upper-bound, while in Section 4 some techniques for finding the SVM solution after the KCV procedure is presented. Finally, in Section 5 we show some experimental results and comparisons of the above techniques, tested on several well-known benchmarking datasets.

2. The Support Vector Machine

Let us consider a dataset composed by l patterns $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_l, y_l)\}$ where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i = \pm 1$. The dimensionality of each pattern $n = \dim(\mathbf{x}_i)$, $\forall i$, represents the number of features, characterizing the dataset. The SVM learning phase consists in solving the following CCQP:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2} \boldsymbol{\alpha}^T Q \boldsymbol{\alpha} + \mathbf{r}^T \boldsymbol{\alpha} \\ & 0 \leq \alpha_i \leq C \quad \forall i \in [1, \dots, l] \\ & \mathbf{y}^T \boldsymbol{\alpha} = 0 \end{aligned} \quad (1)$$

where $r_i = -1 \forall i$, C is a hyperparameter that must be tuned, and Q is a symmetric positive semidefinite $l \times l$ matrix

$$q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \quad (2)$$

where $K(\cdot, \cdot)$ is a Mercer's kernel function, which allows to deal also with nonlinear mappings of the data [17].

After solving the problem (1), the feed-forward classification phase at run-time can be computed as

$$f(\mathbf{x}) = \sum_{i=1}^l y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + b, \quad (3)$$

where \mathbf{x} is a new pattern that must be classified. The class of the new pattern \mathbf{x} is determined according to the sign of $f(\mathbf{x})$. The bias term b can be calculated through the Karush-Kuhn-Tucker (KKT) conditions, which hold at optimality.

The above description allows to identify the first hyperparameter (C) that must be tuned for obtaining the optimal classification performance. In the case of linear SVM this is the only one but, when dealing with nonlinear SVMs, the kernel function gives rise to at least another one: the width of the Gaussian (γ) or the order of the polynomial (p). In general, we will indicate with $\{C, *\}$ the set of hyperparameters to tune, where C is the regularization term, while $*$ indicates other possible hyperparameters.

Some rule-of-thumb method has been suggested for deriving the hyperparameters in a very simple and efficient

way [3]. This approach allows to select the values of the hyperparameters with a single pass through the dataset, but the accuracy of the classification is obviously not the best one. Another approach, which is the most used and effective (practical) procedure, is a hyperparameter exhaustive *grid search*: the CCQP problem is solved several times with different $\{C, *\}$ settings and the generalization error is estimated at each step. Finally, the optimal hyperparameters are chosen in correspondence to the minimum of the estimated generalization error. Obviously, both the coarseness of the grid and the size of the searching space influence severely the quality of the solution and the amount of computation time needed by the search procedure, especially when the generalization error estimate performed at each step is particularly time-consuming.

In any case, given a set of fixed hyperparameter values, the use of the KCV allows to perform the generalization error estimate as we will show in the following section.

3. k -Fold Cross Validation (KCV) and the KCV guaranteed bound

We revise here the analysis performed in [18]. The KCV consists in dividing the training set in k parts, each one consisting of l/k samples: $k-1$ parts are used, in turn, as a training set and the remaining one is used as a validation set. The error performed by the trained SVM on the validation set can be reliably used for estimating π , the true generalization error, because it has not been used for training the model.

In fact, if we consider the error rate on the j -th validation set:

$$\nu_{VAL}^{(j)} = \frac{k}{l} \sum_{i=1}^{l/k} I_i(y_i \neq \hat{y}_i), \quad (4)$$

where $\frac{k}{l}$ is the number of validation patterns, y_i is the true output for the i -th pattern, \hat{y}_i is the SVM output and

$$I_i(y_i \neq \hat{y}_i) = \begin{cases} 0 & \text{if } y_i = \hat{y}_i \\ 1 & \text{otherwise,} \end{cases} \quad (5)$$

we can bound the generalization error by inverting the Cumulative Binomial distribution [19].

For simplicity, we will make use here of the Azuma-Hoeffding [20] inequality, which allows to write the following bound in explicit form:

$$\Pr \left\{ \pi - \nu_{VAL}^{(j)} \geq \varepsilon \right\} \leq e^{-\frac{\varepsilon^2 k}{2l}}. \quad (6)$$

By setting a user defined confidence value δ :

$$\delta = e^{-\frac{\varepsilon^2 k}{2l}} \quad (7)$$

we obtain

$$\varepsilon = \sqrt{\frac{-k \ln \delta}{2l}} \quad (8)$$

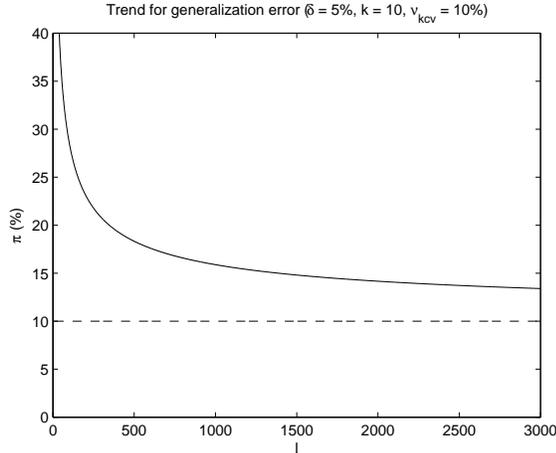


Fig. 1: Typical trend for the generalization error bound as a function of the number of training patterns l with $\nu_{KCV} = 10\%$, $\delta = 5\%$ and $k = 10$. The dotted line indicates the ν_{KCV} value.

and, substituting in Eq. (6), we have:

$$\Pr \left\{ \pi \geq \nu_{VAL}^{(j)} + \sqrt{\frac{-k \ln \delta}{2l}} \right\} \leq \delta. \quad (9)$$

Therefore, for any of the k classifiers, the following bound holds with probability $(1 - \delta)$:

$$\pi \leq \nu_{VAL}^{(j)} + \sqrt{\frac{-k \ln \delta}{2l}}, \quad (10)$$

We can always pick-up randomly one trained SVM to classify a new point and it is possible to show [21] that the performance of the model will be bounded by

$$\pi \leq \nu_{KCV} + \sqrt{\frac{-k \ln \delta}{2l}}, \quad (11)$$

where

$$\nu_{KCV} = \frac{1}{k} \sum_{j=1}^k \nu_{VAL}^{(j)}. \quad (12)$$

Note that k influences not only the training times but also the trade-off between the stability of the error average and the size of the confidence term (i.e. the term under square root). Common practice suggests $k = 5$ or $k = 10$ [22], which usually offer a good compromise. Fig. 1 shows a typical trend of the above bound as a function of the number of training patterns: we stress again the fact that tighter bounds are possible (e.g. decreasing between $O(\frac{1}{\sqrt{l}})$ and $O(\frac{1}{l})$) but we omit them here for the sake of clarity.

4. KCV solution

One of the problem with the k -Fold Cross Validation approach lies in the fact that KCV finds k different SVMs,

each one trained on a subset of the original training set, made up by $(k - 1)l/k$ patterns. From a theoretical point of view, as remarked in the previous section, a rigorous technique for combining them consists in picking randomly one SVM every time a new pattern must be classified: in this case, the run-time error rate is guaranteed to be upper-bounded by Eq. (11) with the user-defined probability. The main drawback in this case is the waste of memory, because, even if at each time only one SVM is used, all the k SVMs must be retained. We will refer to this approach in this paper as the *Random SVM (RDM)* technique.

From a practical point of view, other solutions can be found: one possibility consists in averaging (in some way) the k solutions. An option is to build a new SVM by computing the average of the parameters of the k SVMs [5]: this heuristic results in a large memory saving, so it is worth exploring. We call this approach the *Averaging (AVG)* method.

A third method is often used by practitioners: it consists in building a new SVM using the entire (original) dataset and the same optimal hyperparameters found through the KCV procedure. Even if this could appear as the best solution, it is the less justified from a theoretical point of view. In fact, the KCV estimate is no longer valid for the retrained SVM, because it is different from the k original ones, and, at the same time, no patterns are left outside the training set, which can be used for estimating the generalization error. Some works in trying to study this case from a theoretical point of view have appeared in the literature: the idea is to exploit the stability of the algorithm, that is its ability to perform roughly in the same way when trained on two datasets that differ only by a fraction of patterns. Unfortunately, the proposed methods are of limited practical use [2], [13]. In any case, as this technique is widely used, we will include it in our experiments as the *Retraining (RET)* technique.

Some argue that a better way of performing the retraining of the final SVM on the entire dataset could be derived by noting that the hyperparameters are found on a subset of the original dataset, consisting of $(k - 1)l/k$ patterns. Then, the hyperparameters must be adapted to the larger dataset, by scaling it accordingly:

$$\frac{C'}{l} = \frac{C}{\frac{(k-1)l}{k}}, \quad (13)$$

where C is the optimal hyperparameter obtained with the KCV and C' is the hyperparameter value to be used with the SVM to retrain. The rationale behind Eq. (13) is that the hyperparameter values, normalized by the number of training patterns, have to be the same in both cases. Note that the other hyperparameters are not involved in this resizing, since they are strictly linked to the number of features n , which remains fixed, and not to the number of patterns. We will refer to this method as *Retraining with Heuristic (RWH)*. However, this adaptation of the hyperparameter C

is in contrast with the theory of large margin classifiers. In fact, the SVM problem of Eq. (1) is only a computational simplified formulation of the following:

$$\begin{aligned} \min_{\alpha, \Gamma} \quad & \frac{1}{2\Gamma} \alpha^T Q \alpha + r^T \alpha + \frac{\Gamma}{2w_O^2} \\ & 0 \leq \alpha_i \leq 1 \quad \forall i \in [1, \dots, l] \\ & \mathbf{y}^T \alpha = 0 \\ & \Gamma \geq 0 \end{aligned} \quad (14)$$

where w_O^2 is the maximum margin [2] and $\Gamma = C$ only when the solutions of the two formulations coincide, independently from the number of samples.

All the methods described above are summarized in Tab. 1.

5. Experimental Results

In order to test the methods described in the previous section, we perform several experiments by using some well-known benchmarking datasets. In particular, the datasets used in our experiments are described in Tab. 2: they were introduced by G. Rätsch for the purpose of benchmarking machine learning algorithms [23].

For each replicate of Rätsch's datasets, the following experimental setup is applied:

- the data features are normalized to have zero mean and standard deviation equal to two, so that most of values lie in the range $[-1, +1]$;
- a Gaussian kernel is used;
- a model selection, using k -Fold Cross Validation, with $k = 10$ [22], is performed. The optimal hyperparameters $\{C, *\}$ are found using an exhaustive grid search, where the range and the number of steps are shown in Tab. 3;
- all the k SVMs are stored for benchmarking the *RDM* technique;
- a new SVM, built by averaging the k SVMs parameters, is stored for benchmarking the *AVG* method;
- the best hyperparameters found with the KCV procedure are used for training a new SVM on the whole dataset, which is used for benchmarking the *RET* technique;
- finally, we train a new SVM using the rescaled hyperparameter C' , computed according to Eq. (13), for benchmarking the *RWH* method.

Note that the KCV procedure is performed only on the training sets, so that the test sets are never used for finding the optimal hyperparameters. This guarantees that the test data are independent from the training and the validation steps. To obtain a lower bound on the error rate attainable by a SVM for each dataset, we can select the hyperparameters that minimize the number of misclassifications on the test set. Obviously this figure cannot be used for performance

assessment purposes but it acts as a reference value. We refer to these values as the *Test Set (TS)* error rate.

Tab. 4 presents the average and the standard deviation of the error rate obtained on the Rätsch's test sets with different methods: the second column (TS) represents the error rate with the test set approach (i.e. the best achievable rate). Note that in one case (the Image dataset) the average error for AVG and RWH is lower than the TS rate, but this is due to statistical fluctuations as these two results are characterized by higher standard deviations.

Since it is not easy to compare the various methods, we compute a simple performance index, which describes the relative performance of each method respect to the TS error rate:

$$\frac{\nu_i - \nu_{TS}}{\nu_{TS}}, \quad (15)$$

where ν_i is the error rate obtained with the i -th method on the test set, while ν_{TS} is the reference value. The performance indexes are shown in Tab. 5; the last row indicates the number of times that a method results to be the best performer.

From Tab. 5, it is possible to see that the retraining of the SVM by using the KCV hyperparameters is the best performer in almost half of the cases. Moreover, Tab. 6 shows that the performance index for the RET method on all the thirteen Rätsch's datasets is the lowest one. Surprisingly, the only theoretically justified method (RDM) is the best one in only three cases but shows an average performance which is similar to RET. The worst performing method results to be the AVG one.

As the average error is not robust to outliers, we report here the same analysis using quartile values instead of mean and standard deviation. Tab. 7 shows the values of the performance index computed using the median of the misclassification percentage and in Tab. 8 the corresponding values averaged on on all the thirteen datasets: the RET method still results to be the best one, but the RDM method is now characterized by the second best performance.

By looking at the results it is clear that: (1) the AVG technique, which is often used in practice, is the worst performing one, both considering the mean or the median of the error rate; (2) the RET method seems to be the most effective even though, at the time of this writing, no theoretical result is available to justify its performance; (3) the RDM method is characterized by a good performance (similar to RET method with respect to both mean and median values) and has the advantage that the error rate is guaranteed by Eq. (11) with a user-defined confidence value; (4) the rescaling of the C hyperparameter, despite being used by practitioners, appears to be useless for improving the error rate performance.

Table 1: Methods for building the final SVM after the KCV procedure.

Technique	Short name	Brief review
Random SVM	RDM	All k SVMs, found during KCV, are saved. Each time a new pattern must be classified, a SVM is picked randomly.
Average SVM	AVG	A new SVM is built by averaging the k SVMs parameters, found during KCV.
Retraining SVM	RET	A new SVM is retrained on the whole original dataset, by using the optimal hyperparameters, found with KCV.
Retraining with Heuristic SVM	RWH	A new SVM is retrained on the whole original dataset, by rescaling the hyperparameters.

Table 2: The Ratsch datasets.

Name	N. of features	Training samples	Test samples	Realizations
Banana	2	400	4900	100
Breast-Cancer	9	200	77	100
Diabetis	8	468	300	100
Flare-Solar	9	666	400	100
German	20	700	300	100
Heart	13	170	100	100
Image	18	1300	1010	20
Ringnorm	20	400	7000	100
Splice	60	1000	2175	20
Thyroid	5	140	75	100
Titanic	3	150	2051	100
Twonorm	20	400	7000	100
Waveform	21	400	4600	100

Table 3: Range and number of steps of the grid search procedure.

Hyperparameter	Range	N. of steps
C	$[10^{-1}; 10^5]$	13
γ	$[10^{-4}; 10^5]$	19

6. Conclusions

We have reviewed in this work the k -Fold Cross Validation technique applied to the SVM classification algorithm. Our purpose is twofold: on one hand we want to verify if some techniques, which are often used by practitioners, can be justified from a theoretical point of view and perform as expected; on the other hand we want to benchmark the only theoretical rigorous technique against common practice.

The first point is not to be neglected: the theoretical justification of the method is not only of academic interest but is a way to guarantee (in a statistical sense) the reliability of the result, which is of paramount importance in many fields (e.g. law [24]).

By analyzing the experimental results described in the

previous section, we are now able to answer the three questions raised in the introduction:

- 1) the best ways to combine the k classifiers appear to be the retraining solution (RET) and the randomly chosen model (RDM). The former performs slightly better in practice, while the second one allows to predict the classifier error rate on unobserved data;
- 2) based on the previous comments, we can state that neither a retraining nor a “combination” of the k SVMs are really necessary. However, when memory is an issue, a retraining of the classifier (RET) could be performed if we are not interested in generalization error estimates;
- 3) the experimental results clearly show that rescaling the hyperparameters does not increase the classifier performance.

Our analysis shows clearly that the final user has two main choices. The first alternative (RDM) guarantees the quality of the classifier, which is necessary in several fields like, for example, legal practice [24]: the price to pay for this additional information is a slight reduction in terms of classification performance. The second alternative (RET)

Table 4: Mean and standard deviation for the error rate on Rätsch's datasets.

Dataset	TS		RDM		AVG		RET		RWH	
	mean	st dev								
Banana	11.38	0.67	12.11	1.13	11.85	1.05	12.03	1.14	12.03	1.13
Breast	22.70	4.01	26.56	4.72	26.61	4.67	26.92	4.87	26.96	4.80
Diabetis	22.25	1.52	23.77	1.72	26.14	6.09	23.61	1.65	23.60	1.71
Flare	31.94	1.69	33.11	2.09	33.12	2.05	33.10	2.07	33.13	2.07
German	22.22	1.96	23.71	2.07	23.90	1.97	23.80	2.15	23.78	2.06
Heart	14.35	3.08	17.98	3.69	17.81	3.76	17.75	3.60	17.76	3.52
Image	3.49	0.48	3.73	0.50	3.41	0.61	3.49	0.48	3.46	0.51
Ringnorm	2.40	0.45	2.79	0.63	40.00	18.92	2.68	0.62	2.83	0.74
Splice	11.25	0.69	11.72	0.67	12.75	0.59	11.46	0.60	11.42	0.62
Thyroid	2.78	1.82	5.79	2.36	12.16	10.70	5.95	2.59	5.93	2.56
Titanic	22.05	0.45	22.84	0.94	22.69	0.75	22.68	0.66	22.94	1.74
Twonorm	2.35	0.12	2.72	0.35	2.69	0.43	2.68	0.35	2.71	0.38
Waveform	9.94	0.46	10.65	0.84	15.24	5.96	10.56	0.84	10.60	0.81

Table 5: Performance indexes computed with the average error rate. Best result is in bold face.

Dataset	RDM	AVG	RET	RWH
Banana	0.064	0.041	0.057	0.057
Breast	0.170	0.172	0.186	0.188
Diabetis	0.068	0.175	0.061	0.060
Flare	0.037	0.037	0.036	0.037
German	0.067	0.076	0.071	0.070
Heart	0.253	0.241	0.237	0.238
Image	0.069	-0.023	0.000	-0.001
Ringnorm	0.164	15.682	0.116	0.178
Splice	0.042	0.134	0.019	0.015
Thyroid	1.076	3.363	1.134	1.129
Titanic	0.036	0.029	0.028	0.040
Twonorm	0.158	0.147	0.140	0.156
Waveform	0.072	0.533	0.063	0.067
Best performer	3	2	6	2

Table 6: Mean values for performance indexes of Tab. 5.

RDM	AVG	RET	RWH
0.175	1.585	0.165	0.172

results in a better performing classifier, but at the expense of a violation of the theoretical assumptions. Practitioners can safely use this method in all the applications where a rigorous approach is not necessary. Our results support the need for additional research to fill this gap between theory and practice.

References

- [1] B. Schoelkopf, K.K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, V. Vapnik, "Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers", *IEEE Trans. on Signal Processing*, vol. 45, pp. 2758–2765, 1997.
- [2] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 2001.
- [3] B. L. Milenova, J. S. Yarnus, M. M. Campos, "SVM in Oracle database 10g: Removing the barriers to widespread adoption of Support Vector Machines", *Proc. of the 31st Int. Conf. on Very Large Data Bases*, pp. 1152–1163, 2005.
- [4] S. S. Keerthy, S. K. Shevade, C. Bhattacharyya, K. R. K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design", *Neural Computation*, vol. 13, pp. 637–649, 2001.
- [5] D. Anguita, A. Boni, S. Ridella, F. Riviuccio, D. Sterpi, "Theoretical and practical model selection methods for Support Vector classifiers", in "Support Vector Machines: Theory and Applications", edited by L. Wang, Springer, 2005.
- [6] B. Schoelkopf, A. Smola, *Learning with Kernels*, The MIT Press, 2002.
- [7] J. Shawe-Taylor, N. Cristianini, "Margin distribution and soft margin", in "Advances in Large Margin Classifiers", edited by A. Smola,

Table 7: Performance indexes computed with the median error rate. Best results are in bold face.

Dataset	RDM	AVG	RET	RWH
Banana	0.058	0.031	0.039	0.044
Breast	0.111	0.167	0.167	0.167
Diabetis	0.075	0.090	0.060	0.060
Flare	0.023	0.027	0.023	0.031
German	0.052	0.082	0.075	0.060
Heart	0.357	0.250	0.214	0.250
Image	0.057	0.029	0.000	0.000
Ringnorm	0.129	18.883	0.083	0.112
Splice	0.049	0.146	0.025	0.016
Thyroid	1.000	1.750	1.000	1.250
Titanic	0.028	0.025	0.027	0.027
Twonorm	0.134	0.110	0.113	0.122
Waveform	0.063	0.234	0.050	0.061
Best performer	4	3	7	3

Table 8: Mean values for performance indexes of Tab. 7.

RDM	AVG	RET	RWH
0.164	1.679	0.144	0.169

- P. Bartlett, B. Schoelkopf, D. Schuurmans, The MIT Press, 2000.
- [8] K. Duan, S. S. Keerthy, A. Poo, "Evaluation of simple performance measures for tuning SVM parameters", *Neurocomputing*, vol. 51, pp. 41–59, 2003.
- [9] C. J. C. Burges, "A tutorial on Support Vector Machines for classification", *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.
- [10] B. Efron, R. Tibshirani, *An introduction to the Bootstrap*, Chapman and Hall, 1993.
- [11] B. Efron, R. Tibshirani, "Improvements on Cross-Validation: the 632+ bootstrap method", *Journal of American Statistic Association*, vol. 92, pp. 548–560, 1997.
- [12] D. Anguita, A. Boni, S. Ridella, "Evaluating the generalization ability of Support Vector Machines through the Bootstrap", *Neural Processing Letters*, vol. 11, pp. 51–58, 2000.
- [13] M. Anthony, S. B. Holden, "Cross-Validation for binary classification by real-valued functions: theoretical analysis", *Proc. of the 11th Conf. on Computational Learning Theory*, pp. 218–229, 1998.
- [14] M. Dumler, *Microsoft SQL Server 2008 Product Overview*, Microsoft Corporation, 2008.
- [15] "Cross-Validation (Analysis Services - Data Mining)", in *Microsoft SQL Server 2008 Books Online*, Microsoft Corporation, 2008. Available online at <http://msdn.microsoft.com/>.
- [16] M. Kaariainen, "Semi-supervised model selection based on Cross-Validation", *Proc. of IEEE Int. Joint Conf. on Neural Networks 2006, IJCNN 2006*, pp. 1894–1899, 2006.
- [17] C. Cortes, V. Vapnik, "Support-vector networks", *Machine Learning*, vol. 27, pp. 273–297, 1995.
- [18] D. Anguita, S. Ridella, F. Rivieccio, "K-Fold Generalization Capability Assessment for Support Vector Classifiers", *Proc. of the IEEE Int. Joint Conf. on Neural Networks, IJCNN 2005*, pp. 855–858, 2005.
- [19] M. Kaariainen, J. Langford, "A comparison of tight generalization error bounds", *Proc. of the 22nd Int. Conf. on Machine learning*, pp. 409–416, 2005.
- [20] K. Azuma, "Weighted sums of certain dependent random variables", *Tohoku Math. Journal*, vol. 19, pp. 357–367, 1967.
- [21] A. Blum, A. Kalai, J. Langford, "Beating the Hold-Out: Bounds for K-fold and Progressive Cross-Validation", *Computational Learning Theory*, pp. 203–208, 1999.
- [22] C.-W. Su, C.-C. Chang, C.-J. Lin, "A practical guide to Support Vector classification", Technical report, Dept. of Computer Science, National Taiwan University, 2003.
- [23] G. Raetsch, T. Onoda, K. R. Mueller, "Soft margins for AdaBoost", *Machine Learning*, vol. 42, pp. 287–320, 2001.
- [24] L. Roberge, S. B. Long, D. B. Burnham, "Data Warehouses and Data Mining tools for the legal profession: using information technology to raise the standard of practice", *Syracuse Law Review*, vol. 52, pp. 1281–1292, 2002.

Rule-Based Linear Regression Machine

Olutayo O. Oladunni

Accenture Technology Labs, Accenture, Chicago, IL, USA

Abstract - This paper presents a rule-based linear regression model. The rule(s) is in the form of multiple polyhedral sets to be satisfied by the function on polyhedral regions of the input space, and it is introduced as additional constraints into the formulation of the Tikhonov regularization (ridge regression) model. The resulting formulation leads to a least squares problem that can be solved using matrix methods or iterative methods. Advantages of this formulation include the explicit expressions for the regression parameters; the ability to incorporate and rules directly to the regression equation; computational tractability in providing the regression parameters; and easy implementation without the use of specialized solver-software. Through the computational results of the application problem (telephone data), it is shown that the use of prior knowledge provides a means of reducing the size of the data, as well as improve the function approximation.

Keywords: linear regression, function approximation, rules, optimization.

1 Introduction

In most decision-making problems, we usually try to estimate or predict the values of the dependent variable from known values of the independent variable. But to establish such a relationship between the dependent and independent variables, a training sample of values of both dependent and independent variables must be readily available. These training sample values are used to determine a regression equation that can be used to estimate or predict new (unknown) values of the dependent variable.

The term regression was first used by Sir Francis Galton, an English expert on heritability of size in the 1886, which laid down the foundations of correlation. Galton, using a sample of males and their fathers, found a tendency for exceptionally short fathers to have sons of more average height (taller than their fathers); the opposite was true for usually tall fathers. Galton said that the heights of sons “regressed” to the mean, hence the name regression [1].

Rules expressed as prior knowledge has recently been incorporated into SVM and least squares SVM classifiers, both to improve the classification task and to handle problems where training data may be few or not available [2], [3], [4] and [5]. However, rule incorporation into a regression

formulation is very limited. Mangasarian [6] proposed a methodology for incorporating prior knowledge into a function approximation generated by a linear combination of linear or nonlinear kernels. In that study, prior knowledge was introduced in the form of linear equalities to be satisfied by the function on polyhedral regions of the input space for linear kernels, and on similar regions of the feature space for nonlinear kernels. For standard linear programming formulations, Mangasarian [7] proposed a generalized class of linear programs with constraints in the form of implications (prior knowledge representation).

In this approach, a combination of the rules together with the concept of the least squares support vector machine (LS-SVM) [8] and regularized LS-SVM [9] results in a linear system of equations. Note that the LS-SVM and Regularized LS-SVM concepts considered in this study are different in the sense that, the minimization problem is an unconstrained optimization problem for which the minimizing solution is a solution to a smaller linear system of equations. The idea of modifying the SVM to a LS-SVM that can be condensed to a linear system is a motivation for the development of the proposed rule-based linear regression model. To achieve that, the Tikhonov regularization scheme [10] is employed to conveniently reduce the problem to a smaller linear system for which solutions can be found easily by matrix methods or iterative methods.

The advantages of this method include the explicit expressions for the regression parameters; computational tractability in providing the regression parameters; and easy implementation without the use of specialized solver/software.

This paper is organized as follows. In section 2, Tikhonov regularization rule-based linear regression (RLR) model for estimation or prediction is described. In section 3, a rule-based application problem is given. In section 4, computational results are described, and section 5 concludes the paper.

2 Tikhonov Regularization Rule-Based Linear Regression

Consider a problem of finding a function approximation for a data set in R^n that are represented by measurements & observations denoted by n -dimensional vector x_i and $y_i \in R$

respectively, where $i = 1, \dots, m$ (no. of observations) and y is continuous (output). This problem can be modeled through the function approximation optimization problem:

$$\min_{w, \gamma, \xi} \frac{\lambda}{2} \|w\|^2 + \frac{1}{2} \sum_{i=1}^m (\xi_i)^2, \quad (1)$$

$$s.t. \xi_i = x_i^T w + \gamma - y_i, \quad i = 1, \dots, m$$

x_i are the training sample points, λ is the tradeoff parameter, and ξ_i are error slacks measuring the deviation of each point from the function approximation or regression function. The weights (w, γ) that characterize the linear approximation function below are shown in Figure 1.

$$f(x) = x^T w + \gamma \text{ with } w \in \mathfrak{R}^n, \gamma \in \mathfrak{R}. \quad (2)$$

Problem (1) is minimized parametrically with the tradeoff parameter λ which accounts for the tradeoff between minimum norm and minimum empirical error. Its weight estimate provides a function approximation for the data of interest.

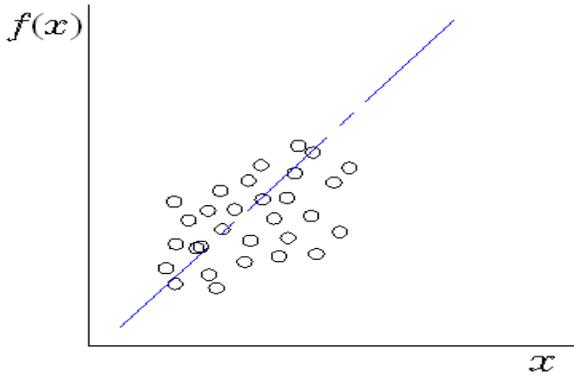


Figure 1. Function approximation problem. The regression function (blue line) and the data points (circles)

Now, suppose that in addition to the given data points, there are rules, conditions which are expected to hold for a particular constant (observation value). We assume that the knowledge sets in an n dimensional space are given in the form of a polyhedral set. Then those are determined by a set of linear inequalities as follows (see Figure 2).

$$Bx \leq b \quad (3)$$

All points x lie in the nonempty polyhedral set defined by (3), and are not necessarily contained in the training data set. Therefore, the following implications must hold for a given w and γ :

$$Bx \leq b \Rightarrow x^T w \geq -\gamma + \eta, \quad (4)$$

where η is the desired target that hold for a specific polyhedral set.

Transforming the proposition (4) into constraints that can be incorporated into regression model (1) can be achieved in two ways: either applying the Nonhomogeneous Farkas Theorem of the Alternative [2], [11] or, using duality in linear programming (LP) [12].

Applying the Nonhomogeneous Farkas Theorem of the Alternative [11], the equivalence of implication (4) is given as:

$$\begin{aligned} B^T u + w &= 0 \\ b^T u - \gamma + \eta &\leq 0 \\ u &\geq 0, \text{ has a solution } (u, w) \end{aligned} \quad (5)$$

Constraints (5) are the equivalence of implication (4), and it's the set of linear equations which will be readily incorporated into the model (1). Note that the equivalence can be verified using LP duality [12].

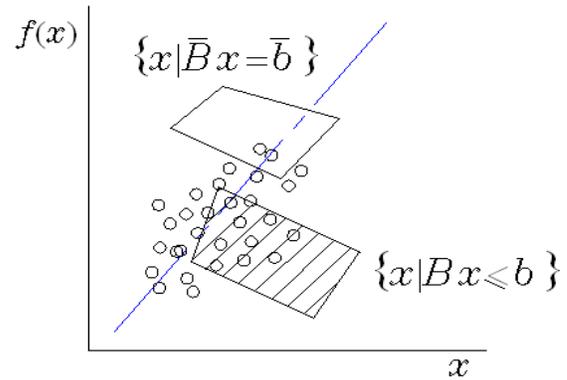


Figure 2. Rule-based function approximation problem. The regression function (blue line), rules or knowledge set (polyhedral plane)

2.1 Incorporating rules into a regression model

Given the rule sets:

$$\{x \mid Bx \leq b\}, \quad (6)$$

where g are the number of rules.

Using Farkas theorem [11] and LP Duality [12], there exist a u such that:

$$B^T u + w = 0, \quad b^T u - \gamma + \eta \leq 0, \quad u \geq 0 \quad (7)$$

Constraint (7) can be incorporated into regression model formulation (1) as additional constraints. The resulting optimization problem is as follows:

$$\begin{aligned} \min_{w, \gamma, \xi, u} \quad & \frac{\lambda}{2} \|w\|^2 + \frac{1}{2} \sum_{i=1}^m (\xi_i)^2 \\ \text{s.t.} \quad & \xi_i = x_i^T w + \gamma - y_i, \quad i = 1, \dots, m \\ & B^T u + w = 0 \\ & b^T u - \gamma + \eta \leq 0 \end{aligned} \quad (8)$$

Problem (8) is a constrained optimization problem referred to as rule-based linear regression (RLR) model. The additional constraints, as they are, assume that the rules satisfy or hold for a specific observation value. However, because we cannot guarantee that such a regression function exists, error slacks (residual variable) are included to drive its variable to zero.

For convenience, problem (8) is rewritten in matrix form (Note that weights can be given to each term of the objective function, constructing a multi-objective optimization (MOP) problem with 4 objectives):

$$\begin{aligned} \min_{w, \gamma, \xi, u, r, p} \quad & \frac{\lambda}{2} \|w\|^2 + \frac{1}{2} [\|\xi\|^2 + \|r\|^2 + \|p\|^2] \\ \text{s.t.} \quad & Aw + \gamma e - y = \xi \\ & B^T u + w = r \\ & b^T u - \gamma + \eta = p \end{aligned} \quad (9)$$

where $u = [u_1^T, \dots, u_g^T]^T$ ($u \in R^{g \times 1}$) is a vector of the multipliers of which each element corresponds to a rule. ξ is a residual error vector accounting for the training data error, and vectors r, p are residual error vectors accounting for the violation of the rules.

The following matrices are defined for data set and rules: Matrix $A \in R^{m \times n}$ is a matrix whose rows is x_i , $e \in R^{m \times 1}$ is a vector of ones. Vector y is a vector whose components are the observations $y_i \in R$. Matrix $B \in R^{g \times n}$ consists of the coefficients of the rule set. Vector $b \in R^{g \times 1}$ consists of components which bound the rule set. $\eta \in R$ is the desired response value, i.e., bound of the function approximation value or regression function value that is satisfied over a specific polyhedral set.

$$A_{m \times n} = \begin{pmatrix} x_1^T \\ \vdots \\ x_m^T \end{pmatrix}, \quad y_{m \times 1} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} \quad (10)$$

For convenience, problem (9) is rewritten in matrix form (RLR):

$$\min_{w, \gamma, u} f_{RLR}(w, \gamma, u) = \left[\begin{aligned} & \frac{\lambda}{2} \|w\|^2 + \frac{1}{2} \|Aw + \gamma e - y\|^2 + \\ & \frac{1}{2} [\|B^T u + w\|^2 + \|b^T u - \gamma + \eta\|^2] \end{aligned} \right] \quad (11)$$

Problem (11) is the final unconstrained RLR model which incorporates the rule sets as defined in (6) into model (1). The tradeoff parameter λ is run through a series or range of values to achieve the best result. If its value increases then the minimum norm is achieved, but at the expense of having a higher training residual error and/or higher rule set residual error. The first term of problem (11) ensures the smoothness of the function, i.e., similar inputs have similar outputs; the second term ensures that the data points are as close as possible to each other and the function approximation; and the third and fourth term ensure the closeness of rules to the function approximation. If data is not available, the second term of the RLR model can be dropped. This result in an estimation function based strictly on rules, and it is useful for situations of which only rules or expert knowledge expressed as rules exists.

Problem (11) is a convex unconstrained optimization problem which has a unique minimum point. The minimizing point of problem (11) is the solution to the following optimality conditions of $f_{RLR}(w, \gamma, u)$ set to equal zero:

$$\frac{df}{dw} = \lambda w + A^T Aw + A^T e \gamma - A^T y + B^T u + w = 0 \quad (12)$$

$$\frac{df}{du} = BB^T u + Bw + bb^T u - b\gamma + b\eta = 0 \quad (13)$$

$$\frac{df}{d\gamma} = e^T Aw - e^T y + e^T e \gamma - b^T u - \eta + \gamma = 0 \quad (14)$$

To provide explicit matrix expressions for w and γ , the optimality conditions need to be rearranged. From equation (13), the following expression is obtained for u :

$$\begin{aligned} u &= -UBw + Ub\gamma - Ub\eta, \\ \text{where } U_{g \times g} &= (BB^T + bb^T)^{-1} \end{aligned} \quad (15)$$

Using u , the location (bias constant) of the function approximation can be estimated by substituting (15) in equation (14):

$$\begin{aligned} \gamma &= q(-vw + f) \\ \text{where } \gamma &\text{ is a scalar} \end{aligned} \quad (16)$$

$$q = \left[e^T e - b^T U b + 1 \right]^{-1} \quad (17)$$

$$v_{1 \times n} = \left[e^T A + b^T U B \right], \quad f = \left[e^T y - b^T U b \eta + \eta \right] \quad (18)$$

With explicit solutions to u and γ already determined under the assumption that U is an invertible matrix, when substituted into equation (12), the solution to the normal vector w (slope) can conveniently be expressed as:

$$w = Q^{-1} z$$

where $w = [w_1, \dots, w_n]^T$ (19)

The linear approximation for estimating a new point or measurement is given by:

$$f(x) = x^T w + \gamma \quad (20)$$

The corresponding linear system to solution (19) is:

$$Qw = z, \quad (21)$$

where

$$Q_{n \times n} = \left[\lambda I + I + A^T [A - eqv] - B^T U [B + bq v] \right] \quad (22)$$

$$z_{n \times 1} = \left[A^T [y - eqf] + B^T U b [\eta - qf] \right] \quad (23)$$

Solution (19) provides an estimate to the linear system of equations in (21). Methods for solving the linear system include matrix decomposition methods, or iterative based methods [12], [13]. Its solution involves the inversion of a smaller dimensional matrix (Q) of the magnitude $n \times n$. Below is an algorithm for RLR.

Algorithm 1. Rule-based Linear Regression

Given a data set & rule set; measurements, observations and rules in R^n , represented by n -dimensional vector $x_i, y_i \in R$, and $\{x | Bx \leq b\}$ respectively, where $i = 1, \dots, m$ (no. of observations) and y is continuous (output).

Weights w and γ for a linear function approximation (linear regression) are computed as follows.

RLR Algorithm:

Step 1: Compute Q from (22) for a specific parameter λ .

Step 2: Compute z from (23).

Step 3: Determine w from (19).

Step 4: Determine γ from (16).

Step 5: Prediction or estimation of a new (unknown) point x by (20).

3 Application Problem

In this section we address the problem of determining the sensitivity of demand for long-distance calls.

Problem Statement: A telecommunication company providing telephone service to the state of Indiana is considering increasing rates for long-distance calls, while reducing charges for certain other services. Any changes in the rate structure by the company must be approved by the Indiana Public Utilities Commission (IPUC), who has asked the company to present data indicating the sensitivity of demand for long-distance calls [1]. The available data consists of 76 quarters (19 years) on the price and quantity of long-distance calls (76 points, 1 attribute). The data is illustrated in Figure 3, where quantity is measured in thousands of calls, and price is an index. The goal is to estimate the demand equation which will be used to predict whether an increase in price will result in an increase in revenue.

Further analyses were made by including personal income (measured in thousands of dollars) as an attribute. The analyses include finding the relationship between price and personal income (76 points, 1 attribute), quantity and personal income (76 points, 1 attribute), and price & quantity versus personal income (76 points, 2 attribute).

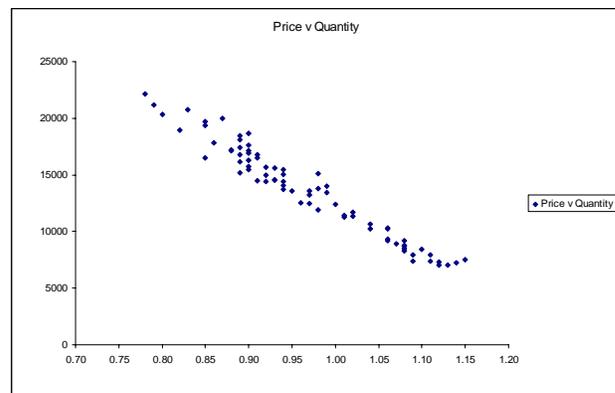


Figure 3. The quantity and price of long-distance calls for Telephone data

Telephone Data Rule: In addition to the telephone data, rules were generated and introduced in the form of linear inequalities to be satisfied by the function on polyhedral regions of the input space. The median value of all variables was used to generate the telephone data rule. This was done to avoid the problems of outliers that undermine the mean values all variables.

Rule 1 - Price and quantity

$$\text{Price} \leq 0.94 \Rightarrow f(x) = \text{Quantity} \geq 14223, \quad (27)$$

Rule 2 - Price and personal income

$$\text{Price} \leq 0.94 \Rightarrow f(x) = \text{Personal income} \geq 70578 \quad (28)$$

Rule 3 - Quantity and personal income

$$\text{Quantity} \geq 14223 \Rightarrow f(x) = \text{Personal income} \geq 70578 \quad (29)$$

Rule 4 - Price, quantity and personal income

$$\begin{aligned} &\text{Price} \leq 0.94, \text{Quantity} \geq 14223 \\ &\Rightarrow f(x) = \text{Personal income} \geq 70578 \end{aligned} \quad (30)$$

4 Computational Results

In this section, the result of the analyzed data set is presented and discussed. The RLR model is used to train the data and rule set. Fifty percent of the data along with the generated rules was trained on the RLR model, and the other fifty percent was used as testing samples. To demonstrate the functionality and appropriateness of the model, the coefficient of determination (R^2) is obtained for the training and testing data. The coefficient of determination measures how much of the variability in the dependent variable (y) is explained by the independent variable (x).

The coefficient of determination is defined below:

$$\begin{aligned} R^2 &= \frac{\text{SSR}}{\text{SST}} \\ &= \text{Amount of variability in } y \text{ that is explained by } x \end{aligned} \quad (31)$$

$$\text{SST} = \sum (y_i - \bar{y})^2, \quad (32)$$

$$\text{SSR} = \sum (y_i - \hat{y}_i)^2, \quad (33)$$

$$\text{SSE} = \sum (\hat{y}_i - \bar{y})^2, \quad (34)$$

where SST is the sum of squares total (total amount of variability in y to be explained in the regression problem), SSR is the sum of squares due to Regression (amount of variability explained by the independent variable x), and SSE is the sum of squares error (amount of variability not explained by the independent variable x).

The tradeoff parameters considered are 0 and $1/m$, where m is the number of observations. The regression equation for

determining the price and quantity relationship is (tradeoff parameter equal to zero) (Table 1 and Figure 4):

$$\text{Quantity} = -42128(\text{Price}) + 54374, \quad (35)$$

Table 1. Regression parameters and coefficient of determination for price v quantity

Run	Tradeoff	w (slope)	y (intercept)	R^2	
				Training data	Testing data
1	0	-42128.00	54374.00	0.9398	0.9616
	1/m	-38506.00	50863.00	0.9327	0.9540
2	0	-42104.00	54259.00	0.9609	0.9433
	1/m	-38870.00	51146.00	0.9552	0.9374

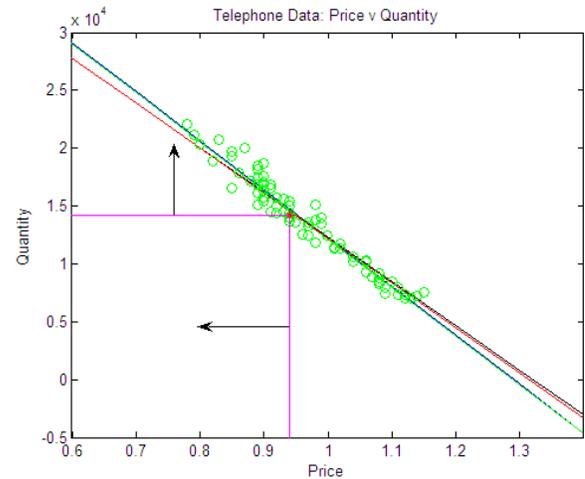


Figure 4. Four regression equation for price v quantity. Sample run 1: blue line ($\Lambda=0$), black line ($\Lambda=1/m$). Sample run 2: green line ($\Lambda=0$), red line ($\Lambda=1/m$). The rule is in magenta

The slope indicates the sensitivity of quantity to price ($\Lambda=0$). The quantity decreases on the average by 42,128 calls for each 1-unit increase in price. Equation (35) also suggests that an increase in price will result in a decrease in revenue. The coefficient of determination (training sample: $R^2=0.9398$, testing sample: $R^2=0.9616$) for tradeoff parameter equal to zero suggests that 93.98% of the training set variability in the quantity of phone calls can be explained by the price variable; and 96.16% of the testing set variability in the quantity of phone calls can be explained by the price variable.

These values are comparable to the simple linear regression model that was used to train all 76 quarters of data (without rules) and reports $R^2=0.9523$. The R^2 values of the rule-based model suggest there is an improvement in the ability to predict future samples considering that only 50% of the data was used to develop the regression equation. The rule specified implies that if the median price index of calls is at most 0.94, then the median quantity of calls should be at least 14,223. Since the median is the middle term of the ranked data, then the rule accounts for the quantity of calls greater than its middle term (14,223), and it should hold for all price indexes lower than its corresponding middle term (0.94). As a result we have fewer points to train along with the rule(s).

Further analyses were conducted to determine more relationships among the variables of the telephone data. The regression equation for determining the price and personal income relationship is (tradeoff parameter equal to zero) (Table 2 and Figure 5):

$$\text{Personal Income} = -69206(\text{Price}) + 134040, \quad (36)$$

Table 2. Regression parameters and coefficient of determination for price v personal income

Run	Tradeoff	w (slope)	y (intercept)	R ²	
				Training data	Testing data
1	0	-69206.00	134040.00	0.9094	0.8522
	1/m	-64701.00	129650.00	0.9057	0.8590
2	0	-69269.00	134040.00	0.9252	0.8045
	1/m	-64992.00	129850.00	0.9218	0.8113

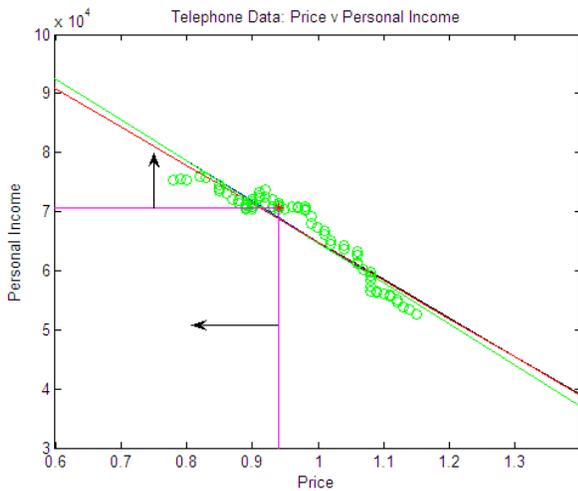


Figure 5. Four regression equation for price v personal income. Sample run 1: blue line (A=0), black line (A=1/m). Sample run 2: green line (A=0), red line (A=1/m). The rule is in magenta

Personal income decreases on the average by 69,206 dollars for each 1-unit increase in price. The coefficient of determination (training sample: R²=0.9094, testing sample: R²=0.8522) for tradeoff parameter equal to zero suggests that 90.94% of the training set variability in the local average personal income can be explained by the price variable; and 85.22% of the testing set variability in the local average personal income can be explained by the price variable. The regression equation for determining the quantity and personal income relationship is (tradeoff parameter equal to zero) (Table 3 and Figure 6):

$$\text{Personal Income} = 1.5052(\text{Quantity}) + 46648, \quad (37)$$

Personal income increases on the average by 1.50 dollars for each 1-unit increase in quantity of calls. The coefficient of determination (training sample: R²=0.8524, testing sample: R²=0.8235) for tradeoff parameter equal to zero suggests that 85.24% of the training set variability in the local average personal income can be explained by the quantity of calls; and 82.35% of the testing set variability in

the local average personal income can be explained by the quantity of calls.

Table 3. Regression parameters and coefficient of determination for quantity v personal income

Run	Tradeoff	w (slope)	y (intercept)	R ²	
				Training data	Testing data
1	0	1.5052	46648.00	0.8524	0.8235
	1/m	1.5052	46648.00	0.8524	0.8235
2	0	1.4988	46797.00	0.8176	0.8645
	1/m	1.4988	46797.00	0.8176	0.8645

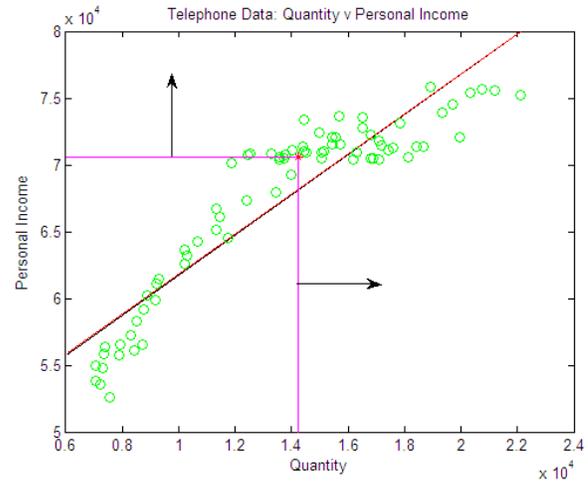


Figure 6. Four regression equation for quantity v personal income. Sample run 1: blue line (A=0), black line (A=1/m). Sample run 2: green line (A=0), red line (A=1/m). The rule is in magenta

The regression equation for determining the price, quantity and personal income relationship is (tradeoff parameter equal to zero) (Table 4):

$$\text{Personal Income} = -64389(\text{Price}) - 0.0625(\text{Quantity}) + 130700, \quad (38)$$

Table 4. Regression parameters and coefficient of determination for price & quantity v personal income

Run	Tradeoff	w ₁ (slope)	w ₂ (slope)	y (intercept)	R ²	
					Training data	Testing data
1	0	-64389.00	-0.0625	130700.00	0.8573	0.9040
	1/m	-23240.00	0.8233	78892.00	0.8362	0.8697
2	0	-69038.00	0.1043	122890.00	0.9085	0.8706
	1/m	-19260.00	0.9931	72209.00	0.8961	0.8262

Personal income decreases on the average by 64,389 dollars for each 1-unit increase in price, holding quantity constant. Personal income decreases on the average by 0.06 dollars for each 1-unit increase in quantity of calls, holding price constant. The coefficient of determination (training sample: R²=0.8573, testing sample: R²=0.9040) for tradeoff parameter equal to zero suggests that 85.73% of the training set variability in the local average personal income can be explained by the price and quantity of calls; and 90.40% of the testing set variability in the local average personal income can be explained by the price and quantity of calls.

5 Conclusions

In this paper a rule-based linear regression (RLR) model is developed and applied to the telephone data. Using the phone call data and rules, we investigated the pairwise relationships between prices, quantity of calls, personal income, and also investigated how price and quantity of calls can help explain the local average personal income of callers. This analysis indicates an inverse linear relationship exist between price v quantity and price v personal income. However, the model suggests there is a positive linear relationship between quantity of calls and the local average personal income of callers. When using both the price and quantity of calls to explain personal income, the relationship appears to be mixed. In Table 4, the first row ($w_2=-0.0625$) suggests a slight inverse influence of quantity on personal income, however, because the value -0.0625 (\$0.06 decrease) is quite small, the inverse effect could be negligible (possibly insignificant), making the results more consist with the positive linear relationship revealed between quantity and personal income in Table 3.

Because of the structure of the RLR model, the results of a decision tree model can be conveniently included into the RLR model as rules. The RLR model assumes a linear relationship between the variables. Future work should include the following:

- Development of a rule based nonlinear and/or kernel regression model.
- Ways of generating rules from data should also be an interesting topic for future work.
- Problems for which there exist only rules or prior knowledge expressed as rules.

6 References

- [1] D.L. Harnett and J.F., Horrell. "Data, Statistics, and Decision Models with Excel". John Wiley & Sons, Inc., 1998.
- [2] G. Fung, O.L. Mangasarian, and J.W. Shavlik. "Knowledge-Based Support Vector Machine Classifiers"; Proceedings Neural Information Processing Systems 2002 (NIPS 2002), Vancouver, BC, Dec 2002.
- [3] G. Fung, O.L. Mangasarian, and J.W. Shavlik. "Knowledge-Based Nonlinear Kernel Classifiers"; in Manfred Warmuth and Bernhard Scholkopf (eds), Proceedings Conference on Learning Theory (COLT/Kernel 03) and Workshop on Kernel Machines, 102-113, Washington, D.C., Aug 2003.
- [4] O.O. Oladunni, T.B. Trafalis, and D.V. Papavassiliou. "Knowledge-Based Multiclass Support Vector Machines applied to Vertical Two-phase Flow"; ICCS 2006, Lecture notes in Computer Science, (V.N. Alexandrov et al., eds.), Part I, LNCS 3991 (Springer-Verlag Berlin Heidelberg) 188-195, 2006.
- [5] O.O. Oladunni. "Least Square Multi-Class Kernel Machines with Prior Knowledge and Applications". Ph.D. dissertation, University of Oklahoma, 2006.
- [6] O.L. Mangasarian, J.W. Shavlik, and E.W. Wild. "Knowledge-Based Kernel Approximation"; Journal of Machine Learning Research, **5**, 1127-1141, 2004.
- [7] O.L. Mangasarian. "Knowledge-Based Linear Programming"; SIAM Journal on Optimization, **15**, 375-382, 2005.
- [8] J.A.K. Suykens and J. Vandewalle. "Least Squares Support Vector Machine Classifiers"; Neural Processing Letters, **9**, 293-300, 1999.
- [9] K. Pelckmans, J.A.K. Suykens, and B. De Moor. "Morozov, Ivanov and Tikhonov Regularization Based LS-SVMs"; In Proceedings of the International Conference On Neural Information Processing (ICONIP 2004), Calcutta, India, Nov 2004.
- [10] A.N. Tikhonov and V.Y. Arsenin. "Solution of Ill-Posed Problems". Winston, Washington D.C., 1977.
- [11] O.L. Mangasarian. "Nonlinear Programming". SIAM, Philadelphia, PA, 2004.
- [12] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. "Nonlinear Programming – Theory and Algorithms". John Wiley & Sons, Inc., 1993.
- [13] J.M. Lewis, S. Lakshmivarahan, and S. Dhall. "Dynamic Data Assimilation". Cambridge University Press, 2006.

Understanding Support Vector Machine Classifications via a Recommender System-Like Approach

David Barbella¹, Sami Benzaid², Janara Christensen³, Bret Jackson⁴, X. Victor Qin⁵, David Musicant⁶

¹Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA

²Department of Computer Science, University of North Carolina, Chapel Hill, NC, USA

³Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA

⁴Ultralingua, Inc., Minneapolis, MN, USA

⁵Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI, USA

⁶Department of Computer Science, Carleton College, Northfield, MN, USA

Abstract—*Support vector machines are a valuable tool for making classifications, but their black-box nature means that they lack the natural explanatory value that many other classifiers possess. Alternatively, many popular websites have shown recent success in explaining recommendations based on behavior of other users. Inspired by these ideas, we suggest two novel methods for providing insight into local classifications produced by a support vector machine. In the first, we report the support vectors most influential in the final classification for a particular test point. In the second, we determine which features of that test point would need to be altered (and by how much) in order to be placed on the separating surface between the two classifications. We call the latter technique “border classification.” In addition to introducing these explanatory techniques, we also present a free-for-download software tool that enables users to visualize these insights graphically.*

Keywords: SVMs, classification, explanations, recommendations

1. Introduction

Support vector machines (SVMs) [1], [2] are a well-known supervised learning technique for performing binary classification. SVMs are very accurate and generalize well to a wide range of applications. Because support vector machines are “black-box” classifiers, the decisions they make are not always easily explainable. By this we mean that the model produced does not naturally provide any useful intuitive reasons about why a particular point is classified in one class rather than another. For instance, consider an SVM used by a bank to determine to whom they will loan money. If a customer’s loan application is rejected and they would like to know why, then it is not very useful to only be able to say that the algorithm came back with a number lower than some required threshold. It would be much more satisfactory to the customer to be able to tell them that they were denied credit because their income is too low and they have six outstanding loans. Decision trees can provide a model that is much more easily interpreted, but cannot always achieve results as accurate as those produced by an SVM.

Recently Fung et al. [3] described the importance of understandable classifications for computer aided diagnosis (CAD). CAD systems provide analysis and interpretation of medical images. Although these systems have been shown to be highly successful in both research labs and clinical settings, doctors are reluctant to trust a diagnosis without receiving a convincing reason for it. Without understandable reasons for the diagnoses produced by these systems, they face problems when undergoing regulation and acceptance in the medical community.

Online recommendation systems have shown recent success in convincing users that the recommendations make sense [4]. Websites such as Amazon.com, MovieLens, Yahoo Launchcast, Netflix, and others explain recommendations in part by telling users “Other people that are similar to you have rated this item highly.” This approach for explanations is easy to understand, and is becoming a familiar approach. Inspired in part by such systems, we propose two techniques to explain classifications provided by support vector machine classifiers for continuous data of relatively low dimensionality. A key distinction between our approach and other approaches for explaining SVM classifiers (see Section 2) is that our techniques explain decisions at the *local level*, i.e. for an individual test point. The first technique involves finding the most relevant support vectors for an individual point, that is those that were most influential in determining the class into which the point was placed. The second technique involves determining the change necessary in a test point’s features to place that point on the separating surface between the two classes. The idea here is that any further change in this direction will place the point on the opposite side of the separating surface, and thus serves as an explanation of how much a test point would need to change in order to be classified in the other class. This technique, which we call border classification, is a variation on the inverse classification technique described below.

2. Related Work

Previous research done in the field of explaining support vector machine classifications includes work in sensitivity

analysis [5], inverse classification [6], and rule extraction [3], [7]–[12]. Of these methods, rule extraction has received the most attention of late. Previously applied to neural networks [13], rule extraction applications have been extended to now include SVMs. SVM rule extraction techniques fall into two broad categories: pedagogical techniques and decomposition techniques [7]. Pedagogical techniques [8] extract rules relating the inputs and outputs of the model, using the trained model to generate training examples which can then be used by a comprehensible learning algorithm (such as decision trees) to create a comprehensible classification model. The principle is that the trained model can create examples that better represent the data than the original dataset which contains errors and noise. Decomposition techniques, on the other hand, are much more reliant on the structure of the SVM. For example, in one approach [3], the SVM hyperplane is approximated with axis-parallel surfaces. Although rule extraction has been known to produce classifiers that are easily comprehensible, they produce secondary models of worse accuracy. Moreover, even though these models may be reasonably understandable from a management perspective, they still lack the simplicity and familiarity to an individual user that recommendation systems have managed to provide. Therefore, instead of attempting to create a new, more easily explainable model based on the model made by the SVM, both of our proposed techniques explain the model made by the SVM directly for an individual point, eliminating the need for an additional model.

Two other methods of understanding classifiers are sensitivity analysis and inverse classification. Sensitivity analysis techniques measure the rate of change in the output of a model caused by the changes in the inputs of the model. This analysis can then be used to measure which input features are most important to achieve accurate output values [5]. Inverse classification techniques take an entirely different approach. Rather than attempting to explain the model, inverse classification describes how one point can be moved into another classification. More formally, the inverse classification problem consists of attempting to find the change in attribute values necessary to move a member of one classification into another classification. A set of prototype cases of each category is required, as well as specification of a similarity function. Our second proposed technique, border classification, is similar to inverse classification in that it finds the minimum change of attributes to switch the classification. However, instead of switching the point to another class, we move the point to the border between classes. Both inverse classification and border classification seek to solve an optimization problem that is highly nonlinear, and thus a global minimum value cannot readily be found. Previous work on inverse classification [6] uses genetic algorithms to solve the problem. We describe how to solve border classification as a nonlinear program, which ensures that the result will be at least a local minimum. Other nonlinear

optimization strategies, including genetic algorithms, could be applied to our approach as well.

The key difference in the conceptualization of inverse classification and border classification can be alternatively explained by considering the algorithms directly. In the case of inverse classification, a set of test points in the opposite class from the test point under consideration is used to seed a genetic algorithm. The goal of the genetic algorithm is to evolve a series of points in this opposite class over a series of generations, and ultimately choose one that is relatively close to the original test point. Thus, the inverse classification paper describes inverse classification as that of finding a point in the opposite class [6]. In some sense, one would hope to do better than this: if one were to find “a point in the opposite class close to a test point,” one could easily find a point even closer by moving in between that point and the separating surface. Therefore, we address this by formulating an optimization problem which constrains the solution to being on the separating surface itself.

Finally, Subianto et al. recently proposed a method for explaining classifiers whose input features are nominal [14]. This idea explains the classifier both at the local level (for an individual data point) and at the global level (for the entire model). The approach ensures local understandability by explaining why each test point was classified the way it was. More specifically, local explanations are defined as “a minimal set of attributes, such that there exists a change of values for the attributes in that set that would change the assigned class” [14]. This goal is similar to the idea of inverse classification and also to our border classification technique. The key difference is that the approach by Subianto et al. only works on discrete-valued datasets, and thus consists of finding a “nearest point” in the opposite class. Border classification differs from this approach in that it is defined for continuous-valued features, and thus it also differs (just as border classification differs from inverse classification) in that it finds a point on the separating surface, not on the opposite side. Subianto et al. also describe a technique for global explainability, which is maintained by attribute weights — i.e., the importance of an attribute in the global classifier in a similar fashion to sensitivity analysis. It is important to note that this method is not restricted to support vector machines, but can be used for any classifier. However, this method is limited by its restriction to discrete data.

3. Notation and Review of SVMs

In this section we define the notation that we will be using to describe support vector machines [1]–[3], [7], and related expressions. Suppose we have a set of m data points $\vec{x}_i \in \mathbb{R}^n$ along with each point's classification y_i , where y_i can take on one of two possible values: -1 or 1 .

The linear support vector machine is defined as the following optimization problem:

$$\min_{\vec{w}, b, \xi_i \geq 0} \frac{1}{2} \|\vec{w}\|_2^2 + C \sum_{i=1}^m \xi_i \quad (1)$$

such that $y_i(\vec{w} \cdot \vec{x}_i - b) + \xi_i \geq 1$

where ξ_i is the error for a given training point \vec{x}_i , \vec{w} is the vector of coefficients for the best separating hyperplane, b is the offset for that hyperplane, and C is a (usually experimentally determined) constant that represents the emphasis that is to be placed on minimizing the error. If C is too large, the emphasis on error avoidance can overwhelm the regularization term ($\|\vec{w}\|_2^2$) and result in overfitting. If C is too small, classification accuracy might be sacrificed.

It is often useful to consider the support vector machine in its equivalent dual formulation [1], [2]:

$$\max_{\alpha_i} -\frac{1}{2} \sum_{i,j} y_i y_j \vec{x}_i \cdot \vec{x}_j \alpha_i \alpha_j + \sum_{i=1}^m \alpha_i \quad (2)$$

such that $\sum_{i=1}^m \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$

By replacing the dot product in the objective function above with a kernel function, we obtain the nonlinear support vector machine [1], [2]:

$$\max_{\alpha_i} -\frac{1}{2} \sum_{i,j} y_i y_j K(\vec{x}_i, \vec{x}_j) \alpha_i \alpha_j + \sum_{i=1}^m \alpha_i \quad (3)$$

such that $\sum_{i=1}^m \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$

Once this problem has been solved, α_i and b (which can be determined from the solution to this problem [1], [2]) can be used to classify test point \vec{x} based on:

$$f(\vec{x}) = \text{sign} \left[\sum_{i=1}^m \alpha_i y_i K(\vec{x}_i, \vec{x}) - b \right] \quad (4)$$

which classifies \vec{x} as either 1 or -1 .

To interpret equation (4), we see that the dual variables α_i represent the importance of a particular support vector within the model. A high value of α_i associated with a particular support vector \vec{x}_i indicates that the kernel value calculated using this support vector will have more influence on the final value for $f(\vec{x})$.

While the kernel function represents an implicit mapping of the points into a higher dimensional space, [1], [2], one can also interpret the kernel $K(\vec{x}_i, \vec{x}_j)$ as a similarity measure between points \vec{x}_i and \vec{x}_j . Different kernels yield different interpretations of similarity. The most popular nonlinear kernel in practice is the Gaussian radial basis kernel:

$$K(\vec{x}_i, \vec{x}_j) = e^{-\gamma \|\vec{x}_i - \vec{x}_j\|_2^2} \quad (5)$$

where γ is a user-chosen parameter. This kernel provides a natural sense of similarity, where a value of 1 indicates that the two points are identical, and a value of 0 indicates that the points are infinitely far apart. If we choose the dot product as the kernel function, we obtain the linear case

of the SVM, as mentioned above. In this case similarity is understood to be related to the cosine of the angle between the two vectors. Since the Gaussian radial basis sense of similarity is somewhat more intuitive, we focus on it throughout our experiments, but our notions of explainability apply to any kernel function.

A number of free and high-quality software packages exist to solve this optimization problem. In our case, we have chosen to leverage SVM^{light} [15] for this purpose, as it is well-known and has several convenient features.

4. Finding important support vectors

When a support vector machine makes a classification, some of the support vectors are more influential than others. This is a function of the support vector's corresponding value of α and of its "similarity" to the test point as determined by the kernel function. Consider the case of an individual applying for a loan. If we are able to identify the support vectors with the largest numerical effect on the classification of the test point (in the direction of its final classification), they allow us to offer an explanation of the form "the individual was rejected for a loan in large part due to his or her similarity to these other individuals that were previously identified as bad credit risks." We therefore define the **pull** $p_i(\vec{x})$ of support vector \vec{x}_i on test point \vec{x} to be a measure of the role \vec{x}_i played in categorizing \vec{x} . Specifically, let:

$$p_i(\vec{x}) = \alpha_i y_i K(\vec{x}_i, \vec{x}) \quad (6)$$

The support vectors with the largest pull values (in magnitude) are the ones that contribute the most to the classification that it receives, and it is these support vectors that provide an "explanation" for the classification for a given test point. The pull is based on the kernel-function similarity between the support vector and the test point, as well as on the α value, which is a measure of how important for classification the support vector is in general. Therefore, presenting a user with these particular support vectors (in an appropriate visualization environment) yields fairly simple information to help a user understand why a classification has been made as it has. In the credit example suggested in Section 1, the support vectors with the largest pull are those individuals who have a strong combination of two factors: they are generally good indicators of whether or not an individual similar to them should receive credit — they have high α values — and they are similar to the loan-seeker in the attributes used to construct the model.

A related problem is that of creating a "report set" of support vectors to report as "explanatory." One extreme is to report all of them, in a list ordered by pull. This list would be sorted in descending order if the test point \vec{x} were classified in class 1, and ascending order if classified in -1 ; this would then place the most relevant support vectors at the top of the list. The other extreme is to simply report the support vector at the top of that list, which is

the one single support vector that had the greatest pull in the direction of the test point's final classification. There are various compromises. After sorting the support vectors by pull values, one could simply pick the top n support vectors on that list. Alternatively, one could look for a transition point when the pull values changed significantly, and report all support vectors above this transition point. In our demonstration software (Section 6), we have chosen to present the top five support vectors on that list. Different applications might find different alternatives to be clearer.

The above approach for choosing explanatory support vectors makes more intuitive sense when using a kernel that always yields non-negative values, such as the Gaussian RBF kernel. In the case of this kernel in particular (and some others), the kernel can be easily thought of as a similarity metric. The top n explanatory support vectors are understood to be the support vectors that dominate due to large α values (the support vectors are globally important), large kernel values (they are similar to the test point), or some combination of the two. This interpretation is more complicated when a kernel that may produce negative values is used, such as the linear kernel. In this case, a support vector in the *opposite* class as the test point might actually sort near the top of the list. Users might consider this confusing; on the other hand, there is a natural interpretation. In the linear case, for example, a negative kernel value indicates some degree of difference between the test point and the support vector, as it requires that the angle between the two vectors is between 90 and 180 degrees. This situation could effectively be described as the support vector "pushing" the test point towards the other class. When using the linear kernel in our software, we currently suppress these "opposite-class" support vectors from being shown, but we believe with proper user training a case could be made for showing them.

5. Finding a nearby border point

Consider again the example of someone seeking a loan who has been rejected. One way of helping this individual understand why a rejection decision was made would be to provide the profile of a fictional individual that would be as similar as possible to this loan-seeker, yet would be just on the border of being "acceptable." Seeing the differences between oneself and a "an approximation to you that would get you accepted for a loan" could provide insight and believability to the classification.

Therefore, given a test point of one classification, we wish to find a nearby point to it on the separating surface. In principle, finding the closest point on the separating surface would be optimal, but due to the highly nonlinear nature of the problem this is not feasible in practice. We therefore focus on finding a point on the border whose distance is *locally* optimal. This nearby point provides a set of relatively minimal feature changes that would result in the point being ambiguously classified. The process of finding this nearby

point, a process that we call "border classification," is found by solving a nonlinear constrained optimization problem. Let \vec{x} be the test point in question, and let \vec{a} be a closest-point candidate. We attempt to find the closest point to \vec{x} while constraining \vec{a} to lie on the separating surface. This can be done by minimizing the square of the Euclidean distance between the two points \vec{a} and \vec{x} :

$$D^2(\vec{a}, \vec{x}) = \sum_{j=1}^n (\vec{a}[j] - \vec{x}[j])^2 \quad (7)$$

Here $\vec{a}[j]$ is the j -th component of \vec{a} , and $\vec{x}[j]$ is the j -th component of \vec{x} . (Minimizing the squared distance simplifies the optimization problem by eliminating the square root that would otherwise be present.) Note that we have chosen to minimize the distance in the original input space, and not in the feature space represented by the kernel. This is intentional: the goal is to present a user with the (locally) minimal change necessary in order to reach the border. While the nearest border point in the feature space has relevance for classification, it doesn't well answer the question for the end-user of "what are the minimal changes that put me on the border?" In order for this information to be understood easily as a comparison, we argue that this calculation should be done in the input space. Of course, if one argued otherwise, our proposed technique could be modified to do so.

The minimization described in (7) is subject to the constraint that \vec{a} lies on the separating surface. Based on the classifier given in (4), the equation for that surface is:

$$\sum_{i=1}^m \alpha_i y_i K(\vec{x}_i, \vec{a}) - b = 0. \quad (8)$$

One convenient side-effect of our formulation of this problem is that we can easily add additional constraints to enhance explainability by taking into account real-world restrictions. For example, certain features (such as age) may be difficult or impossible to change, and therefore one may desire a border classification point whose values for those features are constrained to be the same as for \vec{x} . A solution requiring a change in these "unchangeable" features might not be considered to be particularly useful as an explanation. Furthermore, features can be constrained in other ways to impose physical or other restrictions. For example, physical quantities might be restricted to non-negative values only.

To illustrate an example: suppose that for our loan-seeker, certain features such as age, salary, and outstanding loans might be constrained. It is not helpful to suggest to the user that changing age will help get a loan, as it is not a feature that can be readily adjusted. It is also impossible to earn a negative salary, or have negative outstanding loans. These constraints, combined with the equation for the separating surface, give us the following specific optimization problem:

$$\min_{\vec{a}} \sum_j^n (\vec{a}[j] - \vec{x}[j])^2, \quad (9)$$

Subject to the constraints:

$$\begin{aligned} \sum_{i=1}^m \alpha_i y_i K(\vec{x}_i, \vec{a}) - b &= 0 \\ \vec{a}[q] &= \vec{x}[q] \\ \vec{a}[r] &\geq 0 \\ \vec{a}[s] &\geq 0 \end{aligned} \quad (10)$$

where q , r , and s are the indexes of the dimensions that correspond to age, salary, and outstanding loans, respectively.

One could go further by observing that, for example, it might be easier to pay off loans than to get one's salary increased. Therefore, weights could be added to the objective function (9) to do this. Though we have not implemented this in the current version of our software, it would be a straightforward extension.

In order to solve these minimization problems, the General Algebraic Modeling System (GAMS) was used. We read in the solution values for α_i and b from the model file generated by SVM^{light} during training, and combine them with the other information necessary to format an appropriate GAMS [16] input file with the appropriate variables, constants, and constraints designated. The non-linear solver CONOPT3 [17] is then used to perform a non-linear minimization on the optimization problem, and returns the given coordinates of the locally optimal point \vec{a} . We note that there are a variety of approaches that one might use to improve the quality of this local minimum, including simulated annealing, randomized restarts, genetic algorithms, and other strategies [18]. The inverse classification approach [6], for example, uses genetic algorithms to solve a related problem.

Once a particular value for the border classification point \vec{a} is found, the features for both the user and this nearby point can then be compared in the graphical user interface, granting the user further insight.

We again point out that when solving this non-linear optimization problem, we achieve a distinct goal when compared to the work done on inverse classification using genetic algorithms [6]. Instead of evaluating discrete points of an opposite classification to find a minimal-cost result, we attempt to find a minimal solution by solving an optimization problem with constraints. Furthermore, we are interested in a point on the classification boundary, as opposed to some point of the opposite classification.

6. Results

In order to demonstrate the ideas presented above, we have created a software tool called "SVMzen." SVMzen is specifically targeted to the domain specialist rather than the machine learning specialist. The user interface, shown in a large screenshot at www.cs.carleton.edu/faculty/dmusician/expsvms/screenshot.png, was designed with the intent to be as easy to use and understand as possible. (We have not included the screenshot directly in this

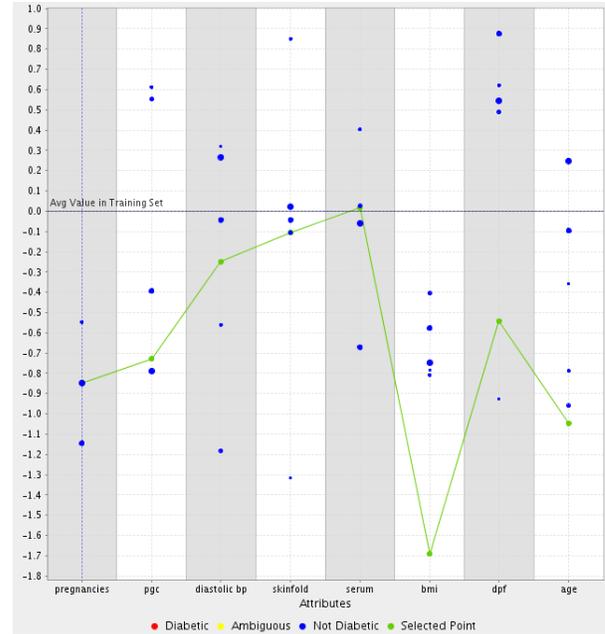


Figure 1: A non-diabetic patient and its five most influential support vectors.

paper due to lack of space.) SVMzen serves as a graphical user interface to build classification models and classify test points (as do Weka [19], LIBSVM [20], and others), but it also displays the *explanations* for these classification models by implementing the proposed techniques.

After a user selects a training set file and chooses parameters for the SVM, SVMzen creates a model file by running SVM^{light} [15]. Next, the user can import a test set file and examine each of the test points individually in the user interface for local understanding of the model. For each selected test point, SVMzen displays the classification and the values for each of its attributes. It also displays an interactive chart which was implemented using jFreeChart [21]. This chart, seen in Figure 1, shows the values for each attribute of the support vectors that had the largest effect on classifying that particular point. Since each feature can have dramatically different scales and ranges, the chart we show has normalized values for each dimension, where we have subtracted the mean and divided by the standard deviation. Sliders to the left of the chart allow the user to modify the values of the test point's attributes and re-classify the point with the new values. This functionality allows the user to manually test the amount of change necessary for each attribute in order to change the classification, and also allows the user to see how sensitive each attribute is to change. Additionally, for a selected test point, SVMzen performs border classification in order to calculate changes that would give the test point an ambiguous classification.

In order to show examples of how the system works, we focus on the Pima Indian Diabetes dataset [22]. The dataset

Attribute	Values
Number of times pregnant	1.00
Plasma glucose	97.00
Diastolic blood pressure	64.00
Triceps skin fold (mm)	19.00
2-Hour serum insulin	82.00
Body mass index	18.2
Diabetes pedigree function	0.30
Age (years)	21.00

Table 1: Medical Data Test Point Values

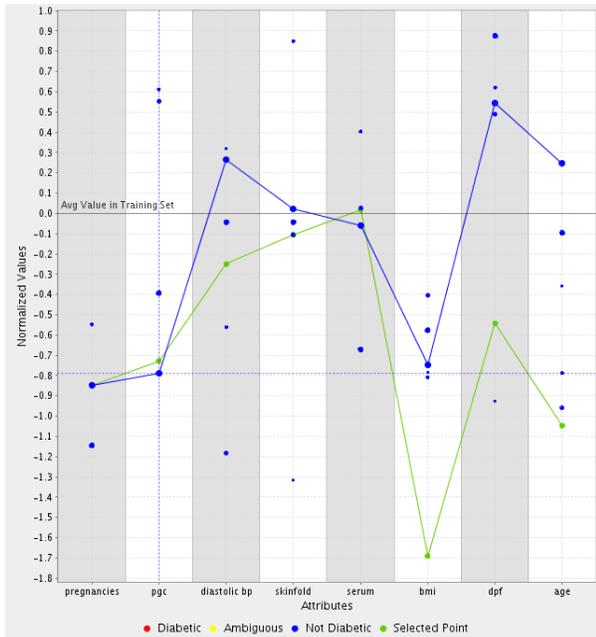


Figure 2: A non-diabetic point and its most influential support vector.

contains eight attributes (as specified in Table 1), and two classifications. We chose to work with the Gaussian radial basis function as a kernel. Based upon accuracies determined by SVM^{light}, we performed some rough experiments on training and test sets in order to pick values for γ and C that yielded reasonable results. For purposes of these visualizations, then, we chose γ to be 0.009, and C to be 1100. In practice, γ and C should be chosen to optimize tuning accuracy via a process such as tenfold cross validation.

We now choose a sample point from this dataset, and discuss the various ways in which SVMzen can help the user understand its classification. The particular point that we have chosen has the attributes listed in Table 1.

Examining the original chart that is generated by SVMzen (Fig. 1) allows us to infer a number of possible explanations for this classification. We notice that the majority of influential support vectors shown have comparatively similar values to our test point for three attributes (triceps skin fold, 2-hour serum insulin, and plasma glucose). Since they have the same classification as our test point, we might begin

Attribute	Suggested Value	Change From Original
Number of times pregnant	3.82	+2.82
Plasma glucose	149.65	+52.65
Diastolic blood pressure	68.84	+4.84
Triceps skin fold (mm)	27.13	+8.13
2-Hour serum insulin	79.96	-2.04
Body mass index	32.01	+13.81
Diabetes pedigree function	0.63	+0.33
Age (years)	21.00	N/A

Table 2: Medical Data Border Classification Values

to conclude that these may be characteristics of a patient with this classification. Our interface allows the user to explore the other features as well. For example, sliding the “pregnancies” slider to 13 (see above-mentioned web-based screenshot) will reclassify this point as diabetic.

Finally, the domain expert might wish to examine some of the risk factors associated with becoming diabetic in the context of this particular patient. Border classification, in a locally minimal fashion, determines the changes required to place the test point on the separating surface between the two classes. On the click of a button, SVMzen provides the values listed in Table 2. (Note that in a similar fashion to the example in Section 5, we have constrained the age of the patient to stay fixed.) This gives the domain expert insight into the patient’s current risk of becoming diabetic by seeing what changes in the patient’s medical records would cause a switch in classification. This also helps the domain expert confirm the diagnosis of a particular patient as the support vector machine is no longer completely a black box, but is now able to provide evidence for why patients are classified as they are.

7. Further Study

There are many directions that additional work on the topic could take. First, while selecting the support vectors with the most significant pull is relatively simple, it is not necessarily the best indicator of which support vectors were actually responsible for putting one particular test point in its current classification. Support vectors with high values of α_i may appear on the list of important support vectors virtually all of the time. In a sense, a test point would not be placed in a given class particularly because of the influence of those support vectors, as they have a strong pull in their direction for *any* test point. This situation would be particularly aggravated when using a Gaussian radial basis kernel with a very small γ , as this tends to homogenize the kernel values across all support vectors. Future work might consider approaches for “filtering out” those support vectors that provide a strong contribution to every test point, and not just to the one under consideration in particular.

Another potential direction for our approach would be to use it in order to determine which features are more

important than others. In the current incarnation of our system, users can use the sliders in SVMzen's interface to modify the value of one feature at a time in order to observe what sorts of changes cause the point to be reclassified. Through such exploratory behavior, the user can gain some insight as to which features play a significant role in the classification. It would be considerably more desirable, however, to automate this process. We also note that our approach in its current form is most usable when the number of features of the dataset is of a size that the user can eyeball all at once (perhaps 25-30 or so). With an automated approach for determining important features, as suggested above, one might show the user only the particularly significant ones, allowing SVMzen to be used on datasets with considerably larger numbers of features.

There are other classification algorithms that share enough similarities with SVMs such that they might also benefit from methods similar to those presented in this paper. Boosting, for example, also results in a classifier that is a weighted combination of terms [23], [24]. The boosted classifier is not a linear combination of kernel functions; rather, it is a combination of simple classifiers. One could look at the "pull" of each simple classifier, attempting again to use it to provide some interpretation as to which simple classifiers played a significant role in classifying an individual point.

There are numerous other variations of support vector machines, and the procedures described here could easily be adapted to many of them. For example, it would be possible to use a similar process to explain the decisions produced by a support vector machine with more than two classes.

8. Conclusion

In this paper we introduce two new techniques for explaining support vector machines on continuous data. Both techniques explain the model on the local level, i.e. for an individual test point, as a recommender system might. The first technique involves finding the support vectors that make the strongest contribution to the classification of a particular test point. The second technique is similar to inverse classification: the goal is to find a relatively minimal change in order to switch the classification of a test point. However, instead of minimally switching the classification (which is not optimally possible in a continuous space), we propose finding the locally minimal change required to move the point to the separating surface of the classes. These techniques add explainability to the results of an SVM classifier in a format with which users of online recommendation systems are quite familiar. We have presented a software tool named SVMzen that allows users to see these explanations graphically. A user can look at a particular test point and determine that the test point was classified in that class due to a specific group of highly weighted support vectors. Furthermore, the user can examine what changes would be necessary to move the test point onto the edge between the

two classes. Additionally, a person who wishes to be placed in the other classification can be told what changes are necessary for that result, which provides prescriptive advice for that individual.

More info about SVMzen is available at www.cs.carleton.edu/faculty/dmusician/expsvm/.

References

- [1] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.
- [2] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*. Cambridge: Cambridge University Press, 2000.
- [3] G. Fung, S. Sandilya, and R. B. Rao, "Rule extraction from linear support vector machines," *Proceedings of the 11th Intl. Conference on Knowledge Discovery in Data Mining*, pp. 38–40, 2005.
- [4] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *CSCW*, 2000, pp. 241–250.
- [5] J. Yao, "Sensitivity analysis for data mining," 2003. [Online]. Available: citeseer.ist.psu.edu/yao03sensitivity.html
- [6] M. V. Mannino and M. V. Koushik, "The cost-minimizing inverse classification problem: a genetic algorithm approach," *Decision Support Systems*, vol. 29, no. 3, pp. 283–300, October 2000.
- [7] D. Martens, B. Baesens, T. van Gestel, and J. Vanthienen, "Comprehensible credit scoring models using rule extraction from support vector machines," *European Journal of Operational Research*, vol. 183, no. 3, pp. 1466–1476, 2007.
- [8] H. Núñez, C. Angulo, and A. Català, "Rule-based learning systems for support vector machines," *Neural Processing Letters*, vol. 24, no. 1, pp. 1–18, 2006.
- [9] J. Diederich, Ed., *Rule Extraction from Support Vector Machines*, ser. Studies in Computational Intelligence. Springer, 2008, vol. 80.
- [10] N. H. Barakat and A. P. Bradley, "Rule extraction from support vector machines: A sequential covering approach," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 6, pp. 729–741, 2007.
- [11] N. H. Barakat and J. Diederich, "Eclectic rule-extraction from support vector machines," *International Journal of Computational Intelligence*, vol. 2, no. 1, pp. 59–62, 2005.
- [12] D. Martens, B. Baesens, and T. V. Gestel, "Decompositional rule extraction from support vector machines by active learning," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 2, pp. 178–191, 2009.
- [13] R. Andrews, J. Diederich, and A. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks," *Knowledge-Based Systems*, vol. 8, pp. 373–389, 1995.
- [14] M. Subianto and A. Siebes, "Understanding discrete classifiers with a case study in gene prediction," *Proceedings of the Seventh IEEE International Conference on Data Mining*, 2007.
- [15] T. Joachims, "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. J. C. Burges and A. J. Smola, Ed. MIT Press, 1999, pp. 169–184.
- [16] GAMS Development Corporation, "GAMS."
- [17] ARKI Consulting & Development, "CONOPT 3.1 Solver."
- [18] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. New Jersey: Pearson Education, Inc., 2003.
- [19] G. Holmes, A. Donkin, and I. H. Witten, "Weka: a machine learning workbench," in *Proceedings of the Second Australian and New Zealand Conference on Intelligent Information Systems*, Brisbane, Australia, 1994, pp. 357–361.
- [20] C.-C. Chang and C.-J. Lin, "LIBSVM Software," 2003, available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.
- [21] D. Gilbert, *The JFreeChart Class Library*. Object Refinery Limited, 2008.
- [22] A. Asuncion and D. Newman, "UCI Machine Learning Repository," 2007. [Online]. Available: <http://www.ics.uci.edu/~mllearn/MLRepository.html>
- [23] R. Meir and G. Rätsch, "An introduction to boosting and leveraging," *Advanced Lectures on Machine Learning (LNAI2600)*, 2003.
- [24] R. E. Schapire, "A brief introduction to boosting," *Proceedings of the Sixteenth Intl. Joint Conference on Artificial Intelligence*, 1999.

View of Boosting as a Search for Randomness Deficiencies

Daniel Burfoot and Yasuo Kuniyoshi

ISI Laboratory, Department of Mechano-Informatics, University of Tokyo, Japan

Abstract—When an optimal statistical model is used to encode a data set as a string of bits, the resulting string will be appear to be random. Conversely, if the bit string shows a randomness deficiency (i.e. is nonrandom), this implies the model is suboptimal and indicates an update that can be used to improve it. This paper presents an abstract algorithm for statistical modeling by searching for randomness deficiencies. The technique constructs a separate model distribution for each data sample, and then incrementally improves the models. In each round the current model CDFs are used to virtually encode the data as a stream of $[0, 1)$ -distributed variables, to which a battery of statistical tests is then applied. Proceeding greedily, at each step the algorithm picks the test that produced the largest randomness deficiency and uses it to update all the model distributions simultaneously. When instantiating this abstract technique to attack the problem of pattern recognition, the result is immediately recognizable as the training phase of a boosting algorithm, where the weak hypothesis functions play the role of the statistical tests for randomness. We call this algorithm FundaBoost; experimental results show that it achieves performance comparable to the well-known AdaBoost algorithm on a set of benchmark problems.

1. Introduction

It is implicit in the work of Shannon that statistical modeling and data compression are nearly equivalent processes: in order to obtain the optimal compression rate for a given data source, one must know its exact distribution. Compared to the philosophically nebulous notion of the “likelihood” of a data set, the compression rate is a tangible and intuitively tractable concept. For this reason insights from information theory are widely used in statistical modeling [1], [2].

Over the past decades a “algorithmic” articulation of information theory has emerged [3]. The insights achieved in this field seem to be deep but primarily theoretical, and have not yet made a substantial impact on the practice of statistical learning. The two flavors of information theory are deeply related, but contain important differences in perspective as well. One such difference in perspective relates to how the two fields approach the problem of compressing data. A traditional information theorist proceeds by choosing a model class and then selecting a set of parameters that minimizes the code length (negative log likelihood) of the data given the model. An algorithmic information theorist might prefer an alternate route: *start* by encoding the data

as a bit stream, and then analyze the stream using a battery of statistical tests. If the result of any test deviates substantially from what would be expected if the stream were completely random, we say it shows a *randomness deficiency*. If a randomness deficiency is detected, it can be exploited to reencode the stream. When this is done three things happen simultaneously: the model is improved, and the bit stream becomes both shorter and more random.

In this paper we present an abstract algorithm for modeling based on the idea of searching for randomness deficiencies. The key conceptual innovation is the realization that it is not necessary to encode the data as bits for the idea to work. Instead we encode the data as variables continuously distributed on the $[0, 1)$ interval; the necessary transformation is specified exactly by the model CDF. Randomness deficiencies in the stream of $[0, 1)$ variables can be exploited to achieve code length savings in the same way as for randomness deficiencies in a bit stream.

After describing the abstract algorithm, we instantiate it to handle the binary pattern recognition problem. The resulting algorithm is immediately identifiable as the training phase of a boosting algorithm where the weak hypotheses play the role of statistical tests for randomness. Experimental results show that the algorithm, which we call FundaBoost, achieves performance comparable to the well-known AdaBoost algorithm [4]. While these results are encouraging, the main purpose of the paper to highlight the new perspective on statistical modeling as a search for randomness deficiencies, and the abstract modeling algorithm (Alg. 1). It should be possible to modify this algorithm to suit a variety of purposes, such as clustering or relationship discovery. Another important advantage of the perspective is that, where a traditional analysis would obtain an optimal set of model parameters and then terminate, our method allows one to continue searching for improvements by applying more and more complicated tests. Of course, a tradeoff must be balanced between the complexity of a test and the improvement it achieves, but in many new applications (such as data mining) very large data sets are available which justify the use of highly complex tests.

2. Background and Related Work

When attempting to predict some event, it is often the case that one cannot define any single rule that makes good predictions with high probability, but one can obtain a large number of rough heuristics that give performance

which is slightly better than random. The goal of boosting is to combine such “weak” hypotheses together to form one “strong” predictor.

Perhaps the most well-known boosting algorithm is called AdaBoost [5]. The key idea of AdaBoost is to associate a weight with each sample, and to increase (decrease) the weight if the sample’s label is predicted incorrectly (correctly) by the previously chosen weak hypotheses. Then, in each boosting round the weak learner that achieves the lowest error on this weighted distribution is selected.

There have been several subsequent studies analyzing AdaBoost and describing variants. It was shown by [2] that the updates to the distribution defined by the weights can be viewed as a problem of finding a new distribution d_{t+1} that is close to the previous one d_t in terms of relative entropy, but which also has the property that the new weak hypothesis has a weighted error of $1/2$. In a similar vein, [6] presented a variant of AdaBoost called InfoBoost, which operates by minimizing the mutual information between the weak hypothesis prediction and the new distribution.

The present work also has conceptual links to the Maximum Entropy (MaxEnt) framework, introduced by [7] and recently used in the field of statistical natural language processing [8]. In MaxEnt one chooses the statistical model that has the maximum entropy while also satisfying a set of empirical constraints determined by the data. The algorithm presented in the current paper can be interpreted as an iterative version of MaxEnt where each iteration updates the model by applying a single new constraint, the empirical value of which is determined by the *encoded* data in the previous iteration. By applying only one constraint per iteration, we obviate the need for a computationally expensive optimization process to find the parameter values. Furthermore, our algorithm provides a mechanism to decide *which* of the potentially large set of empirical constraints (known as context functions in the MaxEnt literature) is the most informative.

3. Statistical Modeling by Fundamental Variable Analysis

Given a data set and a statistical model of it, several questions immediately arise:

- How do we know if our model is good?
- How can we visualize the model quality?
- If the model is imperfect, how can we improve it?

One standard method of answering these questions is to look at the negative log likelihood of the data set given the model. This quantity can be used to compare two models, and can be used to perform gradient descent style algorithms to find good model parameters [9]. However, it does not tell us whether the model class is itself good.

A second method is to sample from the model, and compare the sampled data to the real data. If the sampled

data seems similar to the real data under visual inspection, that provides evidence (but not proof) that the model is good. Sampling can be viewed as a two step process. First a random number generator is invoked to produce $[0, 1)$ uniform random variables. Then the model’s inverse cumulative distribution function (CDF) is used to map these numbers to the space of data outcomes. As an example, imagine you have a distribution corresponding to rolling a fair die. This CDF can be represented as $C = \{c_0 = 0, c_1 = 1/6, c_2 = 1/3, \dots, c_6 = 1\}$. To obtain a sample X from this distribution, you generate a random number $r \in [0, 1)$ and choose the outcome x_i such that $c_{i-1} \leq r < c_i$.

In this paper we consider the opposite process: given a data outcome, we use the model CDF to map the outcome back to find the corresponding $[0, 1)$ value. In the die example, given an outcome $X = 5$, we would choose a value $r \in [2/3, 5/6)$. We call the resulting numbers “fundamental variables”. This word is intended to capture the idea that the $[0, 1)$ variables contain the fundamental randomness of the data, while the structure is expressed by the model. Note how this inverse process depends on both the real data and the model, so that two different models will produce two different sets of fundamental variables from the same data.

By analyzing the distribution of the fundamental variables, we can obtain interesting answers to the three questions posed above. To see this, suppose we generate a set of data D by sampling from a distribution $p(X)$. As we noted above, sampling is just the process of using a model’s inverse CDF to transform from the space of uniform $[0, 1)$ random variables to the data outcome space. If D is generated by sampling from $p(X)$, then $p(X)$ should be a perfect model for D . When we backtransform from the data outcomes of D to the $[0, 1)$ range using the CDF of $p(X)$, the fundamental variables we obtain will be exactly the original set of uniform $[0, 1)$ random variables given to us by the random number generator. If we view them in a histogram, the histogram will be nearly flat.

Now imagine that we generate D by using the distribution $p(X)$ but backtransform using some other (model) distribution $q(X)$. The histogram we now obtain will be uneven, because the model has assigned large space in the probability distribution to outcomes that occur rarely, and vice versa. This unevenness provides direct visual evidence of imperfections in the model. This idea is illustrated in Figure 1. The original data set is created by sampling from a Gaussian mixture model with two components (Figure 1(a)). Our first attempt at modeling the data was a single Gaussian, shown as the dashed line in Figure 1(c). This model is obviously quite imperfect, and this imperfection is illustrated by the highly uneven fundamental variable histogram (FVH) (Figure 1(b)) obtained by backtransforming the data using the model. Furthermore, as we discuss below, the uneven FVH provides a direct way to improve the model. One simply remaps the model CDF by expanding or contracting

regions based on the histogram bin heights (Figure 1(d)).

3.1 CDF Remapping Transformation

The transformation illustrated in Figure 1(d) can be formalized as follows. Let $C = \{c_0 = 0, c_1, c_2, \dots, c_n = 1\}$ be a CDF for a discrete random variable X with n outcomes $\{x_1, x_2, \dots, x_n\}$ such that $p(x_i) = c_i - c_{i-1}$. Given two other CDFs, a source distribution $E = \{e_0, e_1, e_2, \dots, e_m\}$ and a target distribution $E' = \{e'_0, e'_1, e'_2, \dots, e'_m\}$ the remapping transformation is defined as:

$$\begin{aligned} j(i) &= \{j : e_j \leq c_i < e_{j+1}\} \\ c_i &= e_{j(i)} + \beta(e_{j(i)+1} - e_{j(i)}) \\ c'_i &= e'_{j(i)} + \beta(e'_{j(i)+1} - e'_{j(i)}) \end{aligned} \quad (1)$$

Thus $c_i \in [e_{j(i)}, e_{j(i)+1}]$ and $\beta \in [0, 1]$. We will be particularly interested in cases where the CDFs come from a histogram: the source E corresponds to the bin widths and the target E' corresponds to the (normalized) bin heights. Note that if the CDF window of a data outcome x_i is contained entirely within one of the E windows so that $e_{j-1} \leq c_{i-1} < c_i \leq e_j$, then the total change in the probability of x_i can be expressed simply:

$$\begin{aligned} \alpha &= \frac{e'_j - e'_{j-1}}{e_j - e_{j-1}} \\ p'(x_i) &= \alpha p(x_i) \end{aligned} \quad (2)$$

In other words the probability is simply scaled by the same factor α as the histogram bin containing the CDF data outcome window.

3.2 Code Length Savings

We now analyze the bit savings achieved by the histogram based transformation process described above. Suppose we have a sequence of data samples $\{x_1, x_2 \dots x_N\}$. Each data sample x_k is associated with its own model distribution q_k . We have constructed a fundamental variable histogram by mapping each x_k to the $[0, 1)$ unit segment using the CDF of q_k . The FVH has bin widths W_j and normalized bin heights $H_j = N_j/N$. For now, we assume that each data outcome can be unambiguously assigned to a histogram bin. Because of this assumption we are licensed to use the simple update rule of Equation 2, which for the current problem reads:

$$q'_k(x) = q_k(x) \frac{H(x)}{W(x)}$$

Where $H(x)$ and $W(x)$ denote the height and width of the bin to which the data outcome x is assigned. Note that this equation specifies updates for all the outcomes x of each of the distributions q_k , though here we are only interested in the x_k outcome. The before (L) and after (L') code lengths are given by:

$$\begin{aligned} L &= - \sum_{k=1}^N \log q_k(x_k) \\ L' &= - \sum_{k=1}^N \log q_k(x_k) \log \frac{H(x_k)}{W(x_k)} \\ &= - \sum_{k=1}^N \log q_k(x_k) - \sum_k \log \frac{H(x_k)}{W(x_k)} \end{aligned}$$

Thus the total savings is given by:

$$L - L' = \sum_{k=1}^N \log \frac{H(x_k)}{W(x_k)} \quad (3)$$

$$= \sum_j \log \frac{H_j}{W_j} \sum_{k': B(x_{k'})=j} 1 \quad (4)$$

$$= N \sum_j H_j \log \frac{H_j}{W_j} \quad (5)$$

The notationally difficult step in this derivation is the transition from Equation 3 - 4. Since all outcomes that are assigned to the same bin get the same factor, we can rewrite the sum as first going over all bins (j -index) and then over all outcomes assigned to a given bin (k' -index). The step 4 - 5 is justified by the fact that the number of outcomes assigned to a bucket is exactly N times the normalized bucket height H_j . The final quantity can be identified as N times the Kullback-Leibler divergence between the distributions implied by the normalized histogram bin heights and the bin widths. Clearly, if the histogram is flat then the savings will be nearly zero; if it is uneven, substantial savings can be achieved by applying the update. A significantly uneven fundamental variable histogram thus indicates a randomness deficiency that can be exploited.

The above result may seem at first glance to be an obvious corollary of the definition of the Kullback-Leibler divergence, which is the code length penalty paid for using a model q to encode a data source with some other distribution p . What makes it different (in our view) is the fact that here there is no single source model or source distribution: each data sample x_k is encoded with its own model q_k . We are not performing one single update to a single distribution but rather performing N updates simultaneously. This property is important in the boosting algorithm given below, where each data outcome is associated with its own potentially unique model, and we apply a single test to update them all simultaneously.

After updating the models we must also update the fundamental variables, as they depend on the models. The model update has expanded regions with many FV outcomes and contracted regions with few outcomes, so the post-update histogram will be flat or nearly so. That means we can achieve no further savings by applying another

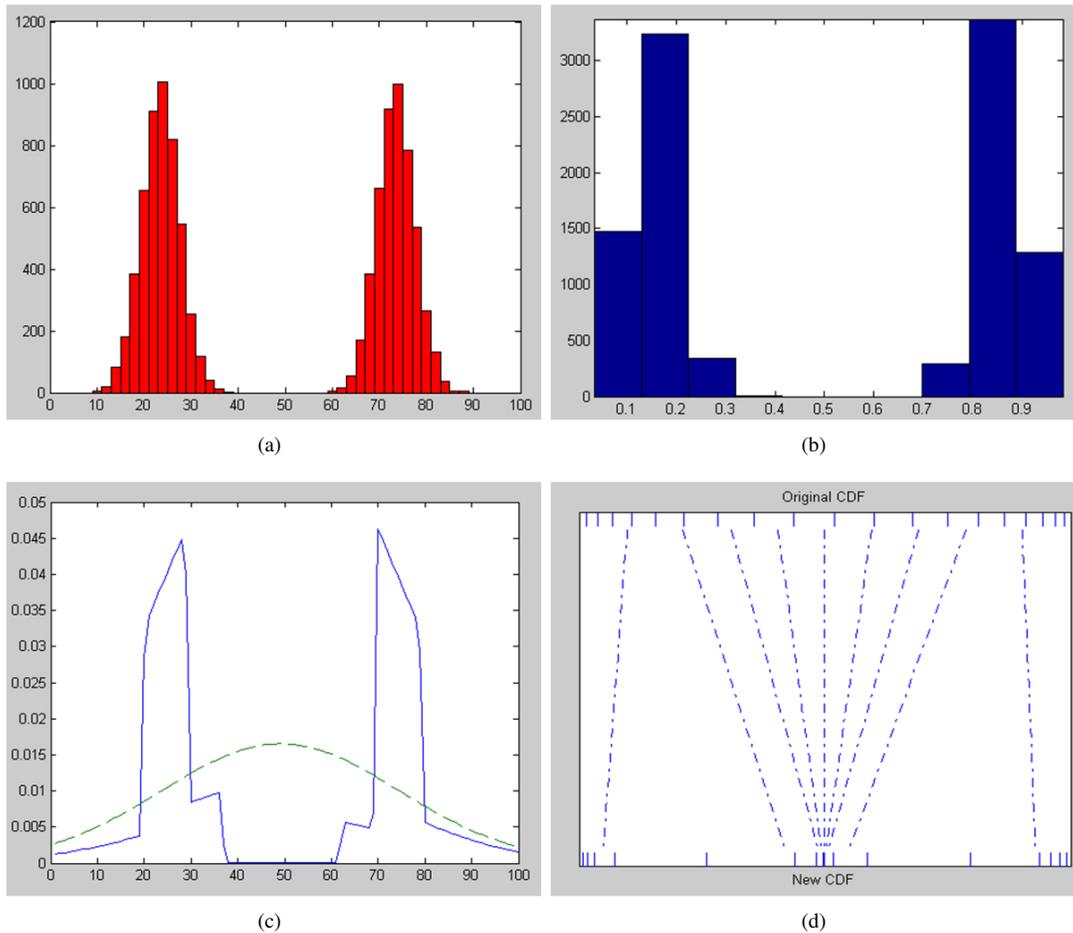


Fig. 1: Illustration of fundamental variable based reencoding. The original data is shown in (a). This data is modeled using a Gaussian with the optimal mean and variance, shown as the dashed line in (c). Backtransforming using this model and the given data yields a fundamental variable histogram (b), with an associated transformation (d). Using the transformation to update the original model yields the solid curve in (c), clearly improving the fit.

transformation to the same set of data outcomes. What we *can* do is define some subset of the full dataset. The full dataset should have a nearly flat histogram, but if we are smart or lucky we may be able to find subsets with highly uneven histograms. We can then update the model distributions for the subset, and achieve another code length savings. To see how this works, consider the following set of fundamental variables:

.41, .45, .06, .32, .05, .14, .27, .48, .48, .33
 .83, .52, .92, .97, .84, .88, .87, .70, .83, .59

If all of these are taken together and viewed in a two bin histogram, the bins will have the same size. However, the above list of FVs is not random. If we separate the first half of the above list from the second half, we will obtain two subsets with very uneven histograms. We can then apply the appropriate transformation to the subsets,

and achieve another savings. The functions that define these subsets are the statistical tests for randomness discussed above. In general, the tests will not depend on the position of the data sample in the list, but rather on some context c_k associated with each data outcome x_k . These contexts should be defined based on the application.

These observations suggest the following abstract iterative algorithm for statistical modeling (Alg. 1). The algorithm maintains an individual model distribution for each data sample. In each round, fundamental variables are obtained by backtransforming the data using the current models. Then a battery of statistical tests \mathcal{W} is applied to the stream of FVs. The test that reveals the largest randomness deficiency in the encoded data, as quantified by Equation 5, is used to update all the models simultaneously.

There are two related technical problems that we have not yet addressed. First, since we are working with discrete variables, there will be a range of values in the $[0, 1)$ interval

Algorithm 1 Abstract modeling algorithm.

given data samples $X = \{x_k\}$, associated contexts $C = \{c_k\}$, statistical test battery \mathcal{W} .
 initialize $Q = \{q_k\}$ as uniform distributions
for $t = 1 : T$ **do**
 $FV = Q \rightarrow \text{backtransform}(X)$
 for $w \in \mathcal{W}$ **do**
 $FV_{sub} = w \rightarrow \text{select}(FV, C)$
 $H = \text{histogram}(FV_{sub})$
 $\text{score}(w) = |FV_{sub}| \cdot \text{KLD}(H)$
 end for
 let $\{w_t^*, H_t^*\}$ be the best test, histogram
for all k **do**
 if $w_t^*(c_k)$ **then**
 $q_k := H_t^* \rightarrow \text{cdf-remap}(q_k)$
 end if
end for
end for
 output final distributions q_k , optimal tests w_t^* ,
 transformation parameters H_t^*

that maps to any given data outcome. So, how do we choose an exact value for the fundamental variable? Second, in practice the assumption that each CDF data outcome window fits into exactly one histogram bin window is not valid. The actual histogram-based remapping will result in some CDF windows being differentially updated: the region falling on one side of a histogram bin boundary will be stretched by some amount, while the region falling on the other side will be stretched by a different amount. Thus, the expression for the codelength savings of Equation 5 is not exact. The answers to these questions should be chosen based on the intended application; we discuss how to handle them for the pattern recognition problem in the next section.

Algorithm 2 The classification algorithm.

Given: learned parameters w_t^*, λ_t^*
 new sample c to classify
 initialize $q = \{0, .5, 1\}$ // model CDF
for $t = 1 : T$ **do**
 if $w_t^*(c)$ **then**
 $q := \lambda_t^* \rightarrow \text{cdf-remap}(q)$
 end if
end for
 if $q_1 > .5$, output TRUE, else output FALSE

4. Boosting Algorithm

In this section we describe FundaBoost, which is the result of instantiating the abstract algorithm to address the problem of binary classification. In this domain, we have a set of binary labels, which we wish to predict using a set of

corresponding “raw” data objects. These correspond to the data samples X and associated contexts C of Algorithm 1. The statistical tests \mathcal{W} are defined based on the raw data objects; these are equivalent to the weak learner hypotheses of boosting. The abstract algorithm is the *training* phase of the boosting process; the outputs are the weak learners w_t^* and parameters H_t^* . Because there are only two outcomes, we use two-bin fundamental variable histograms; thus each round has only one free parameter which we call λ_t^* (equivalent to H_t^*). The w_t^* and λ_t^* obtained from the training process can be used to classify new unseen data objects.

The goal of FundaBoost is to obtain good models $q(X|C)$ of the probability of a label given a data object. The distributions q_k are simply the current model probabilities of a positive or negative outcome. Progress is measured by the codelength (negative log likelihood) that would be required to encode the x_k data using the current models q_k . Model probabilities can be easily transformed into predictions if necessary. Also, note that the training set error is bounded by the current codelength, since each incorrect guess requires a codelength of at least one bit.

As mentioned above, there are two technical problems that must be dealt with to complete the algorithm. The first is how exactly to choose FVs, given that a particular data outcome corresponds to a range of possible values on the $[0, 1)$ interval. One simple method is to select FVs by random sampling on the allowed range. This works well enough, but has the unappealing consequence of injecting randomness into the algorithm. A better solution comes by representing the FV using its full allowed range. When incrementing a histogram count depending on an FV, we simply add the probability that the FV would fall into the bin if it were selected at random from its current range. The second problem is the bin overlap issue, which means that the savings from Eq. 5 is not exact. This does not matter for choosing weak learners, because the optimal weak learner will almost always produce the largest histogram unevenness. However it is a problem for choosing the optimal parameter λ_t^* to use in the CDF remapping step (otherwise we would just use the histogram bin count). Fortunately it is easy to find the optimal λ_t^* given the choice of w_t^* .

To see this, consider the CDF remapping update implied by a two-bin histogram transformation (Equation 1). Let the pre-update CDF be given by $q_k = \{q_k^0 = 0, q_k^1, q_k^2 = 1\}$. Let $q_k^1 = A_k + B_k$, where A_k is the region of the probability window that falls on the left side of the histogram bin partition, and B_k is the region that falls on the right side. Then by following the CDF remapping rule, the post-update CDF will be given by $q_k^{1'} = \lambda A_k + (2 - \lambda) B_k$.

To find the optimal choice for λ using this update scheme, we must take the label data into account. Let variables (a_k, b_k) be equal to (A_k, B_k) if the actual outcome of x_k is true, and $(.5 - A_k, .5 - B_k)$ otherwise. Then the pre-update probability assigned to the correct outcome is $p_k = a_k + b_k$,

and the post-update probability is $p'_k = \lambda a_k + (2 - \lambda)b_k$. Now to find the optimal λ we want the value that maximizes the total codelength. The codelength and its derivatives are:

$$L' = - \sum_{k=1}^N \log(\lambda a_k + (2 - \lambda)b_k) \quad (6)$$

$$\frac{dL'}{d\lambda} = - \sum_{k=1}^N \frac{a_k - b_k}{2b_k + \lambda(a_k - b_k)} \quad (7)$$

$$\frac{d^2L'}{d\lambda^2} = \sum_{k=1}^N \frac{(a_k - b_k)^2}{(2b_k + \lambda(a_k - b_k))^2} \quad (8)$$

The second derivative is clearly always positive and so we can find a root of the first derivative (which is the codelength minimum) using a fast binary search in λ . In summary, in each boosting round FundaBoost selects a weak hypothesis w_t^* based on the FVH unevenness score (Eq. 5), then finds an optimal choice of λ_t^* . Finally all the distributions q_k such that c_k satisfies w_t^* are updated using a CDF remapping transformation with parameter λ_t^* .

5. Experimental Results

We tested the performance of the proposed algorithm in comparison to AdaBoost.M1 on 25 of the datasets from the UCI machine learning repository. As the proposed algorithm currently only works for binary classification problems, the datasets with multiple class labels were modified to work as binary classification problems. For several datasets there was one class much larger than the others; we defined the binary problem to be predicting this class. When there was no such class, we attempted to package similar classes together. For the datasets with prespecified partition of training and test data we used the given partition. For the other datasets without such a partition, we used 10-fold cross validation.

An important part of boosting is the selection of the set of weak hypotheses; the following function bank was used in these experiments. For numeric attributes, a set of 200 threshold points were chosen equally spaced between the minimum and maximum values of the attribute on the training data. An associated weak hypothesis was defined that returns true if the attribute is below the threshold or missing. For nominal attributes, a single function was defined for each possible value that returns true if the attribute is not missing and equal to the specified value. The logical inverse of the above functions were also used. This choice for the weak classifier library is similar to **FindAttRule** described in [5].

In addition to defining the library of weak hypotheses, the critical parameter that must be chosen for boosting algorithms is the number of rounds T . If too small a choice is made for T , one might not achieve a sufficiently powerful classifier; if too large a choice is made then overfitting will occur. AdaBoost seems to be more resistant than other algorithms to the problem of overfitting due to its ability

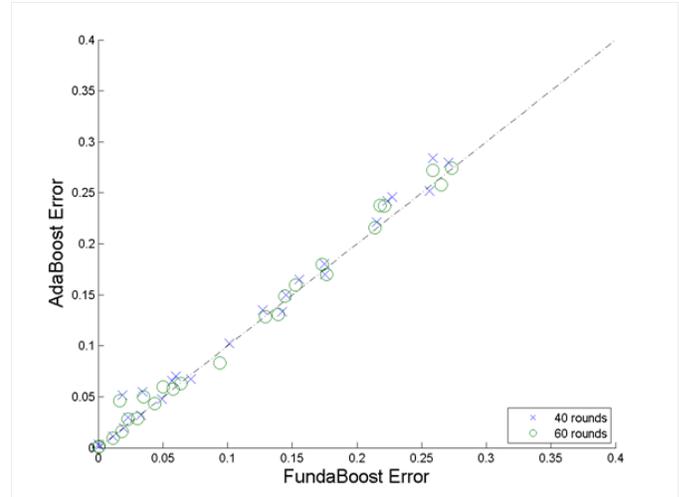


Fig. 2: Generalization error performance comparison of FundaBoost vs. AdaBoost for $T = 40$ and $T = 60$ boosting rounds. Each dot corresponds to one dataset. Points above the line indicate superior performance for FundaBoost.

to increase the margin between training examples [10], but overfitting is still a problem. We tested both algorithms with several choices for the T parameter. Note that the structural similarity of the two algorithms allows clean comparisons: we test both algorithms using the same problem definition, the same set of weak hypotheses, and the same T value.

Table 1 shows a comparison of the performance of the two algorithms for $T = \{20, 100, 200\}$ boosting rounds. The column corresponding to a given value of T shows the number of test errors made by the two algorithms. The “winner” is shown in the last column, with capital letters indicating a difference in error rate of greater than 1%. The final row shows both the total number of significant wins and the total number of overall wins. Figure 2 shows the performance for $T = 40$ and $T = 60$ boosting rounds.

The most striking observation is the very strong advantage of FundaBoost in the low T regime. Another significant detail is that FundaBoost does well on the game based domains Connect4, Krk and Krkp (the latter two are chess-related problems). While better performance may be achieved for higher T values on some domains, on others overfitting has clearly started to occur by $T = 200$. At first we expected AdaBoost to catch up to and then surpass FundaBoost as T became large, but this does not seem to occur. Overall, the results are not revolutionary, but are encouraging and indicate that the basic ideas of the paper are sound.

6. Conclusion

Motivated by ideas from the field of algorithmic information theory, this paper presented a new method for statistical modeling by searching for randomness deficiencies. The key conceptual innovation is a way of transforming data

Table 1: Performance comparison of AdaBoost and FundaBoost

Domain	# test	$T = 20$			$T = 100$			$T = 200$		
		AB	FB	win	AB	FB	win	AB	FB	win
Abalone	4170	924	909	fb	899	900	ab	874	892	ab
Adult	15060	2356	2367	ab	2213	2162	fb	2191	2146	fb
Annealing	100	12	6	FB	6	6	-	5	5	-
BreastW	690	34	34	-	30	31	ab	30	30	-
Car	1720	99	58	FB	81	60	FB	86	60	FB
Connect4	67550	17721	16926	FB	15509	14278	FB	14969	13995	FB
Cont	1470	403	404	ab	416	409	fb	427	412	FB
Cover	100000	24788	23381	FB	23419	21868	FB	22777	21691	FB
CreditA	690	94	96	ab	92	100	AB	97	109	AB
CreditG	1000	256	261	ab	254	257	ab	251	258	ab
Flare	1060	179	191	AB	183	188	ab	186	187	ab
Hypo	3160	33	36	ab	33	40	ab	34	39	ab
Krk	28050	5233	5188	fb	5110	4844	fb	5154	4847	FB
Krqp	3190	174	91	FB	139	51	FB	122	47	FB
Magic	19020	3346	3108	FB	2969	2837	fb	2913	2775	fb
Mushroom	8120	136	19	FB	0	0	-	0	0	-
Satimage	2000	70	77	ab	59	60	ab	59	62	ab
Segment	2310	63	49	fb	29	38	ab	28	39	ab
Shuttle	14500	33	41	ab	11	5	fb	10	5	fb
Sick	2800	80	60	fb	77	71	fb	72	70	fb
Spam	4600	354	338	fb	279	290	ab	267	288	ab
Splice	3190	227	200	fb	180	182	ab	186	174	fb
Vehicle	840	116	104	FB	56	75	AB	49	74	AB
Waveform	5000	772	714	FB	594	618	ab	578	603	ab
Yeast	1480	423	393	FB	393	390	fb	396	395	fb
Total	FundaBoost	10/16			4/11			6/12		
Total	AdaBoost	1/8			2/12			2/10		

into a stream of “fundamental” variables, similar to bits, which represent the essential randomness of the data. The search for randomness deficiencies is performed by applying a battery of statistical tests, each one of which picks out a different subset of the stream. If the fundamental variable histogram of any subset is substantially uneven, it indicates a randomness deficiency that can be exploited to improve the model. Since most statistical modeling techniques do not involve encoding data, they cannot take advantage of this potentially powerful method of model improvement.

Based on the idea of fundamental variables, an abstract modeling algorithm was proposed that works by iteratively updating a set of model distributions. At each step the statistical test which produced the largest randomness deficiency is used to update all the models simultaneously. When the abstract algorithm is instantiated to handle the binary classification problem, the resulting algorithm is immediately identifiable as a form of boosting where the weak hypotheses perform the function of the statistical tests for randomness.

We showed that the performance of this new algorithm compares favorably to AdaBoost on 25 standard machine learning datasets. The performance gains are not revolutionary, but the results show that the technique is basically sound, and justify further investigation. Our future work is to investigate how the abstract algorithm can be modified

for use in different applications.

References

- [1] J. Rissanen, “Modeling by shortest data description,” *Automatica*, vol. 14, pp. 465–471, 1978.
- [2] J. Kivinen and M. Warmuth, “Boosting as entropy projection,” in *Proceedings of the twelfth annual conference on Computational learning theory*. ACM New York, NY, USA, 1999, pp. 134–144.
- [3] M. Li and P. Vitanyi, *An Introduction to Kolmogorov Complexity and Its Applications*. Springer Verlag, 1997.
- [4] Y. Freund and R. Schapire, “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting,” *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.
- [5] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” in *International Conference on Machine Learning*, 1996, pp. 148–156.
- [6] J. Aslam, “Improving algorithms for boosting,” in *Proc. 13th Annu. Conference on Comput. Learning Theory*. Morgan Kaufmann, San Francisco, 2000, pp. 200–207.
- [7] E. T. Jaynes, “Information theory and statistical mechanics,” *Phys. Rev.*, vol. 108, no. 2, pp. 171–190, Oct 1957.
- [8] R. Rosenfeld, “A maximum entropy approach to adaptive statistical language modeling,” *Computer, Speech and Language*, vol. 10, pp. 187–228, 1996.
- [9] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, “A learning algorithm for Boltzmann machines,” *Cognitive Science*, vol. 9, pp. 147–169, 1985.
- [10] R. Schapire, Y. Freund, P. Bartlett, and W. Lee, “Boosting the margin: A new explanation for the effectiveness of voting methods,” *Annals of Statistics*, vol. 26, pp. 1651–1686, 1998.

A Probabilistic Model of Pattern Recognition on Abstract Data

Chun-Hung Tzeng

Computer Science Department, Ball State University, Muncie, IN 47306, USA

Abstract—This paper introduces a mathematical theory for two-class pattern recognition based on a formally defined similarity, which is a reflexive and symmetric binary relation. The knowledge discovering process from a training data set $\{(X_i, Y_i)\}$ searches for a similarity of X_i 's so that similar X_i 's are likely of the same class and to compute a representative clustering of X_i 's. For an observation $X = x$, a classifier first computes the component P_k of the partition generated by the clustering so that $x \in P_k$, and then makes its decision based on the posterior probability $P(Y = 1|X \in P_k)$ on the training data set as in naive Bayes. Both Bayes and Neyman-Pearson Theorems are true for the classifiers. The experiment shows the trade-off between the number of representatives and classifier performance.

Keywords: Similarity, Representative-clustering, Pattern-recognition, Classifier

1. Introduction

Pattern recognition is about predicting the unknown nature of an observation. In a classical model (e.g., [1], [2]), an observation is a d -dimensional vector x . The unknown nature is a class, which is denoted by y and takes values in a finite set $\mathcal{C} = \{0, 1, 2, \dots, M\}$. The task is to create a classifier, which is a function $g : \mathcal{R}^d \rightarrow \mathcal{C}$. The value $g(x)$ is the prediction of y , given x . A probabilistic setting (e.g., [1]) considers a random pair (X, Y) on $\mathcal{R}^d \times \mathcal{C}$, of which a distribution describes the frequency of encountering particular pairs in practice. The error rate of a g is $L(g) = P(g(X) \neq Y)$.

In the two-class problem $\mathcal{C} = \{0, 1\}$, there are other types of error [3]: the false positive rate $L^{(0)}(g)$ and the false negative rate $L^{(1)}(g)$. Given $X = x$, let $\eta(x) = P(Y = 1|X = x)$ be the posterior probability. For a real θ ($0 < \theta < 1$), the classifier g_θ is defined as $g_\theta(x) = 1$ iff $\eta(x) > \theta$. In the classical theory, the classifier $g_{\frac{1}{2}}$ is the Bayes classifier and has the minimal error rate (i.e., the Bayes error). Neyman-Pearson Theorem shows that g_θ for some θ minimizes the false negative rate if the false positive rate is required to be kept under a certain level.

In most cases, the distribution of (X, Y) is unknown. To design a classifier is based on a training data set $\{(X_i, Y_j) | 1 \leq i \leq n\}$. Many classification rules have been proposed (e.g., [1], [2]). For example, the k -nearest-neighbor rule takes a majority vote over the Y_i 's in the subset of k pairs (X_i, Y_i) that have the smallest values $\|X_i - x\|$.

To classify a given observation $X = x$, a classifier first searches for information about x from certain knowledge discovered from the training data set and then makes the decision. The searched information is often incomplete. We call such information *heuristic information*.

A formal probabilistic formulation of heuristic information has been proposed [4], which is represented by a partition $\{P_k\}$ of the random space X . For an observation $X = x$, the search is to compute the particular P_k for which $x \in P_k$. Then the decision is based on the posterior probability $P(Y = 1|X \in P_k)$. The decision will be improved if the heuristic information model is more precise (i.e., the partition is finer).

To discover knowledge from the training data set, similarities play a central role. For example, the k -nearest-neighbor rule uses the Euclidean distance to measure similarities. Most similarity measures are reflexive and symmetric. That is, each x is similar to itself, and y is also similar to x if x is similar to y . This paper uses the two properties to define an abstract similarity. The first step of knowledge discovering is to search for a similarity so that similar X_i 's are likely of the same class. The second step is to compute a data clustering based on the similarity. There are two goals in this process. On the one hand, each cluster contains as many X_i 's as possible and the number of clusters is as few as possible. On the other hand, high percentage of X_i 's in each cluster belong to one class.

This paper introduces a mathematical theory of the approach above for general data, not restricted to the real \mathcal{R}^d . The abstract similarity is called a *tolerance relation* [5]. A space Ω with a tolerance relation ξ is a *tolerance space*, denoted by $\Omega^\xi = (\Omega, \xi)$. The neighborhood of an element x is the set of all elements similar to x , denoted by $\xi(x) = \{u \in \Omega : \xi(x, u)\}$. Let \mathcal{F}_ξ be the Borel field generated by all neighborhoods. The probability measure in the theory is on the measurable space $(\Omega^\xi, \mathcal{F}_\xi)$. The pattern recognition is then represented by a random pair (X, Y) , $X \in \Omega$ and $Y \in \{0, 1\}$. A classifier is a measurable function $g : \Omega \rightarrow \{0, 1\}$. Given $X = x$, the posterior probability $\eta(x) = P(Y = 1|X \in \bar{x})$ is a measurable function, where \bar{x} is the minimal measurable set containing x . The classifiers g_θ 's are defined based on η . Both Bayes and Neyman-Pearson Theorems are true in this abstract model.

On a training data set $\{(X_i, Y_i)\}$, we search for a similarity ξ on X_i 's and compute a representative clustering $\mathcal{R}_\xi = \{R_1, R_2, \dots, R_k\}$, which generates a partition of X_i 's. For a given value $X = x$, we compute the component

$H(x)$ of the partition containing x by comparing x with the representatives. The decision is then based on the posterior probability $P(Y = 1|X \in H(x))$ as in naive Bayes.

To measure the choice of similarity, this paper computes the number of surprising X_i 's (i.e., not similar to any other X_j) and the conditional probability of two X_i 's being of the same type, given that they are similar. The number of surprising records should be as few as possible and the conditional probability should be as large as possible. To measure the performance, this paper uses error rates and the receiver operating characteristic (ROC) analysis.

Good prediction is expected if the training data set contains all typical patterns and the similarity ξ is suitably selected so that the representatives represent most of the typical patterns. The experiment demonstrates the trade-off between computations and classifier performance.

The rest of this paper is organized as follows: Section 2 reviews tolerance space and its data clustering. Section 3 introduces the mathematical theory of classifiers and errors. Section 4 introduces the supervised learning and classifiers. Section 5 describes the experiment. Section 6 is the conclusion.

2. Tolerance Space and Representative Clustering

Let Ω be an abstract set. A binary relation ξ on Ω is a subset of the Cartesian product: $\xi \subset \Omega \times \Omega$.

Definition 2.1 A *tolerance relation* ξ on Ω is a binary relation with the following conditions:

- 1) $(x, x) \in \xi$ for any $x \in \Omega$ (reflexivity), and
- 2) $(x, y) \in \xi \Rightarrow (y, x) \in \xi$ (symmetry).

The pair (Ω, ξ) is called a *tolerance space*, denoted by Ω^ξ . We also use ξ as a predicate: $\xi(x, y)$ iff $(x, y) \in \xi$. If $\xi(x, y)$, we say that x is ξ -similar to y , or x and y are ξ -similar. We omit ξ when there is no ambiguity. Any undirected graph is a tolerance space (and vice versa), where Ω is the set of all vertices, and two vertices are similar if they are the same vertex or adjacent (e.g., Fig. 1).

The minimal tolerance relation is the *discrete tolerance relation* $\{(x, x) \mid x \in \Omega\}$, in which each x is similar to itself only. The corresponding tolerance space is called the *discrete tolerance space* Ω^0 . The maximal one is $\Omega \times \Omega$ and is called the *trivial tolerance relation*, where all elements are similar. The corresponding tolerance space is the *trivial tolerance space* Ω^∞ .

For each $x \in \Omega$, the set $\xi(x) = \{u \in \Omega : \xi(x, u)\}$ is the *neighborhood* of x . Let \mathcal{F}_ξ be the Borel field generated by all neighborhoods. The pair $(\Omega^\xi, \mathcal{F}_\xi)$ is a measurable space and each element of \mathcal{F}_ξ is called a ξ -measurable set. For example, consider $\Omega = \mathcal{R}^n$, the n -dimensional Euclidean space, and the tolerance relation $\xi(x, y)$ iff $\|x - y\| < \varepsilon$ for a fixed given $\varepsilon > 0$, where $\|x - y\|$ is the Euclidean distance.

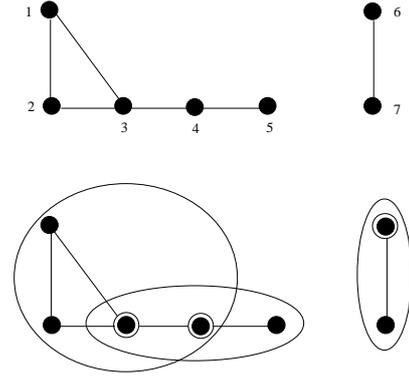


Fig. 1: A tolerance space and a minimal representative clustering.

Then the corresponding \mathcal{F}_ξ is the collection of all Borel sets in \mathcal{R}^n .

For any $x \in \Omega^\xi$, the singleton $\{x\}$ is not always ξ -measurable (i.e., in \mathcal{F}_ξ). In Fig. 1, for example, the singletons $\{3\}$, $\{4\}$, and $\{5\}$ are ξ -measurable, but $\{1\}$, $\{2\}$, $\{6\}$, and $\{7\}$ are not ξ -measurable. We define indistinguishable elements as follows.

Definition 2.2. Two elements x and u in Ω^ξ are *indistinguishable*, denoted by $x \sim u$, if they have the same neighborhood (i.e., $\xi(x) = \xi(u)$).

Note that \sim is an equivalent relation. Let $\bar{x} = \{u \mid u \sim x\}$, the set of all indistinguishable elements of x , which is called the *indistinguishable set* of x and satisfies $\bar{x} = \bigcap_{u \in \xi(x)} \xi(u) - \bigcup_{u \notin \xi(x)} \xi(u)$. In Fig. 1, for example, $\bar{1} = \bar{2} = \{1, 2\}$ and $\bar{6} = \bar{7} = \{6, 7\}$. To avoid advanced measure theory [6], we assume in this paper all spaces are finite, which is sufficient for the application discussed later. The set \bar{x} is the smallest ξ -measurable set containing x .

A function from a tolerance space to another tolerance space $f : \Omega^\xi \rightarrow \Phi^\zeta$ is *measurable* if $f^{-1}(A) \in \mathcal{F}_\xi$ for all $A \in \mathcal{F}_\zeta$. A measurable function maps indistinguishable sets into indistinguishable sets.

Theorem 2.3 If $x \sim y$ in Ω^ξ and the function $f : \Omega^\xi \rightarrow \Phi^\zeta$ is measurable, then $f(x) \sim f(y)$ in Φ^ζ .

Proof: Let $a = f(x)$. Since \bar{a} is ζ -measurable and \bar{x} is the smallest ξ -measurable set containing x , the inverse $f^{-1}(\bar{a})$ is ξ -measurable and hence contains \bar{x} . Therefore, $y \in \bar{x} \subseteq f^{-1}(\bar{a})$, that is, $f(y) \in \bar{a}$ and $f(x) \sim f(y)$. ■

From this theorem, for any real-valued measurable function f on Ω^ξ , $f(x) = f(u)$ if $x \sim u$ (relative to the Borel sets of real numbers). Especially, every real-valued measurable function on the trivial tolerance space Ω^∞ is a constant function.

A function $f : \Omega \rightarrow \Phi^\zeta$ from an arbitrary set Ω to a tolerance space Φ^ζ defines a tolerance relation ξ_ζ on Ω : $\xi_\zeta(x, u)$ for $x, u \in \Omega$ if $f(x)$ and $f(u)$ are ζ -similar in Φ^ζ . The function $f : \Omega^{\xi_\zeta} \rightarrow \Phi^\zeta$ is measurable. For example,

in a data pre-processing $f : \Omega \longrightarrow \Phi$, each similarity on the feature vectors defines a similarity on the original data set Ω .

We introduce a data clustering in a tolerance space Ω^ξ [7]. Each $x \in \Omega$ is a *representative* of its neighborhood $\xi(x)$. A set of elements $\{r_1, \dots, r_k\}$ is a *representative system* of the tolerance space Ω^ξ if the corresponding neighborhoods cover the whole space Ω . A representative system $\{r_1, \dots, r_k\}$ is *minimal* if there is no other representative system with less than k members. The concept of minimal representative system comes from Maak [8]. A representative system forms a *representative clustering* of the tolerance space, in which the clusters are the corresponding neighborhoods. Note that the clusterings in this paper are in general not partitions (e.g., Fig. 1). To search for a minimal representative clustering is intractable in general. A heuristic search has proposed [7], [9]. We use such a *sub-minimal* representative clustering to approximate a minimal representative clustering. In Fig. 1, for example, a minimal representative clustering is computed by the heuristic method.

3. Classifiers and Errors

Pattern recognition is predicting the unknown nature of an observation. Formally, we use x to denote an observation and Ω to denote the space of all possible observations. This paper considers only two classes (e.g., normal and anomaly), denoted by 0 and 1. In pattern recognition, one creates a classifier $g : \Omega \longrightarrow \{0, 1\}$ to represent one's prediction of y given x . The classifier errs on x when $g(x) \neq y$. In the following, we introduce a probabilistic model of pattern recognition on a tolerance space.

Definition 3.1 A *probability measure* on a tolerance space Ω^ξ is a probability measure μ on the measurable space $(\Omega^\xi, \mathcal{F}_\xi)$. The triple $(\Omega^\xi, \mathcal{F}_\xi, \mu)$ is called a probability space of Ω^ξ .

Given a probability space $(\Omega^\xi, \mathcal{F}_\xi, \mu)$, each measurable function $f : \Omega^\xi \longrightarrow \Phi^\zeta$ defines a probability measure μ_ζ on $(\Phi^\zeta, \mathcal{F}_\zeta)$ as follows. For any measurable set $A \in \mathcal{F}_\zeta$, $\mu_\zeta(A) = \mu(f^{-1}(A))$. The triple $(\Phi^\zeta, \mathcal{F}_\zeta, \mu_\zeta)$ is a probability space. For example, the function f is a feature selection and the probability measure of the original data is transformed to the feature vectors.

Let ξ be a tolerance relation on Ω and $(\Omega^\xi, \mathcal{F}_\xi, \mu)$ a probability space. If there is no ambiguity, we omit the ξ in our notation. Let (X, Y) be a random pair taking their respective values from Ω and $\{0, 1\}$. The random pair is defined by a pair (μ, η) , where μ is the probability measure of X and η is the posterior probability, given the value of X . That is, for any $A \in \mathcal{F}$,

$$P(X \in A) = \mu(A),$$

and for any $x \in \Omega$,

$$\eta(x) = P(Y = 1 | X \in \bar{x}).$$

Note that η is measurable (relative to \mathcal{F}) and $\eta(x) = \eta(y)$ if x and y are indistinguishable.

A *classifier* is a \mathcal{F} -measurable function $g : \Omega \longrightarrow \{0, 1\}$, where $\{0, 1\}$ is the discrete tolerance space. The probability of error $L(g) = P(g(X) \neq Y)$ is the *error rate* of g :

$$L(g) = P(g(X) \neq Y) = \int_{\Omega} P(g(X) \neq Y | X \in \bar{x}) d\mu(x).$$

The *false positive rate* $L^{(0)}(g)$ and the *false negative rate* $L^{(1)}(g)$ are

$$L^{(0)}(g) = P(g(X) = 1 | Y = 0),$$

and

$$L^{(1)}(g) = P(g(X) = 0 | Y = 1).$$

The relation among the tree types of error is

$$L(g) = L^{(0)}(g)P(Y = 0) + L^{(1)}(g)P(Y = 1).$$

Definition 3.2 Let $0 \leq \theta < 1$. The classifier g_θ is defined as follows

$$g_\theta(x) = \begin{cases} 1 & \text{if } \eta(x) > \theta, \\ 0 & \text{otherwise.} \end{cases}$$

Note that, if $\eta(x) \in \{0, 1\}$, then $L(g_\theta) = L^{(0)}(g_\theta) = L^{(1)}(g_\theta) = 0$ for $0 < \theta < 1$. The classifier $g_{\frac{1}{2}}$ is the *Bayes classifier* and $L(g_{\frac{1}{2}})$ is the *Bayes Error*. Similarly to the classical theory [1], we have following theorems.

Theorem 3.3 (Bayes) For any classifier g , $L(g_{\frac{1}{2}}) \leq L(g)$.

Proof: Let $A^* = \{x | g_{\frac{1}{2}}(x) = 1\}$ and $A = \{x | g(x) = 1\}$. For any given $x \in \Omega$, we have the conditional probabilities:

$$\begin{aligned} & P(g(X) \neq Y | X \in \bar{x}) \\ &= P(g(X) = 1, Y = 0 | X \in \bar{x}) + \\ & P(g(X) = 0, Y = 1 | X \in \bar{x}) \\ &= 1_A(x)P(Y = 0 | X \in \bar{x}) + \\ & (1 - 1_A(x))P(Y = 1 | X \in \bar{x}) \\ &= 1_A(x)(1 - \eta(x)) + (1 - 1_A(x))\eta(x) \\ &= 1_A(x)(1 - 2\eta(x)) + \eta(x), \end{aligned}$$

and, similarly,

$$P(g_{\frac{1}{2}}(X) \neq Y | X \in \bar{x}) = 1_{A^*}(x)(1 - 2\eta(x)) + \eta(x),$$

where 1_A and 1_{A^*} are the indicators of the sets A and A^* , respectively. Thus,

$$\begin{aligned} & P(g(X) \neq Y | X \in \bar{x}) - P(g_{\frac{1}{2}}(X) \neq Y | X \in \bar{x}) \\ &= (2\eta(x) - 1)(1_{A^*}(x) - 1_A(x)) \geq 0. \end{aligned}$$

Therefore, $L(g) - L(g_{\frac{1}{2}}) =$

$$\int_{\Omega} [P(g(X) \neq Y | X \in \bar{x}) - P(g_{\frac{1}{2}}(X) \neq Y | X \in \bar{x})] d\mu(x) \geq 0. \quad \blacksquare$$

Theorem 3.4 (Neyman-Pearson) For any classifier g and θ ($0 < \theta < 1$), if $L^{(0)}(g) \leq L^{(0)}(g_\theta)$, then $L^{(1)}(g) \geq L^{(1)}(g_\theta)$.

Proof: Assume that $L^{(0)}(g) \leq L^{(0)}(g_\theta)$, which is equivalent to $P(g(x) = 1, Y = 0) \leq P(g_\theta(x) = 1, Y = 0)$. Let $A = \{x|g(x) = 1\}$, $A_\theta = \{x|\eta(x) > \theta\}$, $W = A - A_\theta$, $Z = A_\theta - A$, and $Y = A \cap A_\theta$. Note that all of these sets are measurable, $\eta(x) > \theta$ on Z , and $\eta(x) \leq \theta$ on W . We have $P(g(X) = 1, Y = 0) = \int_\Omega P(g(X) = 1, Y = 0|X \in \bar{x})d\mu(x)$, and $P(g(X) = 1, Y = 0|X \in \bar{x}) = 1_A(x)(1 - \eta(x))$. Therefore,

$$P(g(X) = 1, Y = 0) = \mu(A) - \int_A \eta(x)d\mu(x).$$

Similarly,

$$P(g_\theta(X) = 1, Y = 0) = \mu(A_\theta) - \int_{A_\theta} \eta(x)d\mu(x).$$

The assumption implies that

$$\mu(A) - \int_A \eta(x)d\mu(x) \leq \mu(A_\theta) - \int_{A_\theta} \eta(x)d\mu(x).$$

Since $\int_W \eta(x)d\mu(x) \leq \theta\mu(W)$ and $\int_Z \eta(x)d\mu(x) > \theta\mu(Z)$, we have $\mu(W) \leq \mu(Z)$. Now show that $L^{(1)}(g) \geq L^{(1)}(g_\theta)$, which is equivalent to $P(g_\theta(x) = 0, Y = 1) \leq P(g(x) = 0, Y = 1)$. Since $P(g(X) = 0, Y = 1|X \in \bar{x}) = (1 - 1_A(x))\eta(x)$,

$$\begin{aligned} P(g(X) = 0, Y = 1) &= 1 - \int_A \eta(x)d\mu(x) \\ &= 1 - \int_W \eta(x)d\mu(x) - \int_Y \eta(x)d\mu(x). \end{aligned}$$

Similarly,

$$P(g_\theta(X) = 0, Y = 1) = 1 - \int_Z \eta(x)d\mu(x) - \int_Y \eta(x)d\mu(x).$$

Therefore,

$$\begin{aligned} P(g(X) = 0, Y = 1) - P(g_\theta(X) = 0, Y = 1) &= \int_Z \eta(x)d\mu(x) - \int_W \eta(x)d\mu(x) \\ &\geq \theta(\mu(Z) - \mu(W)) \geq 0. \end{aligned}$$

The Bayes classifier is an optimal classifier if we are required to minimize the error rate. If the false positive rate $L^{(0)}$ is required to be kept under a certain level, then g_θ minimizes the false negative rate $L^{(1)}$ for some θ .

We also consider classifiers relative the Borel subfield \mathcal{D} generated by a partition $\{A_i|i = 1, \dots, n\}$ of Ω , each $A_i \in \mathcal{F}$ and $P(A_i) > 0$. Let $\eta_{\mathcal{D}} = E(\eta|\mathcal{D})$, the conditional expectation of η relative to \mathcal{D} . For $x \in A_i$,

$$\eta_{\mathcal{D}}(x) = \frac{1}{P(A_i)} \int_{A_i} \eta(t)d\mu(t) = P(Y = 1|X \in A_i).$$

A \mathcal{D} -classifier is a \mathcal{D} -measurable function $g^{\mathcal{D}} : \Omega \rightarrow \{0, 1\}$. Note that $g^{\mathcal{D}}$ is constant on each A_i . Let $c_i^{\mathcal{D}} \in \{0, 1\}$ be the value of $g^{\mathcal{D}}$ on A_i . Then

$$g^{\mathcal{D}}(x) = \sum_i c_i^{\mathcal{D}} 1_{A_i}(x),$$

where 1_{A_i} is the indicator of A_i for each i .

Any \mathcal{D} -classifier is a \mathcal{F} -classifier, but not vice versa. For \mathcal{D} -classifiers, we have following similar definitions and properties. The \mathcal{D} -classifier $g_\theta^{\mathcal{D}} = 1$ iff $\eta^{\mathcal{D}}(x) > \theta$, $0 \leq \theta < 1$. The Bayes \mathcal{D} -classifier $g_{\frac{\theta}{2}}^{\mathcal{D}}$ minimizes the error rate of all \mathcal{D} -classifiers. The Neyman-Pearson Theorem is also true; that is, for any \mathcal{D} -classifier $g^{\mathcal{D}}$, if $L^{(0)}(g^{\mathcal{D}}) \leq L^{(0)}(g_\theta^{\mathcal{D}})$, then $L^{(1)}(g^{\mathcal{D}}) \geq L^{(1)}(g_\theta^{\mathcal{D}})$. Since a \mathcal{D} -classifier is also a classifier, $L(g_{\frac{\theta}{2}}^{\mathcal{D}}) \geq L(g_{\frac{1}{2}})$.

In this paper, such a Borel subfield is generated by a representative clustering and represents the heuristic information model of the corresponding heuristic search. The search will simplify the computation with the price of a possibly higher error rate.

4. Supervised Learning

4.1 Training Data Set

This section introduces classifiers based on a training data set $\{(X_i, Y_i), 1 \leq i \leq n\}$. We assume, as in naive Bayes, the data (i.e., $(X_1, Y_1), \dots, (X_n, Y_n)$) is a sequence of independent identically distributed (i.i.d.) random pairs with the same distribution as that of (X, Y) .

Each X_i in the training data set usually consists of several components, each of which may be numerical or categorical. We also assume that the class of X_i is binary; that is, $Y_i \in \{0, 1\}$. Let Ψ be the set of possible values of the variable X and Ω the subset consisting of all X_i 's: $\Omega = \{X_i|1 \leq i \leq n\} \subset \Psi$. We treat Ω as the discrete tolerance space Ω^0 . Let \mathcal{F} be the power set of Ω , the Borel field of the discrete tolerance space. Note that $|\Omega| \leq n$ because it is possible that $X_i = X_j$ for different i and j . For each $x \in \Omega$, consider two counts:

$$f_0(x) = |\{i | 1 \leq i \leq n, X_i = x, Y_i = 0\}|,$$

and

$$f_1(x) = |\{i | 1 \leq i \leq n, X_i = x, Y_i = 1\}|.$$

Note that $\sum_{x \in \Omega} (f_0(x) + f_1(x)) = n$. Let μ be the frequency measure on Ω . That is,

$$\mu(x) = P(X = x) = \frac{f_0(x) + f_1(x)}{n}.$$

The conditional probability of $Y = 1$, given $X = x$, is

$$\eta(x) = \frac{f_1(x)}{f_0(x) + f_1(x)}.$$

Given the training data set, consider a random pair (X, Y) on $(\Omega, \mathcal{F}, \mu)$. For a real θ , $0 \leq \theta < 1$, the classifier g_θ is defined: $g_\theta(x) = 1$ iff $\eta(x) > \theta$. The Bayes error is

$$L(g_{\frac{1}{2}}) = \frac{1}{n} \left(\sum_{f_0(x) < f_1(x)} f_0(x) + \sum_{f_0(x) \geq f_1(x)} f_1(x) \right).$$

Other errors can be computed similarly. Note that the error rates $L(g_\theta) = L^{(0)}(g_\theta) = L^{(1)}(g_\theta) = 0$ if $X_i \neq X_j$ for any $i \neq j$. The Bayes error $L(g_{\frac{1}{2}})$ is the theoretical limitation of the learning. If the Bayes error is too large, the performance of the learning result will be poor. In this case, larger data set and/or more components of information about X are needed.

4.2 Data Pre-processing

Formally, we use a function to represent a data pre-processing (e.g., [2], [10]): $\phi : \Psi \rightarrow \Phi$. We call each possible sample X a record and $\phi(X)$ the *feature vector* of X . All possible feature vectors form the set Φ . Let the image of Ω be $\Phi_m = \phi(\Omega) (\subset \Phi)$. Usually, the function ϕ is not one-to-one. For the frequency of Y_i , we store two numbers for each feature vector $z \in \Phi_m$:

$$n_0(z) = |\{i \mid 1 \leq i \leq n, \phi(X_i) = z, Y_i = 0\}|$$

and

$$n_1(z) = |\{i \mid 1 \leq i \leq n, \phi(X_i) = z, Y_i = 1\}|.$$

That is, $n_0(z)$ and $n_1(z)$ are the numbers of records of class 0 and class 1 in $\phi^{-1}(z) \subseteq \Omega$, respectively. Usually, the size of Φ_m is much smaller than that of Ω for computational purpose. Note that $\sum_{z \in \Phi_m} (n_0(z) + n_1(z)) = n$. The probability measure on Ω is transformed to Φ_m . For each $z \in \Phi_m$, $\mu(z) = \frac{n_0(z) + n_1(z)}{n}$.

Let the partition $\{\phi^{-1}(z), z \in \Phi_m\}$ of Ω generate the Borel subfield $\mathcal{D} \subseteq \mathcal{F}$. For each $x \in \phi^{-1}(z)$, consider the conditional probability $\eta_{\mathcal{D}}(x) = P(Y = 1 | X \in \phi^{-1}(z)) = \frac{n_1(z)}{n_0(z) + n_1(z)}$. Then $\eta_{\mathcal{D}} = E(\eta | \mathcal{D})$, the conditional expectation of η . Any classifier based on the feature vectors is a \mathcal{D} -classifier. We consider the \mathcal{D} -classifiers $g_\theta^{\mathcal{D}}$ for $0 \leq \theta < 1$. We have $L(g_{\frac{1}{2}}) \leq L(g_{\frac{1}{2}}^{\mathcal{D}})$. The data pre-processing simplifies the computation, but may increase the Bayes error.

4.3 Similarity and Representative Clustering

To define similarity depends on each individual task and there are many methods to measure similarities or dissimilarities (e.g., [2]). Here we simply assume a similarity ξ is given on Φ . To measure the suitability of ξ , we compute, given two random similar records X_i and X_j , the conditional probability $P(Y_i = Y_j | \xi(X_i, X_j))$, which is called the similarity *effect* in this paper. The effect should be as near to 1 as possible.

Consider the tolerance space $(\Phi_m)^\xi$ and let $\mathcal{R}_\xi = \{R_1, R_2, \dots, R_k\}$ be a sub-minimal representative clustering computed by the heuristic method mentioned in Section 2.

A record $x \in \Omega$ is called a *surprise* ([11], [12]) if x is not similar to any other record and there is only one i for which $X_i = x$. Therefore, the feature vector of a surprise is always in any choice of \mathcal{R}_ξ . Let the numbers of the surprise records of class 0 and class 1 be denoted by α_ξ and β_ξ , respectively, which are independent of the choice of \mathcal{R}_ξ . The similarity ξ does not provide much useful information about surprises and, therefore, a learning process should eliminate or reduce the number of surprises by adjusting feature vectors and similarities.

4.4 \mathcal{R}_ξ -Classifiers

We design classifiers of X based on \mathcal{R}_ξ . Let \mathcal{C} be the Borel subfield of \mathcal{F} generated by the clusters $\{\phi^{-1}(\xi(R_i))\}$, which is also a subfield of \mathcal{D} : $\mathcal{C} \subseteq \mathcal{D} \subseteq \mathcal{F}$. Each component of the partition in \mathcal{C} is represented by representatives. Let $x \in \Omega$. We search for the set of representatives (in \mathcal{R}_ξ) similar to the feature vector $\phi(x)$, denoted by S_x . The set of representatives not similar to $\phi(x)$ is the complement $N_x = \mathcal{R}_\xi - S_x$. Let $F_x = \bigcap_{R \in S_x} \xi(R) - \bigcup_{R \in N_x} \xi(R)$. Then the component of the partition containing x is the set $H(x) = \{t \mid \phi(t) \in F_x\}$. That is, $H(x)$ is the smallest \mathcal{C} -measurable set containing x .

Let the conditional expectation of η relative to \mathcal{C} be $\eta_{\mathcal{C}} = E(\eta | \mathcal{C}) = E(\eta_{\mathcal{D}} | \mathcal{C})$. Let $x \in \Omega$. In $H(x)$, the number of records of class 0 is $t_0 = \sum_{z \in F_x} n_0(z)$ and the number of records of class 1 is $t_1 = \sum_{z \in F_x} n_1(z)$. Then we have

$$\begin{aligned} \eta_{\mathcal{C}}(x) &= P(Y = 1 | X \in H(x)) \\ &= P(Y = 1 | \phi(X) \in F_x) \\ &= \frac{t_1}{t_0 + t_1}. \end{aligned}$$

The computation is on the feature vectors only.

The Borel field \mathcal{C} is completely determined by \mathcal{R}_ξ and we call a \mathcal{C} -classifier a \mathcal{R}_ξ -classifier. For each θ , $1 \leq \theta < 1$, consider the following \mathcal{R}_ξ -classifier:

$$g_\theta^{\mathcal{R}_\xi}(x) = \begin{cases} 1 & \text{if } \eta_{\mathcal{C}}(x) > \theta, \\ 0 & \text{if } \eta_{\mathcal{C}}(x) \leq \theta. \end{cases}$$

The Neyman-Pearson Theorem is also true for these classifiers and $L(g_{\frac{1}{2}}^{\mathcal{R}_\xi}) \geq L(g_{\frac{1}{2}}^{\mathcal{D}}) \geq L(g_{\frac{1}{2}})$. Computing the posterior probability $\eta_{\mathcal{C}}(x) = P(Y = 1 | X \in H(x))$ needs to compare the feature vector $\phi(x)$ with only k representatives. The computation is simplified.

This paper also uses the receiver operating characteristic (ROC) analysis to study the posterior probability $\eta_{\mathcal{C}}(x)$ by considering the corresponding ROC curve and computing the area under the curve (AUC). The AUC value should be as near to 1 as possible.

4.5 Testing Data Set

Let x be any given record. We use $\eta_{\mathcal{C}}(x) = \frac{t_1}{t_0 + t_1}$ to estimate the posterior probability, where t_0 and t_1 are defined above in the training data. There are two possibilities for

which $t_0 = t_1 = 0$. The first one is that $S_x \neq \emptyset$ but $F_x = \emptyset$. Since S_x still provides information about the record x , we replace F_x by $F_x = \bigcap_{R \in S_x} \xi(R)$. If it is still empty, then we use $F_x = \bigcup_{R \in S_x} \xi(R)$.

The other case is that $S_x = \emptyset$. That is, $\phi(x)$ is not similar to any representative in \mathcal{R}_ξ . We call such a record an *unknown surprise* (w.r.t. ξ). In this case, we use the information about the surprises in the training data set. If $\alpha_\xi + \beta_\xi > 0$, then the posterior probability is estimated by $\eta_C(x) = \frac{\beta_\xi}{\alpha_\xi + \beta_\xi}$. If there are no surprises in the training data set (i.e., $\alpha_\xi + \beta_\xi = 0$), then $\eta_C(x)$ is undefined. That is, the representative system \mathcal{R}_ξ does not provide any useful information about the unknown surprise.

Based on the estimated posterior probability $\eta_C(x)$, we extend the classifier g_θ^C to the testing data set:

$$g_\theta^{\mathcal{R}_\xi}(x) = \begin{cases} 1 & \text{if } \eta_C(x) > \theta, \\ 0 & \text{if } \eta_C(x) \leq \theta, \\ \text{unknown} & \text{if } \eta_C(x) \text{ is undefined.} \end{cases}$$

Consider the discrete tolerance space of records in the testing data set. The posterior probability $\eta_{test}(x)$ is unknown. Let \mathcal{C}_{test} be the Borel subfield generated by the neighborhoods of \mathcal{R}_ξ in the testing data set. Although each extended classifier $g_\theta^{\mathcal{R}_\xi}$ is \mathcal{C}_{test} -measurable, the extended $\eta_C(x)$ is not the actual posterior probability but an estimation. Therefore, the error rates on the testing data set are likely higher than that on the training data set.

If the training data set is large enough to contain typical records and the similarity ξ is well chosen so that there are no or very few surprises, the function η_C will be a good estimation of posterior probabilities on the testing data set and the errors of $g_\theta^{\mathcal{R}_\xi}$ can be well predicted by the errors on the training data set, as shown by the experiment in the next section.

5. Experiments

In our experiments, we use the KDD-99 cup data set [13], which is adapted from the data set of 1998 DARPA Intrusion Detection Evaluation Program [14]. We have loaded 494020 records in our database with 42 attributes, both categorical and continuous. Each record is labeled with normal or certain attack. The 42nd attribute is the label, of which there are 23 different types: one is normal and others are attacks. All attacks are classified as anomalies in this experiment. We use 0 to represent a normal and 1 to represent an anomaly. Our task is to classify records based on the 41 attributes into two classes: anomaly or normal.

5.1 Training Data Set and Pre-Processing

We observe that all *smurf* records and most *pod* records can be completely classified by only two attributes. We exclude those records first. Then using the traditional statistical test of the equality of two distributions, we have selected 9 attributes. In order to make the errors more visible in the

Table 1: Training Data Set

normal	29550
anomaly	7295
total	36845
Bayes error on D_{train}	0
feature vectors	1160
Bayes error on F_{train}	0.0091

experiment, we further exclude 7 feature vectors consisting of more than 2000 records of the same class. In total, we exclude 420262 records (about 85% of the total data set).

From the rest of the records, we choose randomly about half of normals and half of anomalies as the training data set, and the rest as the testing data set. Let the training data set be D_{train} and the testing data set D_{test} . The testing data set D_{test} consists of 29603 normals and 7310 anomalies, 36913 in total. Each record t is mapped into a feature vector $\phi(t)$ with 9 components. Let the set of the feature vectors $\phi(D_{train})$ be F_{train} . The numbers of records in D_{train} and F_{train} and the corresponding Bayes errors are shown in Table 1. Note that the pre-processing reduces the number of the records in D_{train} from 36845 to 1160 (3.1% of 36845) in $\phi(D_{train})$. It also reduces the number of features from 41 to 9. The Bayes error is raised from 0 to 0.0091.

5.2 Tolerance Relations and g_θ -Classifiers

On features vectors (of 9 components) we define a Hamming distance d , in which the absolute differences between numerical values are scaled down to the unit interval $[0, 1]$. For each $\varepsilon > 0$, we define a tolerance relation on the set of feature vectors: $d_\varepsilon(f, g)$ if $d(f, g) \leq \varepsilon$.

Consider the training process on the tolerance space $(F_{train})^{d_\varepsilon}$. Let \mathcal{R}_ε be a corresponding representative clustering. And let α_ε and β_ε be the numbers of surprising normals and anomalies, respectively. We study the \mathcal{R}_ε -classifier $g_\theta^{\mathcal{R}_\varepsilon}$. In the following, we introduce some results for different ε 's: 0.01, 0.25, 0.5, 2.0, 3.0, and 9.0. For $\varepsilon = 0.01$, $(F_{train})^{d_{0.01}}$ is the discrete space, in which all 1160 feature vectors are representatives and the clustering is a partition. For other cases, the clusterings are not partitions. The cluster numbers and computations are significantly reduced, but errors increase. The experiment result is summarized in Table 2 and Table 3 (B- is Bayes and u- is unknown). The result shows the trade-off between the computations and performances. Furthermore, the error functions L , $L^{(0)}$, and $L^{(1)}$ on D_{train} for $\varepsilon = 0.5$ are depicted in Fig. 2. The ROC curves of D_{train} are depicted in Fig. 3. For D_{test} we have very similar results.

If ε is sufficiently large (e.g. $\varepsilon \geq 9.0$), then all records are indistinguishable. That is, the only information about D_{train} is that there are 29550 normals and 7295 anomalies and the posterior probability $\eta(x) = 0.1980$ for any x . Therefore,

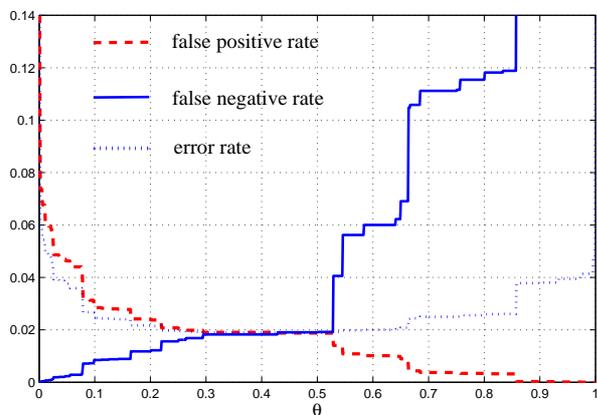
Table 2: Experiment Result for D_{train}

ε	$ \mathcal{R}_\varepsilon $	α	β	effect	AUC	B-error
0.01	1160	236	92	0.9865	0.9995	0.0091
0.25	437	30	34	0.9816	0.9991	0.0122
0.5	240	12	17	0.9732	0.9983	0.0188
3.0	9	0	0	0.7887	0.9671	0.0852
4.0	4	0	0	0.7114	0.8527	0.1769
9.0	1	0	0	0.6824	0.5000	0.1980

Table 3: Experiment Result for D_{test}

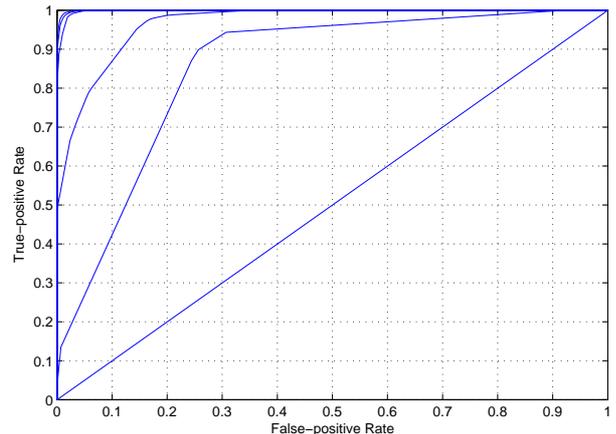
ε	u-surprises	effect	AUC	B-error
0.01	339	0.9851	0.9969	0.0108
0.25	238	0.9794	0.9963	0.0149
0.5	121	0.9709	0.9958	0.0213
3.0	1	0.7769	0.9640	0.0879
4.0	0	0.7110	0.8522	0.1767
9.0	0	0.6824	0.5000	0.1980

the Bayes rule classifies each records as a normal. The error rate is 0.1980, but no anomalies are classified at all, that is, the false negative rate is 1. The similarity effect can also be computed directly from the number of normals and anomalies. The AUC is 0.5, which means this similarity is worthless.

Fig. 2: Error functions on D_{train} for $\varepsilon = 0.5$.

6. Conclusion

This paper introduces a probabilistic model of pattern recognition on a tolerance space. The learning process includes searching for a similarity of the feature vectors

Fig. 3: ROC curves on D_{train} for $\varepsilon = 0.01, 0.25, 0.5, 3, 4, 9$.

and computing a representative clustering. The partition generated by the clustering provides the heuristic information model. Based on the posterior probability, both Bayes and Neyman-Pearson Theorems are true on the training data set. For a record in the testing data set, the set of representatives similar to the feature vector of the record is used to predict the the posterior probability. The experiment demonstrates the trade-off between computations and classification performances.

References

- [1] L. Devroye and G. L. Györfi, *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.
- [2] R. Hastie, T. Tibshirami and J. Friedman, *The Elements of Statistical Learning*. Springer, 2001.
- [3] M. A. E. Maloof, *Machine Learning and Data Mining for Computer Security*. Springer-Verlag, 2006.
- [4] C.-H. Tzeng, *A Theory of Heuristic Information in Game-Tree Search*. Springer-Verlag, 1988.
- [5] E. C. Zeeman, "The topology of the brain and visual perception," in *Topology of 3-Manifolds and related Topics*. Proc. The Univ. of Georgia Institute, 1962, pp. 240–256.
- [6] K. Jacobs, *Measure and Integral*. Academic Press, 1978.
- [7] C.-H. Tzeng and F.-S. Sun, "Data clustering in tolerance space," in *Berthold, Lenz, Bradley, Kruse, and Borgelt, editors, Advances in Intelligent Data Analysis V*. Springer-Verlag, 2003, pp. 297–306.
- [8] W. Maak, *Fastperiodische Funktionen*. Springer-Verlag, 1967.
- [9] C.-H. Tzeng, "Similarity and pattern recognition," in *Proc. Intern. Conf. on Data Mining and Appl.* ICDMA'08, March 2008.
- [10] J. Han and M. Kamber, *Data Mining Concepts and Techniques*. 2nd Edition, Morgan Kaufmann, 2006.
- [11] I. J. Good, *THE ESTIMATION OF PROBABILITIES, An Essay on Modern Bayesian Methods*. Cambridge, Mass.: Research Monograph No.30, MIT Press, 1965.
- [12] —, *The Foundations of Probability and Its Applications*. Minneapolis: University of Minnesota Press, 1983.
- [13] "Kdd-99 cup dataset." <http://kdd.ics.uci.edu/databases/kddcup99/>.
- [14] "Darpa 1998 network intrusion detection dataset." http://www.ll.mit.edu/IST/ideval/data/data_index.html.

On Adaboost and Optimal Betting Strategies

Pasquale Malacaria¹ and Fabrizio Smeraldi¹

¹School of Electronic Engineering and Computer Science, Queen Mary University of London, London, UK

Abstract—We explore the relation between the Adaboost weight update procedure and Kelly's theory of betting. Specifically, we show that Adaboost can be seen as an intuitive optimal betting strategy in the sense of Kelly. Technically this is achieved as the solution of the dual of the classical formulation of the Adaboost minimisation problem. Using this equivalence with betting strategies we derive a substantial simplification of Adaboost and of a related family of boosting algorithms. Besides allowing a natural derivation of the simplification, our approach establishes a connection between Adaboost and Information Theory that we believe may cast a new light on some boosting concepts.

Keywords: Ensembles methods, Statistical Methods, Regression/Classification, Adaboost, Information theory, Optimal betting strategies

1. Introduction

Suppose you want to bet on a series of races of six horses, on which no information is initially available (the same horses participate in all races). The only information available after each race is in the form of the names of a subset of the losers for that race. This implies that you are not informed about the changes to your capital after each race, so your next bet can only be expressed as a percentage of your (unknown) capital. Let us furthermore assume that you are required to bet all of your capital on each race.

Initially, as nothing is known, it seems reasonable to spread one's bets evenly, so that your initial bet will allocate 1/6-th of your capital on each horse.

Suppose now that you learn that horses 1,3, 4, 5 and 6 lost the first race. As your previous bet was optimal¹ (w.r.t. your information at the time) you want the new bet to be the "closest" possible to the previous bet, taking into account the outcome of the race. You could for instance decide to bet a fraction ρ of your capital on the latest "possible" winners (horse 1) and $1 - \rho$ on the latest losers (horses 1,3,4,5,6). This ρ is a measure of how much you believe the result of the last race to be a predictor of the result of the next race.

Notice that, as you have to invest all your capital at each time, $\rho = 1$ (i.e. betting everything on the set of horses

containing the previous winner) would be a bad choice: you would then lose all your capital if the following race is won by any other horse. Table 1 presents an algorithm implementing these ideas: bet $t + 1$ on horse i is obtained from the previous bet t by multiplying it by $\frac{\rho}{\epsilon_t}$ (if the horse was a possible winner) and by $\frac{1-\rho}{(1-\epsilon_t)}$ (if it was a loser); ϵ_t is the total amount bet on the latest possible winners in the previous race t .

If you need to choose a value of ρ before all races, as no information is available, you may well opt for $\rho = 0.5$. This is a "non-committal" choice: you believe that the subset of losers of the each race revealed to you each time will not be a very good predictor for the result of the next race. In our example, where 1,3,4,5 and 6 lost the first race, choosing $\rho = 0.5$ will lead to the bets (0.1, 0.5, 0.1, 0.1, 0.1, 0.1) on the second race. Table 2 shows the bets generated by applying the betting algorithm with $\rho = 0.5$ to a series of 4 races.

In Section 4 we show that the Adaboost weight update cycle is equivalent to this betting strategy with $\rho = 0.5$, modulo the trivial identification of the horses with training examples and of the possible winners with misclassified training data: this equivalence holds under an assumption of *minimal luck*, assumption which we will make more precise in section 5. As will be shown, this derives straightforwardly from a direct solution of the *dual* of the Adaboost minimisation problem. This result, that is the main contribution of our work, establishes the relation between Kelly's theory of optimal betting and Adaboost sketched in Figure 1. It also leads directly to the simplified formulation of Adaboost shown in Table 4.

The rest of this paper is organised as follows: Section 2 gives a brief introduction to the Adaboost algorithm; in Section 3 we outline the theory of optimal betting underlying the strategy displayed in Table 1. The equivalence between this betting strategy and Adaboost is proved in Section 4. In Section 5 we proceed in the opposite direction, and show how convergence theorems for Adaboost can be used to investigate the asymptotic behaviour of our betting strategy. Finally in Section 6 we generalize the correspondence between betting strategies and boosting and show that in the case of arbitrary choice of ρ we recover a known variant of Adaboost [1].

2. Background on Adaboost

Boosting concerns itself with the problem of combining several prediction rules, with poor individual performance,

¹Optimal betting is here understood in the information theoretical sense. In probability terms betting all capital on the horse most likely to win will give the highest expected return, but will make you bankrupt with probability 1 in the long term. Maximizing the return has to be balanced with the risk of bankruptcy; see Section 3.

Table 1: The betting algorithm

Betting Algorithm, ρ is the fraction of the capital allocated to the latest possible winners:

Choose the initial bets $b_{1,i} = \frac{1}{m}$, where m is the number of horses;

For race number t :

- 1) Let W be the set of possible winners in the previous race. Compute $\epsilon_t = \sum_{i \in W} b_{t,i}$;
- 2) Update the bets for race $t + 1$: $b_{t+1,i} = \frac{\rho}{\epsilon_t} b_{t,i}$ if $i \in W$, $b_{t+1,i} = \frac{1-\rho}{1-\epsilon_t} b_{t,i}$ otherwise.

into a highly accurate predictor. This is commonly referred to as combining a set of “weak” learners into a “strong” classifier. Adaboost, introduced by Yoav Freund and Robert Schapire [2], differs from previous algorithms in that it does not require previous knowledge of the performance of the weak hypotheses, but it rather adapts accordingly. This adaptation is achieved by maintaining a distribution of weights over the elements of the training set.

Adaboost is structured as an iterative algorithm, that works by maintaining a distribution of weights over the training examples. At each iteration a new weak learner (classifier) is trained, and the distribution of weights is updated according to the examples it misclassifies. The idea is to increase the importance of the training examples that are “difficult” to classify. The weights also determine the contribution of the each weak learner to the final strong classifier.

In the following, we will indicate the m training examples as $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$, where the \mathbf{x}_i are vectors in a feature space X and the $y_i \in \{-1, +1\}$ are the respective class labels. Given a family of weak learners $\mathcal{H} = \{h_t : X \rightarrow \{-1, +1\}\}$, it is convenient to define $\mathbf{u}_{t,i} = y_i h_t(\mathbf{x}_i)$, so that $\mathbf{u}_{t,i}$ is $+1$ if h_t classifies \mathbf{x}_i correctly, and -1 otherwise. If we now let $d_{t,i}$ be the distribution of weights at time t , with $\sum_i d_{t,i} = 1$, the cycle of iterations can be described as in Table 3.

Each iteration of the Adaboost algorithm solves the following minimisation problem [3], [4]:

$$\alpha_t = \arg \min_{\alpha} \log Z_t(\alpha), \quad (1)$$

where

$$Z_t(\alpha) = \sum_{i=1}^m d_{t,i} \exp(-\alpha u_{t,i}) \quad (2)$$

is the partition function that appears in point 3 of Table 3. In other words, the choice of the updated weights $d_{t+1,i}$ is such that α_t will minimise Z_t .

3. The gambler’s problem: optimal betting strategies

Before investigating the formal relation between Adaboost and betting strategies, we review in this Section the basics of the theory of optimal betting, that motivated the heuristic reasoning of Section 1. Our presentation is a summary of the ideas in Chapter 6.1 of [5], based on Kelly’s theory of gambling [6].

Let us consider the problem of determining the optimal betting strategy for a race run by m horses x_1, \dots, x_m . For the i -th horse, let p_i, o_i represents respectively the probability of x_i winning and the odds, intended as o_i -for-one (e.g., at 3-for-1 the gambler will get 3 times his bet in case of victory).

Assuming that the gambler always bets all his wealth, let b_i be the fraction that is bet on horse x_i . Then if horse X wins the race, the gambler’s capital grows by the wealth relative $S(X) = b(X)o(X)$. Assuming all races are independent, the gambler’s wealth after n races is therefore $S_n = \prod_{j=1}^n S(X_j)$.

It is at this point convenient to introduce the *doubling rate*

$$W(\mathbf{b}, \mathbf{p}) = E(\log S(X)) = \sum_{i=1}^m p_i \log(b_i o_i). \quad (3)$$

(in this information theoretical context, the logarithm should be understood to be in base 2). From the weak law of large numbers and the definition of S_n , we have that

$$\frac{1}{n} \log S_n = \frac{1}{n} \sum_{j=1}^n \log S(X_j) \rightarrow E(\log S(X)) \quad (4)$$

in probability, which allows us to express the gambler’s wealth in terms of the doubling rate:

$$S_n = 2^{n \frac{1}{n} \log S_n} \doteq 2^{n E(\log S(X))} = 2^{n W(\mathbf{p}, \mathbf{b})} \quad (5)$$

where by “ \doteq ” we indicate equality to the first order in the exponent.

As can be shown the optimal betting scheme, i.e. the choice of the b_i that maximises $W(\mathbf{p}, \mathbf{b})$, is given by

Table 2: Bets b_i generated using the betting algorithm (Table 1) with $\rho = 1/2$ by information O_i over four races (w denotes a possible winner).

	x_1	x_2	x_3	x_4	x_5	x_6
b_0	0.1666	0.1666	0.1666	0.1666	0.1666	0.1666
O_0	ℓ	w	ℓ	ℓ	ℓ	ℓ
b_1	0.1	0.5	0.1	0.1	0.1	0.1
O_1	w	ℓ	ℓ	ℓ	w	ℓ
b_2	0.25	0.3125	0.0625	0.0625	0.25	0.0625
O_2	ℓ	ℓ	w	w	ℓ	ℓ
b_3	0.1428	0.1785	0.25	0.25	0.1428	0.0357
O_3	ℓ	w	ℓ	ℓ	ℓ	ℓ
b_4	0.0869	0.5	0.1521	0.1521	0.0869	0.0217

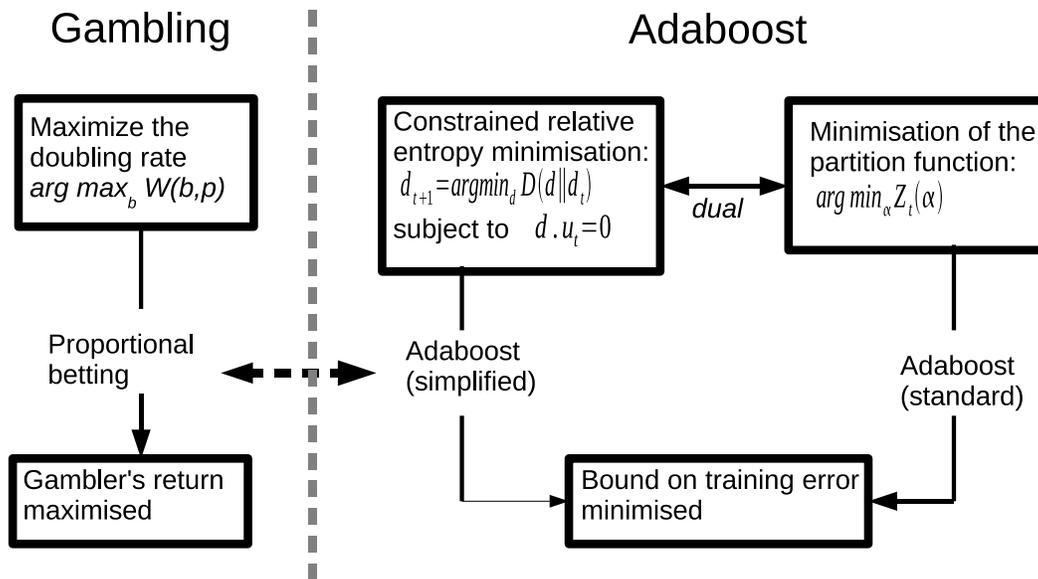


Figure 1: Relation between Kelly's theory of gambling and the standard and dual formulations of Adaboost.

Table 3: The standard formulation of the Adaboost algorithm

The Adaboost algorithm:

Initialise $d_{1,i} = \frac{1}{m}$.

For $t = 1, \dots, T$:

- 1) Select the weak learner h_t that minimises the training error ϵ_t , weighted by current distribution of weights $d_{t,i}$:
 $\epsilon_t = \sum_{i|u_{t,i}=-1} d_{t,i}$. Check that $\epsilon_t < 0.5$.
- 2) Set $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
- 3) Update the weights according to $d_{t+1,i} = \frac{1}{Z_t(\alpha_t)} d_{t,i} \exp(-\alpha_t u_{t,i})$, where $Z_t(\alpha_t) = \sum_{i=1}^m d_{t,i} \exp(-\alpha_t u_{t,i})$ is a normalisation factor which ensures that the $d_{t+1,i}$ will be a distribution.

Output the final classifier:

$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

$\mathbf{b} = \mathbf{p}$, that is to say *proportional betting*. Note that this is independent of the odds o_i .

Incidentally, it is crucial to this concept of optimality that (4) is true with probability one in the limit of large n . Consider for example the case of 3 horses that win with probability $1/2$, $1/4$ and $1/4$ respectively and that are given at the same odds (say 3). In this case, it would be tempting to bet all our capital on the 1st horse, as this would lead to the highest possible expected gain of $3 \times (1/2) = 3/2$. However, it is easy to see that such a strategy would eventually make us bankrupt with probability one, as the probability that horse 1 wins the first n races is vanishingly small for large n , so that the expected value is not a useful criterion to guide our betting. Equation 4, instead, assures us that the capital will asymptotically grow like 2^{nW} . For the proportional betting strategy, in this case, $W = (1/2) \log(3 \times 1/2) + (1/4) \log(3 \times$

$1/4) + (1/4) \log(3 \times 1/6) \approx 0.08$, so that the capital will grow to infinity with probability one [5].

In the special case of fair odds with $o_i = \frac{1}{p_i}$ (fair odds are defined as odds such that $\sum 1/o_i = 1$), the doubling rate becomes

$$W(\mathbf{p}, \mathbf{b}) = \sum p_i \log \left(\frac{b_i}{p_i} \right) = -D(\mathbf{p}||\mathbf{b}) / \log_e 2 \quad (6)$$

where $D(\mathbf{p}||\mathbf{b})$ is the Kullback-Leibler divergence (also known as the relative entropy) of the probability of victory \mathbf{p} and the betting strategy \mathbf{b} .

Note that, since the odds are fair, the gambler can at best conserve his wealth, that is $\max_{\mathbf{b}} W(\mathbf{p}, \mathbf{b}) = 0$. In this case, it follows directly from the properties of the Kullback-Leibler divergence that the maximum occurs for $b_i = p_i$, i.e. that the optimal strategy is proportional betting.

In the following, for convenience, we will refer to Equation 6 as if the base of the logarithm were e instead than 2. This modification is inessential, and for the sake of convenience we will continue to refer to the new quantity as the “doubling rate”.

4. Equivalence of Adaboost and betting strategy

We are now in a position to prove the equivalence of the betting algorithm introduced in Section 1 with the Adaboost weight update strategy, points 2) and 3) in Table 3. Modulo the identification of the horses with training examples and of the possible winners of each race with misclassified examples, this leads to the simplification of Adaboost shown in Table 4. Figure 1 illustrates schematically the relationships between these algorithms.

Our proof is based on the duality relations discussed in a number of excellent papers that cast Adaboost in terms of the minimisation of Kullback-Leibler or Bregman divergences [7], [8], [3]. In these, the Adaboost weight update equations are shown to solve the following problem:

$$\mathbf{d}_{t+1} = \arg \min_{\mathbf{d}} D(\mathbf{d}||\mathbf{d}_t) \quad \text{subject to } \mathbf{d} \cdot \mathbf{u}_t = 0. \quad (7)$$

In information theoretical terms, this can be interpreted as the Minimum Discrimination Information principle applied to the estimation of the distribution \mathbf{d}_{t+1} given the initial estimate \mathbf{d}_t and the linear constraint determined by the statistics \mathbf{u}_t [9].

The key observation is now that the Kullback-Leibler divergence of the d_i and the $d_{t,i}$,

$$D(\mathbf{d}||\mathbf{d}_t) = \sum d_i \log \frac{d_i}{d_{t,i}}, \quad (8)$$

bears a strong formal analogy to Equation 6 above.

For the sake of readability, we discard the index t and simply write d_i for $d_{t,i}$ and d_i^* for $d_{t+1,i}$. The Adaboost dual problem Equation 7 then becomes

$$\mathbf{d}^* = \arg \min_{\mathbf{d}} D(\tilde{\mathbf{d}}||\mathbf{d}) \quad \text{subject to } \tilde{\mathbf{d}} \cdot \mathbf{u} = 0. \quad (9)$$

The constraint $\mathbf{d}^* \cdot \mathbf{u} = 0$ can be rewritten as

$$\sum_{i|u_i=-1} d_i^* = \sum_{i|u_i=+1} d_i^* = 1/2. \quad (10)$$

With these notations, we are ready to prove the following

Theorem 1: The simplified weight update strategy in table 4 leads to the same weights as the original algorithm in Table 3.

Proof: we need to show that the weight update equations in table 4, that are the formal transcription of the betting strategy in Table 2, actually solve the optimisation problem Equation 9.

The constraint Equation 10 allows us to split the objective function in Equation 9 into the sum of two terms:

$$\begin{aligned} D(\mathbf{d}^*||\mathbf{d}) &= \\ &= \sum_{i|u_i=-1} d_i^* \log \frac{d_i^*}{d_i} + \sum_{i|u_i=+1} d_i^* \log \frac{d_i^*}{d_i} \quad (11) \\ &= D_-(\mathbf{d}^*, \mathbf{d}) + D_+(\mathbf{d}^*, \mathbf{d}), \end{aligned}$$

each of which can be maximised separately.

Notice that neither of these terms taken separately represents a Kullback-Leibler divergence, as the partial sums over the d_i^* and d_i are not over distributions. However, this is easily remedied by normalising these sub-sequences separately. To this purpose, we introduce the Adaboost weighted error ϵ_t (see Table 3, point 1), that omitting the t index is equal to $\epsilon = \sum_{i|u_i=-1} d_i$. The first term of the summation can then be rewritten as

$$\begin{aligned} D_-(\mathbf{d}^*, \mathbf{d}) &= \\ &= \frac{1}{2} \sum_{i|u_i=-1} 2d_i^* \log \frac{2d_i^*}{d_i/\epsilon} - \frac{1}{2} \log 2\epsilon \quad (12) \\ &= \frac{1}{2} D_-(2\mathbf{d}^*||\mathbf{d}/\epsilon) - \frac{1}{2} \log 2\epsilon, \end{aligned}$$

where by D_- we mean the Kullback-Leibler divergence computed over the subset of indexes $\{i|u_i=-1\}$. The minimum point is now immediately evident: $2d_i^* = d_i/\epsilon$ or, in the notation of Table 4,

$$d_{t+1,i} = \frac{1}{2\epsilon_t} d_{t,i} \quad \text{if } u_{t,i} = -1. \quad (13)$$

Similarly, the minimization of D_+ leads to $2d_i^* = d_i/(1-\epsilon)$, or

$$d_{t+1,i} = \frac{1}{2(1-\epsilon_t)} d_{t,i} \quad \text{if } u_{t,i} = +1. \quad (14)$$

These are precisely the update equations given in Table 4, which proves the theorem.

We have thus shown that the simplified algorithm in Table 4 is completely equivalent to the original Adaboost algorithm in Table 3. Indeed, the two algorithms generate step by step the same sequence of weights, as well as the same decision function.

A similar simplification of Adaboost, although not so widely used, is already known to researchers in the field (see for instance [10]). However, to the best of our knowledge it has always been derived in a purely computational way by algebraic manipulation of the equations in Table 3. We believe that our derivation as a direct solution of the dual problem Equation 7 is both more straightforward and more illuminating, in that it brings out the direct correspondence with betting strategies represented in Figure 1. Finally, we note that a formally analogous weight update strategy has been employed in online *variants* of the algorithm [11], [12].

Table 4: Applying the proportional betting strategy to the solution of the Adaboost dual problem leads to a simpler formulation of the algorithm, equivalent to the original

<p>Adaboost (simplified):</p> <p>Initialise $d_{1,i} = \frac{1}{m}$. For $t = 1, \dots, T$:</p> <ol style="list-style-type: none"> 1) Select the weak learner h_t that minimises the training error ϵ_t, weighted by current distribution of weights $d_{t,i}$: $\epsilon_t = \sum_{i u_{t,i}=-1} d_{t,i}$. Check that $\epsilon_t < 0.5$. 2) Update the weights: $d_{t+1,i} = \frac{1}{2\epsilon_t} d_{t,i}$ if $u_{t,i} = -1$, $d_{t+1,i} = \frac{1}{2(1-\epsilon_t)} d_{t,i}$ otherwise. <p>Output the final classifier: $H(\mathbf{x}) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(\mathbf{x}) \right)$, with $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$</p>
--

5. Adaboost as a betting strategy

The considerations above suggest an interpretation of the betting algorithm in Table 1 as an iterative density estimation procedure in which an initial estimate, represented by the current bets \mathbf{b} , is refined using the information on the previous race. The updated estimate is then used to place new bets (in the notations of Table 1, $b_{t+1,i} = p_i$), in accordance to the proportional betting strategy.

This interpretation is supported by convergence results for the Adaboost weight update cycle, in either its standard form (Table 3) or its simplified form (Table 4). The application to our betting scenario with partial information is straightforward, modulo the the simple formal identifications $b_i = d_{t,i}$ and $p_i = d_{t+1,i}$.

To prove this, we will first state the following convergence theorem, which has been proved in [7], [8], [3]:

Theorem 2: If the system of the linear constraints has non-trivial solutions, i.e. there is some \mathbf{d} such that $\mathbf{d} \cdot \mathbf{u}_t = 0 \forall t$, then the Adaboost weight distribution converges to the distribution \mathbf{d}^* satisfying all the linear constraints for which $D(\mathbf{d}^* \parallel \mathbf{1}/m)$ is minimal (here by $\mathbf{1}$ we indicate the vector of all ones).

In other words, the weight update algorithm converges to a probability density that is, among all those that satisfy all the constraints, the closest to the original guess of a uniform distribution. This ‘‘approximation’’ property is best expressed by the following ‘‘Pythagorean theorem’’ for Kullback-Leibler divergences, that was derived in [13] in the context of an application to iterative density estimation (a generalisation to Bregman divergences can be found in [8]). For all solutions \mathbf{d} of the system of linear constraints, we have

$$D(\mathbf{d} \parallel \mathbf{1}/m) = D(\mathbf{d} \parallel \mathbf{d}^*) + D(\mathbf{d}^* \parallel \mathbf{1}/m). \quad (15)$$

In terms of our gambling problem, theorem 2 above means that the betting algorithm in Table 1 converges to an estimate $\mathbf{p}^* = \mathbf{b}^*$ of the probability of winning of each single horse. Among all the densities that satisfy the constraints imposed by the information on the previous races, \mathbf{p}^* is the one that

would maximise the doubling rate of the initial uniform bet, $W(\mathbf{p}^*, \mathbf{1}/m)$ (strictly speaking, theorem 2 proves this result only for the case that $\rho = 1/2$; however, generalisation to the case of a generic ρ is straightforward, see [7]).

However, for convergence to occur the information \mathbf{u}_t on the various races must be provided in the order determined by Point 1) in either Table 3 or Table 4, corresponding to the Adaboost requirement that the weak learner should minimise the error with respect to the weights. In the betting case, of course, this appears problematic as the gambler has no influence on the order in which the results occur. The corresponding assumption corresponding to the minimal error requirement would be, in a sense, one of ‘‘minimal luck’’: that is, once we decide on a betting strategy, the next race is the one for which the set of the possible winners has, according to our estimate, the lowest total probability.

When the conditions for convergence are met, the ‘‘Pythagorean’’ equality Equation 15 quantifies the advantage of the estimated probability \mathbf{p}^* over any other density \mathbf{p} compatible with the constraint, if the initial uniform distribution (or indeed any other density obtained in an intermediate iteration) are used for betting:

$$W(\mathbf{p}, \mathbf{1}/m) = W(\mathbf{p}^*, \mathbf{1}/m) - D(\mathbf{p} \parallel \mathbf{p}^*). \quad (16)$$

6. Extension of the simplified approach to Adaboost- ρ

In the following, we generalize the simplified approach outlined in Table 4. In terms of the betting strategy described in the introduction we are now considering a gambler using a value of ρ different from $1/2$. For simplicity we will consider $0 < \rho \leq 1/2$; it is an easy observation that the case $1 > \rho \geq 1/2$ is obtained by symmetry. This generalised constraint produces an algorithm known as Adaboost- ρ , that has been introduced in [1]. This variant of Adaboost has an intuitive geometric and statistical interpretation in terms of handling a degree of correlation between the new weights and the output of the weak learner. It also is an ideal testbed to show how our simplified approach can lead to

straightforward implementations of generalised versions of Adaboost. The question of whether an extension of our approach to the entire class of algorithms described in [7] is feasible is left for future work.

Similarly to what has been done for Adaboost, we directly solve the generalised dual optimisation problem

$$\mathbf{d}_{t+1} = \arg \min_{\mathbf{d}} D(\mathbf{d}^* || \mathbf{d}_t) \quad \text{subject to } \mathbf{d}^* \cdot \mathbf{u}_t = 1 - 2\rho. \quad (17)$$

where $0 < \rho \leq 1/2$. The solution to this problem will provide the simplified weight update rules (that this is indeed the dual of the Adaboost- ρ problem will be clarified in Section 6.1).

Proceeding as in Section 4, we simplify the notation by omitting the index t in $d_{t,i}$. Equation (17) implies the constraints

$$\sum_{i|u_i=-1} d_i^* = \rho, \quad \sum_{i|u_i=+1} d_i^* = 1 - \rho, \quad (18)$$

The objective function Equation 17 can again be decomposed as done in Equation 11:

$$D(\mathbf{d}^*, \mathbf{d} | \rho) = D_-(\mathbf{d}^*, \mathbf{d} | \rho) + D_+(\mathbf{d}^*, \mathbf{d} | \rho), \quad (19)$$

where D_- and D_+ account for the misclassified and the correctly classified training examples respectively.

Writing out the right hand side explicitly yields

$$\begin{aligned} D_-(\mathbf{d}^*, \mathbf{d} | \rho) &= \rho \sum_{i|u_i=-1} \frac{d_i^*}{\rho} \log \frac{d_i^*/\rho}{d_i/\epsilon} - \rho \log \frac{\epsilon}{\rho} = \\ &= \rho D_- \left(\frac{\mathbf{d}^*}{\rho} \parallel \frac{\mathbf{d}}{\epsilon} \right) - \rho \log \frac{\epsilon}{\rho} \end{aligned}$$

and similarly

$$D_+(\mathbf{d}^*, \mathbf{d} | \rho) = (1-\rho) D_+ \left(\frac{\mathbf{d}^*}{1-\rho} \parallel \frac{\mathbf{d}}{1-\epsilon} \right) - (1-\rho) \log \frac{1-\epsilon}{1-\rho}$$

where again D_- and D_+ stand for the Kullback-Leibler divergences calculated over the subsets of indexes $\{i|u_i = \mp 1\}$. It follows straightforwardly that the minimum is achieved for $d_i^* = \frac{\rho}{\epsilon} d_i$ if $u_i = -1$, and $d_i^* = \frac{1-\rho}{1-\epsilon} d_i$ if $u_i = +1$. These effectively are the weight update rules for the generalised algorithm, as shown in Table 5.

6.1 Standard solution of the Adaboost- ρ problem

For comparison, we derive here the standard version of the Adaboost- ρ algorithm in the notations used in this paper (apart for trivial changes of notation, this is the algorithm originally published in [1]).

We first need to obtain the objective function for the minimisation problem associated to Adaboost- ρ , of which Equation 17 is the dual. In other words, we need to determine

a suitable function $Z_{t|\rho}(\alpha)$ which generalises the function in Equation 2, that is to say,

$$\begin{aligned} - \min_{\mathbf{d} | \mathbf{d} \cdot \mathbf{u}_t = 0} D(\mathbf{d} || \mathbf{d}_t) &= \rho \log \frac{\epsilon}{\rho} + (1 - \rho) \log \frac{1 - \epsilon}{1 - \rho} \\ &= \min_{\alpha} \log Z_{t|\rho}(\alpha). \end{aligned} \quad (20)$$

Following the construction outlined in [7] and essentially setting $M_{ij} = u_{j,i} + 2\rho - 1$ in the Lagrange transform, we find

$$Z_{t|\rho}(\alpha) = \sum_{i=1}^m d_{t,i} \exp(-\alpha(u_{t,i} + 2\rho - 1)), \quad (22)$$

that is minimised by

$$\alpha_{t|\rho} = \frac{1}{2} \log \left(\frac{\rho}{1 - \rho} \frac{1 - \epsilon_t}{\epsilon_t} \right). \quad (23)$$

By substituting these values in the equation below it is easy to verify directly that

$$d_{t+1,i} = \frac{\exp(-\alpha_{t|\rho}(u_{t,i} + 2\rho - 1))}{Z_{t|\rho}(\alpha_{t|\rho})} d_{t,i} \quad (24)$$

yields the same weight update rules as given in Table 5, and is indeed the direct generalisation of the update rules in the standard formulation of Adaboost given in Table 3.

The complete standard Adaboost- ρ algorithm is shown in Table 6. Again, we note how the dual solution we proposed in Table 5 presents a considerable simplification of the standard formulation of the algorithm.

7. Conclusions

We discussed the relation between Adaboost and Kelly's theory of gambling. Specifically, we showed how the dual formulation of Adaboost formally corresponds to maximising the doubling rate for a gambler's capital, in a partial information situation. The applicable modification of the optimal proportional betting strategy therefore maps to a direct, intuitive solution of the dual of the Adaboost problem. This leads naturally to a substantial formal simplification of the Adaboost weight update cycle. Previous derivation of similar simplifications are, to the best of our knowledge, purely computational, and therefore, besides being more cumbersome, lack any meaningful interpretation.

The implications of the correspondence between Adaboost and gambling theory works in both directions; as we have shown, it is for instance possible to use convergence results obtained for Adaboost for investigating the asymptotic behaviour of the betting strategy we introduced in Table 1.

We believe that a further investigation of the relation between Adaboost and Kelly's theory, or indeed information theory in general, may lead to a deeper insight into some boosting concepts, and possibly a cross-fertilization between these two domains.

Adaboost- ρ simplified:

Initialise $d_{1,i} = \frac{1}{m}$.

For $t = 1, \dots, T$:

- 1) Select the weak learner h_t that minimises the training error ϵ_t , weighted by current distribution of weights $d_{t,i}$:
 $\epsilon_t = \sum_{i|u_{t,i}=-1} d_{t,i}$. Check that $\epsilon_t < \rho$.
- 2) Update the weights: $d_{t+1,i} = \frac{\rho}{\epsilon_t} d_{t,i}$ if $u_{t,i} = -1$, $d_{t+1,i} = \frac{1-\rho}{1-\epsilon_t} d_{t,i}$ otherwise.

Output the final classifier:

$$H_\rho(\mathbf{x}) = \text{sign} \left(\sum_{t=1}^T \alpha_{t|\rho} h_t(\mathbf{x}) \right), \text{ with } \alpha_{t|\rho} = \frac{1}{2} \log \left(\frac{\rho}{1-\rho} \frac{1-\epsilon_t}{\epsilon_t} \right)$$

Table 5: Adaboost- ρ generalises the Adaboost weight update constraint to the case $\mathbf{d} \cdot \mathbf{u} = 1 - 2\rho$, with $0 < \rho \leq 1/2$

The Adaboost- ρ algorithm:

Initialise $d_{1,i} = \frac{1}{m}$.

For $t = 1, \dots, T$:

- 1) Select the weak learner h_t that minimises the training error ϵ_t , weighted by current distribution of weights $d_{t,i}$:
 $\epsilon_t = \sum_{i|u_{t,i}=-1} d_{t,i}$. Check that $\epsilon_t < \rho$.
- 2) Set $\alpha_{t|\rho} = \frac{1}{2} \log \left(\frac{\rho}{1-\rho} \frac{1-\epsilon_t}{\epsilon_t} \right)$.
- 3) Update the weights according to $d_{t+1,i} = \frac{\exp(-\alpha_{t|\rho}(u_{t,i}+2\rho-1))}{Z_{t|\rho}(\alpha_{t|\rho})} d_{t,i}$,
 where $Z_{t|\rho}(\alpha_{t|\rho}) = \sum_{i=1}^m d_{t,i} \exp(-\alpha_{t|\rho}(u_{t,i}+2\rho-1))$, is a normalisation factor which ensures that the $d_{t+1,i}$ will be a distribution.

Output the final classifier:

$$H_\rho(x) = \text{sign} \left(\sum_{t=1}^T \alpha_{t|\rho} h_t(x) \right)$$

Table 6: The standard formulation of the Adaboost- ρ algorithm

References

- [1] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft margins for adaboost," *Machine Learning*, vol. 42, no. 3, pp. 287–320, 2001. [Online]. Available: citeseer.ist.psu.edu/657521.html
- [2] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Science*, vol. 55, no. 1, 1997.
- [3] M. W. J. Kivinen, "Boosting as entropy projection," in *Proceedings of COLT'99*. ACM, 1999.
- [4] R. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine Learning*, vol. 37, no. 3, 1999.
- [5] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley Interscience, 1991.
- [6] J. J. L. Kelly, "A new interpretation of information rate," *Bell System Technical Journal*, vol. 35, pp. 917–926, July 1956.
- [7] M. Collins, R. E. Schapire, and Y. Singer, "Logistic regression, adaboost and bregman distances," in *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, 2000.
- [8] S. D. Pietra, V. D. Pietra, and H. Lafferty, "Inducing features of random field," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, April 1997.
- [9] S. Kullback, *Information Theory and Statistics*. Wiley, 1959.
- [10] C. Rudin, R. E. Schapire, and I. Daubechies, "On the dynamics of boosting," in *Advances in Neural Information Processing Systems*, vol. 16, 2004.
- [11] N. C. Oza and S. Russell, "Online bagging and boosting," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, 2005, pp. 2340–2345.
- [12] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proceedings of CVPR*, vol. 1, 2006, pp. 260–267.
- [13] S. Kullback, "Probability densities with given marginals," *The Annals of Mathematical Statistics*, vol. 39, no. 4, 1968.

An experimental study on a new ensemble method using robust and order statistics

Faisal Zaman¹, Hideo Hirose²

¹e-mail: zaman@ume98.ces.kyutech.ac.jp, ²e-mail: hirose@ces.kyutech.ac.jp

^{1,2}Department of Information Design and Informatics,
Kyushu Institute of Technology, Iizuka, Fukuoka 820-8502, Japan

Abstract—*In this paper we have proposed a new (robust) ensemble creation method based on subsampling and validation step. Subsampling is used to generate replicates of the training data. To make the base classifier learn adequate information during the training, we used 75% of the training data for each subsample. The validation is based on a validation set (an independent bootstrap sample) automatically generated during the creation of the ensemble. This ensures a better validation of the members in the ensemble. We combine the classifiers using two relevantly new combination methods, named Spread and Trimmean [32]. These two combination rules performed well with Neural Networks. In this paper we used an unstable classifier, the Decision Tree with two stable classifiers, Fisher Linear Discriminant Classifier and Logistic Linear Classifier as base classifiers. Our ensemble creation method is robust in the sense that in each phase of the method (training phase and combination phase), it is robust against irrelevant decision of the any member in the ensemble. We used the κ - error diagram to check diversity property of the classifiers generated by the ensemble; also we have checked the relation between the combination rules and several diversity measures. We have used 15 benchmark datasets to compare misclassification errors of the ensemble with Bagging, Boosting (Adaboost), Rotation Forest and Rot-Boost. The investigation shows that our method improved the accuracy of the stable (linear) classifiers more than bagging and boosting, and its performance with decision tree is also competitive with other ensemble methods.*

Keywords: trimmean combiner, spread combiner, linear classifiers, validation, subsampling

1. Introduction

Ensemble learning is one of the main research directions of Machine Learning and Pattern Recognition in recent years, due to their potential to improve the generalization performance of the predictors [18], [35], [5], [15], [7], [34], [8], [27], [39]. Numerous theoretical and empirical studies have been published to establish the advantages of the predictor decision combination paradigm over the single (individual) predictor [23], [19]. The goal of the ensemble learning is to construct a collection (an ensemble)

of individual (base) classifiers that are diverse and yet accurate. Then the outputs from each ensemble member are combined in a suitable way to create the prediction of the ensemble classifier. The combination is often performed by voting for the most popular class. Most popular among the ensemble creation techniques are Bagging [5], Adaboost [15], Random Subspace Method (RSM) [20], Random Forest [8] and Rotation Forest [27].

Among these techniques, Bagging and Boosting (Adaboost) are perceived as “standard” in constructing classifier ensemble methods. Both approaches build the ensembles by training each classifier on a *customized* data set. Boosting promotes diversity actively whereas Bagging relies on independent re-sampling from the training set. Boosting has been crowned as the “best off-the-shelf classifier” by Leo Breiman [7] himself, the creator of bagging. The secret lies with the ingenious construction of the subsequent training data sets so that classifiers trained on them form a diverse ensemble. *Random Forests* is based on combining two sources of randomness when generating decision trees: (1) each tree is constructed on a bootstrap replicate of the original dataset as in bagging, and (2) a random feature subset of a fixed predefined size is considered for each node at tree construction. *The Random Subspaces method* proposed by T. K. Ho selects random subsets of the available features to be used in training the individual classifiers in an ensemble. *Rotation Forest* is a newly proposed, successful ensemble classifier technique, in which the training set for each base classifier is formed by applying PCA to rotate the original attribute axes. The main idea of Rotation Forest is to simultaneously encourage diversity and individual accuracy within the ensemble: diversity is promoted through doing feature extraction for each base classifier and accuracy is sought by keeping all principal components and also using the whole data set to train each base classifier.

We see that except Rotation Forest, Bagging, Adaboost, RSM, and Random Forest do not prerequisite any data processing to create the ensemble, thus they can be called off-the-shelf ensemble methods. Comparative studies of these methods can be found in [12], [11], [2]. It appears that, on average, among the off-the-shelf ensemble methods AdaBoost is the best method and Rotation Forest is the best ensemble method overall. Recently a new ensemble

technique, “RotBoost” [39] is also found to be performing competently with Rotation Forest, which is a combination of Adaboost and Rotation Forest.

In this paper we have proposed a novel ensemble method which selects the more, “*accurate*” classifiers from the pool of classifiers, which are selected on the basis of a validation step. This validation step consists of a validation set (independent bootstrap sample) automatically created (generated) during the training of the ensemble. The validation step inside the ensemble ensures that the ensemble will be less prone to overfitting. The idea to use an independent bootstrap sample is encouraged by the fact that bootstrapping 37% of the original training data is discarded in each bootstrap sample. It was also checked in our simulation study that the error incurred by out-of-bag samples and the error incurred by an independent bootstrap sample is nearly similar. We have selected 15 of the UCI datasets ([3]) and compare the performance of the new ensemble method with Bagging, Adaboost, Rotation Forest and RotBoost. We have used Fisher Linear Discriminant Classifier [14], Logistic Linear Classifier [1], and CART [4] as the base classifiers.

The rest of the paper is organized as follows; in Section 2 we have reviewed the related methods to our technique and our contribution in developing the new ensemble method, in Section 3 we have described the construction steps of the new ensemble method, in Section 4, the experimental results are presented, where the comparative experiments are done with the other ensemble methods. Section 5 offers the conclusion of the paper with outlines of future objectives.

2. Related Works and Contribution

Our new ensemble method closely resembles with bagging type ensembles, with selection of classifiers while training. Bagging, as every ensemble method, consists of two steps: generating multiple classifiers step (also learning step) and a combination step. Several authors tried to modify bagging in each of these steps.

In 1998 Skurichina and Duin proposed a new bagging algorithm “Nice Bagging” [29], consist of a validation or selection step based on the bootstrap training sample. In the nice bagging only the coefficients of the *nice* classifiers (the bagged linear classifiers having lesser bagged errors than the base classifier) are combined using the average rule. In one of our earlier work [36] we proposed a variation of this method using median combination rule. In [10] authors first proposed to use *trimming* classifiers (selecting classifiers) based on the out-of-bag error of the classifiers (error based on out of bag samples [6]). In that work authors ordered classifiers based on their out-of-bag error and trim the ones showing higher out-of-bag error, naming the method, “Trimmed Bagging.” They used the Linear SVM as the base classifier and showed that trimmed bag improved the performance of the ensemble of the linear SVMs than other ensemble creation techniques.

Bühlman [9] proposed to use *median* instead of average or majority vote as the combination rule; the algorithm is denoted as, “Bragging” (bootstrap robust aggregating). According to Bühlman (2003) the simple median performed slightly better than the other robust location estimators. In that paper the author also used subsampling instead of bootstrapping to create the training samples for the classifiers of the ensemble and define the procedure as Subsample Aggregating (Subagging).

In our earlier work [37] we proposed a robust bagging algorithm with a robust combination rule (median), where the selection is done using the out-of-bag samples as the validation sample and classifiers with lower out-of-bag error (than the training error) is selected. In our another paper [38] we investigated the performance of the new ensemble technique with order statistics combiners (minimum, median and maximum) and mean combiner. Hence we are incorporating the bootstrapped classifiers having lower misclassification error than the base classifier. The resulting aggregated classifier corresponds closely to the nice bagging.

2.1 Contributions

The construction of a good classifier based on a learning sample can be seen as a three-step procedure: learning different rules, selecting the optimal ones, and estimating its misclassification error. In the second step to select the optimal classifier one should consider how the validation of the selection of the optimal classifier is performed. The best way to do this is to use a validation set, independent of the training set. We follow this simple but effective criterion in constructing the new ensemble creation technique. In our new ensemble creation technique we have made several modifications from our earlier works.

- We have strengthened the training phase of the ensemble method by enlarging the training samples
- We have proposed to use an independent bootstrap sample (independent of the bootstrap training sample) in the validation step
- We have used the leave-one-out-cross-validation error (LOOCV) as the rejection threshold for the classifier, which is calculated prior to the ensemble generation.
- We have used trimmean and spread combiner as the combination rules.

These modifications will enforce several advantages in the new ensemble technique:

- Enlarging the training set for each base classifier will ensure increment in accuracy.
- LOOCV error is a true error estimate, so classifiers selected in the ensemble will have higher accuracy.
- Use of robust and order statistic combination rule will hinder any possibility of the ensemble accuracy be contaminated by unusual decision of any of its members.

3. The New Ensemble Method

In this section we will discuss the structure of the new ensemble technique. In our ensemble technique we emphasize on enlarging the training set for adequate learning of the base classifiers, and use independent bootstrap sample for validation set, and also use robust and order statistic combiners to construct a outlier resistant ensemble method. The role of several fixed combination rules (*majority voting, mean, median, min, max, product, average*) in combining multiple classifiers has been studied extensively in the last decade [21], [33], [16].

According to Tumer and Ghosh [32], in the presence of high variability of the classifiers' decisions, the trimmean and spread combiners simply outperform the mean combiner. In stable situations their performance is also equivalent to the mean combiner.

We will discuss about the two combination rule first.

3.1 Trimmean combiner

In this scheme, only a certain fraction of all available classifiers are used for a given pattern. The main advantage of this method over weighted averaging is that the set of classifiers who contribute to the combiner vary from pattern to pattern. Furthermore, they do not need to be determined externally, but are a function of the current pattern and the classifier responses to that pattern. For example let us, for a given pattern \mathbf{x} of size n , $p_{i,j}(\mathbf{x})$ be the probability assigned by the classifier $C_i(\mathbf{x})$ from a pool of classifiers \mathbf{C} , ($i = 1, \dots, L$), to the hypothesis that \mathbf{x} comes from the class w_j , $j = 1, \dots, C$. Then the trimmean combiner to combine all the decisions of the classifiers from \mathbf{C} , is defined as:

$$\mu_j = \frac{1}{n_2 - n_1 + 1} \sum_{m=n_1}^{n_2} p_{m:n,j}(\mathbf{x}), j = 1, \dots, c \quad (1)$$

3.2 Spread combiner

Instead of deleting the extreme values as is the case with the trimmed mean combiner, one can base a decision on those values. The maximum can be viewed as the class with the most evidence for it. Similarly the minimum deletes classes with little evidence. In order to avoid a single classifier from having too large an impact on the eventual output, these two values can be averaged to yield the *spread* combiner. This combiner strikes a balance between the positive and negative evidence. Let $p_{i,min}(\mathbf{x})$ and $p_{i,max}(\mathbf{x})$ be the minimum and maximum probability of the i^{th} classifier (from a pool of classifier \mathbf{C} , of size L), to classify an object \mathbf{x} . Then the spread combiner is defined as follows:

$$\mu_j = \frac{1}{2}(p_{i,min}(\mathbf{x}) + p_{i,max}(\mathbf{x})) \quad (2)$$

3.3 Pseudo code of the new ensemble technique

In Fig.1 we have given the pseudocode of the new algorithm.

INITIAL PHASE:
Calculate the LOOCV error of the base classifier $C(x)$, denote the error as ε_{loocv} .

TRAINING PHASE:
Repeat for $l = 1, 2, \dots, L$

- Take a subsample X^l of the training set X , with 75% of the objects of X .
- Construct the classifier $C^l(x)$ on X^l .
- Calculate the error ε_{1*} of each classifier $C^l(x)$ on an independent bootstrap sample X^{l*} .

SELECTION PHASE:

- Select the classifiers $C^{l*}(x)$ whose $\varepsilon_{1*} \leq \varepsilon_{loocv}$. If this condition is not satisfied train a weak version of the classifier. Denote these classifiers as $C_{valid}^{l*}(x)$.

COMBINATION AND CLASSIFICATION PHASE

- For a given \mathbf{x} , let $p_{l*,j}(\mathbf{x})$ be the probability assigned by the classifier $C_{valid}^{l*}(\mathbf{x})$ to the hypothesis that \mathbf{x} comes from the class w_j . Calculate the confidence for each class w_j , $j = 1, \dots, c$, for c classes, by the trimmean or spread combination rule

$$\mu_j = p_{l*,j}^{trim/spread}(\mathbf{x}), j = 1, \dots, c.$$

- Assign \mathbf{x} to the class with the largest confidence.

Figure 1: Pseudo code of the New Ensemble method

We will define the ensemble using trimmean as, “Trimmean” and the ensemble using spread combiner as, “Spread” from now on.

4. Experimental Results

In this section we present our findings from the experiments with the new ensemble method and several other ensemble methods using 15 benchmark datasets. The aim of our experiments were two fold. First, we wanted to check the performance of the new ensemble method with linear classifiers and compare with the results of Bagging and Boosting. In order to do that we selecte two linear classifiers: Fisher Linear Discriminant Classifier (FLD) and Logistic Linear Classifier (Loglc). Secondly we wanted check how much improvement it makes with CART as the base classifier. We compared its generalization performance with Bagged CART, Adaboost CART, Rotation Forest and Rot-Boost. As diversity among the individual classifiers in an ensemble put impetus in improving the performance of the ensemble [24], we check the diversity of the ensemble using a κ -error diagram. All the experiments were conducted using the Matlab toolbox PRTTOOLS [13]. The CART was implemented using the “treec” algorithm, FLD was implemented using the “fisherc” and Loglc was implemented using the “loglc” algorithm in the toolbox. For Bagging, AdaBoost, Rotation Forest and RotBoost, an ensemble of size 100

trees were trained. We used $L = 20$ as the new ensemble size all through the experiments, as we followed the guide line provided in [17] for the size of the bagging ensemble method.

4.1 Set up of the experiments

A brief description of the datasets are given in Table 1. This selection includes data sets with different characteristics and from a variety of fields. It should be pointed out that the “Auto_mpg” data set was originally a regression task. In Table 1 the first three columns, respectively, give the name, sample size and number of classes and number of features of each data set. For each data set, ten stratified ten-fold cross-validation was performed. A stratified n -fold cross-validation breaks the data set into n disjoint subsets each with a class distribution approximating that of the original data set. For each of the n folds, an ensemble is trained using $n - 1$ of the subsets, and evaluated on the held out subset. As this creates n non-overlapping test sets, it allows for statistical comparisons between approaches to be made.

Table 1: Description of the 15 Data used in this paper

Dataset	Objects	Classes	Features
Auto_mpg	398	2	6
Breastcancer	286	2	10
Biomed	194	2	5
German-credit	1000	2	20
Glass	214	7	9
Cleveland-Heart	297	2	13
Heart-Statlog	270	2	13
Ionosphere	351	2	34
Iris	150	3	4
Liver-disorder	345	2	6
Pima-Diabetes	768	2	8
Sonar	208	2	60
Vehicle	846	4	18
Vote	435	2	16
Wisconsin-breast	699	2	9

4.2 Statistical test

We used t -test for testing the statistical significance of the observed differences in errors of the ensemble methods. In this approach a t -test is conducted on the results of a ten-fold cross validation. This is the most widely used approach for this type of experiment. While the ten folds of the cross-validation have independent test sets, the training data is highly overlapped across folds, and use of the t -test assumes independent trials. We applied the Bonferroni correction, a calculation which raised the critical value necessary for determining significance, in order to compensate for the number of methods used in our experiments. In the Bonferroni correction the value α of an entire set of n comparisons is adjusted by taking the α value of each individual test as α/n . In our experiment the value of $\alpha = 0.05$ and $n = 8$ for the experiment with the linear classifiers and $n = 4$ for the experiment with CART.

4.3 Experiment with linear classifiers

It was demonstrated in the paper [7] that bagging is not useful when applied with linear (stable classifier) as the success of bagging mostly lies on reducing the variance of the unstable classifiers. Then a series of work was done by Duin and Skurichina on ensemble (bagging, boosting and RSM) of linear classifiers [29], [30], [31]. In their work they showed that for critical training sample size the ensemble of linear classifiers are beneficial. In [10], authors showed that their proposed algorithm, “Trimmed Bagging” yield higher relative improvement than bagging, nice bagging and bragging with linear SVMs. In this paper we have constructed ensemble of two linear classifiers FLD and Loglc with our new ensemble method named (Trimmean and Spread) and compared their performance with the ensemble of these classifiers constructed by Bagging and Adaboost. Here our main objective is to find out, a) whether the ensemble methods of the linear classifiers produce significant error reduction than the single classifier, and b) check the performance of the new ensemble method with the linear classifiers.

For each data set and ensemble method, ten 10-fold cross validations were performed. The average prediction errors are reported in Table 2. For comparison with the ensemble of linear classifiers, we display the prediction errors of the base linear classifiers, FLD and Loglc. The results for which a significant difference with base classifier was found are marked with a bullet or an open circle next to them. A bullet next to a result indicates that the ensemble method was significantly better than the individual classifier. Also we use gray shade to mark cells to show that the respective method (cell) has the lowest error for the respective data set (row). An open circle next to a result indicates that the ensemble method was significantly worse than the individual classifier. In the triplet labeled, “Win-Tie-Loss” in the last row of Table 2 the first value, the number of data sets on the ensemble method, was significantly better than the individual classifier; the second one is the number of data sets on which the difference between the performance of ensemble methods and that of the corresponding individual classifier is not significant; and the third one denotes the number of data sets on which the ensemble method was significantly worse than the individual classifier. For each data set, one of the methods was declared “the winner.” For example in auto_mpg dataset the Spread Loglc using spread combiner shows lowest error 0.1118 for this data; this method is the winner. We see that ensembles of FLD and Loglc won in 14 datasets, where as in Pima dataset the individual Loglc classifier showed better performance than other ensembles and individual FLD. Bagging won only in 1 dataset and lost in 5 datasets. It has made significant improvement using FLD in two datasets on the contrary using the Loglc it degrade the performance of the ensemble. Boosting won in 2 datasets using FLD and lost in eleven datasets. The performance

Table 2: Prediction error of single FLD, Loglc and their Ensembles using Trimmean and Spread combiner, Bagging and Boosting

Dataset	FLD					Loglc				
	Single	Trimmean	Spread	Bagging	Adaboost	Single	Trimmean	Spread	Bagging	Adaboost
Auto_mpg	0.1281	0.1213	0.1238	0.1287	0.1181 ●	0.1131	0.1144	0.1118	0.1157	0.1900 ○
Breastcancer	0.2814	0.2654 ●	0.2654 ●	0.2635 ●	0.3270 ○	0.2662	0.2865 ○	0.2806	0.2865 ○	0.2776
Biomed	0.1134	0.1116	0.1228	0.1134	0.1392 ○	0.0979	0.0874 ●	0.0932	0.1031	0.1237 ○
German-credit	0.2470	0.2430	0.2414	0.2459	0.2600 ○	0.2440	0.2481	0.2395	0.2419	0.2900 ○
Glass	0.3346	0.3343	0.3214	0.3548 ○	0.3533 ○	0.3495	0.3676 ○	0.3367	0.3371	0.3888 ○
Cleveland-Heart	0.1650	0.1620	0.1614	0.1658	0.1987 ○	0.1684	0.1627	0.1593 ●	0.1698 ○	0.2357 ○
Heart-Statlog	0.1667	0.1519 ●	0.1590	0.1659	0.2110 ○	0.1556	0.1530	0.1678 ○	0.1615	0.2000 ○
Ionosphere	0.1493	0.1397	0.1269 ●	0.1346	0.1396	0.1168	0.1274	0.1332 ○	0.1226	0.1567
Iris	0.1200	0.1273	0.1180	0.1260	0.1392 ○	0.0330	0.0280	0.0260	0.0460 ○	0.0267
Liver-disorder	0.3304	0.3299	0.3188 ●	0.3264	0.2754 ●	0.3072	0.3151	0.3151	0.3228 ○	0.3101
Pima-Diabetes	0.2451	0.2311 ●	0.2333	0.2284 ●	0.2513	0.2243	0.2277	0.2255	0.2333	0.2526 ○
Sonar	0.2404	0.2548	0.2634 ○	0.2507	0.2075 ●	0.2865	0.2452 ●	0.2620	0.2590	0.2212 ●
Vehicle	0.2348	0.2246	0.2290	0.2273	0.2447	0.2116	0.2070	0.2126	0.2158	0.2400 ○
Vote	0.0575	0.0535	0.0547	0.0575	0.0713 ○	0.0690	0.0581	0.0651	0.0575	0.0644
Wisconsin-breast	0.0395	0.0378	0.0430	0.0404	0.0395	0.0325	0.0310	0.0378	0.0322	0.0527 ○
Win-Tie-Loss		3-12-0	3-11-1	2-21-1	3-3-9		2-12-1	1-12-1	0-11-4	1-5-9

“●” indicates ensembles performed significantly better than base classifier, “○” indicates ensembles performed significantly worse than base classifier

of the new ensemble method is satisfactory (winner in 10 datasets overall, 4 using FLD and 6 using Loglc). Trimmean has significantly improved accuracy than individual classifier in 5 datasets (3 using FLD and 2 using Loglc), Spread has achieved this in 4 datasets (3 using FLD and 1 using Loglc). Both the classifiers performed significantly worse than the individual classifiers using Loglc in 1 and 2 datasets respectively.

4.4 Experiment with CART

In this section we describe our findings of the comparative experiment with our new ensemble creation technique and several ensemble creation technique of CART (Bagging, Adaboost and Rotation Forest), we also include the performance of the single CART as a reference. We denote the CART ensemble using trimmean and spread combiner as TrimCART and SpreadCART respectively. For each data set and ensemble method, ten 10-fold cross validations were performed. The average accuracies and the standard deviations are reported in Table 3. The results for which a significant difference with TrimCART or SpreadCART was found are marked with a bullet or an open circle next to them. A bullet next to a result indicates that TrimCART or SpreadCART was significantly better than the respective method (column) for the respective data set (row). An open circle next to a result indicates that TrimCART or SpreadCART was significantly worse than the respective method. In the triplet labeled, “Win-Tie-Loss” in the last row of Table 3 the first value is the number of data sets on the TrimCART or SpreadCART is significantly better than the other ensemble methods; the second one is the number of data sets on which the difference between

the performance of the TrimCART or SpreadCART and that of the other ensemble methods is not significant; and the third one denotes the number of data sets on which the the TrimCART or SpreadCART is significantly worse than the other ensemble methods. We have used in this paper a corrected estimate of σ proposed by Nadeau and Bengio [26] with a significance level of 5%. In Table 3 we see that TrimCART and SpreadCART is winner only in one and two datasets respectively, though both of the ensemble methods showed generalization performance very similar to bagging and adaboost. In this experiment it is apparent that Rotation Forest is the most accurate ensemble.

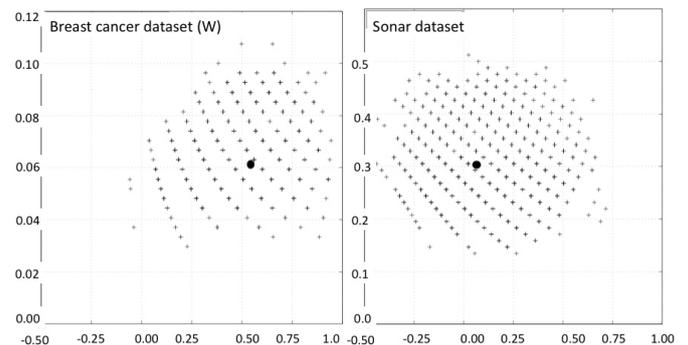


Figure 2: Kappa-error diagrams for Wisconsin Breast Cancer and Sonar Dataset. x-axis= κ , y-axis = average error of each pair.

4.5 Diversity

Margineantu and Dietterich suggested a visualization means for classifier ensembles [25]. Pairwise diversity mea-

Table 3: Mean and Standard deviations prediction error of single CART,CART Ensembles using Trimmean and Spread combiner, Bagged CART, Adaboost CART and Rotation Forest

Dataset	CART					Rotation Forest
	Single	Trimmean	Spread	Bagging	AdaBoost	
Auto_mpg	0.1481 ± 0.023 ●	0.1181 ± 0.012	0.1055 ± 0.014	0.1156 ± .018	0.1195 ± 0.031	0.1118 ± 0.029
Breastcancer	0.3244 ± 0.014 ●	0.2966 ± 0.021	0.2928 ± 0.019	0.2938 ± .035	0.3122 ± 0.023	0.2908 ± .030
Biomed	0.1534 ± 0.019 ●	0.1086 ± 0.011	0.1136 ± 0.018	0.1031 ± .016	0.1082 ± 0.027	0.1013 ± 0.010
German-credit	0.3580 ± 0.137 ●	0.3180 ± 0.053	0.3010 ± 0.075	0.2914 ± 0.064	0.2677 ± 0.0517 ○	0.2895 ± 0.081
Glass	0.2561 ± 0.066 ●	0.2243 ± 0.053	0.2740 ± 0.068	0.2595 ± 0.058 ●	0.2568 ± 0.049 ●	0.2549 ± 0.034
Cleveland-Heart	0.3250 ± 0.048 ●	0.3100 ± 0.034	0.2761 ± 0.037	0.2614 ± 0.043○	0.2527 ± 0.042○	0.1893 ± 0.064○
Heart-Statlog	0.3300 ± 0.022	0.2919 ± 0.021	0.2690 ± 0.0106	0.2193 ± 0.012○	0.2207 ± 0.011○	0.2109 ± 0.011○
Ionosphere	0.1126 ± 0.059	0.1027 ± 0.036	0.1109 ± 0.035	0.0655 ± 0.007○	0.0781 ± 0.006○	0.0551 ± 0.006○
Iris	0.0556 ± 0.007	0.0523 ± 0.006	0.0516 ± 0.005	0.0490 ± 0.006	0.0600 ± 0.007	0.0440 ± 0.005
Liver-disorder	0.3604 ± 0.013	0.3246 ± 0.055	0.3488 ± 0.054	0.3172 ± 0.054	0.3151 ± 0.049	0.3051 ± 0.045
Pima-Diabetes	0.3451 ± 0.017●	0.2811 ± 0.029	0.2433 ± 0.028	0.2443 ± 0.022	0.2755 ± 0.027	0.2553 ± 0.028
Sonar	0.3104 ± 0.018●	0.2088 ± 0.014	0.2834 ± 0.024	0.2165 ± 0.029○	0.1852 ± 0.017○	0.1820 ± 0.030○
Vehicle	0.2848 ± 0.011	0.2646 ± 0.012	0.2790 ± 0.015	0.2516 ± 0.017○	0.2470 ± 0.018○	0.2126 ± 0.011○
Vote	0.0675 ± 0.010	0.0460 ± 0.014	0.0517 ± 0.012	0.0490 ± 0.011	0.0509 ± 0.013	0.0422 ± 0.010
Wisconsin-breast	0.0595 ± .001●	0.0298 ± 0.004	0.0430 ± 0.010	0.034 ± 0.001	0.0320 ± 0.002	0.0288 ± 0.006
Win-Tie-Loss	9-6-0			1-9-5	1-9-5	0-10-5

“●” indicates SpreadCART or TrimCART performed significantly better than other ensembles

“○” indicates SpreadCART or TrimCART performed significantly worse than other ensembles

tures were chosen because “diversity” is intuitively clear for two variables (two classifier outputs in this case). Dependence between two variables can be measured using a correlation coefficient or an appropriate statistic for nominal variables (class labels). The pair-wise diversity measure used in [25] is the interrater agreement kappa(κ). Kappa evaluates the level of agreement between two classifier outputs while correcting for chance. If the classifiers are independent then $\kappa = 0$, but independence among the classifiers are not amenable in multiple classifier systems. Even more desirable is “negative dependence” $\kappa < 0$, whereby classifiers commit related errors so that when one classifier is wrong, the other has more than random chance of being correct. An ensemble of L classifier generates $\frac{L(L-1)}{2}$ pairs of classifiers C_i, C_j . On the x-axis of a kappa-error diagram is the κ for the pair and on the y-axis is the E_{ij} =averaged individual error of C_i, C_j . As small values of κ indicate better diversity and small values of E_{ij} indicate better accuracy, the most desirable pairs of classifiers will lie in the bottom left corner. Fig.2 shows the kappa-error diagrams for two data sets. All ensembles consisted of 100 classifiers (CART), therefore there are 4,950 dots in each plot. The ensembles were trained on the training sets and the kappa-error diagrams were calculated on the testing sets. The black circle in each of the diagram indicates the cloud centroid of the points in the diagram. It is clear from the graph that diversity span of the method is large and also the accuracy limit of the classifiers are also small. This implies that the new ensemble method

make an agreement between both accuracy and diversity. In Table 4 we have presented the correlation between the two combination rules and three diversity measures ($Q =$ The Q statistic, $DF =$ Double-Fault and $\theta =$ Difficulty). Several authors studied exhaustively to find out the relation (correlation) between the combination rules and diversity measures [28], [22] We used the Wisconsin Breast Cancer dataset for this experiment. From Table 4 we can say that there is no strong correlation between the diversity measures and combination methods.

Table 4: Correlation between Trimmean and Spread Combination rule with several diversity measures

Combination Rules	Diversity Measures		
	Q	DF	θ
Trimmean	0.18	-0.26	-0.34
Spread	0.23	-0.16	-0.28

5. Conclusions

This paper presents a novel ensemble classifier generation method using two combination rules: spread and trimmean. In this paper we have achieved some novel findings from our experiments. Firstly we have made an extensive study on ensemble of linear classifiers. We found out that our new ensemble method is able to reduce error of linear classifier ensembles (Table 2). It has performed superiorly than bagging and boosting while used with linear classifiers. Secondly we have checked the relation (correlation) of two new

combination rules with several diversity measure. Thirdly we have used the κ -error diagram to check the diversity-accuracy pattern of the new ensemble method. From the diagram we get the idea that the new ensemble method generates diverse classifiers within a wide range. The new ensemble method was also compared with other decision tree ensemble creation techniques with CART as the base classifier. From the benchmark experiment it can be concluded that the new ensemble method performed better than bagging and boosting and also showed potential compared with rotation forest. Future research for the development of the method include, but are not limited to:

- Check the relationship between bias-variance of the ensemble with different combination rules. If a significant relationship can be found out, then it would be easier to find out why for different combination rule the performance of an ensemble deviates.
- Use other stable and unstable classifiers e.g., SVM, nearest neighbor classifiers, neural networks to evaluate their performance with the new ensemble.

References

- [1] J. A. Anderson, *Logistic Discrimination*, Handbook of Statistics 2: Classification, Pattern Recognition and Reduction of Dimensionality, P. R. Krishnaiah and L. N. Kanal, ed. North Holland, Amsterdam, 1982.
- [2] R. E. Banfield, L. O. Hall, K. W. Bowyer, D. Bhadoria, W. P. Kegelmeyer, and S. Eschrich, "A comparison of decision tree ensemble creation techniques," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 173–180, 2007.
- [3] C. L. Blake and C. J. Merz, (1998) UCI Repository of Machine Learning Databases, <http://www.ics.uci.edu/mllearn/MLRepository.html>.
- [4] L. Breiman, J. Freidman, R. Olshen and C. Stone, *Classification and Regression Trees*. Belmont, CA: Wadsworth, 1984.
- [5] L. Breiman, "Bagging Predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [6] L. Breiman, "Out-of-bag estimation," Statistics Department, University of Berkeley CA 94708, Tech. Rep., 1996.
- [7] L. Breiman, "Arcing Classifiers," *Annals of Statistics*, vol. 26, no. 3, pp. 801–849, 1998.
- [8] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [9] P. Bühlman, "Bagging, subbagging and bragging for improving some prediction algorithms," *Recent Advances and Trends in Nonparametric Statistics*, M. G. Arkitas, and D. N. Politis eds., pp. 9–34, Elsevier, 2003.
- [10] C. Croux, K. Joosens and A. Lemmens, "Trimmed Bagging," *Computational Statistics & Data Analysis*, vol. 52, no. 2007, pp. 362–368, 2007.
- [11] T. G. Dietterich, "An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization," *Machine Learning*, vol. , no. 2, pp. 139–157, 2000.
- [12] T. G. Dietterich, "Ensemble Methods in Machine Learning," in *Proc. Conf. Multiple Classifier Systems*, pp. 1–15, 2000.
- [13] R. P. W. Duin, PRTOOLS 4.1.4, A *Matlab Toolbox for Pattern Recognition*, Pattern Recognition Group, Delft University, Nederland, 2008.
- [14] R. A. Fisher, "The Precision of Discriminant Functions," *Annals of Eugenics*, vol. 10, no. 4, 1940.
- [15] Y. Freund and R. E. Schapire, "Experiments with a New Boosting Algorithm," in *Proc. 13th Int'l Conf. Machine Learning*, pp. 148–156, 1996.
- [16] G. Fumera, and F. Roli, "A Theoretical and Experimental Analysis of Linear Combiners for Multiple Classifier Systems," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, pp. 942–956, 2005.
- [17] G. Fumera, F. Roli and A. Serrau, "A Theoretical Analysis of Bagging as a Linear Combination of Classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1293–1299, July 2008.
- [18] L. K. Hansen and P. Salamon, "Neural Network Ensembles," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, Oct. 1990.
- [19] T. Hastie, R. Tibshirani, J. Freidman, *The elements of statistical learning: data mining, inference and prediction*, New York, Springer Verlag, 2001.
- [20] T. K. Ho, "The Random Subspace Method for Constructing Decision Forests," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, Aug. 1998.
- [21] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On Combining Classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.
- [22] L. I. Kuncheva, M. Skurichina, and R. P. W. Duin, "An experimental paper on diversity for bagging and boosting with linear classifiers," *Information Fusion*, vol. 3, pp. 245–258, 2002.
- [23] L. I. Kuncheva, *Combining Pattern Classifiers. Methods and Algorithms*, John Wiley and Sons, 2004.
- [24] L. I. Kuncheva and C. J. Whitaker, "Measures of Diversity in Classifier Ensembles," *Machine Learning*, vol. 51, pp. 181–207, 2003.
- [25] D. D. Margineantu and T. G. Dietterich, "Pruning Adaptive Boosting," in *Proc. 14th Int'l Conf. Machine Learning*, pp. 211–218, 1997.
- [26] C. Nadeau and Y. Bengio, "Inference for the Generalization Error," *Machine Learning*, vol. 62, pp. 239–281, 2003.
- [27] J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: A new classifier ensemble method," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1619–1630, Oct 2006.
- [28] C. Shipp and L. Kuncheva, "Relationships between combination methods and diversity methods in combining classifiers," *Information Fusion*, vol. 3, pp. 135–148, 2002.
- [29] M. Skurichina, R. P. W. Duin, "Bagging in Linear Classifiers," *Pattern Recognition*, vol. 31, no. 7, pp. 909–930, 1998.
- [30] M. Skurichina, R. P. W. Duin, "The Role of Combining Rules in Bagging and Boosting," *Advances in Pattern Recognition*, F. J. Ferri, J. M. Inesta, A. Amin, P. Pudil eds., LNCS, vol. 1876 .pp. 631–640, Springer Verlag, Berlin, 2000.
- [31] M. Skurichina, R. P. W. Duin, "Bagging, Boosting and the Random Subspace Method for Linear Classifiers," *Pattern Anal. Applications*, vol. 5, pp. 121–135, Springer-Verlag, London, 2002.
- [32] K. Tumer and J. Ghosh, "Robust combining of disparate classifiers through order statistics," *Pattern Analysis & Applications*, vol. 5, no. 2, pp. 189–200, Springer, 2002.
- [33] L. Xu, A. Krzyzak, C. Suen, "Methods of combining multiple classifiers and their application to handwriting recognition," *IEEE Trans. Systems, Man and Cybernetics*, vol. 22, pp. 418–435, 1992.
- [34] G. I. Webb, "MultiBoosting: A Technique for Combining Boosting and Wagging," *Machine Learning*, vol. 40, no. 2, pp. 159–196, 2000.
- [35] D. H. Wolpert, "Stacked Generalization," *Neural Networks*, vol. 5, pp. 241–259, 1992.
- [36] M. F. Zaman and H. Hirose, "Robust bagging algorithm in small sample classification," in *3rd International Nonlinear Sciences Conference*, Tokyo, Japan, 2008.
- [37] M. F. Zaman and H. Hirose, "A Robust bagging method using median as a combination rule," in *Proc. 7th IEEE International Conference on Computer and Information Technology CIT'08*, pp. 55–60, 2008.
- [38] M. F. Zaman and H. Hirose, "A New Bagging Method using Four Ms with Application to Linear Classifiers," in *Proc. 4th World Conf. of IASC*, pp. 1777–1785, 2008.
- [39] C. X. Zhang and J. S. Zhang, "RotBoost: A technique for combining Rotation Forest and AdaBoost," *Pattern Recognition Letters*, vol. 29, no. 2008, pp. 1524–1536, 2008.

Large Experiment and Evaluation Tool for WEKA Classifiers

D. Baumgartner and G. Serpen

Department of Electrical Engineering and Computer Science, University of Toledo, Toledo, OH, USA

Abstract - This paper presents a new Windows®-based software utility for WEKA, a data mining software workbench, to simplify large-scale experiment and evaluation with many algorithms and datasets in the classification context. The proposed tool, LEET (Large Experiment and Evaluation Tool) makes it possible to accomplish a variety of tasks that are presently rather difficult or impractical through the standard WEKA interfaces. This includes allowing comparison of classifiers across multiple experiments, tracking execution time, calculating diversity measures, and summarizing the characteristics of many datasets. We have tested and validated LEET as part of a study with 50+ machine learning classification/ensemble algorithms, 46 datasets, and calculation of a variety of performance measures. With WEKA providing the algorithm implementations, LEET facilitates the execution and evaluation of large-scale experiments with greater ease than any existing interface.

Keywords: WEKA, large-scale experiment/evaluation, automation script, classification, diversity measure

1 Introduction

Large-scale machine learning and data mining experiments that process a significant number of algorithms through an equally large number of datasets pose unique challenges for efficient and effective management. Making large-scale experiments more controllable is of paramount interest since they typically tend to have very long run-times. Another aspect of interest is availability of tools to, at least partly, automate the entire experimentation process.

In machine learning and data mining, WEKA (Waikato Environment for Knowledge Analysis) provided the first comprehensive software workbench for research and development. Although it has been subjected to on-going improvements over the past decade, it is often necessary to develop custom scripts to facilitate experiments that may also require calculation of performance measures not readily available as part of the existing WEKA implementation. We present a Windows®-based GUI application, LEET (Large Experiment and Evaluation Tool), that, while using WEKA as the main computing engine, simplifies this process. The focus of LEET is on large-scale experiment and evaluation. Furthermore, LEET has some features that are difficult or impractical to perform within the existing WEKA interfaces.

The WEKA website lists over 50 related projects to the data mining workbench; however the utility of these are

mostly limited to extensions/enhancements of existing algorithms, addition of new algorithms, interfaces for other software, or support for specific problem domains [8]. LEET offers features that are not only unique for WEKA, but also other 3rd-party software that we currently know of.

The remainder of this paper is organized as follows. First, WEKA and its interfaces are presented. Second, LEET is explained and its features are examined along with its interface. Finally, the conclusions for this paper are given.

2 WEKA: A machine learning toolkit

WEKA (Waikato Environment for Knowledge Analysis) is a machine learning and data mining software tool written in Java and distributed under the GNU Public License [9]. The goal of the WEKA project is to “*build a state-of-the-art facility for developing machine learning techniques and to apply them to real-world data mining problems*” [8]. It contains several standard data mining techniques, including data preprocessing, classification, regression, clustering, and association. Although most users of WEKA are researchers and industrial scientists, it is also widely used for academic purposes.

A brief discussion of WEKA is offered in this section. First, the background and history of the program are given, followed by a discussion of its interfaces, and finally a summary of the limitations of the interfaces in the context of large-scale experiment and evaluation of classifiers.

2.1 Background and history

The WEKA project has been funded by the New Zealand since 1993. The initial goal at this time was as follows [6]:

“The programme aims to build a state-of-the-art facility for developing techniques of machine learning and investigating their application in key areas of New Zealand economy. Specifically we will create a workbench for machine learning, determine the factors that contribute towards its successful application in the agriculture industries, and develop new methods of machine learning and ways of assessing their effectiveness.”

In 1996, a mostly C version of WEKA was released, while in 1999 it was redeveloped and released in Java to support platform independence. Today, there are three versions of

WEKA available to the public [8]. The GUI version (6.0) is the most recent release. The developer version (3.5.8) allows users to obtain and modify source code to add content or fix bugs. The book version (3.4.14) is as described in the data mining book released by Witten and Frank [9].

2.2 Interfaces

WEKA has four interfaces, which are started from the main GUI Chooser window, as shown in Fig. 1 (this discussion is based on WEKA version 6.0). The Explorer, Knowledge Flow, and Simple CLI (Command Line Interface) all handle data preprocessing, classification, regression, clustering, and association. The Experimenter handles only classification and regression problems. Each interface has different utilities and different benefits/detriments. Following, the features of each interface are described. This review focuses on the classification aspect, since that is the context of this paper.

2.2.1 Explorer

The Explorer is possibly the first interface that new users will run simulations in. It allows for data visualization and preprocessing. Execution of a classifier allows for only a single algorithm on a single dataset. Training and testing data can be either from the original dataset (exact data, cross-validation, or percentage split) or a different dataset. The

output results can range from simple statistics like the percent of correctly classified instances to a textual or graphical model of the classifier, entropy evaluation measures, and a full listing of predictions for each instance.

2.2.2 Experimenter

The Experimenter allows a user to set up and execute a set of simulations. There is no data visualization or preprocessing, but many algorithms can be executed on many datasets. Training and testing data are limited to only cross validation and percentage split. The results of an experiment can easily be stored, but the user has no control over what the stored contents are.

The Experimenter only stores a specific set of performance measures. That is, the predictions for each instance are not saved. Although this makes storage of results more efficient, no new measures can be calculated since the predictions are not known. Analysis in the Experimenter places the results in a tabular manner (e.g. each row is a dataset and each column is a classifier), where the performance measure of interest can be changed. Furthermore, the paired t-test, which is a commonly accepted statistical significance test for comparing classifiers [4], is implemented and can be applied to the table. This allows for comparison of classifiers across multiple datasets.

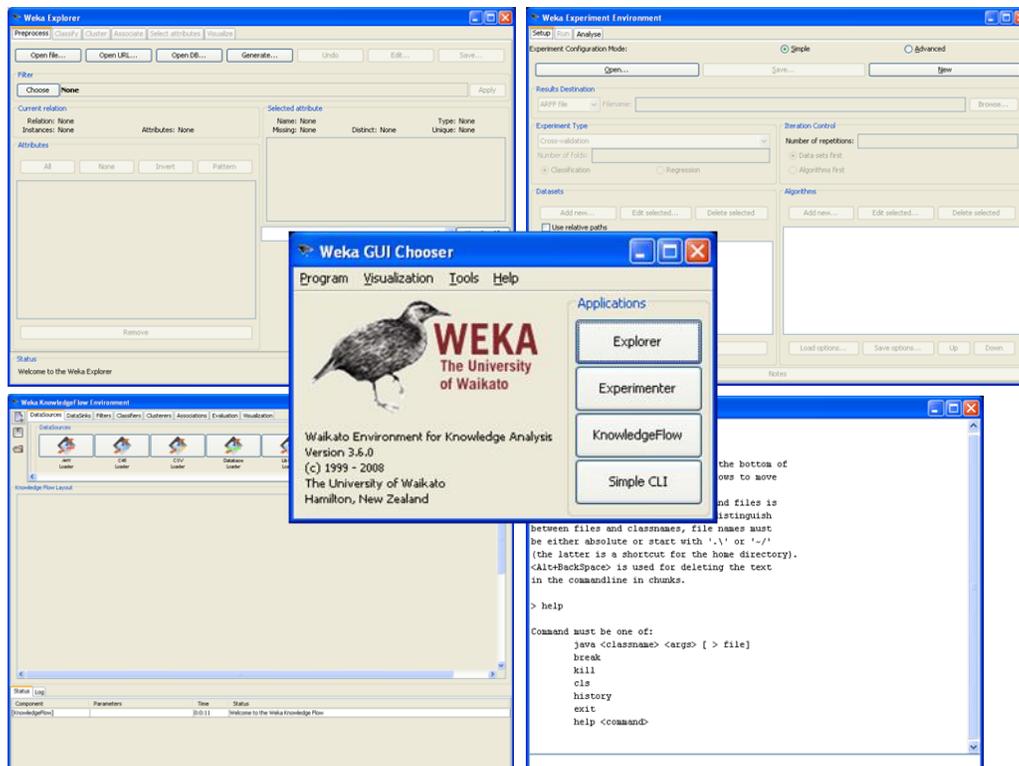


Fig. 1: The interfaces of WEKA. GUI Chooser at center, Explorer at top-left, Experimenter at top-right, Knowledge Flow at bottom-left, and Simple CLI (Command Line Interface) at bottom-right.

There are a few usability issues that the authors have noticed with the Experimenter interface. First, the results of simulations are not saved to the hard drive until the entire experiment is complete (for the ARFF and CSV file types, which are the most common). If WEKA is closed prematurely or crashes before the execution of the entire set of simulations is complete, then all of the results are lost. When an experiment is expected to take a long time to complete because of a large number of classifiers and/or datasets, there is a risk of hours, days, or even weeks of simulations being lost. Splitting up experiments into smaller pieces makes the possibility of this occurring less likely, but it leads to the second issue with the Experimenter usability.

In a typical research undertaking, it may not always be possible to determine all simulations that need to be performed. For example, after running an initial set of experiments a researcher may find a new dataset that should be included in their analysis. The Experimenter allows for classifier comparisons within only a single experiment output file. Merging the outputs of different experiments is possible, although it is difficult, time-consuming and error prone. A more flexible alternative to comparing classifiers across experiments is desirable.

2.2.3 Knowledge flow

The Knowledge Flow offers a data-flow inspired interface for WEKA [2]. It is a graphical alternative to Explorer, although not all functionality from the Explorer is available in Knowledge Flow and vice-versa. For the purposes of this paper, no distinction is made between the two interfaces.

2.2.4 Simple CLI

The Simple CLI provides a command-line interface that allows direct execution of WEKA commands [2]. This is included within the GUI of WEKA for operating systems that do not provide their own command-line interfaces. Simple CLI provides full access to all WEKA features through calls to the Java classes, although it still has the memory limitations of the other interfaces. The output is similar to what can be obtained from the Explorer, and it can be easily piped to an external file for storage.

2.3 Executing large experiments with WEKA

Each interface has a specific purpose and utility. Whereas the Explorer and Knowledge Flow are tailored to beginning users, the Experimenter and Simple CLI target more advanced users. We believe that the Experimenter is too restricting on the data that it outputs (i.e. no predictions for each instance) and not scalable for very large sets of simulations (i.e. interruption before execution is finished will lead to loss of results). The Simple CLI has memory limitations and it is rather inefficient to use since each simulation must be manually entered in. The only other

alternative is to use the operating system command-line to call WEKA.

The WEKA documentation reads, “*while for initial experiments the included graphical user interface is quite sufficient, for in-depth usage the command-line interface is recommended, because it offers some functionality which is not available via the GUI – and uses far less memory*” [2]. In the context of this statement, the command-line interface being referred to is of the operating system and not the Simple CLI. Instead of executing each simulation manually, a user can create a script to execute them sequentially.

Traditionally, researchers develop their own scripts that are catered to the specific needs of their work. Things such as folder/file structure, type of results to store, and calculation of performance measures need to be handled before simulation can begin, which can be costly with regards to time and money. Whereas the community supports WEKA with extensions to functionality (e.g. adding classifiers), there are no publicly available resources for interfaces to handle large-scale experiments [8].

3 Large experiment and evaluation tool for WEKA classifiers

We present a Windows®-based GUI application, named LEET (Large Experiment and Evaluation Tool), which facilitates execution and evaluation of large-scale classification experiments with WEKA¹. Its features are grouped into three tasks.

1. Executes classification experiments using WEKA’s built-in classifiers.
2. Evaluates the executed experiments to obtain performance measures.
3. Evaluates datasets to calculate characteristics.

LEET is created in response to requirements formed during our research of classification ensembles and the limitations of the WEKA interfaces discussed earlier. Specifically, we require the ability to run large-scale experiments and collect the predictions for each instance so that performance measures not found in WEKA can be calculated. Traditionally, researchers must create static scripts that execute experiments and parse the result files. Instead, we have developed LEET as a GUI application that makes the experiment and evaluation process simpler.

The current implementation of LEET has been validated in versions 3.5.8 and 3.6.0 of WEKA. It can execute any classifier that WEKA allows and calculates the prediction accuracy average, prediction accuracy standard deviation, execution time, and diversity of classifiers.

¹ LEET is available at <<https://sourceforge.net/projects/largeexperiment/>>.

Furthermore, LEET offers a utility to calculate the characteristics (e.g. number of instances, classes, and features) for a large set of datasets; something that cannot be done in WEKA.

In the remainder of this section LEET is described. First, specifications of the environments in which LEET is developed, and can consequently run are presented. Second, the tasks that the program performs are elaborated upon. Finally, validation of LEET is offered through its use in our own large-scale research experiments.

3.1 Environment

The development of LEET was originally intended for in-house use only, so it was developed in the environment that we are most familiar with – C# in Visual Studio 2008. Consequently, the program requires the free .NET Framework 3.5, and unless third party software is used, LEET is restricted to Windows XP and Windows Vista.

LEET is an application outside of the WEKA interface. This has two main benefits. First, as previously mentioned, using the operating system command-line to call WEKA uses memory more efficiently than running the traditional interfaces. Second, LEET can work across multiple versions of WEKA since there is no integration of code. The functionality of LEET depends only on the WEKA input and output. The same may not be true if the programs were integrated – the GUIs alone of the three current versions of WEKA would need separate implementations to support an additional interface such as LEET. The concept of support for multiple versions has already been validated, as LEET was originally developed for version 3.5.8 of WEKA, but has been verified to work without modification for the newest 3.6.0 version.

3.2 Features of LEET

The LEET application is shown in Fig. 2. There are three main areas in the window. The menu area at the top contains information about the program and the program settings. The settings contains properties such as the WEKA install directory, default directory for the experiments to be saved, and the maximum heap size of the Java virtual machine that WEKA runs on. The area at the bottom with the three tabs is where the experiments and evaluations are performed. The area between the prior two is for dataset selection. Since dataset selection is a common operation for all three tabs, having it at a single location helps to reduce the clutter and increase usability of the interface. The functionality of each tab is explained below.

3.2.1 Execute - Experiments

The Execute – Experiments tab of LEET is shown in Fig. 2. This component serves as a GUI for creating and executing command-line simulations in WEKA. There are two main text boxes for input. The user enters classifier

configurations in “Classifiers”. These can be formed in the Explorer interface within WEKA or manually typed. Each classifier must have an alias associated with it, which is specified in “Classifier Alias”. Notice in Fig. 2 that there are three J48 classifier configurations and three classifier aliases. The aliases are used to build the directory structure that stores the simulation results.

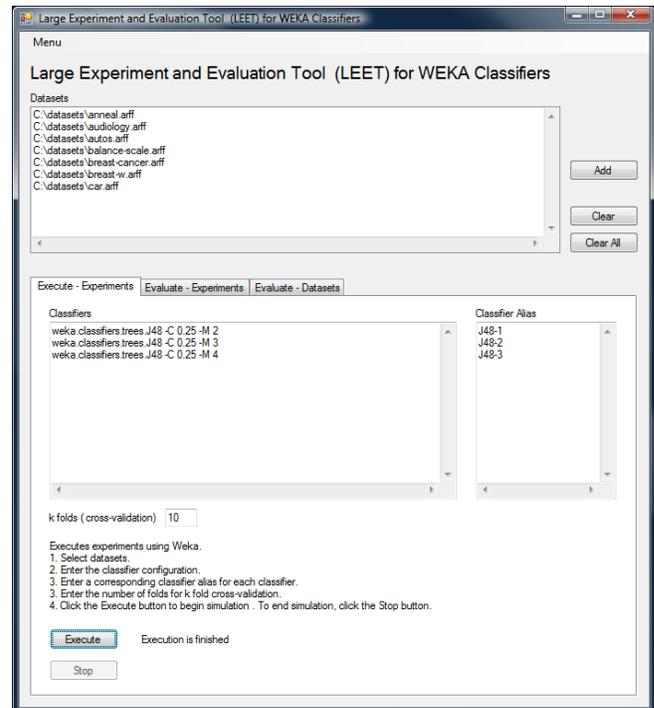


Fig. 2: The Execute – Experiments tab of LEET.

The last parameter needed for an experiment is the number of folds in cross-validation. Although this is the only sampling method supported in LEET, it is the most common and accepted method in the literature [4]. The number of folds can be specified in the “k fold (cross-validation)” text box, and is defaulted to 10.

The “Execute” button starts the experiment. The flowchart for the experiment process is shown in Fig. 3. Each classifier is run on the specified datasets. After a simulation (i.e. a classifier and dataset pair) is complete, the execution time and prediction results are stored in files within a directory named by the classifier alias.

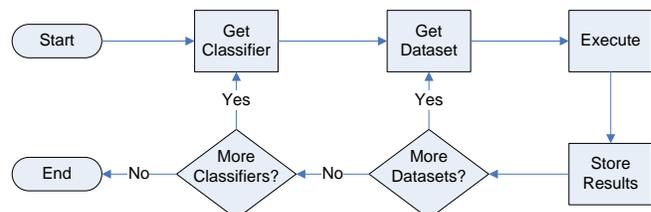


Fig. 3: Flowchart for the execution of experiments in LEET.

Execution times for simulations of a classifier are aggregated in a single file. Each line in this file specifies the execution time in “hours:minutes:seconds” format for a different dataset. Individual files store the prediction results for each simulation. These files are named by the dataset used in the simulation. A partial sample of a prediction results file is shown in Fig. 4. The first line shows the classifier configuration, while the second displays the date/time of the execution. The remainder of file contains the predictions as output by WEKA. Notice that the actual and predicted class information is provided. Furthermore, the final column specifies the probability of the prediction. This extra data facilitates calculations of more advanced measures than simply accuracy.

```

Classifier: weka.classifiers.trees.J48 -C 0.25 -M 2
Execution: 2/18/2009 3:52:22 PM

==== Predictions under cross-validation ====

inst#   actual      predicted   error   prediction
  1     3:cochlear  3:cochlear    0.941
  2     3:cochlear  3:cochlear    0.941
  3     3:cochlear  3:cochlear    0.941
  4     3:cochlear  3:cochlear    0.941
  5     3:cochlear  3:cochlear    0.941
  6     3:cochlear  3:cochlear    0.941
  7     7:cochlear  7:cochlear    0.857
  8     7:cochlear  7:cochlear    0.857
  9     20:otitis_  16:mixed_c    +    0.429
 10     4:cochlear  15:mixed_c    +    0.6
...     ...         ...           ...     ...

```

Fig. 4: Partial sample of the prediction file stored after a simulation in LEET.

The experiment execution within LEET differs from WEKA’s Experimenter in three main ways:

- The predictions for each instance are stored. This allows for calculation of any performance measure, instead of a predetermined set of performance measures.
- The execution time for the simulation is stored. This is an important factor to consider in real-world scenarios.
- The results are stored after each simulation instead of waiting until all simulations are complete. Consequently, LEET is less sensitive to losing progress in program/computer crashes since only a single simulation is interrupted rather than a whole set. Resuming the experiment is as simple reselecting the datasets and entering the classifier configuration and aliases of those that need completed. This supports setting up and running large-scale experiments.

3.2.2 Evaluate - Experiments

The Evaluate – Experiments tab of LEET is shown in Fig. 5. Here, the “Classifier Alias” corresponds to the aliases entered earlier when the experiments were executed.

The performance measures supported in the current version of LEET include prediction accuracy average, prediction accuracy standard deviation, execution time, and diversity. This set of measures are required for our current research with classifier ensembles, and besides the prediction accuracy values, they are not calculated within WEKA. Since the actual predictions for each instance are stored rather than a pre-specified set of measures, additional measures can be calculated in the future without requiring simulations to be re-executed.

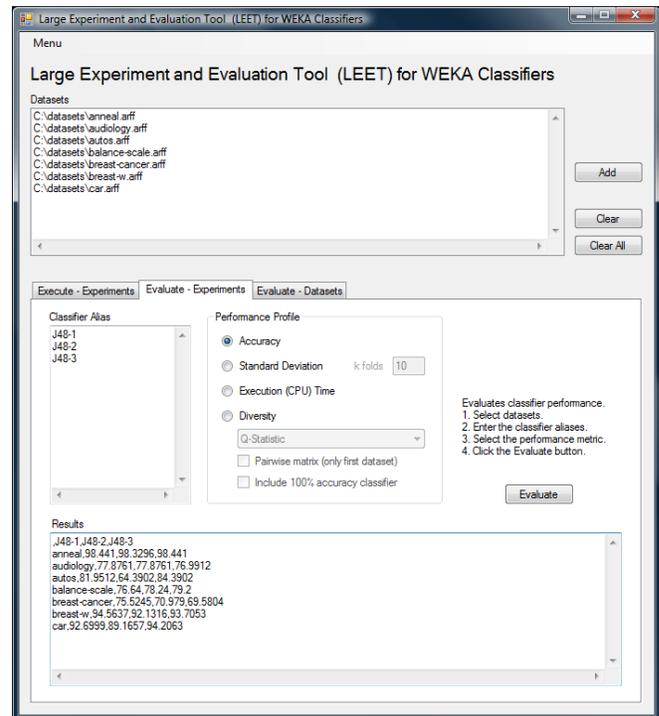


Fig. 5: The Evaluate – Experiments tab of LEET.

Diversity between classifiers is a concept of classification ensembles. Those included in LEET are Q-statistic, disagreement, weighted count of errors and correct, and exponential error count [1,3]. Furthermore, there are options to output the diversities in a matrix form, which can be used as input for a method of visualizing the diversity between classifiers in a two-dimensional space [5].

The results of the performance measure are output in a CSV format in the “Results” text box. This allows for easy copy-and-paste into a spreadsheet application, where further processing can be performed. Statistical significance tests have not been implemented within the current version of LEET because of their algorithmic complexity.

It is common for researchers to require simulations that are supplemental to their original experimentation (e.g. a new dataset is explored). These two sets of results need to be handled as one to allow for evaluation. While WEKA’s Experimenter requires a manual merging of result files to accomplish this, LEET makes the process simple. Any

classifiers that have been executed with LEET can be considered as a single group for evaluation.

3.2.3 Evaluate - Datasets

WEKA has the ability to visualize and give characteristics about datasets via the Explorer, but this can only be performed for one dataset at a time. Large research projects may contain 10's or even 100's of datasets, and it is customary in the literature to present some basic characteristics of each dataset [7]. LEET offers an alternative to WEKA that makes obtaining characteristics for many datasets simple.

The Evaluate – Datasets tab of LEET is shown in Fig. 6. The only parameter for this is choosing between number and percent for the feature outputs. For all of the selected datasets, the following characteristics are output to the “Results” text box: number of instances, number of classes, number/percent of unary features, number/percent of binary features, number/percent of nominal features, number/percent of numeric features, percent of majority classes, and percent of minority classes. Like in the experiment evaluation, the output is in CSV format which allows for easy exporting to a spreadsheet for formatting.

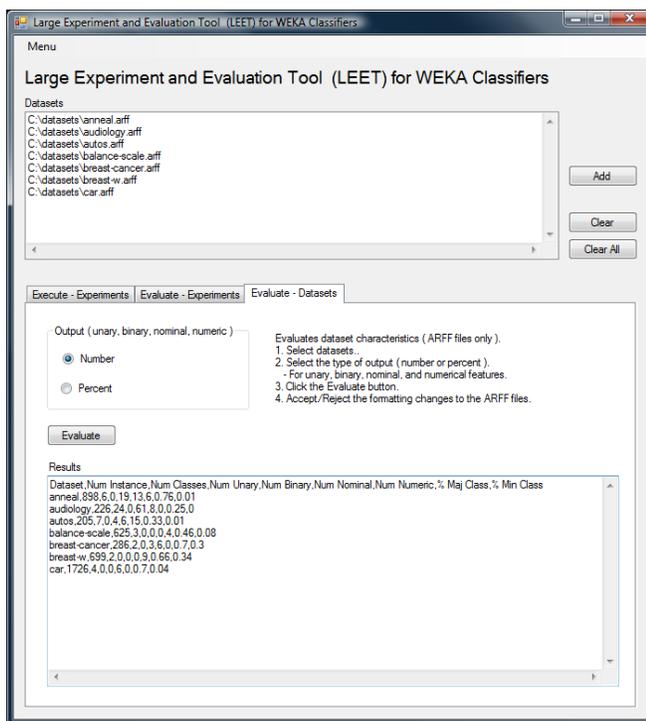


Fig. 6: The Evaluate – Datasets tab of LEET.

3.3 Large-scale experiments with LEET

LEET has played an important role in our current research with classification ensembles. It not only makes the process of experimentation and evaluation simpler, more controllable, and manageable but also provides an opportunity to verify the functionality of LEET. Our

research endeavors require experiments with 50+ single-classifier and classification ensembles; each of which is executed on 46 datasets. To our knowledge, this is among the largest empirical studies in the literature.

Furthermore, we have used the experiment evaluation functionality within LEET to create groups of classifiers for which performance comparison is carried out. Since the output of LEET is in CSV format, exporting the data to other programs for further processing is trivial. For example, we have generated a spreadsheet that automatically calculates statistical significance between classifiers within the output generated by LEET.

4 Conclusions

Traditionally, data mining experiments are large and difficult to manage effectively. Researchers must create static scripts to execute and calculate performance measures. The same is true with WEKA, a popular and powerful data mining workbench. In this paper, we have presented LEET as an application for executing and evaluating large-scale classification experiments using WEKA.

LEET can execute any classifier within WEKA and stores the results in a manner to allow for the calculation any performance measures. LEET calculates the average and standard deviation of classifier prediction accuracy, but also includes execution time and diversity measures, which are not currently available in WEKA. Furthermore, LEET offers a utility to calculate the characteristics (e.g. number of instances, classes, and features) for a large set of datasets; something that cannot be done in WEKA.

The development of LEET has been concurrent with our research in classification ensembles. This has allowed for in-depth validation of the features of the application. LEET is a work-in-progress, and additional performance measures are likely to be added – especially as our research of classifiers encourages. Porting LEET to Java to allow for platform independence would be beneficial for the community. Another possible modification is to integrate LEET into WEKA, although being external does have benefits, as discussed in earlier in this paper.

5 References

- [1] M. Aksela and J. Laaksonen. (2006). Using diversity of errors for selecting members of a committee classifier. *Pattern Recognition*, 39, 608-623.
- [2] R. R. Bouckaert, E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald, and D. Scuse. (2008). *WekaManual for Version 3-6-0*. University of Waikato, New Zealand. Retrieved from <<http://www.cs.waikato.ac.nz/~ml/weka/index.html>>.

- [3] L. I. Kuncheva. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. Hoboken, NJ: John Wiley & Sons, Inc.
- [4] T. M. Mitchell. (1997). *Machine Learning*. McGraw Hill.
- [5] E. Pezalska, R. P. W. Duin, and M. Skurichina. (2002). A discussion on the classifier projection space for classifier combining. *Multiple Classifier Systems, Lecture Notes in Computer Science*, 2364, 137-148.
- [6] B. Pfahringer. (2007). Weka: A tool for exploratory data mining [Power-Point slides]. University of Waikato, New Zealand. Retrieved from <<http://www.cs.waikato.ac.nz/~ml/weka/index.html>>.
- [7] A. Shawkat and K. A. Smith. (2006). On learning algorithm selection for classification. *Applied Soft Computing*, 6, 119-138.
- [8] WEKA: The University of Waikato. Retrieved February 11, 2009 <<http://www.cs.waikato.ac.nz/~ml/weka/index.html>>.
- [9] I. H. Witten and E. Frank. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. 2nd ed., Morgan Kaufmann, San Francisco.

Formal Model for Fault Recognition in Modern Telecommunication Networks

Jacques-H. Bellec, and Tahar-M. Kechadi
 University College Dublin, School of Computer Science,
 belfield campus, dublin 4, Ireland.
 (email: Jacques.Bellec@ucd.ie, Tahar.Kechadi@ucd.ie)

Abstract—The 3G networks allow voice, video and internet access over mobile devices. As those networks are wide and complex, their management has to be tackled by an efficient methodology. When a fault appears in the network, it can lead to a cascade of faults, which more often mask the original one. Moreover, components in error state do not provide information on the fault, but only on the local effect of the fault. Previous attempts to create an efficient fault recognition technique have failed due to the lack of an efficient model and by assuming too many hypotheses on the nature of telecommunication networks. In our approach, we define a model that captures the key notions and relationships, common to most telecommunication networks. We illustrate and experiment our model with real data generated by 3G telecommunication networks.

I. INTRODUCTION

Who could disagree that electronic devices have invaded our lives for the past 15 years? Indeed, mobile phones, PDAs, laptops and other devices are daily used by millions of people and are now connected to the internet by wide telecommunication networks. Modern 3G networks allow voice, video and internet access to users who can use more and more services. However, when a fault appears in a network, a disruption of one or several services can follow, leading to a considerable loss of bandwidth, network connections, quality of service, etc. Faults in large, complex and heterogeneous networks are unavoidable, but a quick detection is profitable to everyone. The problem of fault recognition in telecommunication networks has been tackled using various approaches. Available techniques are not efficient enough to cope with the complexity of modern networks and are based on the skills of human experts in network fault recovery. The aim of this paper is to describe a model for fault recognition in telecommunication networks that can be used for further network analysis.

II. BACKGROUND

The main problem is to identify the origin of a fault from a continuous flow of alarms. Today's network management systems integrate event correlation engines to address these issues [1], [2]. The problem of an automatic identification of correlated events has been tackled from various perspectives with different techniques, but usually their results are not satisfactory [3], [4], [5], [6]. This problem presents many challenges; 1) Wide telecommunication networks are composed of heterogeneous hardware devices and configuration policies, so their topological information is partially known

and cannot be considered reliable. 2) The alarms provide incomplete information about the faults. The alarms are generated by components being in error-state because of the presence of a fault nearby, but one fault can trigger other faults thus increasing the number of alarms and masking the original faults. Moreover, a fault can be categorised as unknown by the components, as only a few error messages are possible. In this case it does not provide any information about the nature of the fault. Some alarms can be destroyed or lost in the network, so they do not appear in the log files. 3) Some sub-networks can be configured with particular policies and, therefore, produce different alarm messages for the same fault. Given these problems, we believe that a good fault recognition solution should not require the use of any given topological maps. The output of a fault recognition system should be a set of clusters/groups that contain highly-correlated events of alarms that lead to at least one fault for each group.

In the last two decades many techniques have been developed in attempt to obtaining a good solution for the fault recognition problem. One of the main weaknesses of these techniques is that either they put some major limitations by simplifying the model or provide partial solutions. For instance, some of them as presented in [7], [8], assume the availability of the network topology, performance log files, previous alarm log files, detailed faults/alarms training sets, and the presence of some experts during decision making phases [9], [10]. Others assume a predefined model with a specific topology, management architecture, alarm format [11], [12], or independent faults can not happen in a time window t [13], [14]. Usually, they do not have a general view of the fault recognition problem and they are limited to only temporal correlations [15], [16]. However, a very few techniques take into account the alarms' behaviour, the network topology, the real-time constraints, which we believe are the key attributes.

III. FORMAL MODEL

In the following, we describe some key notions we use in our approach.

A **fault** is a “disorder” or failure occurring in the hardware or software of the network [17]. An **alarm** is an external notification of a potential failure. These notifications may originate from management systems or from probes widespread over the network. Fault behaviour can be characterised by

three states: permanent, intermittent or transient. Intermittent faults occur on a discontinuous or periodic manner, causing degradation of a service during that time. Transient faults can cause a temporary and minor degradation to a service. Permanent faults need some repair actions in order to restore the system normal behaviour. An **error** is an abnormal state which is the visible effect of a fault. A single fault can generate several errors. An error can be located by identifying the **event**, namely the set of alarms which have been sent to notify the presence of the error. A **warning** is an abnormal state which may be followed by an error. A **symptom** is seen as a set of warnings and errors which characterise a fault. An alarm has a set of attributes such as:

- Event time stamp: the time the alarm was issued.
- Logged time: the time the alarm was recorded.
- Perceived severity: the importance of the alarm effect which is ranging between critical and indeterminated.
- Alarm ID: the unique serial number of the alarm.
- Alarm Key: is composed by all alarm attributes.
- Node ID: the unique serial number of the node in a subnetwork.
- Event Type: indication of the nature of what happened.
- Probable Cause: indication of why it happened.
- Specific Problem: clarifies what happened.

Definition 1: A symptom S_i is defined by a triplet $\langle W_i, E_p, E_s \rangle$ where W_i is a set of alarms gathered in an event warning before the fault occurs, E_p is a set of alarms which can be viewed as the primary errors, and E_s is a set of secondary errors. These sets can be empty and we assume that they are finite and disjoint.

Definition 2: A fault F_i is defined by $\langle T_i, S_i, t, p \rangle$, where T_i is the nature of a fault, S_i is the symptom of a fault, t and p are the time and place when and where the fault occurred respectively.

Definition 3: A fault F_i is correlated to F_j if there is a causal relationship between them noted $F_i \rightarrow F_j$ or $F_i \rightarrow F_j$.

Let A be a set of all alarms recorded in the dataset. An alarm is defined by its message content, place, and the time of its occurrence. Formally it can be defined as follows.

Definition 4: An alarm a_k is defined by a triplet $\langle \lambda(a_k), \omega(a_k), \delta(a_k) \rangle$ where $\lambda(a_k)$ is a set of parameters identifying the alarm content (message), $\omega(a_k)$ is a set of topographical parameters (the place in the network) and $\delta(a_k)$ is a set of temporal parameters (the time when it was produced and registered).

Definition 5: Two alarms are said to be **identical** if they originate from the same network component with the same embedded message. $(a_i \equiv a_{i'}) \iff \lambda(a_i) = \lambda(a_{i'})$ and $\omega(a_i) = \omega(a_{i'})$.

The notion of identical alarms will be used to eliminate all redundant alarms. However, two identical alarms occurring at different times may refer to two different errors in the system. Therefore, we need to study the behaviour of the alarms in order to distinguish between redundant alarms and the others.

Definition 6: The **behaviour** of an alarm a_i , $\beta(a_i)$, is defined by the number of identical alarms to a_i , its period, and a period threshold; $\beta(a_i) = \langle |a_i|, \rho_{a_i}, \iota_{a_i} \rangle$.

The period of an alarm is defined to be the average time between any two consecutive identical alarms. The period threshold is the maximum acceptable rate between two identical alarms in the same group. For instance, $\beta(a_i) = \langle 1, 0, 0 \rangle$ is a single alarm. All *periodic* identical alarms are gathered in the same group. Two identical alarms belonging to two different groups are called *aperiodic*. Because we assume that if the time between two consecutive alarms is higher than the threshold value, they are considered to report different errors or faults. In our study, we consider that single and twin alarms are aperiodic and therefore they are in different groups. The problem of identifying periodic and aperiodic alarms is not straightforward. Due to a network delay or overloading, some identical alarms may be lost or delayed and the remaining identical alarms will be classified in different groups. This problem can be aggravated by the presence of overlapping events, leading to a wrong estimation of the period between identical alarms of the same group.

To solve this problem, we develop an algorithm based on fuzzy logic to improve the accuracy of each group of alarms. This algorithm is called *Score-Matching* (SM). The main advantage of this technique is that it assigns a degree of confidence to each group, which can be used later during the clustering phase.

Definition 7: Two alarms are said to be **similar** if they almost have the same content. $a_i \sim a_{i'} \iff \lambda(a_i) = \lambda(a_{i'}) + \Delta_\lambda$

Definition 8: Two alarms a_i and a_j are said to be **correlated** if they are **similar** and if they verify the following:

$$D(a_i, a_j) = \sqrt{\sum_{i=1}^n \left(\frac{d_i^{min}}{MAX(d_i^{min})} \right)^2 \cdot w_i}$$

$$\psi_{e_k}(D(a_i, a_j)) \geq \alpha$$

where $d_i^{min} = |\lambda_i(a_i) - \lambda_i(a_j)|$
 ψ is the membership function of the set e_k which contains a_j , α is the degree of confidence in the process, and w_i is the weight which expresses the importance of the parameter p_i . All values d_i are normalized using their maximum values which gives the same importance to all factors even if they present different scaling metrics.

Definition 9: Two alarms are said to be **strictly correlated** if they are **identical** and they appear in the same time neighborhood. Those alarms belong to the same set e_i . $a_i \cup a_{i'} \in e_i \iff \omega(a_i) = \omega(a_{i'}) + \Delta_\omega$ and $\delta(a_i) = \delta(a_{i'}) + \Delta_\delta$

Definition 10: An alarm a_i is **correlated** to a set e_i if there exists an alarm a_j in this set which is correlated to a_i . $a_i \in e_i \iff \exists a_j \in e_i, \psi_{e_i}(D(a_i, a_j)) \geq \alpha$

Let A be a set of alarms and let e_k be a fuzzy set in A , characterized by a membership function $\psi_{e_k}(a_i)$ which assigns to each element of A a value between $[0, 1]$. Let α be the minimum degree of confidence of belonging to a set

e_k .

$$a_i \in e_k \iff \psi_{e_k}(a_i) = \psi_{e_k}(\lambda(a_i), \omega(a_i), \delta(a_i)) \geq \alpha$$

In [18] and [9], the authors have considered some of the alarm correlation types as compression; reducing multiple occurrences of an alarm into a single set called an event, removing the warning and non-interesting alarms from the dataset. In our model, we complete and develop this notion of event.

Definition 11: An **event** is a set of strictly correlated alarms. Given e_i a set of alarms in A , we have $e_i = \sum_{j=1}^m a_j$ where all the alarms a_j are "strictly correlated".

The process of constructing events can be seen as part of the preprocessing phase. Indeed, the event recognition is also a compression of the data, as only one alarm is required to represent the full set. Its goal is to classify the events by importance into three categories. Some of the attributes of an event are:

- Event ID : unique ID,
- Start Time: minimum time of all its alarms,
- End Time: maximum time of all its alarms,
- Relevance: maximum score of its alarms,
- Place : extrapolated from the density of alarms,
- Alarm list: list of all the alarm IDs,
- Code Type: interpreted nature of the event (Primary, secondary or tertiary),
- Nb Alarms : number of embedded alarms.

Although some dimensions like the time and the place are directly extracted from the alarms, the relevance is more complicated to find out. After the system has been configured with some general priority rules which are expressed in the IF() THEN() ELSE() format, a score representing the overall importance can be calculated for each event. The scoring function χ uses static attributes defining the nature of the alarms of an event. This function is a fuzzy membership function that assigns a value for an event, which is

$$\chi(e_i) = \text{MAX} \chi(a_j) = \text{MAX} \chi(\lambda(a_j)) \quad (1)$$

In order to identify the most relevant events, three categories are created:

- 1) Primary $\iff \chi(e_i) \geq \alpha$,
- 2) Secondary $\iff \chi(e_i) \in [b, \alpha]$,
- 3) Tertiary $\iff \chi(e_i) \in [0, a]$.

Where $\alpha \in [1, 0]$ is the degree of confidence, a and b are two constants, with $a \leq b \leq \alpha$. Primary events are the most important; they describe hardware or software critical errors, so they can be viewed as primary symptoms. Tertiary are less interesting events because they describe some telecommunication failures which are the most common and minor effects. Finally, secondary events are moderately interesting, they might describe some errors produced in cascade from an original fault. Although this classification is made according to two constants a and b , it can be tuned up to refine the quality of each set. The fuzzy correlation system EDM provides results with a degrees of confidence α chosen by the user at the beginning of the procedure.

Definition 12: An event is said to be critical, relevant, interesting or primary if its membership value is strictly superior to a user defined threshold α .

$$e_i \in \Gamma(E) \iff \alpha < \chi(e_i) \leq 1$$

Definition 13: Two events are said to be correlated if there exists at least one alarm from each set correlated together. $e_i \cup e_j \in C_k \iff \exists a_m \in e_i$ and $a_n \in e_j$ where $\psi(D(a_i, a_j)) \geq \alpha$

A **cluster** is a set of correlated events. The number of clusters can be calculated dynamically or chosen in advance, but must be significantly low compared to the number of raw input alarms. Basically, the number of clusters represents more or less the number of faults that should be solved by the operator. A cluster can be described by using several characteristics as presented in [19]. Given a cluster $C_m \in C$ of N events $\{e_{m1}, e_{m2}, \dots, e_{mN}\}$, we have:

$$\text{centroid} = O_m = \frac{\sum_{i=1}^N e_{mi}}{N} \quad (2)$$

$$\text{radius} = R_m = \sqrt{\frac{\sum_{i=1}^N (e_{mi} - O_m)^2}{N}} \quad (3)$$

$$\text{diameter} = D_m = \sqrt{\frac{\sum_{i=1}^N \sum_{j=1}^N (e_{mi} - e_{mj})^2}{N * (N - 1)}} \quad (4)$$

The centroid is defined as the middle of the cluster and it does not need to be an actual event in the cluster. A cluster is a final result according to a degree of trust α . The aggregated events which compose the clusters are said to be close enough with a certain degree of trust α and with a specified scope. We call C the set of all possible cluster c_k , $C = \sum_{k=1}^n c_k$, and Ψ_{c_k} the membership function of c_k .

$$\exists e_i \in E, \Psi_{c_k}(e_i) \geq \alpha \Rightarrow (e_i \in c_k). \quad (5)$$

Some attributes of a cluster are as follow:

- Cluster ID : unique ID
- Start Time: minimum time
- End Time : maximum time
- Score: calculated according to the sum of all events scores which composed the cluster.
- Place : extrapolated from the density of the events
- Rep list: the main relevant events
- Event list : the list of all embedded events
- Nb Alarms : number of embedded alarms

Definition 14: An **episode** is a projection of an event on one dimension. This dimension is the alarm content λ .

$$\Phi(e_i) = \begin{cases} \Phi(\sum_{j=1}^n a_j) \\ \Phi(\sum_{j=1}^n \langle \lambda(a_j), \omega(a_j), \delta(a_j) \rangle) \\ \sum_{j=1}^n \lambda(a_j) = p_i \end{cases}$$

An episode can be related to a warning, a primary error or a secondary error. It can also be marked as validated or unvalidated by the user. This idea has been already introduced in [20], where the authors aim to analyze sequences of events, and to discover recurrent combinations of events, called

frequent episodes. Although their technique is mainly based on the generation of episodes, we use them to consolidate our clustering process by taking into account redundant patterns. Namely, when we test if an alarm belongs to an event, we look if it appears in the set of known or frequent episodes and we can increase or decrease the confidence for this alarm to belong to this event.

Although networks are continuously evolving, some fault prototypes can be isolated and stay valid for a while. However, their validity time is unknown making systems only based on frequent patterns easily outdated. Fault prototype can make short term prediction on the presence and nature of a fault.

A **scenario** is a fault prototype, namely a validated sequence of correlated episodes. It is a projection of a valid cluster on one dimension λ . A scenario can be composed by one or several episodes. They are extracted during the cluster validation phase. A scenario is different from a cluster, as it does not take into account the place and time of the alarms but just the nature (core) λ . Scenarios are stored in a database with a corresponding frequency and degree of confidence. They can be very helpful when dealing with real-time constraint correlation when working on alarm stream. For example, when a set of incoming events match a known scenario, possible fault descriptions and remedies are proposed to the network operator. However the validity of a scenario is always changing with the evolution of the network making the update of the fault scenario database quite complex.

Definition 15: We call Υ_i the scenario describing a fault type characterized by $\langle T_i, [p_j, p(j+1), p(j+2), \dots, p_k], \eta_i, \epsilon_i \rangle$ where T_i is the nature of the fault, $[p_j, \dots, p_k]$ is the set of the relevant episodes describing the fault, η_i is the frequency of this prototype and ϵ_i is the confidence.

We define **Inter-correlated** events by the fact that they belong to different classes. By Inter-correlation constraint, we focus on discovering multi-level correlation among the different natures of alarm. For instance, a primary event and a secondary event can be said inter-correlated.

We define **Intra-correlated** events by the fact: all of the events in the correlation rules must belong to the same class of event. By Intra-correlation constraint, we can discover the correlation relations in the same level of importance in the telecommunication network. For example, two secondary events can be intra-correlated.

The **scope** defines the network area that the system will take into account when correlating alarms. The correlation process can be done for a particularly interesting scope defined by the network operator. It can be viewed as a constraint which limits the correlation process to a certain level of the network hierarchy or a set of places in the network. It can be worthwhile for the network operator, to get only a close up to a certain scale of the network, and so have a more accurate view of what is going on there.

IV. FORMAL SPACES

The pattern recognition process from raw data to pre-processed data is characterised by a large reduction of the number of elements. It includes several distinct levels of abstraction as the input is transformed through the following spaces [21], [22]:

Input space A : Clustering is based on observations about the objects under consideration. The input space captures all the known alarms generated by the network components. Let N denote the number of alarms in the input space, and let a particular alarm have D associated attributes. The number of attributes can vary between alarms.

Feature space E : The transformation $\gamma : A \rightarrow E$ removes some redundancy by feature selection and extraction. Feature selection implies that a subset of the input dimensions that is highly relevant to the problem is selected. For example, the attributes like time, space, severity and probable cause are the most useful to find an explanation on the nature of the fault. A feature extraction transformation may extract the hidden information like the behaviour of an alarm and the behaviour of a set of alarms. Features or attributes may be distinguished by the type and number of values they can take:

- binary: Boolean true or false,
- nominal: finite number of categorical labels with no inherent order, such as the severity,
- ordinal: mappable to the cardinal numbers with the smallest element and an ordering relation, such as the place,
- continuous: real valued measurements such as the time,
- combination of different types.

The transformation to feature space may render some alarms unusable, due to missing data or non-compliance with input constraints. Thus, we denote the corresponding number of objects as n with $n \leq N$ and the number of dimensions as d with $d \leq D$.

Similarity space S : It is an intermediate space between the feature space E and the output space C . The similarity transformation $\phi : E \times E \rightarrow S$ translates a pair of internal object-centered descriptions into terms of features into an external relationship-oriented space S . While there are $n \times d - dimensional$ descriptions (e.g., $d \times n$ matrix $X \in E^n$), there are $(n-1)n/2$ pairwise relations (e.g., $n \times n$ matrix $S \in S^{(n \times n)}$). In our work, we mainly use a distance function $D(a_i, a_j)$. The dual notions of distance and similarity can be interchanged. However, their conversion has to be done carefully in order to preserve a overall coherence of the system. Similarities $\delta \in [0, 1] \subset \mathbb{R}$ and distances $d \in [0, \infty] \subset \mathbb{R}$ can be related in various non-linear, monotonically decreasing ways.

Output space C : In partitioning the output space is a vector of nominal attributes providing cluster membership to one of k clusters. To represent a clustering, we denote it either as a set of cluster $\{C_l\}_{l=1}^k$ or as a n -dimensional label vector ρ , where $a_j \in C_l \iff \rho_j = l$. Many approaches to the output transformation $\Phi : E^n \rightarrow C^n$ or $\Phi : S^{(n \times n)} \rightarrow C^n$ which is the actual clustering algorithm have been investigated in the

literature. However, as they do not present enough skills to deal with alarm data, we introduce two new complementary algorithms in the following section.

Hypothesis space H : It is the space where a particular clustering algorithm searches for a solution. H depends on the general tendency or preference of a particular algorithm. An evaluation function $\theta : H \rightarrow \mathbb{R}^+$ can work purely based on the feature and or similarity space, but can also incorporate external knowledge such as user given categorization.

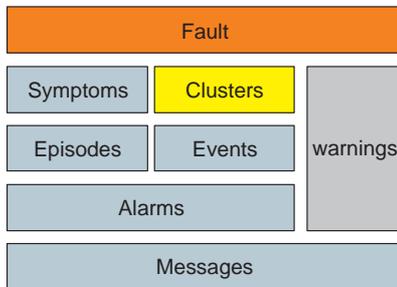


Fig. 1. Notions used in the model.

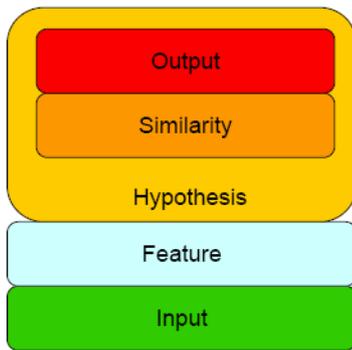


Fig. 2. Formal spaces for the fault recognition model.

We have now introduced the key notions of our model, it is now time to explain in detail the problem we try to solve. As we said, the main goal is to present to the operator a small number of alarms, highly representative of the faults which appeared in the network. To do so, we have to find the correlations rules for the alarms. These rules are most of the time unknown to the operator, who identify the characteristic symptoms of the fault and then, the fault itself, according to his knowledge about the network, and the history of fault recovery. When a fault appears, the different symptoms which follow are a cascade of errors. Errors are not directly viewable, but if we retrieve good correlation rules we would be able to identify the events which are the external manifestation of the errors. And this, according to a degree of trust, whose limit is specified by the user. The first step consists into the identification of the errors which follow the fault appearance. As we have just a limited view of what is going on in the network, we can try to approximate the errors by identifying the events of alarms. Then a correlation between these events must be done, to recognize dependent

and independent events and then present to the operator some plausible sets of events which can represent the faults with a strong degree of trust.

V. EXPERIMENTATIONS

This model has been used to build a technique called Behavioural Proximity Discovery (BPD). The BPD technique has been implemented in a prototype named Network Alarm Correlation Software (NACS). Figure 3 presents the main components of the software. Such architecture is needed for

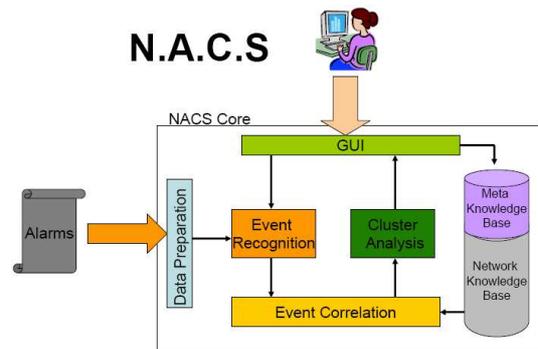


Fig. 3. The NACS Overview

a fast response when crucial faults occur in the network. The network knowledge base contains a set of all the plausible scenarios previously discovered in the data set, and is constantly updated according to the network user's choices. The meta knowledge base contains statistic descriptions of the scenarios embedded in the knowledge base and allows an easy access to the scenarios with targeted patterns. In this section, we present some results generated by the BDP technique using several data samples produced by a modern telecommunication network. The samples DS1 represent the activities of 6 main areas of a simulated network, each of them containing 5 cells, during a time window of 2 hours. 4 alarm types are used with different values of importance. We introduce some randomly generated perturbations to the simulated network for each data set. The sample DS2 and DS3 represent the activity of an Italian telecommunication network for a 24 hours period. Figure 4 shows the results generated by BDP with the data set DS1. 7 clusters are identified as being the main faults which appeared in the network during the time window of 2 hours. Figure 5 shows 5 independent clusters, generated from DS2. Figure 6 represents a dense alarm activity from DS3, which makes clusters hard to visualize. Therefore Figure 7 shows a close up on these 5 clusters with a circular display.

VI. CONCLUSIONS

In this paper we present a formal model for fault recognition in modern telecommunication networks. This model is the basis for any fault recognition technique which need to cope with the complexity of alarm relationships. It provides a general framework, defining the different notions and the

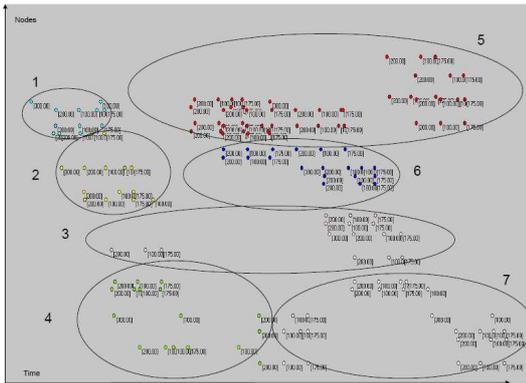


Fig. 4. Results generated with DS1.

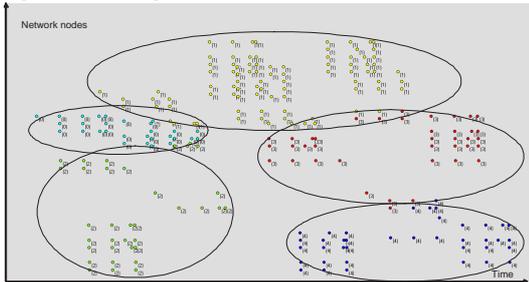


Fig. 5. Results generated with DS2.

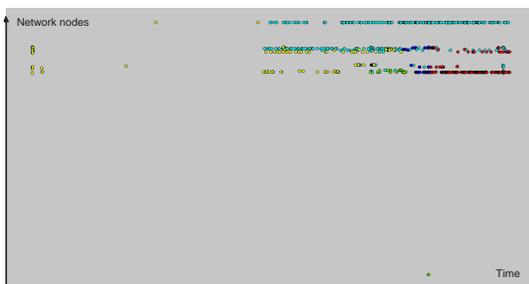


Fig. 6. Results generated with DS3.

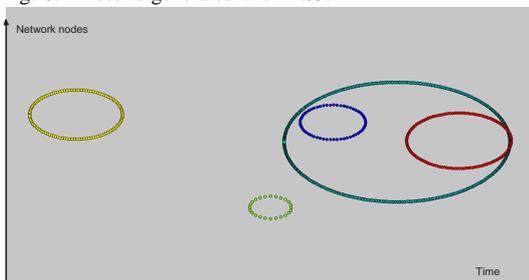


Fig. 7. Results generated with DS3, circular display.

articulations of an adaptive and scalable behaviour-based approach. The benefits are a high efficiency of fault detection since diagnosis knowledge is encoded in a behaviour representation, a powerful and evolutive representation of the main probable faults, a low demanding system on human intervention and network topology knowledge, a real time reaction to face of the alarm storms and to keep the network safer. This model has been implemented in a software called NACS to provide the main roots of faults which appeared in the network in the form of clusters. This model does not require any standard of knowledge on the nature of the networks and performs better than existing techniques when evaluated with real-world data sets.

REFERENCES

- [1] K. Yamanishi and Y. Maruyama, "Dynamic syslog mining for network failure monitoring," in *11th ACM SIGKDD int. conf. on Knowledge discovery in data mining, (KDD '05)*. New York, NY, USA: ACM Press, 2005, pp. 499–508.
- [2] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 4, pp. 443–471, 2003.
- [3] R. Brunet, "Temporal alarm correlation in communication networks," Master's thesis, Faculty of Engineering, Carleton University, 1998.
- [4] T. Li, F. Liang, S. Ma, and W. Peng, "An integrated framework on mining logs files for computing system management," in *11th ACM SIGKDD int. conf. on Knowledge discovery in data mining*, Chicago, Illinois, USA, May. 9-12 2005.
- [5] M. Möller, S. Tretter, and B. Fink, "Intelligent filtering in network management systems," in *Integrated Network Management*, 1995, pp. 304–315.
- [6] R. Vaarandi, "Sec - a lightweight event correlation tool," in *IEEE workshop in IP operations and management*, 2002, pp. 111–115.
- [7] P. Fröhlich, W. Nejd, K. Jobmann, and H. Wietgreffe, "Model-based alarm correlation in cellular phone networks," in *5th Int. Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'97)*, Haifa, Israel, Jan.12-15 1997, pp. 197–204.
- [8] Y. Pencolé and M.-O. Cordier, "A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks," *Artificial Intelligence.*, vol. 164, no. 1-2, pp. 121–170, 2005.
- [9] G. Jakobson and M. Weismann, "Real time telecommunication network management: extending event correlation with temporal constraints," in *4th int. symposium on integrated network management*, Santa Barbara, CA, USA, 1995, pp. 290–301.
- [10] K. Hatonen, M. Klemettinen, H. Mannila, P. Ronkainen, and H. Toivonen, "Tasa: Telecommunication alarm sequence analyser or how to enjoy faults in your network," in *IEEE/IFIP Network Operations and Management Symposium (NOMS'96)*, Kyoto Japan, April 1996, pp. 520–529.
- [11] E. Marilly, A. Aghasaryan, S. Betge-Brezetz, O. Martinot, and G. Delegue, "Alarm correlation for complex telecommunication networks using neural networks and signal processing," in *IP Operations and Management, 2002 IEEE Workshop*, 2002, pp. 3–7.
- [12] R. Sasisekharan, V. Seshadri, and S. Weiss, "Data mining and forecasting in large-scale telecommunication networks," *IEEE expert*, vol. 11, pp. 37–43, 1996.
- [13] C.-S. Chao and A.-C. Liu, "An alarm management framework for automated network fault identification," *Computer Communication*, vol. 27, pp. 1341–1353, april 2004.
- [14] C. Dousson, "Alarm driven supervision for telecommunication network: On-line chronicle recognition," *Annales des Telecommunications*, vol. 51, pp. 501–508, sept/oct 1996.
- [15] R. Gardner and D. Harle, "Alarm correlation and network fault resolution using kohonen self-organising map," in *IEEE Global Telecom. Conf.*, vol. 3, New York, NY, USA, 1997, pp. 1398–1402.

- [16] M. Klemettinen, H. Mannila, and H. Toivonen, "A data mining methodology and its application to semi-automatic knowledge acquisition," in *8th Int. Workshop on database & expert systems applications (DEXA'97)*, Toulouse, France, Sept 1-5 1997.
- [17] R. Gardner and D. Harle, "Methods and systems for alarm correlation," in *IEEE GLOBECOM '96*, London, UK, Nov. 18-22 1996, pp. 136-140.
- [18] G. Jakobson and M. Weismann, "Alarm correlation," *IEEE Network*, vol. 7, no. 6, 1993.
- [19] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: An efficient data clustering method for very large databases," in *ACM Intl. Conf. on management of data*, 1996, pp. 103-114.
- [20] H. Mannila, H. Toivonen, and A. Verkamo, "Discovery of frequent episodes in events sequences," *Data mining and knowledge Discovery*, vol. 1, pp. 259-286, 1997.
- [21] S. Kumar, "Modular learning through outspace decomposition," Ph.D. dissertation, The university of texas at austin, 2000.
- [22] A. Strehl, "Relationship-based clustering and cluster ensembles for high-dimensional data mining," Ph.D. dissertation, The University of Texas, 2002.

Evaluating Algorithms for Concept Description

Cecilia Sönströd, Ulf Johansson and Tuve Löfström

Abstract—When performing concept description, models need to be evaluated both on accuracy and comprehensibility. A comprehensible concept description model should present the most important relationships in the data in an accurate and understandable way. Two natural representations for this are decision trees and decision lists. In this study, the two decision list algorithms RIPPER and Chipper, and the decision tree algorithm C4.5, are evaluated for concept description, using publicly available datasets. The experiments show that C4.5 performs very well regarding accuracy and brevity, i.e. the ability to classify instances with few tests, but also produces large models that are hard to survey and contain many extremely specific rules, thus not being good concept descriptions. The decision list algorithms perform reasonably well on accuracy, and are mostly able to produce small models with relatively good predictive performance. Regarding brevity, Chipper is better than RIPPER, using on average fewer conditions to classify an instance. RIPPER, on the other hand, excels in relevance, i.e. the ability to capture a large number of instances with every rule.

I. INTRODUCTION

IN the data mining task concept description [1], the aim is to gain insights. The focus is not to produce models with high predictive accuracy, but to adequately describe the most important relationships in the data. Recommended techniques for this task are rule induction and conceptual clustering. In many concept description situations, what is actually needed is a highly understandable classification model, indicating that rule induction is most suitable. However, most rule induction algorithms focus on optimizing predictive performance, i.e. accuracy, often at the expense of comprehensibility, and few, if any, include direct ways for the user to control the tradeoff between accuracy and comprehensibility. Of course, the term comprehensibility is far from unproblematic, since many factors influence whether a model is understandable or not. Often, model size is used to estimate comprehensibility, with the implicit assumption that small models are easier to interpret.

In [2], we proposed that concept description models should be evaluated using accuracy and three properties that capture different aspects of comprehensibility. Hence, the four criteria for evaluating concept description models become:

- Accuracy: The model should have high accuracy on

unseen data to guarantee that relationships found hold in general

- Brevity: The model should classify as many instances as possible with few conditions
- Interpretability: The model should express conditions in a clear and readable way
- Relevance: Only those relationships that are general and interesting should be included in the model

The most common representation language for transparent models is decision trees, and many successful decision tree algorithms exist. Looking at the above criteria, it is clear that standard decision tree algorithms such as C4.5 [3] and CART [4] are capable of fulfilling the accuracy criterion. Intuitively, the decision tree representation also seems suitable for obtaining good brevity, since a balanced decision tree will make many different classifications with relatively few conditions used for each path to a leaf node. However, regarding the two other criteria, decision trees have some serious drawbacks, since the representation leads to models that are hard to survey, and typically also contain many branches that classify only a few instances. Looking at a decision tree, especially in its textual representation, a decision-maker will have a hard time finding and interpreting the most important relationships; indeed, he would probably trace the sequence of tests and write out a separate rule, consisting of a conjunction of tests, for branches of interest.

The above example is an argument for the alternative representation decision lists, or ordered rule sets. A decision list is, in essence, a maximally unbalanced decision tree, where each rule, containing one or more tests, directly classifies a number of instances. Those instances not covered by one rule are tested on the next rule, and this proceeds until a default rule classifies all remaining instances. This representation is especially suitable for concept description, since it admits a rule construction method prioritizing high coverage for the top rules, thereby obtaining good brevity and relevance by capturing the most general relationships in the dataset with very few conditions; or, put in another way, describing the most important relationships with very simple rules. Furthermore, since the top rules in a decision list are very easy to identify, a decision list algorithm with high coverage in its top rules will also have good interpretability. Finally, the default rule used in decision lists also provides a mechanism for increasing both interpretability and relevance, since models can avoid formulating rules that only classify a few instances.

Ultimately, a decision list algorithm suitable for concept description should consistently obtain accuracy and brevity comparable to standard decision tree algorithms, with its representation offering benefits for interpretability and relevance. It must be noted that this is a tough challenge, since industrial-strength decision trees are very good at optimizing accuracy and has a representation that lends itself very well to good brevity

II. BACKGROUND

Most decision list algorithms are based on the sequential covering algorithm, which in each step constructs a rule covering a number of instances and then removes those instances from the dataset before the next rule is found. This procedure is repeated until some stop criterion is met, when a default rule, classifying all remaining instances, is formulated. Examples of decision list algorithms based on this idea are AQ [5], CN2 [6], IREP[7] and RIPPER [8].

RIPPER (*Repeated Incremental Pruning to Produce Error Reduction*) is widely used and considered to be the most successful decision list algorithm to date [9], and is reported to have good performance regarding both accuracy [10], running time and ability to handle noise and unbalanced data sets [11]. RIPPER works by constructing rules only for the minority class(es) and using the default rule for the majority class. For multiclass problems, rules are constructed for classes in order of class distribution. In [9], most of RIPPER's power is attributed to its optimization (post-pruning) procedure and the authors argue that post-pruning is probably the best way to obtain good predictive performance for decision list algorithms.

However, some of the properties that give RIPPER good performance regarding accuracy and speed are detrimental to its concept description ability. First, and foremost, for binary problems with relatively even class distributions, rules are only formulated for the class that happens to have the lowest number of instances, and hence no explicit description (other than as the negation of the rules for the minority class) is offered for instances belonging to the other class. This will also often result in RIPPER obtaining relatively poor brevity even for quite short rule sets. Furthermore, RIPPER is built to optimize accuracy and will sometimes do so at the expense of comprehensibility, without any clear possibility for the user to control this tradeoff via parameter settings.

We have previously, see [12], introduced the decision list algorithm Chipper, aimed at performing concept description. Chipper is a greedy algorithm based on sequential covering, and the basic idea is to, in each step, find the rule that classifies the maximum number of instances using only one split on one attribute. The algorithm uses a parameter, called *ignore*, to control the tradeoff between accuracy and coverage. This parameter specifies the acceptable misclassification rate for each rule, expressed either as an absolute number of instances or as a proportion of remaining instances in the data set. The *ignore* parameter can thus be

used to control the granularity of the rules produced, with high *ignore* values being used for finding rules with high coverage (possibly with lower rule accuracy) and low values for more specific rules. The other main parameter in Chipper is called *stop* and is used to determine the proportion of instances that should be described by rules before the default rule is formulated. Thus, a *stop* value of 80% means that the default rule will be formed when at least 80% of the instances are covered by the rule set.

In [12], Chipper was evaluated on 9 binary datasets from the UCI machine learning repository [13], which were chosen to enable interpretation of rule sets found, and compared to two standard algorithms for generating transparent models; RIPPER and C4.5. Results were very encouraging, with Chipper obtaining accuracy comparable to other techniques, but having superior comprehensibility.

However, it was noted that Chipper sometimes performed very badly regarding accuracy and also was very sensitive to the value of the *ignore* parameter. This is, of course, not entirely a bad thing, since it means that the parameter works as intended, letting the user choose the granularity of the rule set. In some cases, though, it is of course desirable to have the option of finding "the best possible" model without having to manually search for optimal parameter settings.

The main purpose of this study is to conduct a more thorough evaluation of both a slightly improved version of Chipper, but also RIPPER and C4.5 regarding suitability for concept description. It is of course interesting to investigate how existing algorithms perform regarding brevity and relevance, especially for decision trees which are not normally evaluated on these criteria.

III. METHOD

The main change in Chipper from [12] is that the algorithm is able to handle multiclass problems and is now implemented in Java and incorporated in the WEKA [9] data mining framework. Implementation in WEKA has facilitated solving the problem with excessive sensitivity to the value of the *ignore* parameter, by using the built-in meta procedure for cross-validation parameter selection (CVParameterSelection). CVParameterSelection uses internal cross-validation to find the best values for one or more parameters within a specified range. Obviously, most standard techniques, such as RIPPER and C4.5, use an internal cross-validation procedure as a part of their algorithm to optimize models on accuracy. Using this procedure to set parameter values in Chipper gives the desired flexibility regarding parameter use, since the user can either specify a given range or set specific values.

Finally, using techniques all implemented in the same framework means an improved ability to carry out comparisons by using controlled experiments with fixed folds for all techniques.

A. Datasets

26 publicly available dataset from the UCI repository [13] were used for the experiments. As seen in Table 1 below, where the dataset characteristics are presented, both binary and multiclass problems are used, and the datasets have different properties regarding number of instances, as well as the number of continuous (*Cont.*) and categorical (*Cat.*) attributes.

TABLE 1. DATA SETS USED

Data set	Instances	Classes	Cont.	Cat.
Breast cancer	286	2	0	9
Cmc	1473	3	2	7
Colic – Horse	368	2	7	15
Credit – American	690	2	6	9
Credit – German	1000	2	7	13
Cylinder	540	2	18	21
Diabetes – Pima	768	2	8	0
E-coli	336	8	7	0
Glass	214	6	9	0
Haberman	306	2	2	1
Heart – Cleveland	303	2	6	7
Heart – Statlog	270	2	10	3
Hepatitis	155	2	6	13
Ionosphere	351	2	34	0
Iris	150	3	4	0
Labor	57	2	8	8
Liver disorders – BUPA	345	2	6	0
Lymphography	148	4	3	15
Sick	3772	2	22	7
Sonar	208	2	60	0
Tae	151	3	3	2
Vehicle	846	4	18	0
Votes	435	2	0	16
Wine	178	3	13	0
WBC - Wisconsin	699	2	9	0
Zoo	101	7	16	1

B. Experiments

In the experiments, Chipper is compared to two techniques producing transparent models, but using different representations. The chosen decision tree algorithm is again C4.5 implemented as J48 in WEKA and the decision list technique is RIPPER, implemented as JRip in WEKA. The motivation for these choices is that the two algorithms are standard techniques for their respective representations.

In the first experiment, the aim is to investigate whether Chipper is able to obtain acceptable predictive performance, measured as accuracy and *area under the ROC-curve* (AUC), over a large number of datasets. The motivation for including both accuracy and AUC is that they measure quite different things; while accuracy is based only on the final classification, AUC measures the ability of the model to rank instances according to how likely they are to belong to a certain class; see e.g. [14]. AUC can be interpreted as the probability of ranking a true positive instance ahead of a false positive; see [15].

In this experiment, Chipper is used with parameter settings favoring accuracy, using `CVPParameterSelection` for

both for *stop* and *ignore*. *Ignore* had values between 0.5% and 5%, with 10 different values and *stop* was between 70% and 95%, with 6 different values. In this experiment, 10 x 10-fold cross-validation is used, with identical folding for all three techniques, for measuring accuracy and AUC. J48 and JRip were used with their default settings in WEKA, which means that J48 trees are pruned.

In the second experiment, the tradeoff between accuracy and comprehensibility is investigated by using Chipper with settings favoring comprehensible models, meaning lower *stop* values and higher *ignore* values. The choice was to use *stops* between 60% and 80%, and *ignores* between 4% and 8%. For this kind of rule set, where the aim is to present the most important relationships in the data, the evaluation should primarily be on comprehensibility, with overall model accuracy serving only to guarantee that the relationships found will hold in general.

C. Evaluation of comprehensibility

For measuring the brevity aspect of comprehensibility, we have previously, in [2], suggested the *classification complexity* (CC) measure. The classification complexity for a model is the average number of tests (i.e. simple conditions using only one attribute) needed to classify an instance. A low value thus means good brevity.

The interpretability and relevance aspects, however, have not as yet been formulated as numeric measures. The most problematic of these is arguably interpretability; in Chipper, interpretability is ensured by the very simple representation language, allowing only one test in each rule. For relevance, a measure would have to differentiate between models containing few and many rules, but also be more refined than just model size. A model should be deemed to have high relevance when every part of the model classifies a large number of instances. To make comparison between different representations possible, we introduce the term *classification point*, and define it as a place where instances are assigned class labels in a classification model. For a decision list, the classification points then consist of all rules, and for a decision tree, every leaf is a classification point. We propose that relevance can be measured by calculating the average number of instances that reach each classification point in a model. Thus, a high value will represent good relevance. This in essence, means calculating the load factor for each rule or leaf. This measure will simply be called *relevance* and is calculated using (1) below:

$$relevance = \frac{\#instances\ in\ dataset}{\#classification\ points} \quad (1)$$

Comprehensibility for the models produced in both experiments is consequently evaluated using classification complexity (CC), and the relevance measure introduced above.

IV. RESULTS

The results regarding accuracy and AUC from Experiment 1 are shown in Table 2 below. Bold numbers indicate the best result for a specific dataset.

TABLE 2. RESULTS ACCURACY AND AUC, 10X10CV

Data set	J48		JRip		Chipper	
	Acc	AUC	Acc	AUC	Acc	AUC
Breast-cancer	70.5%	0.59	71.5%	0.60	67.8%	0.54
Cmc	52.3%	0.69	52.4%	0.64	48.2%	0.66
Colic	85.3%	0.85	85.1%	0.83	77.6%	0.78
Credit - A	85.2%	0.87	85.2%	0.87	85.1%	0.91
Credit - G	70.6%	0.64	72.2%	0.63	70.1%	0.65
Cylinder	73.2%	0.76	67.1%	0.66	66.9%	0.71
Diabetes	74.5%	0.75	75.2%	0.72	76.0%	0.79
E-coli	82.8%	0.96	81.4%	0.95	71.1%	0.95
Glass	67.6%	0.79	66.8%	0.80	61.5%	0.73
Haberman	72.2%	0.58	72.4%	0.60	76.3%	0.61
Heart - C	78.2%	0.78	80.0%	0.81	71.2%	0.74
Heart - S	78.2%	0.79	78.8%	0.80	72.0%	0.73
Hepatitis	79.2%	0.67	78.1%	0.62	80.8%	0.74
Ionosphere	89.7%	0.89	89.2%	0.89	87.2%	0.85
Iris	94.7%	0.99	93.9%	0.99	93.1%	0.99
Labor	78.4%	0.72	83.7%	0.82	78.9%	0.73
Liver	65.8%	0.65	66.6%	0.65	63.2%	0.63
Lymph	75.6%	0.77	76.3%	0.40	79.2%	0.46
Sick	98.9%	0.96	98.3%	0.94	96.5%	0.94
Sonar	73.6%	0.75	73.4%	0.75	75.5%	0.78
Tae	57.4%	0.75	43.8%	0.60	42.1%	0.60
Vehicle	72.3%	0.76	68.3%	0.77	59.9%	0.73
Vote	96.6%	0.98	95.8%	0.96	94.7%	0.97
Wine	93.2%	0.97	93.1%	0.96	91.9%	0.97
WBC	95.0%	0.96	95.6%	0.96	93.8%	0.95
Zoo	91.6%	1.00	86.6%	0.93	87.9%	1.00
#Wins	13	14	9	10	5	9

As can be seen from the table, J48 performs best overall, both regarding accuracy and AUC. J48 obtains the highest accuracy on 13 out of 25 datasets, and JRip performs slightly better than Chipper. However, for some datasets, there are significant differences in accuracy, with Chipper quite often losing by a large margin. This is probably due to there being no global rule set optimization targeting accuracy in Chipper, whereas the two other techniques spend quite a lot of effort on post-pruning and optimizing their models on accuracy, using internal cross-validation.

When measuring predictive performance using AUC, J48 still stands out, winning 14 datasets, albeit 5 of them drawn with other techniques. In contrast to accuracy, there is very little difference between Chipper and JRip on the AUC measure.

The comprehensibility results from Experiment 1 are shown in Table 3 below. Classification complexity (CC) is given for the number of tests, *size* is measured simply as the total number of tests in the rule set, serving as an indicator of overall model complexity, and relevance (*Rel*) is calculated according to (1) above. All these measures are given for the model constructed over the entire dataset that WEKA outputs.

When calculating CC, everything using more than 30

conditions is lumped together as if it were in the default rule; the only dataset where this happens is Vehicle for both JRip and Chipper. Obviously, J48 has an advantage for the CC measure, because of its different representation.

TABLE 3. COMPREHENSIBILITY RESULTS FOR EXPERIMENT 1

Data set	J48			JRip			Chipper		
	Size	CC	Rel	Size	CC	Rel	Size	CC	Rel
Breast-c	21	5.1	13	4	4.6	95	8	7.8	32
Cmc	131	8.9	11	14	12.6	295	13	10.0	105
Colic	5	1.7	61	6	5.4	92	13	5.0	26
Credit - A	24	3.8	28	7	6.0	173	7	2.4	86
Credit - G	98	9.2	10	11	5.3	333	8	5.1	111
Cylinder	66	23	8	6	6.0	180	22	9.9	23
Diabetes	19	3.7	38	9	8.0	192	10	3.8	70
E-coli	21	4.4	15	19	16.7	34	13	4.8	24
Glass	29	5.9	7	18	14.9	31	25	13.1	8
Haberman	10	2.3	28	3	3.9	153	12	7.1	24
Heart - C	17	3.6	17	6	5.3	76	11	6.7	25
Heart - S	17	3.0	15	8	6.7	54	7	4.2	34
Hepatitis	10	2.9	14	5	5.3	39	6	3.2	22
Ionosphere	17	5.8	20	2	2.4	117	13	7.7	25
Iris	4	2.1	30	3	2.3	38	3	2.2	38
Labor	6	2.7	8	4	3.7	19	5	3.2	10
Liver	25	4.7	13	4	4.2	115	13	7.7	25
Lymph	13	5.0	11	8	7.5	25	13	5.3	11
Sick	22	3.2	164	10	9.5	943	2	1.4	1257
Sonar	17	4.4	12	9	7.6	42	5	3.2	35
Tae	33	6.4	4	1	1.8	76	20	9.0	7
Vehicle	97	7.2	9	43	23.1	53	49	18.3	17
Vote	5	1.6	73	6	5.2	145	3	1.6	109
Wine	4	2.3	36	4	3.9	60	5	2.5	30
WBC	13	2.7	50	9	7.4	140	4	1.9	140
Zoo	8	2.8	11	6	5.8	20	6	2.7	14
#Wins	2	15	0	17	5	25	10	7	4
Average	28.2	4.9		8.7	7.1		11.4	5.8	

As can be seen from the table, J48 is able to use its representation to obtain very good classification complexity, despite having by far the largest average model size. However, J48 performs very badly on relevance, not winning a single dataset. This, together with the very good classification complexity, indicates that the decision trees produced by J48 contain a lot of branches that classify very few instances each, which means that the overall comprehensibility of the model is limited.

Turning to a direct comparison between Chipper and RIPPER, the results are very clear. Chipper is superior to RIPPER on classification complexity, performing better on 19 out of 26 datasets and also obtaining a lower average number of tests. It is, of course, slightly iffy to calculate an average like this, but what this average says is that over all datasets, Chipper needs just under six tests to classify an instance. JRip performs best on relevance, winning all but one datasets, which of course is a consequence of the smaller model size. Indeed, RIPPER rule sets typically consist of between 3 and 4 rules and seldom contain rules classifying only a few instances.

Below are some sample rule sets for Chipper and JRip, where the differences between how the two techniques work become very apparent.

```

IF Color_intensity <= 3.4 THEN 1 [55/0]
IF Proline >= 885.0 THEN 0 [49/0]
IF Flavanoids <= 1.22 THEN 2 [42/0]
IF Magnesium <= 94.0 THEN 1 [11/0]
IF Proline >= 680.0 THEN 0 [10/0]
DEFAULT: 2 [11/5]

```

Figure 1: Chipper sample rule set for Wine

```

IF Flavanoids <= 1.39 AND
  Color_intensity >= 3.85 THEN 3 [47/0]
IF Proline >= 760 AND
  Color_intensity >= 3.52 THEN 1 [57/0]
DEFAULT: 2 [74/3]

```

Figure 2: JRip sample rule set for Wine

```

IF FATIGUE == 0.0 THEN LIVE [54/2]
IF PROTIME >= 51.0 THEN LIVE [34/1]
IF ALBUMIN >= 4.1 THEN LIVE [12/1]
IF BILIRUBIN >= 3.5 THEN DIE [8/1]
IF SEX == 0.0 THEN LIVE [7/0]
IF ALBUMIN <= 2.6 THEN DIE [5/0]
DEFAULT: LIVE [35/16]

```

Figure 3: Chipper sample rule set for Hepatitis

```

IF ALBUMIN <= 3.8 AND
  ALBUMIN <= 2.8 THEN DIE [13/2]
IF PROTIME <= 42 THEN DIE [15/7]
IF SPIDERS = yes AND
  BILIRUBIN >= 2 THEN DIE [11/4]
DEFAULT: LIVE [116/6]

```

Figure 4: JRip sample rule set for Hepatitis

In the Hepatitis rule set, Chipper's emphasis on brevity is very clear, with top rules covering many more instances than in the corresponding JRip rule set. For a concept description task, this is clearly a desirable property. For Wine, JRip builds just one rule for each class, obtaining very good relevance, but slightly worse classification complexity than Chipper, due to having the top rule for the minority class and using multiple conditions in both rules.

The results from Experiment 2, where Chipper is set to optimize comprehensibility, are shown in Table 4 below. For comparison between the two settings, Chipper accuracies from Experiment 1 are included in the table.

TABLE 4. CHIPPER ACCURACY AND SIZE RESULTS FOR EXPERIMENT 2

Data set	Chipper Exp 1		Chipper Exp 2	
	Acc	Size	Acc	Size
Breast-cancer	67.9%	8	68.7%	5
Cmc	48.2%	13	51.1%	12
Colic	77.7%	13	76.8%	2
Credit - A	85.7%	7	85.3%	2
Credit - G	70.1%	8	71.0%	7
Cylinder	69.0%	22	68.2%	16
Diabetes	76.7%	10	73.1%	4
E-coli	71.7%	13	67.3%	9
Glass	60.3%	25	62.6%	17
Haberman	74.3%	12	74.9%	9
Heart - C	71.1%	11	73.5%	4
Heart - S	71.6%	7	69.5%	7
Hepatitis	82.6%	6	82.2%	6
Ionosphere	87.4%	13	87.9%	2
Iris	92.7%	3	88.7%	3
Labor	80.6%	5	74.2%	2
Liver	63.4%	13	61.7%	9
Lymph	80.1%	13	74.1%	9
Sick	97.8%	2	94.5%	2
Sonar	76.7%	5	65.2%	2
Tae	42.5%	20	48.6%	7
Vehicle	60.7%	49	57.2%	5
Vote	95.1%	3	95.8%	2
Wine	94.1%	5	88.3%	3
WBC	95.0%	4	91.3%	1
Zoo	90.9%	6	88.7%	4
#Wins	17	4	9	26
Average		11.4		5.8

Here, there are several striking results. First of all, Chipper actually obtains higher accuracy on 9 datasets with settings prioritizing short and clear rule sets. This is obviously an indication that there maybe is a problem with over-fitting when using the settings optimizing accuracy. It is also notable, but not surprising, that a considerable loss of accuracy occurs for datasets where accuracy is very high, see e.g. the WBC, Wine and Iris datasets, since Chipper is now allowed to formulate much less accurate rules.

Regarding rule set size, rules are on average much smaller for these Chipper settings; indeed, rule set size is decreased for all but 4 datasets. A really good example of how a considerable reduction in rule set size only leads to a relatively small loss of accuracy is seen in the Vehicle dataset, where an almost incomprehensible model consisting of 49 rules is replaced by a much more comprehensible 5-rule model, with only a small loss in accuracy.

```

IF HOLLOWES RATIO <= 186
  THEN 2 [147/60]
IF SCALED VARIANCE_MINOR <= 310
  THEN 3 [133/54]
IF DISTANCE CIRCULARITY <= 76
  THEN 2 [125/40]
IF ELONGATEDNESS >= 42
  THEN 3 [111/29]
IF MAX.LENGTH RECTANGULARITY >= 169
  THEN 0 [77/26]
DEFAULT: 1 [253/132]

```

Figure 5: Sample Chipper rule set for Vehicle

The comprehensibility measures for Experiment 2 are given below in Table 5. Again, Chipper and JRip results from Experiment 1 are included for reference.

TABLE 5. BREVITY AND RELEVANCE RESULTS FOR EXPERIMENT 2

Data set	JRip		Chipper Experiment 1		Chipper Experiment 2	
	CC	Rel	CC	Rel	CC	Rel
Breast-cancer	4.6	95	7.8	32	3.6	48
Cmc	12.6	295	10.0	105	8.6	113
Colic	5.4	92	5.0	26	1.7	123
Credit - A	6.0	173	2.4	86	1.8	230
Credit - G	5.3	333	5.1	111	3.7	125
Cylinder	6.0	180	9.9	23	8.4	32
Diabetes	8.0	192	3.8	70	2.7	154
E-coli	16.7	34	4.8	24	4.6	34
Glass	14.9	31	13.1	8	10.0	12
Haberman	3.9	153	7.1	24	5.6	31
Heart - C	5.3	76	6.7	25	2.8	61
Heart - S	6.7	54	4.2	34	4.2	34
Hepatitis	5.3	39	3.2	22	3.2	22
Ionosphere	2.4	117	7.7	25	1.9	117
Iris	2.3	38	2.2	38	2.0	38
Labor	3.7	19	3.2	10	2.0	19
Liver	4.2	115	7.7	25	6.0	35
Lymph	7.5	25	5.3	11	6.1	15
Sick	9.5	943	1.4	1257	1.3	1257
Sonar	7.6	42	3.2	35	2.0	70
Tae	1.8	76	9.0	7	4.4	19
Vehicle	23.1	53	18.3	17	3.7	141
Vote	5.2	145	1.6	109	1.5	145
Wine	3.9	60	2.5	30	2.1	45
WBC	7.4	140	1.9	140	1.3	350
Zoo	5.8	20	2.7	14	2.5	20
#Wins	3	20	3	1	22	12
Average	7.1		5.8		3.8	

The overall picture in this experiment is that the reduction in size carries over to classification complexity, i.e. that these shorter rules significantly improve the ability to classify a majority of instances with few conditions. The same holds for relevance, where Chipper is now significantly better, but still not as good as RIPPER.

Below are some sample Chipper rules from Experiment 2, once again illustrating that rules have good brevity, relevance and interpretability.

```
IF plas <= 107.0 THEN 0 [289/36]
IF plas >= 156.0 THEN 1 [117/23]
IF mass <= 28.2 THEN 0 [102/18]
IF preg >= 8.0 THEN 1 [44/11]
DEFAULT: 0.0 [216/87]
```

Figure 6: Sample Chipper rule set for Diabetes

```
IF physician-fee-freeze == 0
  THEN 0 [247/2]
IF synfuels-corporation-cutback == 0
  THEN 1 [139/3]
DEFAULT: 1 [49/19]
```

Figure 7: Sample Chipper rule set for Vote

V. CONCLUSIONS

The results show that there is a definite tradeoff between accuracy and comprehensibility when performing concept description, with no technique or representation being superior in all areas. C4.5 produces very high predictive performance, and also uses its tree representation to obtain very good classification complexity, which means that C4.5 trees are accurate with good brevity. However, these models are often quite big and contain many relationships that only hold for a few instances, and which are of very limited value in a concept description task. Indeed, these spurious components may even obscure the important relationships found in the data by making the model hard to survey and interpret.

RIPPER performs quite well on accuracy and also manages to use its representation with conjunctive rules to obtain very good relevance, with typical models containing few rules, each with high coverage. Unfortunately, RIPPER is inherently unable to really perform well on brevity, i.e. classification complexity, since rules are only formulated for the minority classes. Although this is a clear advantage for unbalanced problems where the minority class is the one of interest, it is in general not suitable for concept description.

Chipper, finally, manages to obtain competitive accuracy on some datasets, even though it does not contain any post-processing of rule to optimize accuracy on the whole model. Chipper clearly outperforms RIPPER on brevity, especially when using parameter settings favoring comprehensibility, which reduce both model size and classification complexity, without significant loss of accuracy. In fact, over all datasets, Chipper models in Experiment 2 use only 4 conditions on average to classify an instance.

Overall, the conclusion is that both decision trees and decision lists have some properties that are suitable for concept description, but that no algorithm performs well on all evaluated criteria. Clearly, some work remains to handle the tradeoff between predictive performance and different aspects of comprehensibility when performing concept description.

VI. DISCUSSION AND FUTURE WORK

Since Chipper, although designed for concept description, has proved capable of sometimes obtaining accuracy on par with techniques aimed at predictive performance, it would of course be interesting to see if Chipper could be extended with the aim of increasing predictive accuracy, but keeping the core idea of greedy search for rules with high coverage. It is then natural to extend the representation language to include intervals, conjunctions and disjunctions. An argument for this was seen in [16], where Chipper was evaluated on a medical dataset and where it became apparent that the representation language, allowing only a single condition per rule, was too simple to capture some of the

important relationships in the dataset. Another possibility, when aiming to increase predictive power, is to use different rule selection criteria, more capable of handling the tradeoff between coverage and accuracy. Obviously, when seeking to improve predictive accuracy, the issue of over-fitting must be handled, especially since the present study indicates that this is already a problem when Chipper is used with prediction settings.

There are of course many possible extensions that would improve concept description ability. It could perhaps be argued that the current *ignore* parameter, although having a very marked effect on rules produced, is not all that intuitive. Since it sort of depends on the number of instances in the dataset, it is quite hard to predict what kind of rule accuracy a specific *ignore* value will produce. For decision-makers, a parameter formulated directly in terms of rule accuracy would be easier to use efficiently, especially since domain constraints (such as cost of false positives and negatives) often can be translated into rule accuracy. Obviously, in this context, rule selection could be based on precision and/or recall instead of accuracy, with the ability to prioritize one class.

To a lesser extent, the non-intuitive reservation also applies to the *stop* parameter. An alternative way of formulating this would be that the user could just specify the desired number of rules before the default rule is applied.

When looking at the results and rule sets from the Sick dataset, which is extremely unbalanced with 3772 instances and the class distribution of 3541/231, it is obvious that RIPPER's approach of finding rules only for the minority class and then applying the default rule to classify the remaining instances works extremely well. With such dataset, the minority class will also almost always be the one that decision-makers are interested in obtaining a concept description of. Hence, a natural extension to Chipper is to let the user determine explicitly which class(es) should be described by rules and which class(es) should only be captured by the default rule.

REFERENCES

- [1] The CRISP-DM Consortium, CRISP-DM 1.0, www.crisp-dm.org, 2000.
- [2] C. Sönströd, U. Johansson and R. König, "Towards a Unified View on Concept Description", *The 2007 International Conference on Data Mining (DMIN'07)*, Las Vegas, NV, 2007.
- [3] J. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufman, 1993.
- [4] L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, 1984.
- [5] J. Hong, I. Mozetic, and R.S. Michalski, "AQ15: Incremental learning of attribute-based descriptions from examples, the method and user's guide", Report ISG 85-5, UIUCDCS-F-86-949, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1986.
- [6] P. Clark and T. Niblett, "The CN2 induction algorithm", *Machine Learning*, 3:261-283, 1989.
- [7] J. Fürnkranz and G. Widmer, "Incremental Reduced Error Pruning", *Proceedings of the 11th International Conference on Machine Learning (ICML'94)*, p.70-77, San Mateo, CA, 1994.
- [8] W. Cohen, "Fast Effective Rule Induction", *Proceedings of the 12th International Conference on Machine Learning (ICML'95)*, p. 115-123. Tahoe City, CA, 1995.
- [9] J. Fürnkranz and P. Flach, "An Analysis of Stopping and Filtering Criteria for Rule Learning", *Proceedings of the 15th European Conference on Machine Learning*, 123-133, 2004.
- [10] I. H. Witten and E. Frank, *Data Mining – Machine Learning Tools and Techniques*, 2nd ed, Morgan Kaufman, 2005.
- [11] P. Tan, M. Steinbach and V. Kumar, *Introduction to Data Mining*, Pearson Education, 2006.
- [12] U. Johansson, C. Sönströd, T. Löfström and H. Boström, "Chipper – A Novel Algorithm for Concept Description", *10th Scandinavian Conference on Artificial Intelligence*, IOS Press, pp. 133-140, Stockholm, Sweden, 2008.
- [13] C. L. Blake and C. J. Merz, UCI Repository of Machine Learning Databases, University of California, Department of Information and Computer Science, 1998.
- [14] T. Fawcett, "Using rule sets to maximize roc performance", *15th International Conference on Machine Learning*, pp. 445-453, 2001
- [15] A. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms", *Pattern Recognition*, 30:1145-1159, 1997.
- [16] C. Sönströd, U. Johansson, U. Norinder, and H. Boström, "Comprehensible Models for Predicting Molecular Interaction with Heart-Regulating Genes", *7th International Conference on Machine Learning and Applications (ICMLA '08)*, Orlando, FL, IEEE press, pp. 559 – 564, 2008.

Greedy Learning: Using Advantages of Distribution Functions to Improve Generalization of ANNs

Kristina Davoian and Wolfram-M. Lippe

Abstract—In this paper we analyze the benefits of utilizing different distribution functions in the learning strategy with respect to Artificial Neural Networks' (ANNs) generalization ability. We present a modification of the recently proposed network weight-based evolutionary algorithm (NWEA), by involving three distribution functions in mutation. Each parent in the population generates three offspring by applying Gaussian, Cauchy and uniform distributions. The utilization of the greedy learning implies that the advantages of the distribution functions in terms of step size will direct the algorithm towards good generalized ANNs.

Keywords: Neural Networks, Evolutionary Algorithms, Supervised Learning

I. INTRODUCTION

DESPITE the diversity of the evolutionary search methods applicable to Artificial Neural Networks' (ANNs) parameters optimization, there is no proof that any of them is beneficial in finding the appropriate ANNs for a various range of problem domains. Furthermore, some evolutionary approaches, such as genetic algorithms (GAs), are faced with internal difficulties (e.g. the permutation problem caused by crossover [1, 2]), which often complicate the evolution of good generalized ANNs. For this reason the *mutation-based self-adaptive strategies*, i.e. Evolutionary Programming (EP) [3-6] and Evolutionary Strategies (ES) [7-9], have found consideration in the theory of ANNs' learning.

The key aspects that EP and ES concentrate on are the self-adaptive methods for changing the strategy parameters and the distribution used in mutation. Both approaches utilize the standard normal distribution and similar self-adaptive methods¹. Later, Yao *et al.* [14] established that the distribution in the mutation strategy is crucial in the determination of the mutation step size, and proposed a novel EP technique, called the Fast Evolutionary Programming (FEP) [15, 16], which adopts the self-adaptation strategy of the classical EP (CEP), but uses the Cauchy distribution instead of the Gaussian one. Further, Yao [17] introduced the combinations of CEP and FEP approaches referred to as the Improved Fast Evolutionary

Programming (IFEP) and the Mixed Fast Evolutionary Programming (MEP). The IFEP and MEP aim at mixing the different search biases of Gaussian and Cauchy mutations at the chromosome (IFEP) and the gene (MEP) levels. Both techniques belong to the greedy algorithms: they generate two offspring from each parent, one by Gaussian mutation and the other by Cauchy mutation, and select the fittest individual as the surviving offspring.

This work is devoted to the improvement of the learning in ANNs. We introduce a modification of the early proposed network weight-based EA (NWEA) strategy [18], referred further to as greedy NWEA (GNWEA), which is advanced with the idea of greedy selection. In contrast to the CEP and FEP approaches, which are independent search methods, NWEA was developed specially for ANNs' learning. The main feature of NWEA consists in the special approach to the adjustment of the random values. The strategy parameter in the classical techniques is evolved during the evolution alongside with the object parameters. In comparison to that, the adaptation strategy in NWEA consists of two components, which bear the information about the position of an individual in the search space and based of this knowledge, bias the improvement towards the perspective regions of the search space. In this paper NWEA was extended with the using of three distribution functions: Gaussian, Cauchy and uniform. The algorithm generates three offspring from each parent. All offspring are kept till the $\lambda = 3 \times m$ (m is the size of the parental population) offspring are created, and take part in the forming of a new generation.

The goal for this work is to investigate the impact of the mixed search biases of three distributions on the generalization in ANNs. For this purpose, the preliminary experimental studies on the breast cancer and heart disease classification problems were provided. The generalization results of ANNs, evolved with GNWEA were compared with those, evolved with the NWEA.

The rest of the paper is organized as follows: Section II provides a brief overview of Artificial Neural Networks and the learning paradigm used in this paper. Section III discusses the well-known CEP and FEP learning algorithms and maintains the advantages of using Gaussian and Cauchy distributions to generate random values. Section IV described the main steps of the greedy learning algorithm based on the NWEA adaptation strategy. Section V describes the experiments and analyses the obtained results. Section VI concludes this paper.

K. Davoian and W.-M. Lippe are with the Department of Mathematics and Computer Science, University of Münster, Einsteinstr. 62, 48149 Münster, Germany (phone: +49-2518333796, fax: +49-2518333755, emails: kristina.davoian@uni-muenster.de, lippe@math.uni-muenster.de).

¹ The self-adaptive method for modifying the strategy parameters was introduced by Rechenberg and Schwefel [12, 13] for ES and independently, by Fogel [10, 11] for meta-EP (detailed described in sections III and IV-A).

II. THEORETICAL BACKGROUND

A. Artificial Neural Networks

An Artificial Neural Network (ANN) [1, 2, 19] is a mathematical model, which consists of a set of interconnected processing elements, called neurons or nodes, where each neuron receives, modifies and transfers signals. The links between neurons are called connections. Each connection has a corresponding value, called the *connection weight*, which is used to modify an incoming signal. Feed-forward networks considered in this paper, refer to a type of ANNs, where information is transferred in one direction and all connections are directed from input neurons towards output neurons. The outgoing signal of each neuron is calculated by means of *activation function* (sometimes also referred to as *transfer function*), which is expressed as follows:

$$y_i = f_i \left(\sum_{j=1}^n x_j w_{ij} - \theta_i \right)$$

where y_i is the output of neuron i , x_j is the j -th input signal to neuron i , w_{ij} is a connection weight between neurons j and i , θ_i is an internal threshold value² of the neuron i , i.e. value, that must be achieved by the weighted sum of incoming signals to activate the neuron. Mathematically, the threshold value shows the position of the highest increment of the monotony increasing activation function. The transfer function used by all the nodes is sigmoid function:

$$f(t) = \frac{1}{1 + e^{-t}}$$

Evolutionary Artificial Neural Networks (EANNs) belong to a subclass of ANNs, in which evolutionary search procedures, i.e. EAs are used to evolve parameters of ANNs. The distinct feature of EAs is their adaptability to the environment and changes in that environment, which is specified by the EAs' evolution process.

B. Learning in ANNs

The central issue in theory of ANNs is *learning*, also known as training process accomplished by using *examples* or *training data*. It consists of the self-modification of a network (e.g. network's parameters) according to some learning rule. Usually, learning in ANNs is formulated as an optimization process for the purpose of finding an optimal set of connection weights (weights training) according to specific optimality criteria. However, principally learning can involve optimization of other parameters of ANNs [19].

The study in this paper is limited to the widely applied learning paradigm – *supervised learning*. Supervised learning is provided by the comparison between the actual output of ANN and the expected correct output. Obtained *training error (fitness)* calculated by the *error function*,

² Threshold values (or biases) are usually implemented as connection weights with fixed input -1

estimates the worth of the network. Basically, supervised learning is formulated as the minimization of the error function, such as the mean square error (MSE) or the root mean square error (RMSE) between the expected and the actual outputs over all training examples of a considered task.

III. ADVANTAGES OF THE GAUSSIAN AND CAUCHY DISTRIBUTION FUNCTIONS IN EVOLUTIONARY LEARNING

There are two main evolutionary learning strategies basically used to train ANNs: the classical Evolutionary Programming (CEP) [3] based on the Gaussian distribution and the Fast Evolutionary Programming (FEP) [15], based on the Cauchy distribution. In order to establish the role of a distribution function in these algorithms, let us have a look on their evolution process.

Each of both algorithms creates an initial population of μ chromosomes randomly. Each chromosome is represented by a pair of real-valued vectors (x_i, η_i) , $\forall i \in \{1, \dots, \mu\}$, where $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ is the set of n connection weights of the network and $\eta_i = (\eta_{i1}, \eta_{i2}, \dots, \eta_{in})$ is the vector, denoted to as a control (strategy) parameter, responsible for self-adaptation. After evaluation of fitness the creation of new individuals is performed. Each parent (x_i, η_i) creates an offspring (x_i', η_i') , by: for $j \in \{1, \dots, n\}$

$$x_i'(j) = x_i(j) + \eta_i(j)N_D \quad (1)$$

$$\eta_i'(j) = \eta_i(j) \exp(\tau' N(0,1) + \tau N_j(0,1)) \quad (2)$$

The learning rates τ and τ' are commonly set to:

$$\tau = \left(\sqrt{2\sqrt{n}} \right)^{-1}$$

$$\tau' = \left(\sqrt{2n} \right)^{-1}$$

$x_i(j)$, $x_i'(j)$, $\eta_i(j)$ and $\eta_i'(j)$ denote the j -th component of the vectors x_i , x_i' , η_i and η_i' , respectively. $N_j(0,1)$ in Eq. (2) denotes a random value with mean 0 and variance 1 and is generated anew for each value of j . The value of N_D indicates a random number generated according to either Gaussian (for CEP) or Cauchy (for FEP) distribution. Hence, for the CEP Eq. (1) can be rewritten:

$$x_i'(j) = x_i(j) + \eta_i(j)N(0,1) \quad (3)$$

where $N(0,1)$ is a uniformly distributed random value with mean 0 and variance 1.

Similarly, Eq. (1) for FEP is given by:

$$x_i'(j) = x_i(j) + \eta_i(j)\delta_j \quad (4)$$

where δ_j is a Cauchy random number with the scale parameter³ $\gamma = 1$, which is generated anew for each value of j .

³ γ is a scale parameter in the one-dimensional Cauchy probability density function, defined by the following equation:

$$f(x) = \frac{\gamma}{\pi(\gamma^2 + x^2)}, \quad -\infty < x < \infty$$

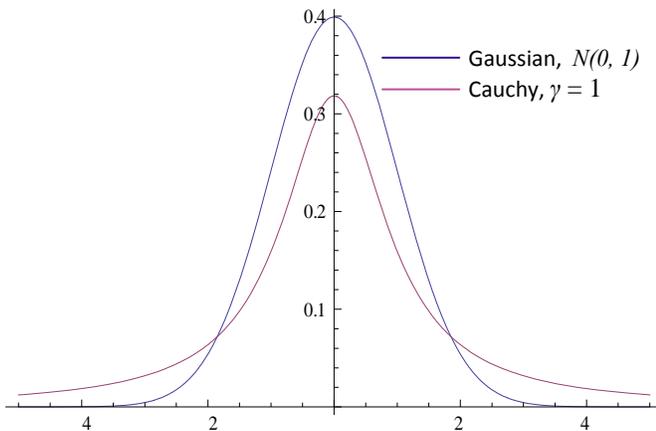


Fig. 1 Gaussian and Cauchy probability density functions

It is obvious, that both CEP and FEP use the same mutation scheme to modify individuals and the same self-adaptation strategy to correct mutation step size. The only difference is distribution used to generate random numbers (Fig. 1). Therefore, it is reasonable to claim that the features of algorithms' performances are caused by a type of used distribution. Let us discuss the advantages of using both distributions.

The expected length of Gaussian (with $\mu = 0$ and $\sigma^2 = 1$) and Cauchy (with $\gamma = 1$) jumps can be calculated by integrating the probability density functions:

$$E_G(x) = \int_0^{+\infty} x \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx = \frac{1}{\sqrt{2\pi}} = 0.399$$

$$E_C(x) = \int_0^{+\infty} x \frac{1}{\pi(1+x^2)} dx = +\infty$$

Apparently, the Cauchy distribution enables longer jumps than the Gaussian one. At first sight it seems that longer jumps in the search space induce quicker convergence, and so the Cauchy distribution is preferable in the searching strategy. However, this assumption is wrong. The analytical studies in [14] provided to investigate when large jumps are beneficial, showed that long jumps are advantageous only when the global optimum is far away from the current search point. In other words, long jumps are effective when the distance between the global optimum and the current point is larger than mutation's step size. On the other hand, the Cauchy distribution will no longer be beneficial when the distance between the neighborhood of the global optimum and the current point is smaller than the step size of the mutation. This implies that the use of small jumps is more effective near the neighborhood of the global optimum. Hence, the Gaussian distribution increases the probability of finding the optimum when the distance between the current point and the neighborhood of the global optimum is small.

IV. LEARNING ALGORITHM

A. Self-Adaptation in Evolutionary Algorithms

As we have seen from the previous section, the distribution plays an important role in the modification of population's individuals. However, a distribution has a spontaneous character; hence the use of distribution only is not enough to improve individuals efficiently. In order to bias the distribution towards appropriate regions of the search space self-adaptation mechanisms have been proposed. Self-adaptation aims at adjusting the setting of control parameters, which can be of various forms [20], during the run of the algorithm.

The self-adaptation mechanism, described in the previous section by Eq. (2) was introduced by Rechenberg and Schwefel [12, 13] in the area of evolutionary strategies, and independently, by Fogel [10, 11] for evolutionary programming. The basic step in this mechanism consists of *a mutation of mutation parameters themselves*, i.e. the evolution of strategy parameters alongside with the object parameters. The modification of the control parameters is realized by multiplication with a random variable.

In contrast to the classical evolutionary algorithms, the NWEA algorithm [18], designed to evolve ANNs' parameters, uses different self-adaptation approach to find an optimum. The self-adaptation in NWEA comprises two control parameters, which *incorporate information about the position of an individual on the genotype and phenotype level*. The first component includes information about worth of a chromosome according to its fitness. The second component adds information about an ANN topology, the genotype encodes, i.e. information about position of an individual in the ANN architecture space. Alike ES and EP, the NWEA approach relies on mutation and does not utilize crossover at all.

B. Basic steps of the Learning Algorithm

The evolution with the NWEA algorithm is implemented as follows:

- 1) Create an initial population consisting of μ randomly generated chromosomes. Each chromosome $x_i = (x_1, x_2, \dots, x_n)$, $\forall i \in \{1, \dots, \mu\}$, represents one possible set of connection weights⁴, where m is a population size, n – a total number of connections between neurons and $x_j \in [-1.0; 1.0]$, $j \in \{1, \dots, n\}$.
- 2) Evaluate the fitness of each individual from a population according to the objective function.
- 3) In contrast to other EP approaches, which apply probabilistic selection methods for choosing parents for reproduction, NWEA “allows” all parental chromosomes to take part in the creation of new individuals. It is worth noting that during mutation only

⁴ In case of simultaneous evolution of ANN's weights and architectures, each chromosome consists of a set of vectors according to the ANN's connectivity matrix

one gene in the parental chromosome changes its value.

Offspring chromosomes are created by application of the following equation to every chromosome in the population:

$$x_i'(j) = x_i(j) \left(1.0 + N_w(l, n) \cdot N_F \cdot N_{Rand}^D \right) \quad (5)$$

where $x_i'(j)$ is a gene randomly chosen out of a chromosome x_i , and mutated; N_w – value that implicitly describes an ANN's internal structure, N_F – fitness, determined by the error function (MSE or other) of x_i , and N_{Rand}^D is a random value, that can be Gaussian, Cauchy or normally distributed. The mutation is provided at the chromosome level.

The components N_F and $N_w(l, n)$ represent the adjustment components in the mutation strategy. They add knowledge about position and worth of a chromosome in search space in order to achieve optimal improvement of chromosomes at each stage of evolution. The component N_F in Eq. (5) represents the genotype information, i.e. error of the mutated chromosome, which changes dynamically for every mutated chromosome. It enables the control of a randomly generated value and the adjustment of the mutation strength to an individual depending on its fitness, i.e. the higher the error of a chromosome, the higher the step size.

The value $N_w(l, n)$ is defined by Eq. (6), which depends on the number of hidden layers l and the average number of neurons on hidden layers n in the given ANN. This value is distributed by the Fermi-Dirac-like function and is calculated according to the following formula:

$$N_w(l, n) = A_1 + \frac{l}{2} + \frac{B_1 - \frac{l}{2}}{1 + \exp\left(\frac{n - \mu}{T_1}\right)} \quad (6)$$

The value μ is similar to the chemical potential in the original Fermi-Dirac function (as cited in [18]) and depends on the number of hidden layers:

$$\mu = A_2 + \frac{B_2}{1 + \exp\left(\frac{l - B_2}{T_2}\right)} \quad (7)$$

The coefficients $A_1 = 3.0$, $B_1 = 2.0$, $T_1 = 0.4$, $A_2 = 1.2$, $B_2 = 3.2$, $T_2 = 0.6$ were obtained by the approximation of the results in [18] so that the NW values never become negative (the coefficients T_1 and T_2 correspond to the temperature in the original Fermi-Dirac function). For each ANN the quantity N_w is calculated only once and does not change its value during the evolution, if we consider the evolution of connection weights in the environment determined by an ANN architecture; in

case of simultaneous evolution of architectures and connection weights it becomes a new value every time when ANN's architecture is changed.

The main advantage of incorporating phenotype information in mutation is that it comprises detailed knowledge about the ANN's topology⁵ and thus enables to improve the values of connection weights in respect to a given phenotype. It is known from the theory of evolution that individuals with favorable traits, determined by the genotype, are more likely to survive and reproduce ("survival of the fittest") and the fitness of every individual is defined by the individual's ability to adapt to the environment. From such point of view, the NWEA approach is an abstraction, which involves the knowledge of an environment (phenotype) and adapts genotype of every individual to it, and thus, increases the fitness of chromosomes of every next generation.

Each time mutation takes place three times and produces three different offspring: once using normally distributed value N_{Rand}^{DG} , once – Cauchy random number N_{Rand}^{DC} and once – uniformly distributed random value ($N_{Rand}^{DU} \in [-1.0, 1.0]$). The benefits of using Gaussian and Cauchy distributions were described in previous section. In addition to that, we also utilize the uniformly distributed random values, as we keep out modification of NWEA to a minimum, and the original NWEA strategy is based on the continuous uniform distribution. Although there is no stronger motivation for using uniform distribution in this modification, it adds an additional randomness in the evolution process.

- 4) Evaluate the fitness of a new individual based on the objective function.
- 5) Repeat the process from point (3) until λ ($\lambda \geq 3 \times m$) new chromosomes are created.
- 6) Create new population of m individuals: new population is created according to the $(\mu + \lambda)$ -ES elitist method, which chooses μ best individuals from both parental and offspring chromosomes based on their fitness. This is accomplished by applying 2-tournament selection method that selects a group of individuals (usually four) from both parental and offspring populations, and compares their fitness. The individual with the higher fitness reaches the offspring population.
- 7) Repeat the process from point (3) until some halting criteria are satisfied.

Thus, by creating offspring population our greedy modification of the NWEA strategy selects the current best

⁵Although mutation strategy in (5) incorporates the knowledge about ANN's internal structure, it gives detailed information about considering topology, since the number of neurons in input and output layers is determined by a solving problem

individuals. On the other hand, the risk of trapping is local optima is minimal, since the random values initially have long and short step sizes.

V. EXPERIMENTAL STUDIES

The experimental studies of GNWEA were provided for two simple benchmark data sets for the purpose of studying generalization accuracy of EANNs. During the evolution the ANNs' weights and architectures have been optimized simultaneously, in order to reduce the noise in the fitness evaluation. For each problem 50 runs of GNWEA were provided. Following initial parameters were used for these experiments: the population size 30, the maximum number of generations 250, the number of hidden nodes for each individual was chosen uniformly at random between 1 and 3. The algorithm stopped when the maximal generation was reached. The results of GNWEA were compared with those of NWEA, which is shown to be more efficient than other searching techniques [21].

A. Breast cancer diagnosis

The breast cancer data set was originally obtained from Dr. William H. Wolberg at the University of Wisconsin Hospitals, Madison. The data set consists of 699 examples of which 458 (65.5%) are benign examples and 241 (34.5%) are malignant examples. Each example contains nine attributes: clump thickness, uniformity of cell size, uniformity of cell shape, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, mitoses. The goal of the data set is to classify a tumour as either benign or malignant based on these attributes.

In our experiments, the whole data set was divided into three subsets, as suggested by Prechelt [22]: a training set, a validation set, and a testing set. The first set was used to train EANNs. The validation set was explored as a pseudo-testing set in order to evaluate the fitness of networks during evolution. This prevents overtraining of the network and improves its generalization ability. During this process ANN's learning is carried out until the minimal error on the validation set (and not on the training set) is achieved. Finally, the testing data were considered to evaluate the performance of the evolved ANNs. 349 examples of the given breast cancer data set were used as training data, the following 175 examples as validation data, and the final 175 patterns as training data.

The error function (fitness) was calculated according to the equation, proposed by Prechelt [22]:

$$E = 100 \cdot \frac{O_{\max} - O_{\min}}{N \cdot P} \sum_{p=1}^P \sum_{i=1}^N (o_{pi} - t_{pi})^2$$

where O_{\min} and O_{\max} are the minimum and maximum values of output coefficients in the problem representation. N is the number of output nodes, P is the number of patterns, o_{pi} and t_{pi} are the actual and desired outputs of node i for pattern t

correspondingly.

Table I presents classification results for breast cancer diagnosis. The error rate in the Table I shows the percentage of incorrect classifications, produced by the ANNs. Table II shows the architecture of evolved ANNs as well as generation numbers (min and mean) at which the optimal results were obtained.

B. Heart disease diagnosis

The heart disease data set was obtained from Cleveland Clinic Foundation and was supplied by Robert Detrano of the V.A. Medical Center, Long Beach, CA. The data set consists of 270 examples. The heart disease original data set consisted of 303 examples, but 6 of them contained missing class values and were excluded from the database. Other 27 examples of the remained data were eliminated as they retained in case of dispute.

Each example in the database contains 13 attributes, which present results of medical tests provided on patients: age, sex, chest pain type, resting blood pressure, cholesterol, fasting blood sugar < 120 (true or false), resting electrocardiogram (norm, abnormal or hyper), max heart rate, exercise induced angina, oldpeak, slope, number of vessels colored and thal (normal, fixed, rever). These attributes have been extracted from a larger set of 75. The goal of diagnosis is to recognize the presence or absence of heart disease given the attributes. Initially, the data set considered four different degrees of the heart disease to classify the predicted results. Later, modification in the problem definition suggested reducing the number of predicted values on two and categorizing results into two classes: presence or absence of illness.

For this set of experiments we applied the same equation to calculate fitness values as for the breast cancer diagnosis problem. Tables III and IV report the best and the average prediction errors and evolved EANNs' architectures for heart disease problem, respectively.

VI. DISCUSSIONS AND CONCLUSIONS

In this paper we proposed a greedy learning algorithm, which is the improvement of the NWEA strategy for ANNs parameters optimization. The goal for this was to explore how different step sizes determined by the different distributions affect the evolution process and the quality of the obtained ANNs.

The preliminary experimental studies were carried out for the breast cancer and heart disease benchmark data sets. The obtained results indicated that the utilization of greedy technique improves the generalization ability of ANNs. Statistical analysis of data reported in Tables I-IV with t -test showed that the GNWEA algorithm was able to find better results than NWEA for both problems. While the differences between evolved ANN architectures (Tables II and IV) are insignificant (still GNWEA performed better), the differences between the error accuracies on both training and testing sets are found to be not quite significant for

TABLE I
COMPARATIVE RESULTS OF THE PREDICTION ACCURACY FOR BREAST CANCER DIAGNOSIS

		NWEA			GNWEA		
		min	max	mean	min	max	mean
Training set	Error	1.418	3.650	2.722	1.183	2.659	2.329
	Error rate	0.01698	0.04672	0.03921	0.00744	0.02451	0.0224
Validation set	Error	0.052	1.018	0.557	0.037	0.637	0.349
	Error rate	0.00000	0.01072	0.00547	0.00000	0.00902	0.00562
Testing set	Error	0.178	3.546	1.413	0.054	2.899	1.217
	Error rate	0.00000	0.03397	0.01384	0.00000	0.02687	0.00467

TABLE III
COMPARATIVE RESULTS OF THE PREDICTION ACCURACY FOR HEART DISEASE DIAGNOSIS

		NWEA			GNWEA		
		min	max	mean	min	max	mean
Training set	Error	7.489	12.166	11.007	5.763	12.005	8.963
	Error rate	0.07879	0.15184	0.12477	0.04831	0.12069	0.09446
Validation set	Error	11.746	14.301	12.450	9.271	12.158	9.677
	Error rate	0.12124	0.19706	0.15935	0.08113	0.13812	0.10543
Testing set	Error	10.126	13.842	12.266	7.009	12.932	10.633
	Error rate	0.13195	0.17997	0.15165	0.09385	0.14889	0.11676

TABLE II
ANN ARCHITECTURES FOR BREAST CANCER DIAGNOSIS

	NWEA	GNWEA
Connections (min)	14	14
Connections (max)	78	82
Connections (mean)	36	29
Hidden nodes (min)	0	0
Hidden nodes (max)	4	4
Hidden Nodes (mean)	1.3	1.3
Generation (min)	101	87
Generation (mean)	139.1	118

TABLE IV
ANN ARCHITECTURES FOR HEART DISEASE DIAGNOSIS

	NWEA	GNWEA
Connections (min)	28	26
Connections (max)	202	192
Connections (mean)	88.4	78.3
Hidden nodes (min)	2	2
Hidden nodes (max)	8	8
Hidden Nodes (mean)	4.3	3.7
Generation (min)	127	106
Generation (mean)	172.2	157.9

breast cancer diagnosis (Table I) and extremely significant for heart disease problem (Table III).

Although we did not provided the analysis of the proposed algorithm' complexity, it is essential to discuss briefly the computational costs it incurs. Obviously, GNWEA requires much time to create new individuals than NWEA. In order to create offspring by Gaussian and Cauchy mutations, a computer program first needs to transform the pseudo-random values into Gaussian and Cauchy random numbers. Let us assume that time, the computer program needs to execute one operation is t . Thus, time needed to generate the pseudo-random uniform numbers is:

$$t(f_U) = t$$

Hence, the expected time of generating Gaussian and Cauchy numbers is calculated by:

$$t(f_G) = t(f_C) = t_{transf}(t(f_U)) = 2t$$

This simplification shows that GNWEA needs

approximately five times more computational time than NWEA to create the random values (which, however, does not mean that GNWEA is five time slower than NWEA). Besides temporal expenses, GNWEA requires much computational memory to create three times more offspring, as the size of offspring population of GNEWA exceeds the size of offspring population of NWEA in three times.

In spite of that, the computation time and resources are not at highest importance while solving classification/prediction problems, where generalization, i.e. an ANN's ability to find good results on both training and testing sets is crucial. From this point of view, the results obtained for GNWEA are promising and encourage further studies on data sets with large number of attributes.

REFERENCES

- [1] X. Yao, „Evolving artificial neural networks”, in *Proceedings of the IEEE*, IEEE Press, pp. 1423-1447, 1999

- [2] X. Yao, „A review of evolutionary artificial neural networks”, *International Journal of Intelligent Systems*, 8(4):539–567, 1993
- [3] D. B. Fogel, L. J. Fogel, and V. W. Porto, “Evolving neural networks,” *Biological Cybern.*, vol. 63, no. 6, pp. 487–493, 1990
- [4] D. B. Fogel, E. C. Wasson, and V. W. Porto, „A step toward computer-assisted mammography using evolutionary programming and neural networks,” *Cancer Lett.*, vol. 119, no. 1, p. 93, 1997
- [5] V. W. Porto, D. B. Fogel, and L. J. Fogel, “Alternative neural network training methods,” *IEEE Expert*, vol. 10, pp. 16–22, Mar. 1995
- [6] A. P. Topchy and O. A. Lebedko, „Neural network training by means of cooperative evolutionary search,” *Nuclear Instrum. Methods in Phys. Res., Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 389, nos. 1–2, pp. 240–241, 1997
- [7] I. Rechenberg, „Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution“, Fromman-Holzboog Verlag, Stuttgart, Germany, 1973
- [8] T. Bäck, F. Hoffmeister, H.-P. Schwefel, „A survey of evolutionary strategies“, in *Proc. of the 4th International Conference on Genetic Algorithms*, pp. 2–9, San Diego, CA, July 1991.
- [9] G. W. Greenwood, „Training partially recurrent neural networks using evolutionary strategies,” *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 192–194, Feb. 1997
- [10] L. J. Fogel, A. J. Owens, and M. J. Walsh, „Artificial intelligence through simulated evolution”, Wiley, New York, 1966
- [11] L.J. Fogel. „Autonomous automata”, *Industrial Research*, 4:14–19, 1962
- [12] I. Rechenberg, „Cybernetic solution path of an experimental problem”, *Royal Aircraft Establishment*, Farnborough, page Library Translation 1122, 1965
- [13] H.-P. Schwefel, „Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik“, *Diplomarbeit*, Technische Universität Berlin, 1965
- [14] Yao, G. Lin and Y. Liu, „An Analysis of evolutionary algorithms based on neighborhood and step size”, *Evolutionary Programming VI: Proc. of the Sixth Annual Conference on Evolutionary Programming (EP97)*, Lecture Notes in Computer Science, Vol. 1213, pp.297–307, Springer-Verlag, Berlin, 1997
- [15] X. Yao, Y. Liu, „Fast Evolutionary Programming”, in *Proc. of the Fifth Annual Conference on Evolutionary Programming*, MIT Press, pp. 451–460, 1996
- [16] X. Yao, Y. Liu, G. Lin, „Evolutionary programming made faster,” *IEEE Transactions on Evolutionary Computation*, IEEE Press, 1996
- [17] X. Yao and Y. Liu, „Scaling up evolutionary programming algorithms,” *Evolutionary Programming VII: Proc. of the Seventh Annual Conference on Evolutionary Programming (EP98)*, Lecture Notes in Computer Science, Vol. 1447, Springer-Verlag, Berlin, pp.103–112, 1998
- [18] K. Davoian, W.-M. Lippe, „Including phenotype information in mutation to evolve artificial neural networks”, in *Proc. of the IEEE International Joint Conference on Neural Networks (IJCNN'07)*, Orlando, USA, 2007
- [19] W.-M. Lippe, „Soft-Computing mit Neuronalen Netzen, Fuzzy-Logic und Evolutionären Algorithmen”, Springer-Verlag, Berlin Heidelberg, 2006
- [20] S. Meyer-Nieberg, H.-G. Beyer, „Self-Adaptation in evolutionary algorithms“, *Studies in Computational Intelligence*, Springer-Verlag, Berlin/Heidelberg, Vol.54, ISBN: 978-3-540-69431-1, pp.47–75, 2007
- [21] K. Davoian, A.Reichel, W.-M. Lippe, „Time series prediction in parallel evolutionary artificial neural networks”, in *Proc. of the 2007 International conference on Data Mining (DMIN'07)*, CSREA Press, pp. 10–15, 2007
- [22] L. Prechelt, „Proben1—A set of neural network benchmark problems and benchmarking rules.” Fakultät für Informatik, Universität Karlsruhe, Karlsruhe, Germany, Tech. Rep. 21/94, Sept. 1994

Generation of "Weak" Models in Stochastic Discrimination

I. Skrypnik

Department of Computer Science and Information Systems, University of Jyväskylä, Jyväskylä, Finland

Abstract – A stochastic discrimination method is a localized multi-model learning technique with a great potential for contemporary data. The theory proves reaching complete class discrimination under a few conditions applied to a collection of the component “weak” models. Achieving an agreement between those conditions on real data requires heuristic findings and algorithmic tricks. Known implementations of this method use various model types, stopping criteria, and employ different solutions for the uniformity-enrichment conflict of model generation. Model design, in particular, greatly influences effectiveness of the method. It was found that generation of hyperrectangular and hyperspherical models leaves some instances out of reach as uniformity stabilizes. The paper suggests improvements to these models using cell based and k-NN based approaches. Parallel generation of models with different enrichment degree is proposed to increase chances of obtaining satisfactory collection of models. Experiments show that k-NN approach is three times faster than cell based approach.

Keywords: Stochastic Discrimination, Local Learning, Ensemble, Classification, Data Mining.

1 Introduction

The theory of stochastic discrimination (SD) [8-10] is relatively new to a data mining community yet has been successfully applied to different pattern recognition problems [4,5,7]. Ceredyn Systems [2] used it as a technological base for data mining solutions in pharmaceutical and bioinformatics applications. Favorable results were reported on preliminary comparison of the derived SD method with neural networks, SVM, boosting, and bagging [2].

In the SD theory, given a solvable class discrimination problem, the SD method covering the instance space with “weak” models would get arbitrarily close to the best solution in affordable time [8,9]. Discrimination of classes is guaranteed by maintaining rigorous conditions of *uniformity* and *enrichment* for the collection of *projectable* component models. The models are generated by a pseudo-random process and cover the data instances of each particular class approximating its shape and separating it from the others.

The SD theory has been elaborated in [1,3], but the method itself has not been widely explored on different data mining applications. The strict mathematical assumptions used in the SD theory are rarely met in real data sets [5]. Practical application of the method requires weakened assumptions and suggests specific algorithmic solutions.

This paper focuses on the implementation issues and suggests two modifications of hyperrectangular and hyperspherical models using cell based and *k*-Nearest Neighbor based approaches. Both implementations are evaluated on synthetic classification problems with heavily interleaved classes and non-uniform density of instances. The experiments show that suggested modifications along with parallel model generation decrease generation time and promote convergence of the process. Analysis of advantages and disadvantages of different model types presented in the paper contributes to the research on further advance of this promising data mining technique.

Section 1 introduces basic concepts of the SD theory. Section 2 discusses model generation with different model types. Section 3 presents suggested approaches. Section 4 describes parallel model generation with closing streams and experiments on synthetic data sets with different class boundaries.

2 SD theory: the basic concepts

2.1 A “weak” model in SD

A notion of a “weak” model used in SD is somewhat different from a “weak” classifier used in ensemble techniques. A “weak” model in SD describes a region in a feature space, which is relatively small compared to the region bounded by the entire set of training instances. There is no requirement for a prediction accuracy level of a “weak” model in SD, but it has to maintain *projectability*, that is to capture class boundaries in a particular region overlooking the training instances and convey the true class boundaries in unseen instances of this region. A model covers majority of instances of a particular class, according to model’s *rating* and the *enrichment* condition. A collection of models in SD reflects decision regions of a classifier. As a “weak” model covers majority of one particular class being *projectable*, it can be considered as a local classifier. Unclassified instances from the test set covered by this model are assigned to the majority class of this model.

2.2 Projectability, uniformity, and enrichment

Projectability of models with respect to unclassified instances of the test set (*TE*) is based on the assumption that in the training set (*TR*), proportion of instances of a certain class in a local region covered by a model is close to actual proportion in that region taking into account unseen instances. This assumption is justified for a majority of

natural problems. SD assumes that TR is *representative* sustaining *descriptive power of the component models*. It guarantees similar accuracies for classifiers having the same enrichment degree of the component models [6]. Minimum and maximum bounds placed on a model size help to maintain projectability.

For each instance q_i from TR the *coverage* $N(q_i, M_t)$ is a current number of models in the collection M_t that include q_i , where $t=1..T$ is the number of models in the collection. $Y(q_i, M_t)$ is a ratio of $N(q_i, M_t)$ over t . $Y(q_i, M_t)$ is a probability of q_i being covered by a model M , $M \in M_t$. As M_t expands, the values of $Y(q_i, M_t)$ for each q_i converge to 0.5 [6,11], if the model generation process is not biased toward any particular q_i .

The uniformity condition can be illustrated by a simple example from [6], when 1-dimensional TR includes 10 instances and each model M includes 5 instances. Then the number of all possible models in M_t (without repetitions) is 252. When M_t includes 252 models, each instance q_i has the same coverage $N(q_i, M_t)$ and $Y(q_i, M_t)=0.5$, i.e. a probability of every instance in TR being covered by any model included to M_t is 0.5. That is models in M_t cover every instance uniformly. A *uniform coverage* of TR , or *uniformity*, is a fundamental principle of the SD theory.

Eventually, SD deals with classification problems. Thus, an additional restriction would be to cover only instances of one particular class. As shown above, $Y(q_i, M_t)$ values converge disregarding class membership of the instances included to a model. If class membership of instances is incorporated, $Y(q_i, M_t)$ converges to two distinct values for a two-class problem. According to the theory, each model should be *enriched* toward a particular class, that is to cover majority of that class. The class to be enriched is *positive*; the other one is *negative*. Instances from TR covered by a model are called *positives* or *negatives* according to their class label. A fraction of positives covered by the model M is $frac_{p,M} = pos_M / n_p$, where pos_M is a number of positives covered by M and n_p is the number of positives in TR . The fraction of negatives covered by M is $frac_{n,M} = neg_M / n_n$ respectively. These fractions are considered as an *enrichment degree* of a particular model M . *Rating* (pos_M, neg_M) is a combination of positives and negatives in M , then $frac_{p+n,M} = (pos_M + neg_M) / (n_p + n_n)$.

In practice, creation of a full models collection using a stochastic model generation process is an unfeasible task. Regulation over the generation process is needed in order to obtain a *nearly* uniform coverage in a reasonable time. Some regulation can be established using threshold values related to uniformity and enrichment conditions. Introduction of model

size bounds μ and η , $0 < \mu \leq \eta \leq 1$, limits $frac_{p+n,M}$ so that $\mu \leq frac_{p+n,M} \leq \eta$. The enrichment condition for a candidate model M , such that $\mu \leq frac_{p+n,M} \leq \eta$, is described by the inequality $|frac_{p,M} - frac_{n,M}| \geq \beta$, where β is an enrichment threshold, $0 < \beta < 1$.

Collection of models enriched toward one class is a common setting for the enrichment threshold. Higher degree of enrichment increases accuracy, but collection of highly enriched models from a random stream takes more time. Due to uniformity and enrichment conflict, the number of acceptable models becomes considerably limited. Preliminary estimation of β may help to prevent this situation. The uniformity condition is much more difficult to satisfy. Strict uniformity requires that every instance in TR is covered by *the same number of models of every particular rating* within specified μ, η , and β . For practical tasks, this strict condition is weakened to *near uniformity* [5]. In particular, μ, η , and β settings may imply acceptance of models with certain rating only [6]. On the other hand, this restriction elongates the generation process, eventually promoting the uniformity and enrichment conflict. The number of acceptable models can be very small to obtain even a nearly uniform coverage. A solution is suggested in Subsection 4.1.

2.3 Classification rule

The *base random variable* $X(q_i, M_j)$ is defined via the membership function $C_{M_j}(q_i)$. $C_{M_j}(q_i)=1$ iff q_i is covered by the model M_j accepted at j -th iteration, and 0 otherwise.

$X(q_i, M_j) = \frac{C_{M_j}(q_i) - frac_{n,M_j}}{frac_{p,M_j} - frac_{n,M_j}}$. Every time after accepted

M_j , a discriminant $Y(q_i, M_t) = \frac{1}{t} \sum_{k=1}^t X(q_i, M_k)$ is updated with respect to current collection of models M_t . As M_t expands, the value $Y(q_i, M_t)$ changes for each q_i and converges to two distinct values, 1 and 0, for positives and negatives accordingly. Separation of trends happens before all possible models restricted by selection of μ, η , and β are included. The classification rule is obtained using the Central Limit Theorem using the base random variable. Instances are assigned to a positive class if $Y(q_i, M_t) \geq 0.5$.

For prediction, the Y values are determined considering coverage of the test instances by models in M_t calculated using description of model boundaries. In order to obtain the final prediction for multi-class problems the probability variables Y are estimated in "one-against-all" or "one-against-one" subproblems of class decomposition. The final class value comes from the subproblem, where Y is the

largest [10]. Details of the SD theory can be found in [5,6,10].

3 Generation of models

3.1 Approaches to model generation

A “weak” model should have a simple representation, easy for storage and enumeration of covered instances [6]. A few variants have been suggested: (1) interval based hyperrectangular model, regions bounded by two parallel hyperplanes perpendicular to one or several randomly selected dimensions, (2) half-spaces bounded by a threshold on the randomly selected dimension, (3) half-spaces bounded by hyperplanes equidistant to a pair of randomly selected instances of different classes, (4) hyperspherical models centred at randomly selected instances and drawn with random radii calculated over all dimensions, and (5) Hamming distance based model [5,6,11,12]. It was also proposed to derive a model from a union or intersection of several models of the same type [6,7]. The above model types fall into three categories: hyperrectangular (1) - (3), hyperspherical (4), and binary (5).

Hyperrectangular models of type (1) are constructed using the intervals of values on each dimension, one or several randomly selected dimensions (1). An interval $[l_j, u_j]$ is randomly chosen on a dimension j , $j = 1..P$. Then models are regions bounded by two parallel hyperplanes perpendicular to the selected dimension and intersecting the axis in the interval ends. A new instance q is covered by a model, if it's value in the j^{th} feature falls into the corresponding interval of values, $q^j \in [l_j, u_j]$.

In models of type (2), a threshold value on a selected dimension(s) creates a dividing hyperplane. In [6], the threshold value t_j is selected randomly within the range of possible values on a randomly chosen dimension j . A hyperplane intersects this dimension orthogonally in the point corresponding to a selected threshold value. Thus, models are half-spaces bounded by hyperplanes. A new instance q is covered by a model, if this instance resides in the same half-space, that is $q^j \in [\min_j, t_j]$ or $q^j \in [t_j, \max_j]$. Thresholds may be placed on several randomly selected features.

Models of type (4) are half-spaces bounded by hyperplanes equidistant to a pair of instances [6,12]. Two instances from different classes q_0 and q_1 are randomly selected, then the *difference vector* is computed as the normal vector of a hyperplane that separates the instances from the same class as q_0 , $C(q_0)$, and the instances from the set of instances belonging to a different class $\bar{C}(q_0)$. The model is represented by a difference vector to be stored together with an additional value κ , which specifies location of the

discriminating hyperplane along the difference vector with respect to q_0 and q_1 . If $\kappa = -1$, the hyperplane intersects the instance belonging to the set $\bar{C}(q_0)$; if $\kappa = 1$ the hyperplane intersects the instance belonging to the set $C(q_0)$, and with $\kappa = 0$ it crosses the midpoint. Whether an instance is covered depends on which half of the divided space it resides. An alternative method of computing the separating hyperplane is to use a random vector for the projections of instances q_0 and q_1 , rather than the difference vector. This removes the constraint that the hyperplane is perpendicular to the difference vector.

With models (1) – (4) additional restrictions are needed in order to maintain projectability and the generation process. The number of dimensions to be restricted and the magnitude of the restriction are related to class boundaries, density and other characteristics of a data set. Restricting one dimension at a time with one threshold may lead to a model capturing too many instances and losing projectability. With restrictions on every dimension the generation process may end up with empty models. Some control on projectability for models (1) - (2) may be gained if the threshold values on a selected dimension are obtained from the corresponding values on two randomly selected instances of different classes.

If the applied conditions are too severe for the model generation process, it was suggested to accept the best model after a predefined number of iterations. However, the resulting collection of models may not satisfy conditions required by the SD theory by far, therefore the final prediction result is rather random. The above considerations brought up two modifications of known model types implemented with parallel model streams of different size and enrichment degree, the cell based and k -Nearest Neighbor based models, considered in Subsections 3.2 and 3.3.

3.2 Interval-based generation and its cell-based modification

The parameters of the SD method are the minimum and maximum model sizes (or μ and η) as for projectability, the enrichment threshold β , the uniformity threshold ω , the classification rule probability threshold α , and the number of iterations I . For example, parameter settings for the basic SD algorithm with $N = 200$ instances in TR are: the minimum model size $0.03 * N$, the maximum model size $0.06 * N$, $I = 1000$, $\beta = 0.01$, $\omega = 0.3$, and $\alpha = 0.5$. The main steps of model generation are outlined below.

*** Algorithm SD_basic ***

```

For (j = 0; j < P; j = j + 1){
    min[j] = determineFeatureMin(j);
    max[j] = determineFeatureMax(j);
}
// start collection of models

```

```

While (i < I) {
// start generation of an acceptable model
While (modelAccepted = false) {
  q0 = randomPoint(TR);
  hrect = generateSubintervals(min,max,q0);
// hrect [0] - lower ends
// hrect [1] - upper ends
  tmpmod = getModelInstances(hrect[0],hrect[1]);
// start uniformity and enrichment test
  enrichment = getEnrichment(tmpmod);
  if(enrichment > β){
    uniformity = getCoverageRatio(tmpmod);
    if(uniformity > ω)
      modelAccepted = true;
    else
      modelAccepted = false;
  }
  else{
    modelAccepted = false;
  } // end uniformity and enrichment test
} // end generation of an acceptable model
updateCoverage(tmpmod);
updateProbability(tmpmod);
models[i] = tmpmod;
} // end collection of models
// start classification
For (i = 0; i < N; i = i + 1){
  For (j = 0; j < I; j = j + 1){
    if(coveredByModel(j, test_q[i]) = true)
      updateProbability(test_q[i]);
  }
  prob[i] = calculateTestProbability(test_q[i]);
  classifyTestPoint(prob[i], α);
} // end classification

```

Generation of models as regions bounded in selected dimensions in our algorithm follows the steps used [13] with a few changes.

- Set the number of trials e for each iteration (for example, $e=2000$);

- Determine the maximum hyperrectangular region covering the entire data $R^P = \prod_{j=1}^P (v_{1,j}, v_{T,j})$, where $v_{1,j}$ and $v_{T,j}$ are minimum and maximum values of feature j ;

- Set the coefficient $\lambda = 1.1$ and determine $R^\lambda = \prod_{j=1}^P \lambda(l_j, u_j)$;

- Set the minimum interval percentage coefficient $\rho_{\min} = 0.005$ (the initial value is 0.5 % of the interval between $v_{1,j}$ and $v_{T,j}$);

- Set the maximum interval percentage coefficient $\rho_{\max} = 0.125$ (the initial value is 12.5 % of the interval between $v_{1,j}$ and $v_{T,j}$);

- Set the coefficient $\rho_{decay} = 0.9999$ to decrease ρ_{\max} during each of e trials.

The models are formed as follows. Let R^1 be the smallest hyperrectangular region containing TR . Then let R^λ be the rectangular region in R^P such that R^λ and R^1 have the same centre, R^λ is similar to R^1 and the width of R^λ along each dimension is λ times the corresponding width of R^1 . A subinterval $[l_j, u_j]$ of $[v_{1,j}, v_{T,j}]$ is then generated in

each dimension j by taking a random q_0 as a center, and assigning a width of the subinterval $(l_j, u_j) = \varepsilon(v_{1,j}, v_{T,j})$, where ε is a random value such that $\rho_{\min} \leq \varepsilon < \rho_{\max}$. Finally, a subinterval $[l_j, u_j]$ is adjusted to reside entirely within R^λ . ρ_{\max} is multiplied by ρ_{decay} each time the new model is added to the collection, until $\rho_{\max} = \rho_{\min}$. Instead of using trials and ρ_{decay} , adjustments to the model size are performed inside of the generation procedure, as shown below. The hyperrectangular region is increased or decreased in each dimension with $0.01(v_{1,j}, v_{T,j})$.

```

* Procedure generateSubintervals(v1, vT, q0) *
// external size smin, smax, rho_min, rho_max
lambda = 1.1;
coeff = randomNumber(rho_min, rho_max);
// start for all dimensions
For (j = 0; j < P; j = j + 1){
  length = vT,j - v1,j; // min-max length
  delta = (length * coeff) / 2;
  upper_j = q0_j,max + delta;
  lower_j = q0_j,min - delta;
  delta2 = length * lambda;
  delta3 = length * (1 - lambda);
  highest_j = vT,j + delta2;
  lowest_j = v1,j - delta2;
  if (lower_j < lowest_j)
    lower_j = lowest_j;
  if (upper_j > highest_j)
    upper_j = highest_j;
} // end for all dimensions
tmpmod = getModelInstances(lower, upper);
currmodsize = getCurrModSize(tmpmod);
// start adjusting the model size
While (!(smin < currmodsize < smax)) {
// start for all dimensions
  For (j = 0; j < P; j = j + 1){
    if (currmodsize < smin) {
      upper_j = upper_j + delta3;
      lower_j = lower_j - delta3;
    }
    if (currmodsize > smax) {
      upper_j = upper_j - delta3;
      lower_j = lower_j + delta3;
    }
    if (lower_j < lowest_j)
      lower_j = lowest_j;
    if (upper_j > highest_j)
      upper_j = highest_j;
  } // end for all dimensions
  tmpmod = getModelInstances(lower, upper);
  currmodsize = getCurrModSize(tmpmod);
} //end adjusting the model size
return lower, upper;

```

Adjustments for a model size in the described algorithm may take a very long time without pre-estimation for model sizes. The problem is aggravated by the conflict of uniformity and enrichment conditions. We suggest a cell based model modification that rescales each dimension to facilitate model generation increasing "resolution" of data.

Given a real positive number d , one can obtain a list of hyperrectangles of side length d_j , $j = 1 \dots P$, covering

instances of TR . The volumes of cells are given by $\prod_{j=1}^P d_j$. The total number of cells H is strictly a function of how many partitions we choose to have on each dimension. The number of cells in j^{th} dimension is given by $Q_j = \text{int} \left[\frac{v_{T,j} - v_{1,j}}{d_j} \right]$, where $v_{1,j}$ and $v_{T,j}$ are minimum and maximum values of feature j . The total number of cells is given by $H = \prod_{j=1}^P Q_j$. The algorithm based on the Sameer's framework [14] provides a rule of assigning instances to cells of the predetermined size. The coordinates of each cell are computed and instances are assigned to cells. The same rule is used during model generation. The goal is to obtain non-empty cells, thus, the data set is partitioned to obtain resolution $B = 1 \dots y$, where $y \geq 0$. Increasing resolution we increase the number of cells and decrease the number of instances in them. Similar to [14], a starting value for the parameter $y \geq 64$ is used. At the end, most cells likely will contain a single instance.

The instance is assigned to a cell by known bottom left vertex coordinates. The cell in the position m_j from the bottom in the dimension j , $0 \leq m_j \leq Q_j$, is labeled $H(m_1, \dots, m_p)$. The bottom left cell is $H(0, \dots, 0)$ and the top right cell is $H(Q_1 - 1, \dots, Q_p - 1)$. The coordinates of the bottom left cell $H(m_1, \dots, m_p)$ are given by $(v_{1,1} + m_1 d_1, \dots, v_{1,p} + m_p d_p)$, where $v_{1,1}$ and $v_{1,p}$ are minimum values of the 1st and P^{th} features accordingly.

3.3 k -Nearest Neighbor based models

In this model type the distance to the k^{th} neighbor of a randomly selected initial instance is used as a radius. If the number of nearest neighbors is greater than k , the instances are selected in the direction of higher density. The hyperrectangular frames are drawn by the intervals of feature values of the instances included to the model. The algorithm of model generation is presented below.

```

* Procedure generateKNNModel(q0) *
// generates a model of the fixed size s
// start model generation
nlist = findNeighbours(q0, s-1);
tmpmod = getListInstances(q0, nlist);
intervals = drawHRect(tmpmod);
tmpmod2 = getModelInstances(intervals);
currmodsize = getCurrModSize(tmpmod2);
// start adjusting the model size
if (currmodsize != s) {
  list2 = findEqualDistancePoints(q0, nlist);
  newlist = reduceTowardsHigherDensity(q0, nlist s, list2);
} // end adjusting the model s
mod = getModelInstances(q0, newlist);
} //end adjusting the model size
return mod;

```

4 Multi-stream model generation

4.1 Parallel model generation with closing streams

The uniformity condition implies that uniformity should be enforced for every particular model rating [6]. The entire collection of models can be divided onto several groups with models of particular size and rating. The number of such groups can be obtained from the minimum and maximum model sizes and the enrichment threshold. In the parallel model generation approach, streams of models with different ratings grow towards uniform coverage independently. However, the probabilities are updated using the entire collection of models. This approach increases chances to generate a sufficient number of models for a more accurate prediction. Having the minimum and maximum model sizes and the enrichment threshold, streams that will be closed before generation starts can be foreseen. Considering k nearest neighbors within the minimum and maximum model size for every instance from TR , the enrichment threshold suggests whether to keep a particular steam open for generation. In order to reduce costs of this approach the decision about any particular stream of models can be made after the first model satisfying the enrichment condition in the stream is found.

There is always a conflict between conditions of enrichment and uniformity that may result in stopping the model generation process before the desired number of iterations performed. The generation process may come to an endless loop or stop due to unsuccessful choice of the model size and enrichment condition. In case of k -Nearest Neighbor based generation, it may happen due to restricted number of variants that start from the random instance q_0 , as the process becomes more deterministic. Thus, our algorithm has to close streams during the generation process, indicating that all possible models satisfying enrichment and uniformity conditions have been collected and no new acceptable model can be generated. Consistency of model collection in the particular stream can be verified each time after the generation process made a large number of unsuccessful attempts to generate an acceptable model through iterations. The SD implementations considered in this paper evaluate streams after this number is twice greater then the number of positive points in TR .

In order to promote uniform coverage of a positive class, coverage for q_i is calculated after each successful addition of a new model to M_i and a search for "weak" instances is performed. An instance is labeled as "weak" if its coverage is less than the average coverage of the instances of its class. A new model will be accepted if, in addition to the enrichment condition, it covers a particular ratio of "weak" instances. Then, the uniformity threshold can be set to a ratio

of “weak” instances covered by a new model to the number of “weak” instances at the moment.

4.2 Experimental study

In order to investigate the cell based and the k -Nearest Neighbor based implementations of model generation in SD, some experiments has been performed. The goal is to see how quickly the sufficient number of acceptable models has been generated; track unreachable instances with both model types, and finally, evaluate prediction accuracy. The data sets were selected due to their simplicity, representativeness and previous usage in the literature to test SD and the other learning algorithms. Five synthetic data sets used in this study are two-dimensional, representing different degree of non-linearity of the class boundaries and intersection of the class distributions. Square [11], Concentric [13], Sinuous [7], Clouds and 2D-Gauss [13] data sets are shown in Fig. 1-5 respectively. The first three data sets have non-intersecting linear or non-linear class boundaries, while the last two have intersecting class boundaries.

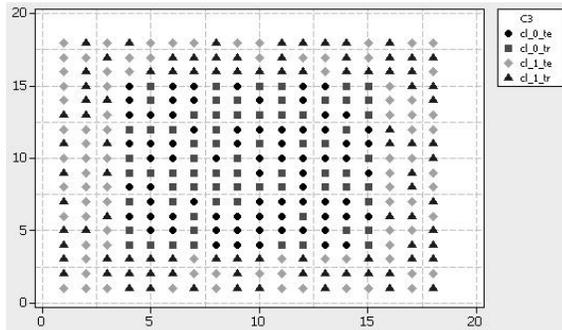


Fig. 1. The Square data set, 2 classes, linearly separable. Class 0 and class 1 in the training set are marked cl_{0_tr} and cl_{1_tr} ; class 0 and class 1 in the test set are marked cl_{0_te} and cl_{1_te} respectively.

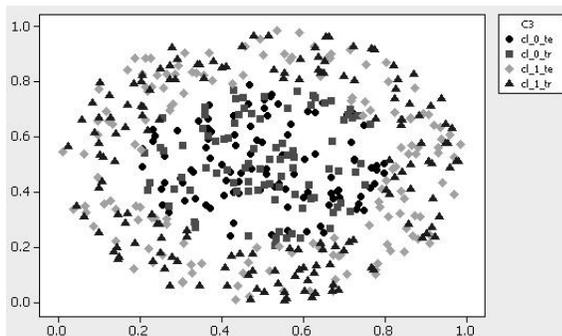


Fig. 2. The Concentric data set, 2 classes, 250/250 instances in training (TR) and test (TE) sets, non-intersecting classes. Class 0 and class 1 in TR are marked cl_{0_tr} and cl_{1_tr} ; class 0 and class 1 in TE are marked cl_{0_te} and cl_{1_te} respectively.

In order to demonstrate easily interpretable results, the cross-validation majority technique was not applied in the experiments. The experiments were carried out in a single run keeping track of the model generation process. For each 2-class subproblems the minority class is the class towards

which the models are enriched. For the data sets with more than 2 classes “one-against-all” encoding was applied.

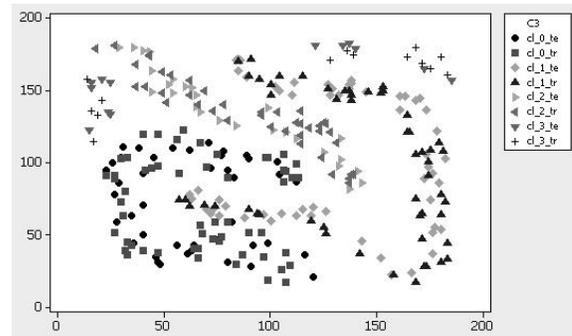


Fig 3. The Sinuous data, 4 classes, non-linearly separable classes with wide margins between them. 3% of the original data is used; TR -56/54/41/14, TE - 42/58/27/13 instances. Classes 0 - 3 in TR are marked cl_{0_tr} - cl_{3_tr} , classes 0 - 3 in TE are marked cl_{0_te} - cl_{3_te} respectively.

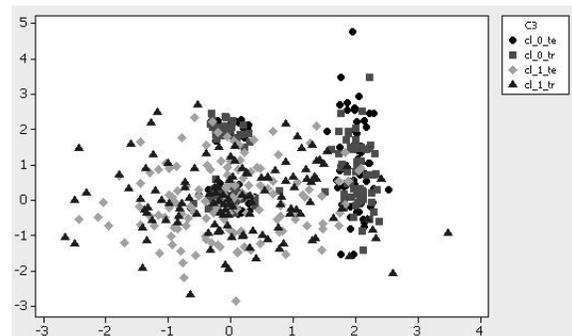


Fig. 4. The Clouds data set, 2 classes, 250/250 instances in training (TR) and test (TE) sets, heavily interleaved classes. Class 0 and class 1 in TR are marked cl_{0_tr} and cl_{1_tr} ; classes 0 and 1 in TE are marked cl_{0_te} and cl_{1_te} respectively.

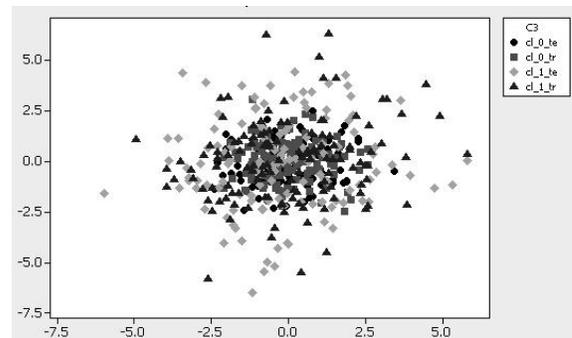


Fig 5. The 2D-Gauss data set, 2 classes, 250/250 instances in training (TR) and test (TE) sets, heavily interleaved classes. Class 0 and class 1 in TR are marked cl_{0_tr} and cl_{1_tr} ; classes 0 and 1 in TE are marked cl_{0_te} and cl_{1_te} respectively.

Selection of minimum and maximum model sizes depends on the geometrical data characteristics (density, class shapes / class boundaries and the width of margins between classes) as well as on the model generation method used. Preliminarily, the experiments have been carried out with a fixed model size limited to 4 instances per model, without

pre-estimation for the model size as a parameter. It creates 4 streams for the 2-class data (Square, Concentric, Clouds and 2-D Gauss) and 3-4 streams for different one-against-all subproblems of the 4-class data (Sinuous).

Prediction accuracy in Table 1 is obtained after 200, 500, 1000 and 5000 iterations of model generation using k -Nearest Neighbour based and cell based generation methods. On the selected data sets the cell based models provided slightly better results than k -Nearest Neighbour based models, except for the Sinuous data set. However, these results cannot be accepted as statistically meaningful.

Table 1. Prediction accuracy and model generation time

Data set	# of mod-s	SD, k -NN based model generation				SD, Cell based model generation			
		200	500	1000	5000	200	500	1000	5000
Square	Acc.	66.67	67.90	70.99	70.99	80.86	80.25	80.86	80.86
	Time	17	44	154	302	150	293	1252	5380
Concentric	Acc.	64.00	66.0	66.8	66.8	76.4	85.2	86.4	79.2
	Time	6	12	62	185	354	7664	11082	80386
Sinuous	Acc.	62.14	62.14	62.14	62.14	59.28	57.86	55.71	54.29
	Time	250	424	1317	8236	983	1889	4271	137453
Clouds	Acc.	64.80	67.20	67.20	67.20	74.80	76.0	79.6	78.8
	Time	6	14	32	204	456	2793	7352	30649
2D Gauss	Acc.	66.00	66.00	65.60	66.00	66.40	68.80	68.80	58.80
	Time	57	138	552	6305	1346	2663	5100	14908

Both implementations have demonstrated slight accuracy decrease or no accuracy increase for 5000 iterations over 1000 iterations. Such performance is a result of the increased coverage of the instances that reside on the class boundaries. Wider margins between classes imply better performance for the SD method. The main achievement is that the model generation process is about 3 times faster with k -Nearest Neighbor based models compared to cell based models.

5 Conclusions

There are few known implementations of the SD method following the theory. All of them perform along the lines predicted by the theory with varying success on different data sets. This research attempts to uncover implementation details of model generation, the most time consuming and critical part of the method, and make a step towards its improvement.

The paper discusses several model types suggested in the literature. Performance of model generation in SD with two model types was thoroughly analyzed while searching for better implementation. As a result, some improvements are suggested. The cell based implementation requires more time for model generation, and is very sensitive to parameter settings established after pre-estimation of the minimum and maximum model size, the enrichment and uniformity thresholds made on a validation data set.

k -Nearest Neighbors based model generation is more deterministic due to the restricted number of possible model variants coming from random choice of the initial instance. It

creates several restrictions, for example, the number of possible models satisfying both the uniformity and enrichment conditions appears to be less than for the cell based implementation. Not all of the potential models are reachable due to the conflict between the uniformity and enrichment conditions. However, the model generation process is about 3 times faster with k -Nearest Neighbor based models compared to cell based models maintaining the same accuracy level. The parallel model generation approach increases chances to get at least one combination of positive and negative instances in a particular model size stream, such that enough models can be collected for an acceptable accuracy level.

6 References

- [1] R. Berling, "An alternative method of stochastic discrimination with applications to pattern recognition". Ph.D. dissertation, Dept. Math., State Univ. of New York at Buffalo, NY, 1994.
- [2] Ceredyn Systems Web-page. Available: <http://web.archive.org/web/20060428221610/>
- [3] D. Chen, "Estimates of classification accuracies for Kleinberg's method of stochastic discrimination in pattern recognition". Ph.D. dissertation, Dept. Math., State Univ. of New York at Buffalo, NY, 1998.
- [4] D. Chen, X. Cheng, "A simple implementation of the stochastic discrimination for pattern recognition", *LNCS*, vol. 1876, 2000, pp. 882-887. [*Proc. Joint IAPR Int. Workshop on Advances in Pattern Recognition*, Alicante, Spain, 2000].
- [5] D. Chen, P. Huang, and X. Cheng, "A concrete statistical realization of Kleinberg's stochastic discrimination for pattern recognition, Part I. Two-class classification", *The Annals of Statistics*, vol. 31, no. 5, pp. 1393-1412, 2003.
- [6] T. K. Ho, "A numerical example on the principles of stochastic discrimination", *Tutorial lecture notes*, presented at the 2004 Int. Conf. Pattern Recognition, Cambridge, U.K.
- [7] T. K., Ho and E. Kleinberg, "Building projectable classifiers of arbitrary complexity", in *Proc. 13th Int. Conf. on Pattern Recognition*, Vienna, 1996, pp. 880-885.
- [8] E. Kleinberg, "Stochastic discrimination", *The Annals of Math. and Art. Intel.*, vol. 1, pp. 207-239, 1990.
- [9] E. Kleinberg, "An overtraining-resistant stochastic modeling method for pattern recognition", *The Annals of Statistics*, vol. 24, no. 6, 1996, pp. 2319-2349.
- [10] E. Kleinberg, "A mathematically rigorous foundation for supervised learning", *LNCS*, vol. 1857, 2001. [*Proc. 1st Int. Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2000].
- [11] E. Kleinberg, "On the algorithmic implementation of stochastic discrimination", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 5, pp. 473-490, May 2000.
- [12] D.C. Lambert. Stochastic discrimination utilities (Version 0.91) [Computer Program]. Chicago, IL.
- [13] Project ELENA, UCL Microelectronics Laboratory, Université catholique de Louvain, Belgium. Available: <http://www.dice.ucl.ac.be/neural-nets/Research/Projects/ELENA/elena.htm>
- [14] S. Singh, M. Singh, and M. Markou, "Feature selection for face recognition based on data partitioning", in *Proc. 16th Int. Conf. on Pattern Recognition*, vol. 1, Quebec, 2002, pp. 680-683.

Regression based Incremental Learning through Cluster Analysis of Temporal data

Syed Zakir Ali¹, P.Nagabhushan², and Pradeep Kumar.R³

¹and ²Department of Studies in Computer Science, University of Mysore, Mysore, Karnataka, India

³ Amphisoft Technologies Private Limited, Coimbatore, Tamil Nadu, India

Abstract - The creation of overall knowledge employing "learning at one go" is not possible when the flow-in of data is temporal. In this research work, a new regression based incremental learning model which employs local cluster analysis is proposed to build up the overall knowledge by successive updating process. It is also proposed to devise a Zero-Memory incremental learning model that does not re-indent the past data in the process of knowledge updation, enabling the minimization of operational memory and other computational overheads.

Keywords: Incremental Learning, Zero-Memory Learning, Local Clustering, Knowledge updation, Regression Analysis, Temporal data.

1 Introduction

One time learning or *learning at one go* has remained a traditional method of learning in Pattern Recognition/Machine Learning for a very long time. With the increase in size of data and with the emergence of Data Mining, the methods of learning have to deviate from the traditional *one shot* learning. In other words learning process cannot afford to remain as non-incremental.

One shot learning to extract the most useful knowledge is possible provided the entire data is available in one go and also the volume of data is computationally manageable. However this is untrue when the availability of data depends on arrival of data at delayed time instants as in case of temporal flow-in of data [3]. On the contrary this is blessed with a boon of availability of computationally manageable batches of data, which can be subjected to local cluster analysis to get the knowledge about that batch of data. The overall knowledge can be built by incremental updation. This process is generally termed as incremental learning in literature [7] [10] [22].

Langley P in [10] has observed that most research on incremental learning rests on the assumption that the learner should make use of the learnt knowledge at any stage of learning.

In machine learning community, knowledge improvement in incremental learning is achieved in three forms [11] depending on the volume of data from which the earlier knowledge was generated, is re-processed or is available for

re-processing. They are: (i) Zero memory learning, where none of the earlier processed data is re-called or re-indent (ii) Full memory learning where all the past data are retained and re-processed and (iii) Partial memory learning where some of the past data which are of importance and are likely to play a vital role in the learning process are retained and re-processed. The problem of memory based learning approaches is the requirement of large capacity memory [20]. However, Zero-memory learning is the most optimal in terms of space requirements; and it has been very difficult to achieve [11].

In computer vision and pattern recognition, learning is classified as supervised, unsupervised, semi-supervised and reinforcement [12] [13]. The unsupervised learning, also called as clustering is very much useful in several exploratory pattern-analysis, grouping and decision making including image segmentation and data mining [2]. In semi-supervised and reinforcement learning one can feel a flavor of the beginning of incremental learning. In reinforcement learning, results generated at intermediate levels will be sent as a feedback to the next level of processing.

To incorporate the concepts of incremental learning in data mining, Michalski in [16] has observed that provision should be made for inserting background knowledge and knowledge about the goal into the knowledge generation process and has coined the process as Incremental Knowledge Mining. In [16], the following model for the knowledge generation process is proposed:

$$DATA + PRIOR_KNOWLEDGE + GOAL \rightarrow NEW_KNOWLEDGE$$

The work of Christophe [4], traces many motivating reasons for incremental learning. It has been observed in [4] that learning takes place over time rather than as a one shot experience. It is argued that incrementality is rather ubiquitous in learning.

Incremental data clustering has been explored in [5] [8] [12]. In [3], a model for dynamic/on-line clustering of data is proposed. The approach for clustering is based on some observations about how the arrival of new data can perturb/change the existing clusters and how this change in clusters is evaluated quantitatively. It is observed that (i) the judgment of a cluster splitting situation is difficult and computationally unattractive, (ii) density is an important

parameter for the merging as well as for the creation of clusters and (iii) in the mechanism of dynamic clustering, if a cluster is not detected or if two clusters are not merged at some stage, the results may be modified at a later stage. This is considered as an advantage of dynamic clustering.

In [17], a clustering algorithm which uses class labels as knowledge is presented. Here it is important to note that classification represents an important *source of knowledge*.

In [14], an incremental version of the Density Based Spatial Clustering of Applications of Noise (DBSCAN) algorithm for clustering the data is proposed. It is argued that DBSCAN is one of the most efficient algorithms on large databases and is applicable to any database containing data from metric space to a spatial database or to a WWW-log database. Further it is said that insertions/deletions of objects affects the current clustering only in the neighborhood of this object because of density based nature of DBSCAN. Because of the advantages, in this research, for local cluster analysis we have used DBSCAN algorithm.

All the above works concentrate on the incremental agglomeration of data. While extracting the knowledge of clusters of the subsequent batches, every element of the cluster is reprocessed again. In other words, the process is a Full –Memory learning [11]. Rather if the new data is just added to the knowledge of each cluster to get the updated knowledge; it would have saved a lot of computational effort. To the best of our knowledge, we could not find efforts for incremental agglomeration of knowledge packets in the knowledge generation process.

In this paper, we concentrate on achieving incremental learning in the context of data arriving temporally, where incremental learning is the only possibility or a better alternate. Here, incremental learning is achieved through incremental clustering. We propose a model and an algorithm for incremental addition of knowledge packets for updation of knowledge. The net effect is that we do not propose to indent the past data in the updation process. In other words, we aim at Zero-Memory learning [11]. Regression based distance measure [18][19] is used to compute the distance between the knowledge packets. Section 2 introduces our approach. In Section 3, the new model is devised and the algorithm is presented. In Section 4, experimental results are provided. Section 5 provides conclusion and further avenues.

2 The Proposed Approach

During incremental learning, we believe that due to arrival of new data one or more of the following situations may arise. These are inline with the situations proposed by Chaudhuri in [3] for dynamic data clustering.

- (i) *Absorption of data*: Knowledge packets created from the available data are sufficient and are justified. No new/additional knowledge can be derived with the addition of new samples.
- (ii) *Merging of Knowledge packets*: If $K1 \rightarrow K2$ and $K2 \rightarrow K1$ or if a single knowledge appears as two knowledge components earlier because of incomplete learning due to

lack of samples, then in the subsequent learning process, two knowledge packets get merged in to one.

- (iii) Subsequent learning with additive samples could result in the *formation/creation of an additional knowledge packet*.
- (iv) The new inflow of samples may not create a new knowledge, may not support the existing knowledge, but could presently appear to be *stray* instances. However with the arrival of next set of samples in a further slab, the knowledge scenario could get updated, because of the interaction of the new samples with the so called stray instances.
- (v) The continued addendum of samples into existing knowledge packets could cause uneven distribution of knowledge (density of samples) within a knowledge packet ending in the splitting of one knowledge packet in to two distinct packets of knowledge with a clear boundary/separation between them. The cluster splitting situation is a challenging task even when the complete data is available. Since we have to work only on the knowledge parameters there should be some mechanism at the time of merging itself which can avoid splitting of the cluster at a later stage.

In case of *data arriving temporally*, a batch / chunk of data would be available at the end of some interval of time. Hence in this case there cannot be any ambiguity in the sequence of data chunks / packets to be processed. In general, from an i^{th} data batch $[B]_i$, i^{th} knowledge set $[K]_i$ can be derived. Sooner $[K]_i$ is available, if it has to be merged with the existing knowledge $[Knowledge]_{i-1}$ which is the one updated up to the end of $(i-1)^{th}$ stage, then,

$$[Knowledge]_i \leftarrow f([Knowledge]_{i-1}, [K]_i);$$

$$\text{where, } [Knowledge]_{i-1} \leftarrow f([Knowledge]_{i-2}, [K]_{i-1}),$$

$$[Knowledge]_{i-2} \leftarrow f([Knowledge]_{i-3}, [K]_{i-2}) \text{ and so on.}$$

Finally overall knowledge is:

$$[Knowledge] \leftarrow [Knowledge]_n \text{ if } n^{th} \text{ stage is the last stage.}$$

In [18] [19], it has been shown that the knowledge of any kind of data (generic data) can be conveniently represented by a histogram and the knowledge assimilated in histograms can further be compressed by converting them to regression lines. Apart of the number of elements in each cluster, mean and standard deviation, we can take regression lines as knowledge parameters. For an n-dimensional data space, if the current batch $[B]_i$ shows up ‘k’ knowledge packets $[Cl_1] [Cl_2] \dots [Cl_k]$ knowledge structure $[k]_i$ is as shown in table I.

Table I. Knowledge Structure $[K]_i$ of batch B_i

$[B]_i$	No. of elements	f_1		f_2		...	f_n	
$[Cl_1]$	n_1	${}^i\mu_1^1$	${}^i\sigma_1^1$	${}^iL_1^1$	${}^i\mu_1^2$	${}^i\sigma_1^2$	${}^iL_1^2$...
$[Cl_2]$	n_2	${}^i\mu_2^1$	${}^i\sigma_2^1$	${}^iL_2^1$	${}^i\mu_2^2$	${}^i\sigma_2^2$	${}^iL_2^2$...
...
$[Cl_k]$	n_k	${}^i\mu_k^1$	${}^i\sigma_k^1$	${}^iL_k^1$	${}^i\mu_k^2$	${}^i\sigma_k^2$	${}^iL_k^2$...

Where, μ – Mean, σ – Standard deviation/Density,
 L – Regression Line in terms of slope and intercept, Cl – Cluster

3 The New Computational Model

The basic model for updation of knowledge after the processing of i^{th} batch $[B]_i$ is:

$$[Knowledge]_i \leftarrow f([Knowledge]_{i-1}, [K]_i) \quad \text{--- (1)}$$

Where, $[K]_i$ is the set of knowledge packets obtained by the local cluster analysis on the batch $[B]_i$ and $[Knowledge]_{i-1}$ is the set of knowledge packets obtained by the incremental agglomeration up to the end of $(i-1)^{\text{th}}$ batch.

Let us assume that updated $[Knowledge]_{i-1}$ has 'p' knowledge packets up to the end of $(i-1)^{\text{th}}$ batch.

$$[Knowledge]_{i-1} = \{[CL]_1, [CL]_2, \dots, [CL]_p\} \quad \text{--- (2)}$$

Where $[CL]_x = \{\mu_x, \sigma_x, L_x\}$;

Let us assume that the present batch of data $[B]_i$ has resulted in k number of knowledge packets.

$$[K]_i = \{[C]_1, [C]_2, \dots, [C]_k\} \quad \text{--- (3)}$$

Where $[C]_j = \{\mu_j, \sigma_j, L_j\}$;

The entire process of representation of knowledge packet in terms of histogram, transformation of histogram to normalized histogram then to cumulative histogram and subsequently to normalized regression line is presented in detail by Pradeep Kumar and Nagabhushan in [18] [19]. For the sake of completeness, a summary of the regression distance measure is made available in the appendix.

The creation of $[Knowledge]_i$ is obtained by updating $[Knowledge]_{i-1}$ based on the relation (1). To enable this it is required to find the nearest knowledge packets one from $[Knowledge]_{i-1}$ and the other from $[K]_i$. The nearest two knowledge packets $[CL]_x \in [Knowledge]_{i-1}$ and $[C]_j \in [K]_i$ is accomplished by measuring the distance between the corresponding lines of regression $L_x \in [Knowledge]_{i-1}$ and $L_j \in [K]_i$. If two clusters $[CL]_x$ and $[C]_j$ are mergeable then upon the proposed merging operation,

$$\text{new_}\mu = \left(\frac{(n_j * \mu_j) + (n_x * \mu_x)}{n_j + n_x} \right) \quad \text{--- (4)}$$

$$\text{new_}\sigma = \sqrt{((n_j * (\sigma_j^2 + d_j^2)) + (n_x * (\sigma_x^2 + d_x^2))) / (n_j + n_x)} \quad \text{--- (5)}$$

Where, $d_j^2 = (\mu_j - \text{new_}\mu)^2$; $d_x^2 = (\mu_x - \text{new_}\mu)^2$;

new_L in terms of slope and intercept is obtained as follows:

$$\text{new_}S = ((n_j * S_j) + (n_x * S_x)) / (n_j + n_x) \quad \text{--- (6)}$$

$$\text{new_}I = ((n_j * I_j) + (n_x * I_x)) / (n_j + n_x) \quad \text{--- (7)}$$

Where: S_j is the slope of the line L_j and S_x is the slope of the line L_x , I_j is the intercept of line L_j and I_x is the intercept of line L_x .

Proofs for these formulations are well established results in statistics [15].

The merging is allowed based on the following criterion:

Let $q_1 = (\text{new_}\sigma - \sigma_j)$; and let $q_2 = (\text{new_}\sigma - \sigma_x)$; then $p = \min(q_1, q_2)$;

Merging of the two batches can be done if $(q < = \text{predefined range})$; Here the *predefined range* is the prior knowledge [16] about the density of the dataset.

At the end of this procedure $[Knowledge]_i$ is created which in the worst case upon no merge happening results in $(p+k)$ packets of knowledge. Upon complete merging of the packets the resulting number of knowledge packets produced is $\min(p,k)$. Therefore the number of knowledge packets will be in between $[\min(p,k), (p+k)]$. In fact it is possible that the number of knowledge packets at each level could be less than the number of packets as worked out above, because of induction of continued merging process. This aspect, is not considered while designing the new model.

The above process of updating continues till all n batches are processed. Finally $[Knowledge] \leftarrow [Knowledge]_n = f([Knowledge]_{n-1}, [k]_n)$. However the process of merging has to be continued to get the optimal number of knowledge packets at the end. Finally the process is terminated when the desired numbers of knowledge packets are produced or the distance between the resultant knowledge packets is greater than the predefined threshold.

While designing the new model, we have also stressed upon maintaining the zero-memory status, i.e., at any stage of combining $[Knowledge]_{i-1}$ with $[k]_i$, the data of batch $[B]_{i-1}$ or batches prior to $[B]_{i-1}$ will not be re-indented. This keeps the updating algorithm computationally simple and the amount of operational memory required is kept low. However because of this the samples which could not contribute to the formulation of knowledge packet in an earlier batch, are treated as outliers and are simply rejected. This is in fact a demerit because all samples are not made to participate in the creation of overall knowledge, and it is possible that such rejected outliers together might create altogether some new knowledge packets.

It should also be observed that a suitable strategy is presented for merging the knowledge packets, but not for splitting a packet. Our argument is that since merging operation is prevented under the unfavorable conditions, the question of subsequent splitting does not arise.

Algorithm: Zero Memory Learning

Input: Temporal flow-in of data. Let us assume that a batch of data samples is available at i^{th} time instant.

Consider the first/initial batch of data; Set the parameters *Eps* – the radius that delimitate the neighborhood area of a point and *MinPts* – the minimum number of points that must exist in the *Eps*-neighborhood. Apply the DBSCAN clustering algorithm to find natural groups/clusters in the given data. Outliers to be removed strictly; let us say 'p' natural clusters are obtained. The 'p' clusters are the 'p' knowledge packets.

```

do for all the clusters of initial batch
  extract_knowledge to obtain 'p' knowledge_packets;
end do

do for all the knowledge packets of the initial batch
  find_distance between the knowledge_packets and record
  them in the distance_matrix;
end do

do for the batches of data arriving sequentially
  Apply the same clustering algorithm (which was applied
  to initial batch of data); let us say 'k' natural clusters are
  obtained.
do for the 'k' natural clusters obtained
  extract_knowledge;
  find_distance between this packet of knowledge to
  all the knowledge_packets available in the initial
  batch;
  Record the packet number which shows the
  minimum distance; the packet number is
  destination_pkt_number and the minimum distance is
  min_val;
  In the distance_matrix, search the row/column
  represented by the destination_pkt_number for a
  value less than min_val.
if found
  Move this knowledge packet as an additional
  knowledge packet of the initial batch; This
  increases the knowledge_packets of the initial
  batch by one.
  Re-compute the distance_matrix;
else Check density_of_the_proposed_merge;
  if density_of_the_proposed_merge is in a pre-
  defined acceptable range,
  Merge this knowledge packet with the
  knowledge packet of the initial batch which is
  represented by the destination_pkt_number;
  Re-compute the distance_matrix;
else Move this knowledge packet as an additional
  knowledge packet of the initial batch; This
  increases the knowledge_packets of the initial
  batch by one.
  Re-compute the distance_matrix;
  end if;
end if;
end do;
end do;
do while number of knowledge packets > Goal
  find the packets with minimum distance;
  if they can be merged as per the
  density_of_the_proposed_merge, merge them;
  reduce the number of knowledge packets by one;
  Re-compute the distance_matrix;
else set minimum distance to a very high value;
  end if;
end do;

```

Algorithm: extract_knowledge

```

do for each cluster
  do for each dimension of cluster
    get the mean ( $\mu$ ); get the standard deviation ( $\sigma$ );
    construct a histogram; normalize the histogram and
    keep the number_of_elements as knowledge;
    fit a first order polynomial to the normalized histogram
    to obtain a regression line ( $L$ );
  end do;
end do;

```

4 The Experiment

For the sake of clear understanding and a proper visual demonstration of the proposed method, we have used a synthetic dataset of 5000 points in a 2-dimensional space with five classes of similar sizes with a very high density. The dataset is as shown in figure 1.

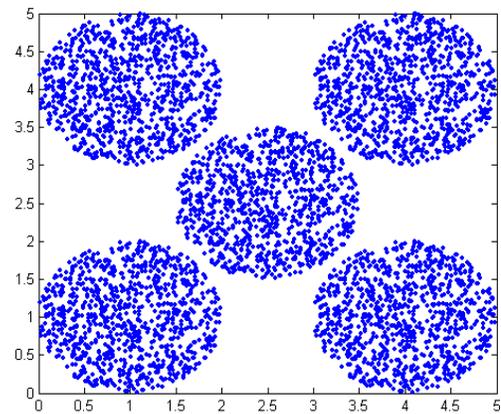


Fig 1. Initial Dataset

We randomly arranged the samples of the dataset and divided the dataset into four batches of equal size and processed the four batches one at a time in a sequential order simulating the temporal arrival of data. For DBSCAN, *Eps* was set to 0.25 and *MinPts* was set to 20. We performed experiments on this synthetic dataset by dividing it into more number of batches as well as with different sequence of batches and we obtained similar results. Hence we restrict our discussion to the dataset which was divided into four batches of equal size. Batch B_1 is considered as the initial batch. Upon applying DBSCAN, we obtained 22 clusters which are as shown in figure 2.

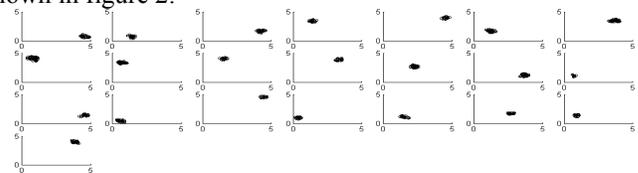


Fig 2. Twenty-Two clusters/Know Packets of batch B_1

In figure 2, first row indicates cluster numbers/knowledge packet numbers 1 to 7, the second row indicates 8 to 14, third row indicates 15 to 21 and the final row indicates packet

number 22. The same concept is followed for the other figures. Out of the 1250 elements of the batch B_1 , 566 elements were marked as outliers (because of the stringent parameters Eps and $MinPts$ of DBSCAN) and were thrown out as per the requirements of zero-memory learning [11].

From batch B_2 , we obtained 19 clusters which are as shown in figure 3.

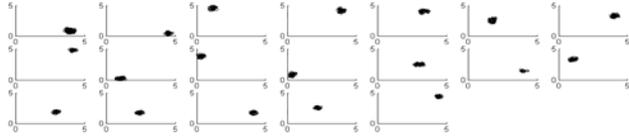


Fig 3. Nineteen Clusters of batch B_2

Out of these 19 clusters, 14 got merged in some of the knowledge packets created from batch B_1 , which is now K_1 , using the regression distance measure [18] [19]. The status of knowledge packets at the end of B_2 is as shown in figure 4.

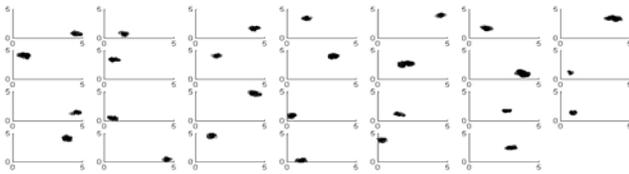


Fig 4. Status of knowledge packets at the end of B_2

Similarly the clusters obtained by B_3 is as shown in fig 5, status after B_3 is as shown in fig 6, clusters obtained by B_4 is as shown in fig 7 and status after B_4 is shown in fig 8.

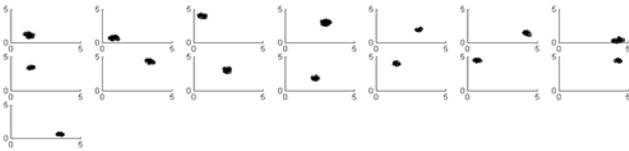


Fig 5. Fifteen clusters of batch B_3

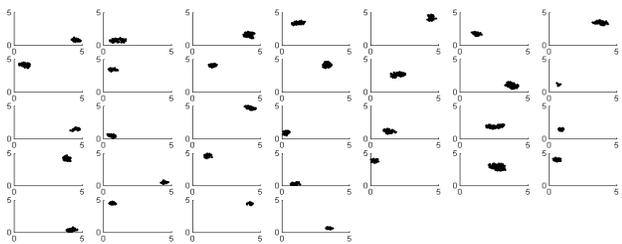


Fig 6. Status of knowledge packets at the end of B_3

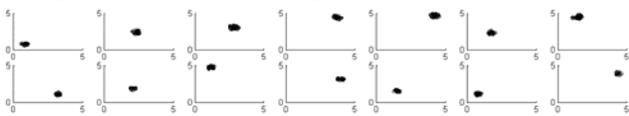


Fig 7. Fourteen clusters of batch B_4

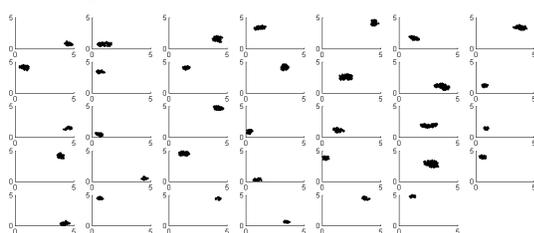


Fig 8. Status of knowledge packets at the end of B_4

For the sake of clarity the cluster state which shows the merger of clusters into the knowledge packets is as shown in table II. Columns 1, 2 and 3 of table II indicates packet numbers of batch B_2 , packet number of K_1 with which the corresponding packet of batch B_2 got merged and packet numbers at the end of batch B_2 (which is now K_2) respectively. Similarly, columns 4, 5 and 6 are for batch B_3 and columns 7, 8 and 9 are for batch B_4 .

Table II. Knowledge Generation Process

B_2 Pkts	Merger with K_1	K_2 Pkts	B_3 Pkts	Merger with K_2	K_3 Pkts	B_4 Pkts	Merger with K_3	K_4 Pkts
1	13	1	1	19	1	1	2	1
2	0*	2	2	2	2	2	12	2
3	0*	3	3	0*	3	3	27	3
4	22	4	4	27	4	4	0*	4
5	11	5	5	20	5	5	17	5
6	12	6	6	3	6	6	12	6
7	7	7	7	0*	7	7	24	7
8	17	8	8	4	8	8	13	8
9	0*	9	9	11	9	9	20	9
10	0*	10	10	27	10	10	0*	10
11	18	11	11	20	11	11	7	11
12	0*	12	12	10	12	12	6	12
13	3	13	13	0*	13	13	14	13
14	4	14	14	0*	14	14	5	14
15	20	15	15	0*	15			15
16	20	16			16			16
17	3	17			17			17
18	12	18			18			18
19	5	19			19			19
		20			20			20
		21			21			21
		22			22			22
		23			23			23
		24			24			24
		25			25			25
		26			26			26
		27			27			27
					28			28
					29			29
					30			30
					31			31
					32			32
								33
								34

* indicates unmerged clusters.

Dynamic merging of the nearest knowledge packets repeatedly yields the five knowledge packets as shown in figure 9.

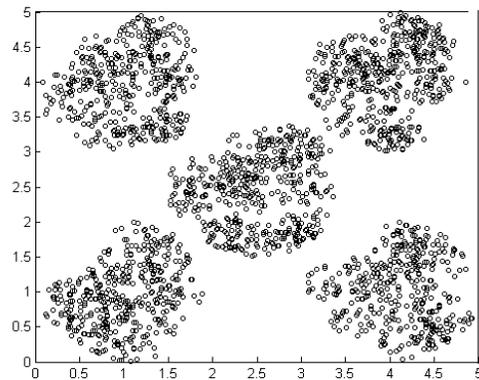


Fig 9. Final Five Knowledge packets

The details of the number of clusters/knowledge packets, outliers at each stage, number of unmerged clusters are summarized in the table III.

Table III. Summary of Knowledge Packets

Batch No	Number of elements	Outliers	Number of knowledge packets	Number of unmerged knowledge packets	Number of knowledge packets at the end of each batch
B_1	1250	566	22	Not applicable	22
B_2	1250	660	19	05	27
B_3	1250	776	15	05	32
B_4	1250	858	14	02	34

Number of elements in each of the final knowledge packet is 388, 402, 417, 470 and 463 respectively. Table IV gives the summary of deviation of knowledge parameters of all the five knowledge packets and table V gives the average summary.

Table IV. Summary of deviation of Knowledge parameters

Knowledge Parameter	Values		Know. Packet 1		Knowledge Packet 2		Knowledge Packet 3		Knowledge Packet 4		Knowledge Packet 5	
	Actual	Computed	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
μ	1.00	0.99	1.00	0.99	4.00	0.99	2.50	2.50	1.00	3.99	4.00	4.00
Diff(%)	1	5	0.99	0.94	4.11	1.03	2.50	2.42	1.05	3.96	3.96	4.07
σ	0.54	0.45	0.54	0.45	0.54	0.45	0.54	0.45	0.54	0.45	0.54	0.43
Computed	0.42	0.44	0.41	0.48	0.41	0.48	0.45	0.49	0.41	0.49	0.41	0.48
Diff(%)	22	2	24	6.6	24	6.6	16.6	8.8	24	8.8	24	11.6
L (slope)	0.14	0.14	0.21	0.14	0.21	0.14	0.29	0.29	0.14	0.21	0.21	0.21
Computed	0.14	0.13	0.20	0.15	0.20	0.15	0.29	0.28	0.15	0.22	0.22	0.21
Diff(%)	0	7	4.7	7.1	4.7	7.1	0	3.4	7.1	4.7	4.7	0
L (Intercept)	0.49	0.50	0.29	0.50	0.29	0.50	0.17	0.17	0.49	0.29	0.29	0.29
Computed	0.49	0.52	0.28	0.47	0.28	0.47	0.17	0.15	0.45	0.29	0.30	0.28
Diff(%)	0	4	3.4	6	3.4	6	0	11.7	8.1	0	3.4	3.4

Table V Summary of average deviation in knowledge parameters

Knowledge parameter	f_1	f_2
μ	0.107	0.053
σ	0.123	0.030
L (slope)	0.006	0.005
L (Intercept)	0.012	0.016

Even after a loss of 57% of the samples as per the requirements of the zero-memory learning [11], we are able to get back the five classes and the average deviation of less than 5% in μ and L is meager. This indicates the power of regression line as a knowledge representative and is very much suitable for zero-memory learning. However, a deviation of 22% in σ of f_1 has to be improved.

We have also tested our concept on the standard iris dataset [21] by carefully avoiding the overlapping samples. The standard iris dataset has 150 points in 4-dimensional space. First 50 samples belongs to class 1; the second 50 belongs to class 2 and the third 50 belongs to class 3; Class 1 is clearly separable from class 2 and 3, whereas class 2 and 3 are unseparable. We have selected 40 samples from class 1, 30 non overlapping samples each from class 2 and 3 respectively. These samples were arranged in random order and divided into four batches of equal size and processed in sequential order to simulate the temporal arrival of data. For clustering using DBSCAN, the parameters *Eps* and *MinPts* were set to 0.75 and 02 respectively. Finally we got three clusters/knowledge packets with 39, 28 and 29 elements respectively as some of the elements were identified as outliers and were discarded as per the requirements of zero-memory learning [11].

5 Conclusions

A new conceptual model is presented for incremental addition/update of knowledge packets to the already existing knowledge packets resulting in updated repository of knowledge packets. The results obtained are quite encouraging and it is expected that the proposed model will provide a new dimension to incremental clustering. Since regression based distance measure is used, the model can support multi dimensional generic datasets/databases. The advantage of zero memory learning algorithm is that it is computationally simple and the operational memory required is kept low. The demerit of the algorithm is that the samples which could not contribute to the formation of a knowledge packet in an earlier batch are treated as outliers and are simply rejected, which could have been of importance in the subsequent stages of learning.

6 Appendix

The regression distance measure invented by Pradeep Kumar and P. Nagabhushan [18] [19] is summarized below. Consider a histogram H with 10 bins; $H = \{b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8, b_9, b_{10}\}$ where b_i is the frequency count of the bin centered at C_i . For example the corresponding histogram for the data say $A = \{10\ 30\ 40\ 50\ 40\ 30\ 20\ 20\ 30\ 10\}$ is as shown in fig 10.

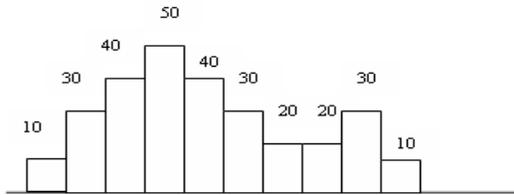


Fig 10. A Sample Histogram

Now a cumulative frequency distribution is computed for each of these 10 centers resulting in the cumulative histogram (CH); $CH = \{cn_1, cn_2, cn_3, cn_4, cn_5, cn_6, cn_7, cn_8, cn_9, cn_{10}\}$ where $cn_i = \sum (cn_k)$ for $k = 0$ to i ; for the histogram of fig 10, CH becomes $\{10\ 40\ 80\ 130\ 170\ 200\ 220\ 240\ 270\ 280\}$ and the cumulative histogram is as shown in fig 11.

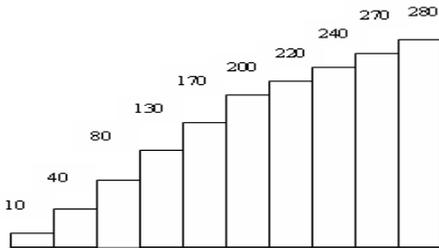


Fig 11. Cumulative Histogram for the histogram of fig 10
CH is then normalized by dividing cn_i for $i = 1$ to 10 by cn_{10} . Now 10 points are marked on the top of each bin in the CH corresponding to the bin centers and a first order polynomial is fitted across these 10 points to obtain regression line with y_i ranging between 0 and 1 and x_i 's range is decided by the minimum and maximum co-efficient values at a particular scale. The regression line fitting is done as shown in fig 12.

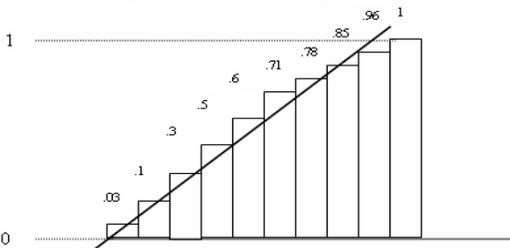


Fig 12. Regression Line fitting on a normalized histogram
Distance between such obtained lines can be computed by finding the distance between the two lines as well as by finding the behavior of the two lines. The calculation of distances is as shown in fig 13.

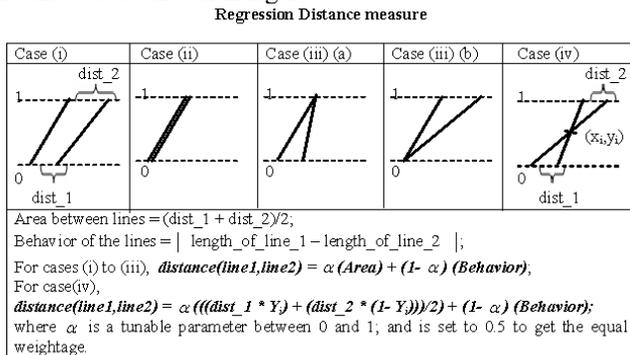


Fig 13. Distance between two lines

7 Acknowledgement

First Author wishes to thank the executives and the staff members of the Caledonian College of Engineering, Sultanate of Oman; Ms. Pavitha of the department of Statistics, Mr. Manjunatha S. and Mr. Prabhudev Jagdeesh - the Research scholars of the Department of Studies in Computer Science, University of Mysore for their continuous encouragement and support.

8 References

- [1] Anil K Jain & Dubes R.C, "Algorithms for Clustering data", Prentice Hall, Englewood Cliffs, NJ, 1998.
- [2] Anil K Jain, M.N Murthy and P.J Flynn, "Data Clustering: A Review", ACM Computing surveys, vol 31, No. 3, September 1999.
- [3] Chaudhuri B.B. "Dynamic Clustering for Time Incremental Data", Pattern Recognition Letters 15, pp 27-34, 1994.
- [4] Christophe G.C, "A Note on the Utility of Incremental Learning", AI Communications, vol 13, Issue 4, pp 215-223 Dec 2000.
- [5] Fazli C, Edward A.F, Cory D.S and Robert K.F "Incremental Clustering for Very Large Document Databases: Initial MARIAN Experience", An International Journal of Information Sciences—Informatics and Computer Science, vol 84, Issue 1-2, 1995.
- [6] Han & Kamber, "Data Mining – Concepts and Techniques", Second Edition, Elsevier, pp 51 -56; 2006.
- [7] Kaufman K.A & Michalski R.S, "Initial Considerations towards Knowledge Mining", Machine Learning and Inference Laboratory, George Mason University, 2004.
- [8] Khalid M. Hammouda, Mohammed S. Kamel, "Incremental Document Clustering Using Cluster Similarity Histograms", IEEE/WIC International Conference on Web Intelligence, pp 597-601, Halifax, Canada, October 2003.
- [9] Klaus P Jantke, "Types of Incremental Learning", Technical Report SS-93-06 Published by The AAAI Press, Menlo Park, California, 1993.
- [10] Langley P, "Order Effects in Incremental Learning", P. Reimann & H. Spada (Eds), "Learning in Humans and Machines: Towards an Interdisciplinary Learning Science", Elsevier, Amsterdam, 1995.
- [11] Maloof A.M & Michalski R.S, "Incremental Learning with partial instance memory", Artificial Intelligence 154, 95-126, 2004.
- [12] Martin Ester, Hans-Peter Kriegel, Jorg Sander, Michael Wimmer, Xiaowei Xu, "Incremental Clustering for Mining in a Data Warehousing Environment" Proceedings of the 24th VLDB conference New York, 1998.
- [13] Martin H.C.Law, "Clustering, Dimensionality Reduction And Side Information", Doctoral Dissertation of the Michigan State University, 2006.
- [14] Masseglia F, Poncelet P, Teisseire M, "Incremental Mining of Sequential Patterns in Large Databases", Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004
- [15] Michael J Parik, "Advanced Statistics from an elementary point of view", Elsevier Academic Press; ISBN 13:978-0-12-088494-0;
- [16] Michalski. R.S, "Knowledge Mining: A Proposed New Direction", Invited talk at the Sanken Symposium on Data Mining and Semantic Web, Osaka university, Japan, March 10-11, 2003.
- [17] Narasimhamurthy .M. & Sridhar V, "A Knowledge-based Clustering algorithm", Pattern Recognition Letters 12, 511-517, 1991.
- [18] Pradeep Kumar R, "Wavelets for Knowledge Mining in Multi Dimensional Generic Databases", PhD Thesis of the University of Mysore, 2006.
- [19] Pradeep Kumar R & P Nagabhushan, "Multi Resolution Knowledge Mining Using Wavelet Transform" Engineering Letters, 14:1;EL_14_1_30; 2007
- [20] Seiichi ozawa and Nikola kasabov, "Incremental Learning on Chunk Data for Online Pattern Classification Systems" IEEE Transactions on Neural Networks; 2008.
- [21] UCI Machine Learning Repository; <http://archive.ics.uci.edu/ml/datasets/Iris>
- [22] Widmer, G. and Kubat, M., "Learning in the presence of Concept Drift and Hidden Concepts", Machine Learning, 23, pp 69-101, 1996.

Study of Dhaka Stock Exchange- Efficient Market Hypothesis

Jarka Arefin, Rashedur M Rahman

Department of Electrical Engineering and Computer Science
North South University, Dhaka, Bangladesh

Abstract - *The efficient-market hypothesis (EMH) (Fama et.al [2]) asserts that efficient markets are informationally efficient or all information should reflect on stock prices. No one can earn excess profit using any kind of information in efficient market. There are three form of efficiency in market: Strong form, Semi Strong form and Weak form. We test the Efficient Market Hypothesis for Dhaka Stock Exchange (DSE) [6] during the period 2003 to 2005. We use excess return market model and forecasting techniques to test the Semi Strong and Weak form efficiency in DSE respectively. Results show that the DSE is not an efficient market either in semi-strong or weak form.*

Keywords: Efficient Market Hypothesis, Semi Strong form efficiency, Weak form Efficiency, Forecasting, Econometric Method, Neural Network.

1 Introduction

The efficient-market hypothesis (EMH) asserts that efficient markets are informationally efficient or that prices on traded assets such as stocks, bonds, or property will reflect all known information. The efficient-market hypothesis states that it is impossible to outperform or earn excess return consistently from the market by using any information that the market already knows. According to EMH there are three forms of efficiency: strong form, semi strong form and weak form efficiency. In strong form efficient market no group of investors can earn excess profit by using any public or private information not even any monopolistic group. Semi-strong-form efficiency implies that stock prices adjust to new information available very rapidly and in an unbiased fashion such that no excess returns could be earned by trading on that information. Semi-strong-form efficiency implies that neither fundamental analysis nor technical analysis techniques will be able to produce excess returns. Excess returns could not be earned by using investment strategies based on historical stock prices in weak form efficient market. Therefore, in weak form efficient market share index is random. There is no pattern for share index.

In this research, we test the validity of Efficient Market Hypothesis for Dhaka Stock Exchange. We test the DSE for two form of efficiency, i.e., semi-strong form and weak form. Two techniques are used to test the efficiencies. The first

technique is Excess Return Model and the second one is forecasting technique. Excess return market model shows that many stocks in DSE have excess returns. But in Semi strong form efficient market no one can earn excess return using any kind of public information (e.g., dividend, price to earning ratio, etc.). So, DSE does not hold semi strong form efficiency. We also test the Weak Form Efficient Market Hypothesis for DSE during the same period from 2003 to 2005 using two forecasting techniques. We use neural network and econometric method to forecast the change in market index which shows that change in market index follows a trend whereas in weak form the change in market index should random. We forecast 30 and 60 day ahead change in market index. Econometric model gives better result for 30 day ahead forecasting for change in market index than Neural net. Neural net gives better forecasting for 60 day ahead forecasting than Econometric model. As there is a trend in change in the market index Dhaka Stock Exchange is not efficient in weak form.

Rest of the paper is organized as follows. Section 2 describes the theoretical framework for EMH. Section 3 focuses on the datasets used in this research. Section 4 discusses and analyzes the empirical results. Finally, Section 5 concludes and gives direction of future research.

2 Theoretical Framework

2.1 Efficient Market Hypothesis

The **efficient-market hypothesis (EMH)** (Fama et.al [2]) asserts that efficient markets are informationally efficient or that prices on traded assets such as stocks, bonds, or property, reflect all known information. The efficient-market hypothesis states that it is impossible to consistently outperform the market or earn the excess profit by using any information that the market already knows. Prices in the EMH can only be affected by unknown information in the present or which will appear random in the future.

The EMH was developed by Professor Eugene Fama at the University Of Chicago Graduate School Of Business in his Ph.D. thesis in the early 1960s.

2.2 Different forms of EMH

The efficient market is the one where prices of securities reflect all the information quickly and accurately. Efficient Market Hypothesis states that it is impossible to earn excess return using any kind of information both public and private. There are three common forms in which the efficient-market hypothesis is commonly stated: weak-form efficiency, semi-strong -form efficiency and strong-form efficiency. They are described below:

Weak-form efficiency: Excess returns cannot be earned by using investment strategies based on historical share prices. Technical analysis techniques will not be able to consistently produce excess returns, though some forms of fundamental analysis may still provide excess returns. Share prices exhibit no serial dependencies. There are no patterns to asset prices. This implies that future price movements are determined entirely by unexpected information and therefore are random.

Semi-strong-form efficiency: Semi-strong-form efficiency implies that share prices adjust to publicly available new information very rapidly and in an unbiased fashion, such that no excess returns can be earned by trading on that information. Semi-strong-form efficiency implies that neither fundamental analysis nor technical analysis techniques will be able to reliably produce excess returns. To test for semi-strong-form efficiency is to test the lag of the adjustment of announcing information. If there is lag for adjustment of public information and investors can earn excess profit using that information then the market is not semi strong form efficient.

Strong-form efficiency: Share prices reflect all information, public and private, and no one can earn excess returns. To test for strong-form efficiency, a market exists where no group of investors cannot consistently earn excess returns over a long period of time. Using public or private information no excess profit can not be made in strong form efficient market. In the strong form efficient market all non public information such as some insider information which is only accessible to certain monopolistic group should reflect in the stock prices. It is the highest form of efficiency of the market which encompasses both semi strong form and weak form efficiency.

2.3 Random Walk Hypothesis

The random walk hypothesis is a financial theory which states that stock market prices evolve according to a random walk and the prices of the stock market cannot be predicted in an efficient market. Economists have historically accepted the random walk hypothesis. They have experimented with several tests and continue to believe that stock prices are completely random because of the efficiency of the market.

Burton Malkiel, Professor of Economics and Finance at Princeton University, first introduced the term random walk

hypothesis in his book "A Random Walk Down Wall Street" (Malkiel et al. [3]).

2.4 Excess Return Market Model

We test the DSE for semi strong form of efficiency. We use the excess return market model for this purpose. The model is based on Capital Asset Pricing Model (CAPM) [5] and single index model. We briefly describe those models below. More details could be found elsewhere [5].

The CAPM is the equilibrium relationship between expected return and beta of asset. It says that the expected rate of return on an asset is a function of two component of the required rate of return-the risk free rate and the risk premium.

$$E(R_i) = \text{Risk-free rate} + \text{Risk premium}$$

$$= r_f + (E(R_m) - r_f) \beta_i$$

$$E(R_i) = r_f + (E(R_m) - r_f) \beta_i$$

where,

$E(R_m)$ = The Expected rate of return on the market portfolio.

β_i = the beta coefficient for asset i

$E(R_i)$ = the required rate of return on asset i

Risk premium for security i = (Market Risk premium) β_i

This relationship states that an investor requires a return on a risky asset equals to the return on a risk-free plus a risk premium and the greater the risk, the greater is the risk premium. The Security Market Line (SML) line (in Figure 1) graphically depicts the CAPM. If one plots annual return vs beta one will get the SML. Intercept is the risk free rate r_f and Slope is the market premium $(E(R_m) - r_f)$.

In Single Index Model, stock prices normally go up and down together with some common factors. These factors could be political (war or peace), economical (change in the real interest rate) or even international (oil crisis). Market index responds to some common macroeconomic factors and includes well diversified portfolios. Thus the rate of return R_{it} on stock i is related to market index R_{mt} by a linear equation of the form:

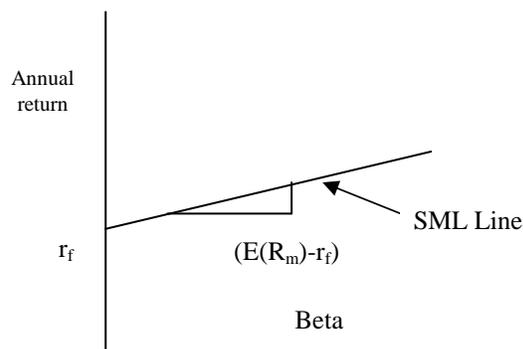


Figure 2: Security Market Line

Market index responds to some common macroeconomic factors and includes well diversified portfolios. Thus the rate of return R_{it} on stock i is related to market index R_{mt} by a linear equation of the form:

$$R_{it} = \alpha_i + \beta_i R_{mt} + \xi_{it}$$

Here,

R_{it} is the return on security i

R_{mt} is the market index for period t

α_i is the intercept term

β_i is the slope term

α and β can be obtained from regression analysis by plotting

R_{it} vs R_{mt}

ξ_{it} the random residual error

Now we first consider the market Model

$$R_{it} = \alpha_i + \beta_i R_{mt} + \xi_{it}$$

Subtract R_f from both sides

$$R_{it} - r_f = \alpha_i - r_f + \beta_i R_{mt} + \xi_{it}$$

Next, add and subtract $\beta_i r_f$ from the right hand side and rearrange to give

$$R_{it} - r_f = \alpha_i - r_f (1 - \beta_i) + \beta_i (R_{mt} - r_f) + \xi_{it}$$

$$R_{it} - r_f = \alpha^* + \beta_i (R_{mt} - r_f) + \xi_{it} \dots (1)$$

$$\alpha^* = \alpha_i - r_f (1 - \beta_i)$$

Equation 1 gives the market model where the return on the asset and the return on the market are expressed in excess of the risk-free rate r_f . This re-expressed market model is called the excess return market model. From the SML we see that CAPM imposes the restriction $\alpha_i=0$

we will test $\alpha_i=0$ in every regression.

if $\alpha^* > 0$ that is

$$\alpha^* = (R_{it} - r_f) - \beta_i (R_{mt} - r_f) > 0$$

which indicates that the asset is yielding an excess expected return higher than the CAPM predicts. One might think that it is a good model to hold. The CAPM would predict that this stock is under priced because its excess return is higher than what the CAPM predicts. In order for the expected return to fall, the current price of the stock needs to rise.

2.5 Econometric Method

We use two forecasting techniques, namely, Autoregressive Integrated Moving Average (ARIMA) and Neural Network techniques to predict the market index from historical indices. The theory behind those techniques is presented below.

2.5.1 ARIMA Model

ARIMA processes are mathematical models used for forecasting the data or better understand the data. ARIMA is an acronym for AutoRegressive, Integrated, Moving Average. Each of these phrases describes a different part of the mathematical model.

ARIMA processes have been studied extensively and are a major part of time series analysis. They were popularized by George Box and Gwilym Jenkins in the early 1970s; as a result, ARIMA processes are sometimes known as Box-Jenkins models. Box and Jenkins [1] effectively put together in a comprehensive manner the relevant information required to understand and use ARIMA processes.

The ARIMA approach to forecasting is based on the following ideas:

1. The forecasts are based on linear functions of the sample observations;
2. The aim is to find the simplest models that provide an adequate description of the observed data. This is sometimes known as the principle of parsimony.

Each ARIMA [1] process has three parts: the autoregressive (AR) part; the integrated (I) part; and the moving average (MA) part. The models are often written in shorthand as ARIMA(p,d, q) where p describes the AR part, d describes the integrated part and q describes the MA part. p,d,q are integer which are zero or greater than zero. Each of those parts are described in detail below.

AR: This part of the model describes how each observation is a function of the previous p observations. For example, if p = 1, then each observation is a function of only one previous observation. That is,

$$X_t = c + \sum_{i=1}^p \phi_i X_{t-i} + \epsilon_t$$

where X_t represents the observed value at time t, X_{t-1}, X_{t-2}, \dots represents the previous observed values at time t-1, t-2, ... and ϵ represents some random error and c is constant and ϕ_1, ϕ_2, \dots are the parameters of the model.

I: This part of the model determines whether the observed values are modeled directly, or whether the differences

between consecutive observations are modeled instead. If $d = 0$, the observations are modeled directly. If $d = 1$, the differences between consecutive observations are modeled. If $d = 2$, the differences of the differences are modeled. In practice, d is rarely more than 2.

MA: This part of the model describes how each observation is a function of the previous q errors. For example, if $q = 1$, then each observation is a function of only one previous error. That is,

$$X_t = \varepsilon_t + \sum_{i=1}^q \theta_i \varepsilon_{t-i}$$

where the $\theta_1, \dots, \theta_q$ are the parameters of the model and the $\varepsilon_t, \varepsilon_{t-1}, \dots$ are the error terms.

Combining these three parts gives the diverse range of ARIMA models.

There are also ARIMA processes designed to handle seasonal time series, and vector ARIMA processes designed to model multivariate time series. Other variations allow the inclusion of explanatory variables. ARIMA processes have been a popular method of forecasting because they have a well-developed mathematical structure from which it is possible to calculate various model features and to better understand the data in which the data fit best in the model.

2.5.2 The Hypothesis Test for Correlation test

Null hypothesis $\rho_k=0$ (there is no correlation at lag k)
 Alternative hypothesis $\rho_k \neq 0$

If the p -value for this test at lag $k > .05$ the null hypothesis is true. But if p -value is at lag $k < .05$ there is significant lag at lag k and alternative hypothesis is true.

2.6 Artificial Neural Network

Neural networks are one of the applications developed from artificial intelligence research. Neural networks were developed through studies of neuron perceptions [5]. In a sense neural networks are computer programs that are trained to behave and learn analogously to a human brain remembering some information from the past. A network is built with many neurons called nodes which are modeled on neurons from biological organisms. It is a system composed of many simple processing elements whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes. The network learns the patterns and develops the ability to forecast. BackPropagation (BP) is one of the most commonly used training algorithms for multi-layer neural networks .

A fully connected, layered, feed-forward network is depicted in Figure , where x_i, h_i, o_i represent unit activation levels of input, hidden, and output units, respectively. Weights on the connections between the input and hidden layers are denoted by $w1_{ij}$, while weights on connections between the hidden and output layers are denoted by $w2_{ij}$. The neurons marked with “1” are threshold neurons and their activation value is set to 1.

Figure 3’s network has three layers, although it is possible and sometimes useful to have more layers. Each unit in one layer is connected in the forward direction to every unit in the next layer. Activations flow from the input layer, through the hidden layer, to the output layer. The knowledge of the network is encoded in the weights on connections between units. A network typically is initialized with a random set of weights. The network adjusts its weights each time it processes an input-output pair.

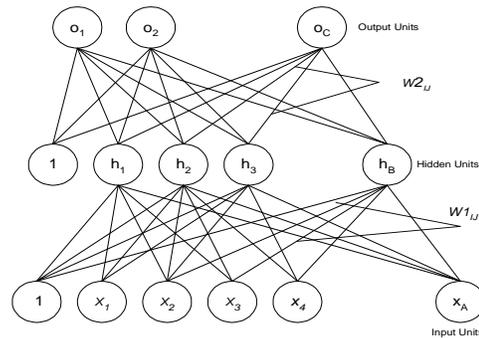


Figure 3: Multilayer FeedForward Neural network Architecture

2.7 Performance Metric

Mean Squared Error (MSE) is used to measure the accuracy of a forecasting model. This measure is the average of the sum of the squared differences between the forecast values and the actual time series values.

$$\text{Sum of the Squared error} = \sum (Y_t - F_t)^2$$

$$MSE = \frac{\sum (Y_t - F_t)^2}{n}$$

where Time series Value at time $t=Y_t$
 Forecast value at time $t=F_t$
 Forecast Error= $(Y_t - F_t)$

Root mean squared error=SQRT(MSE)

3 Data Set and Some Sample Calculation

Data has been taken from Dhaka stock Exchange's website from <http://www.dsebd.org>. We use general market index for Dhaka Stock Exchange from 2003 to 2005 for testing the Efficient Market Hypothesis. The change in monthly market index are used for forecasting $R_t = \ln(P_t/P_{t-1})$. Where P_t stands for monthly share index for month t , P_{t-1} stands for monthly share index for month $t-1$ and \ln stands for natural logarithm.

Table 1 and Table 2 represent the one and two month ahead forecasting data that has been used to predict market index by neural network and ARIMA model.

Months	General Index	Return(ln)
Jul-03	799.98	
Aug-03	793.12	-0.008612193
Sep-03	789.74	-0.004270757
Oct-03	800.59	0.013645178
Nov-03	920.61	0.139687538
Dec-03	967.88	0.050071619
Jan-04	961.18	-0.006946416
Feb-04	953.81	-0.007697206
Mar-04	973.88	0.020823603
Apr-04	1112.19	0.132798231
May-04	1185.83	0.064111907
Jun-04	1318.92	0.106370269
Jul-04	1,289.14	-0.022837891
Aug-04	1,513.29	0.160310759
Sep-04	1633.02	0.076144973
Oct-04	1710.45	0.046325432
Nov-04	1,877.05	0.092944902
Dec-04	1971.31	0.048996901
Jan-05	1843.95	-0.066788287
Feb-05	1835.62	-0.004527711
Mar-05	2030.39	0.100845594
Apr-05	1537.87	-0.277829551
May-05	1648.28	0.069333978
Jun-05	1713.17	0.038613136
Jul-05	1510.11	-0.12616296
Aug-05	1,613.17	0.066018691
Sep-05	1,673.21	0.03654275
Oct-05	1694.62	0.01271459
Nov-05	1694.42	-0.000118028
Dec-05	1694.42	0

Table 1: Data used for One month ahead forecasting by Arima and neural net.

Month	General Index	Return(ln)
Sep-03	789.74	-0.01288295
Nov-03	920.61	0.153332716
Jan-04	961.18	0.043125203
Mar-04	973.88	0.013126396
May-04	1185.83	0.196910138
Jul-04	1289.14	0.083532378
Sep-04	1633.02	0.236455732
Nov-04	1877.05	0.139270334
Jan-05	1843.95	-0.017791386
Mar-5	2030.39	0.096317883
May-5	1648.28	-0.208495573
Jul-05	1510.11	-0.087549824
Sep-5	1673.21	0.102561441
Nov-5	1694.42	0.012596562

Table 2: Data used for Two month ahead forecasting by Arima and neural net

4 Results and Analysis of Results

4.1 Semi strong form efficiency testing using Excess Return Market Model

The average excess return for each stock for this period should be zero if semi efficient market hypothesis holds. In semi efficient market publicly available information (dividend yield, price to earning ratio) can not be used for making abnormal profits. We try to derive the excess return using excess return market model.

Stock	$E(R_i) - r$	$(E(R_m) - r) \beta_i$	Excess return higher than CAPM predicts
GlaxoSmith Kline:	15.06322	3.612075525	11.45114
SquarePharma:	47.51202	-0.91648161	48.4285
BABTC:	52.27777	8.72348245	43.55429
Meghna Cement:	0.57267	-0.107295408	0.679965
Renata:	25.0417	-0.004360075	25.04606
National Tubes:	15.44094	-3.592331991	19.03327
Singer:	36.80532	6.687653689	30.11767
Uttara Finance:	-2.3157	-2.579574374	0.263874

Table 3: Excess return of Stocks

Many efficient stocks have excess return $(R_{it} - r_f) - \beta_i (R_{mt} - r_f) > 0$. Square Pharma has excess return 48.43 %

,BABTC has excess return 43.55% and Renata has excess return 25.05% etc. As there is excess return higher than CAPM Semi Strong form efficient market does not exist in DSE.

4.2 Forecasting using Neural net and Arima for testing weak form efficiency

Weak form efficiency of the market holds when we can not predict the share prices from its historical information. Arima(0,1,0) is to be fitted in a time series data if it holds random walk model. Our data does not fit in Arima (0,1, 0) model. So market return does not follow random walk model.

Data fits in Arima (1,0,2) model and we accept the p-value less than .05. If p value is less than .05 there is an autocorrelation lag exists. AR1 coefficient is 0.73606455 with p-value 0.00000386 and MA1 coefficient is 0.99744266 with p-value 0 and MA2 coefficient is -0.46540761 with p-value 0.00564925 which is less than .05. Also total sum of squared errors and RMS is calculated from residuals which is the difference between actual data (market return) and fitted data by Arima model.

Arima	Coeff	StErr	p-value
AR1	0.73606455	0.15935449	0.00000386
MA1	0.99744266	0	0
MA2	-0.46540761	0.16817053	0.00564925

Table 4: Arima coefficient for one month ahead forecasting

Our neural network has three layers. The number of neurons in the first layer is equal to total number of input variables which are the market indexes .The second layer has 25 hidden inputs and the number of nodes in the third layer is equal to the number of nodes in the first layer which give the market returns. By trial and error method we found that 25 hidden nodes with one layer give the least MSE.

In the network specification stage we adjust a number of default parameters or values that influence the behavior of the training process. In neural network we specify error rate .001 and momentum .6 and gradient descend .1 and epochs 300. Here is a comparison of Arima with neural network for 30 day ahead forecasting.

One month ahead Model	Sum of squared errors	RMS Error
Arima	0.025217359	0.04584154
Neural Network	0.073130571	0.07806545

Table 5: Forecasting quality of Arima and neural network for 30 day ahead forecasting.

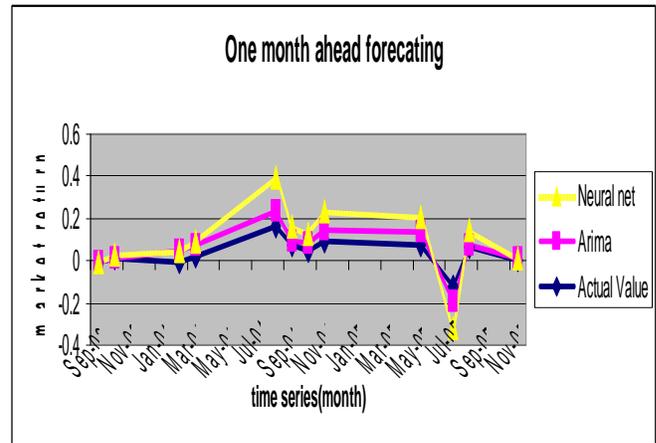


Figure 4: Forecasting for 30 day ahead market return using arima and neural network.

In Neural net Sum of squared error is 0.073130571 and in Arima sum of squared error is 0.025217359 and in Neural net RMS error is 0.078065449 and In Arima RMS error is 0.04584154. Arima shows better forecasting as Sum of squared errors and RMS error is less than Neural net model.

For 60 day ahead forecasting Arima model data fits in the model Arima (0,1,1) which gives least RMS error. For Arima (0,1,1) model we get the following coefficient term. We accept coefficient whose p value is less than .05. We accept MA1 which has coefficient term 0.76990259 whose p value is 0.01151542. So, data fits in the Arima (0,1,1) model which shows a trend.

Arimaa	Coeff	StErr	P-value
MA1	0.7699025	0.30471206	0.01151542

Table 6: Arima coefficient for two month ahead forecasting.

In neural network we use one layer hidden multilayer feed forward network with 25 nodes with error rate .001 and momentum .6 and gradient descend .1 and epochs 300.

Table 7 depicts the comparison of Arima with neural network for 60 day ahead forecasting.

Two month ahead	Total sum of squared errors	RMS Error
Arima	0.36391124	0.246276
Neural	0.115500103	0.138744431

Table 7: Forecasting quality of Arima and neural network for 60 day ahead forecasting.

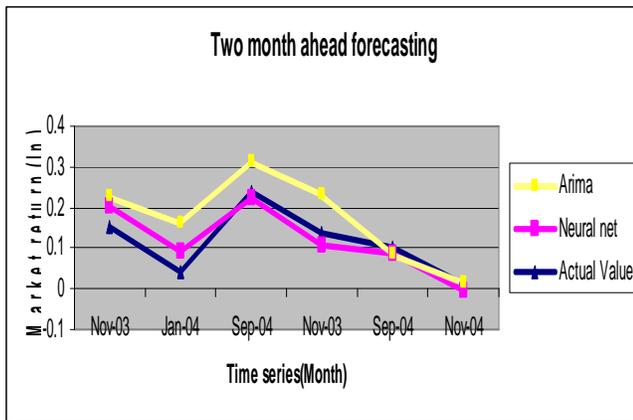


Figure 5: Forecasting for 60 day ahead market return using Arima and neural network.

In Neural net Sum of squared error is 0.115500103 and in Arima sum of squared error is 0.36391124 and in Neural net RMS error is 0.138744431 and In Arima RMS error is 0.246276. Neural net shows better forecasting as Sum of squared errors and RMS error is less than Arima model

So our empirical study shows that change in market index can be forecasted using neural net and Econometric (Arima) model. Market index for Dhaka Stock Exchange follows a trend. There is a pattern for market index movement. Therefore, Dhaka Stock Exchange is not an efficient market as market index does not follow random pattern

5 Conclusion

In semi efficient market publicly available information (dividend yield, price to earning ratio) can not be used for making abnormal profits. We try to derive the excess return using excess return market model. As many efficient stocks have excess return DSE is not efficient in semi strong form.

In weak form efficient market price index exhibit no serial dependencies, so there are no patterns to change in market index. This implies that future market index movements are determined entirely by unexpected information and therefore are random. We use neural network and econometric method to forecast the change in market index which shows that change in market index follows a trend where in efficient market change in market index is random. From the empirical study we see that change in market index shows a trend for DSE, future change in market index can be predicted. But according to weak form efficient market hypothesis stock market prices follow a random walk and thus the prices of the stock market cannot be predicted. So our result shows that change in market index is not random, Dhaka stock exchange is not a weak form efficient market.

References:

1. Box,G. and Jenkins, G. M. (1970) Time series analysis: Forecasting and control, San Francisco:Holden- Day.
2. Fama,E. (1965), "The behavior Of Stock Market Prices", Journal of Business,vol38, p.34-105.
3. Malkiel, Burton G. (1973). A Random Walk Down Wall Street, 6th, W.W. Norton & Company, Inc.
4. Rosenblatt,F.(1962). Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Washington, D.C.: Spartan Books.
5. Sharpe,W., Linter and Mossin "Capital Asset Prices: A theory of Market equilibrium under conditions of risk" The Journal of Finance,Vol-19(September 1960)
6. DataSet: <http://www.dsebd.org>

Identifying Subscribers with Multi-split Fuzzy Decision Trees

G. Jaya Suma*, Dr. M. Shashi **

*Dept of Computer Science and Engineering, GITAM Institute of Technology

GITAM University, Visakhapatnam, India.

**Dept of Computer Science and Systems Engineering, College of Engineering,

Andhra University, Visakhapatnam, India.

ABSTRACT

Crisp decision trees depend on sharp split points to partition the data based on continuous valued attributes. This crisp partitioning cannot handle the compatibility of the boundary points with the partitions on both sides of the split points which may lead to misclassification. Such situations may better be handled by fuzzy decision trees which provide blurred boundaries around the fuzzy partitions. Some of the existing learning algorithms for the development fuzzy decision trees rely on pre-fuzzifying the attributes before constructing the fuzzy decision trees. The drawback of this approach lies upon the difficulty of finding the correct number of fuzzy partitions for each attribute and consequently poor performance due to rule redundancy.

This paper suggests dynamic fuzzification of crisp decision trees to generate multi-split fuzzy decision trees. Fuzzification approach proposed in this method is localized to consider the data distribution around the split points while demarking the fuzzy regions around that split points. Improved performance was observed while comparing the proposed algorithm with that of crisp decision trees and pre-fuzzified decision trees. The proposed dynamic fuzzification algorithm is used to identify subscribers based on their activity in using an Online Learning System during a trial free period of four weeks before subscribing the Online Learning System.

Keywords: Crisp Decision Tree, Fuzzy Decision Tree, Dynamic Fuzzification, Split points, Online Learning System.

1. INTRODUCTION

Decision trees are very successful in building classifier with nominal attributes. They are less effective for ordinal and continuous attributes. Efficient algorithms have been developed to induce

a reasonably accurate, albeit suboptimal, decision trees in a reasonable amount of time. These algorithms usually employ a greedy strategy that grows a decision tree by making a series of locally optimum decisions for partitioning the data. Such algorithms include ID3, C4.5 and CART [3, 10, and 11].

ID3 is a popular and efficient method of decision tree construction for classification of symbolic data and cannot operate for numerical values. Real life problems deal with non-symbolic (numeric, continuous) data. They must be discretized prior to attribute selection. However CART and C4.5 do not require prior partitioning.

In C4.5 algorithm, a decision tree is grown in a recursive fashion by partitioning the training records into successively smaller subsets in order to reduce impurity. If all the records in subset belong to the same class, then it is a leaf node labelled with the class label.

If data set contains records that belong to more than one class an attribute test condition is selected to partition the records into smaller subsets. A child node is created for each outcome of the test condition and the records in data set are distributed to the children based on the outcomes. The algorithm is then recursively applied to each child node. A stopping condition is needed to terminate the tree growing process [2].

There are many measures that can be used to determine the best way to split the records. These measures are defined in terms of the class distribution of the records before and after splitting. The measures developed for selecting the best split are often based on the degree of impurity of the child nodes. The smaller the degree of impurity, the more skewed the class distribution. Impurity measures include entropy, gini index, classification error and gain ratio [3].

Different attribute test conditions are to be applied for different attribute type. For binary attributes the test conditions generate two potential outcomes. A nominal attribute can have many values. Its test conditions can be expressed either as binary or multi-way splits. Some decision tree algorithms such as CART produce only binary splits by considering nearly 2^{k-1} possibilities for a nominal attributes with k outcomes. Other algorithm like ID3 produces multi-way splits for nominal attribute one split each for a distinct value of the corresponding attribute. Ordinal attributes can also produce binary or multi splits. Ordinal attribute values can be grouped as long as the grouping does not violate the order property of the attribute values.

For continuous attributes the test condition can either be expressed as a single comparison test ($A \leq v$) or ($A > v$) for binary split or a range query like $v_i \leq A < v_{i+1}$ for multiple values of i to model a multi split associated with continuous attribute, A . For selecting the value of v for the binary split, the decision tree algorithm must consider all possible split positions and it selects the one that minimises impurity. For the multi split, the algorithm must consider all possible ranges of continuous values. Whether a crisp decision tree has binary splits or multi-splits it suffers with the fundamental weakness of crisp partitioning of data based as continuous valued attributes.

The main fundamental weakness of crisp decision trees is sharp split points, it uses to partition the data based on continuous valued attributes. This crisp partitioning cannot handle the compatibility of the boundary points with the partitions on both sides of the split points which may lead to misclassification. Such situations may better be handled by fuzzy decision trees which provide blurred boundaries around the fuzzy partitions [5].

The Organisation of this paper is as follows: Section 2 briefly discuss as fuzzy decision trees. Section 3 describes the methodology of proposed algorithm. Section 4 describes application. Section 5 presents experimental analysis and conclusion.

2. FUZZY DECISION TREES

Fuzzy decision trees are significantly different from the crisp decision trees which allow gradual transitions among fuzzy outcomes. They result in an unknown record traversing along multiple paths ended with possibly different class labels. It also maintains a better degree of transparency on how the decision outcome was reached.

The main challenges to fuzzy approach to decision tree construction are

1. Number of fuzzy partitions used to represent the continuous valued attributes while building a fuzzy decision tree.
2. To determine the size and shape of the membership function for each fuzzy partition.

The fuzzy tree methodologies proposed by *Sisson* and *Chong*[13] and *Umano et al*[12] require the data to have been pre-fuzzified before the fuzzy decision trees are induced. *Umano et al* took the help of human experts to fuzzify the attributes before construction of the fuzzy decision tree. They relied on the expert intuition to estimate the number of fuzzy regions and shape and size of membership function required for each attribute.

Pre-fuzzification approach first requires the range of each attribute to be partitioned into fuzzy intervals which were then used to quantize the training set from which a crisp decision tree is induced. Pre-fuzzification can lead to extensive pre-processing time in fuzzifying the data. This approach is static as it is done irrespective to the purpose of the task. Due to individual subjective perception of domain experts conflicting opinions might cause more uncertainty into the tree. *Janikow's* methodology [7] is another pre-fuzzification based fuzzy decision tree construction method which lacks clarity on the criteria for correct number of fuzzy partitions for a given attribute. Creation of more number of fuzzy partitions from which fuzzy rule bases are directly generated causes redundant rules. This can add additional complexity to the rule base and ultimately lead to poor interpretability of the fuzzy model resulting in lost decision transparency.

2.1 FUZZY INFERENCE ALGORITHM AND ITS LIMITATIONS

K. Crockett et al have proposed an alternative approach to create fuzzy decision trees from crisp decision trees. Fuzzy inference algorithm (FIA) [1] fuzzifies attributes after the construction of crisp decision tree dynamically. Hence it provides a highly optimised rule set that does not over fit the training data and has good generalization ability over the domain. In this approach the C4.5 algorithm is used to construct the crisp decision tree.

FIA fuzzifies each branch of a binary split created by C4.5 [10, 11] by identifying the fuzzy region around the split point [1]. The induced fuzzy tree provides fuzzy rules for classification as suggested by various paths of the tree from root to leaf nodes.

The main drawback of FIA is that it is limited to fuzzifying binary splits generated by C 4.5 decision tree. Another drawback is that the standard deviation of the attribute is either added or subtracted to the value of split point for finding the boundaries of fuzzy region. This unnecessarily extends the fuzzy region into crisp regions and reduces classification accuracy. This paper proposes a new dynamic fuzzification approach for fuzzifying the crisp decision trees to generate multi-split fuzzy decision trees.

3. METHODOLOGY FOR PROPOSED ALGORITHM FOR MULTI-SPLIT DYNAMIC FUZZIFICATION

Development of multi-split fuzzy decision tree using the proposed algorithm is depicted in figure 1.

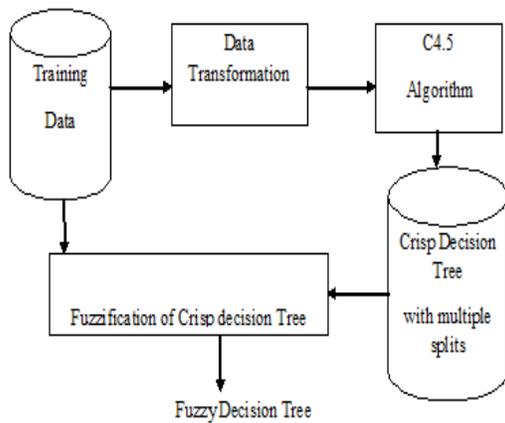


Figure 1: Development of multi-split Fuzzy Decision Tree

A multi-split fuzzy decision tree has some of its non-leaf nodes split into more than two branches. A continuous attribute A with multiple split points v_1, v_2, v_3 can be represented either as a non leaf node with multiple splits as shown in figure 2a or it can be represented as series of binary splits on it as shown in figure 2b.

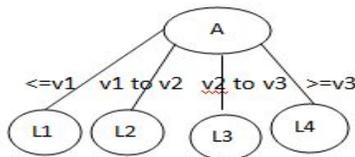


Figure 2 a

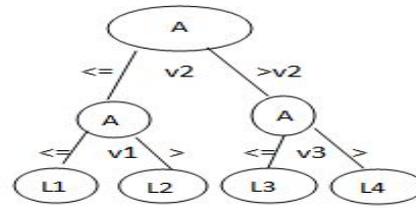


Figure 2 b

During the dynamic fuzzification of a crisp decision tree our algorithm suggests a fuzzification method to deal with multi-split non-leaf nodes/attributes. A separate membership function is defined for each fuzzy condition like $A \leq v_2$ or $A > v_1$. A branch of a fuzzy non-leaf node is represented by two consecutive fuzzy conditions on a single attributes represented by the non-leaf node like $v_1 < A \leq v_2$. The spread of attribute A around the split point v is considered for defining the fuzzy membership function. The membership function over the domain dm, \dots, dn of i^{th} attribute, A_i , where dt_i represent the branching threshold of attribute A_i . A specific value x of attribute A_i passing through the tree would then be assigned a membership grade based on its proximity to dt_i using sigmoid function.

3.1 PROPOSED ALGORITHM FOR DYNAMIC FUZZIFICATION

K. Crockett [1] has considered standard deviation for whole range of training records. By calculating standard deviation for the whole set of values of attribute the lower bound (dm) and upper bound (dn) values are beyond the maximum and minimum values of the whole training set. In the proposed method for dynamic fuzzification the standard deviation is calculated around the split point dt within its local fuzzy region. Hence it is required to demarcate the fuzzy regions around various split points of multi-split non leaf node such that the fuzzy regions are non-overlapping. It may be observed that associated with each split point there are two fuzzy conditions, namely *less than or equal to* and *greater than* the split point with separate fuzzy regions around the same split point. The boundaries of the fuzzy region are calculated, firstly identifying the locality of dt and then finding the local mean and standard deviation within the locality. The boundaries of the locality are calculated by adding and subtracting 15% of gap between split points to implement " \leq " fuzzy condition and 10% of the gap for implementing " $>$ " fuzzy condition.

Mathematically the limits of the locality and the boundaries of fuzzy region separately for each fuzzy condition are calculated using the formulae given below.

Locality

1. For less than or equal to fuzzy condition of dt.

$$\text{Lower limit} = dt_1 - (dt_2 - dt_1) * 15\%$$

$$\text{Upper limit} = dt_1 + (dt_2 - dt_1) * 15\%$$

2. for greater than fuzzy condition of dt

$$\text{Lower limit} = dt_1 - (dt_2 - dt_1) * 10\%$$

$$\text{Upper limit} = dt_1 + (dt_2 - dt_1) * 10\%$$

The spread of the data within the locality of each fuzzy condition in terms of Local mean (β) and standard deviation (S.D) are calculated to find the boundaries of fuzzy region, dm and dn as given below.

Fuzzy Region

The fuzzy region of *less than or equal to* dt fuzzy condition starts at

$$dm_l = \text{local mean} - 2 * \text{S.D} \text{ and ends at } dn_l = \text{local mean} + 2 * \text{S.D.}$$

Similarly the boundary of fuzzy region for *greater than* dt fuzzy condition is given below

$$dm_g = \text{local mean} - 2 * \text{S.D}$$

$$dn_g = \text{local mean} + 2 * \text{S.D.}$$

Using fuzzy sigmoid function S, the fuzzy membership grades are calculated separately for the *less than or equal to* fuzzy condition and *greater than* fuzzy condition as given below.

For fuzzy condition *less than or equal to* dt

$$S(x; \underline{dm}_l, \beta, \underline{dn}_l) = \begin{cases} 1 & \text{if } x \leq \underline{dm}_l \\ 1 - 2 * ((x - \underline{dm}_l) / (\underline{dn}_l - \underline{dm}_l))^2 & \text{if } \underline{dm}_l \leq x \leq \beta \\ 2 * ((x - \underline{dn}_l) / (\underline{dn}_l - \underline{dm}_l))^2 & \text{if } \beta \leq x \leq \underline{dn}_l \\ 0 & \text{if } x \geq \underline{dn}_l \end{cases}$$

For fuzzy condition *greater than* dt

$$S(x; \underline{dm}_g, \beta, \underline{dn}_g) = \begin{cases} 0 & \text{if } x \leq \underline{dm}_g \\ 2 * ((x - \underline{dm}_g) / (\underline{dn}_g - \underline{dm}_g))^2 & \text{if } \underline{dm}_g \leq x \leq \beta \\ 1 - 2 * ((x - \underline{dn}_g) / (\underline{dn}_g - \underline{dm}_g))^2 & \text{if } \beta \leq x \leq \underline{dn}_g \\ 1 & \text{if } x \geq \underline{dn}_g \end{cases}$$

Given the Crisp decision tree and training data set the overview of the dynamic fuzzification process is as in figure 3.

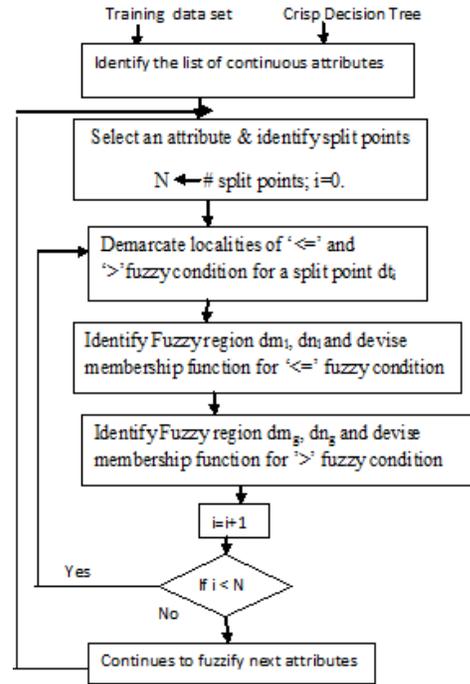


Figure 3: Fuzzification of Continuous attributes in Crisp decision tree

3.2 CLASSIFICATION OF UNKNOWN EXAMPLE

An unknown example to be classified traverses along multiple paths of the fuzzy decision tree with varying degrees of membership along the branches. A pre-selected fuzzy inference technique is then used to combine membership grades down the paths within the tree using t-norms, and then t-conorms [1, 4] are used to aggregate the strength in support of a class label over all the paths that end with leaf nodes of the corresponding class label. The unknown example is classified into the class with highest support. A wide variety of parametric models are available for t-norms and t-conorms for flexible implementation of fuzzy conjunction and disjunction where as Zadeh's conjunction and disjunction operators [9] are non-parametric and hence rigid. In this work Yager's parametric t-norms and t-conorms [8] are used except for the implementation of range fuzzy condition, which required Zadeh's conjunction operator. [14].

4. APPLICATION

In this paper we have considered the task of predicting possible subscribers for an Online Learning System (OLS) [6]. An OLS is a study community that helps the students to clear their doubts and solve their home work problems. A student registers as a free member or a paid member to access the OLS. A free member is the one who can access the OLS for a stipulated time period without any payment. A paid member is the one who accesses OLS by payment. A paid member can subscribe the OLS either for a month or year. After completion of stipulated time period of free membership, a free member can either continue to use the OLS by converting to a paid member by paying the subscription charges or discontinue. Our interest is to identify possible subscribers after completion of stipulated time period of free membership, based on the *activity* of the student, number of *subjects* he is accessing, *difficulty* of the problems that are being accessed by the student etc during the free membership period. Each of these attributes have multiple responses taken weekly over a period of free membership of one month. We have developed a system that uses dynamic fuzzification approach for classifying free users of OLS in order to identify possible subscribers.

This system predicts whether the customer will be a subscriber or discontinues based on the available OLS data [6]. In OLS data the explanatory variables are *activity* of the customer, number of *subjects* covered, the *difficulty* level of problems the customer is accessing and usage of *Answer Board* in terms of number questions he is asking and challenges. The target variable is whether customer will be a monthly member, yearly member or discontinues. The weekly activity values are normalized by taking logarithm with base2. The ratio of number of complex to easy problems attempted gives a continuous level for attribute *difficulty*. The attributes *subject*, *answer board*, *activity* and *challenges* are count variables. The values of the continuous attributes *activity* and the *difficulty* are taken as a weighted summation of their weekly values.

A new categorical attribute "*pattern of activity*" is constructed to identify the usage pattern of customer. The pattern of activity indicates whether the customer is a "*consistent*", "*Aggressive*" or "*uninterested*" customer. It is estimated in terms of the mean and standard deviation (S.D) of multiple responses for the attribute *activity* over the period of four weeks. A customer is said to be

1. *Consistent* if activity of 4th week is between mean-S.D and mean + S.D.

2. *Aggressive* if activity of 4th week is greater than mean+ S.D.
3. *Uninterested* if activity of 4th week is less than mean-S.D.

5. EXPERIMENTAL ANALYSIS AND CONCLUSION

In the analysis the data of 498 customers as training data set and collection of 190 records of different customers as test set, have been considered. A crisp decision tree is constructed using C4.5 algorithm with Weka tool. Figure 4 depicts the multi split crisp decision tree developed based on data of 498 customers. It may be observed that the attribute "*activity*" appears repeatedly along the paths which can better be implemented as a multi-split attribute. Classification accuracy of the crisp decision tree on the training set was found to be as high as **90.02%** where as the test set classification accuracy is only **66.2%**. This situation calls for a more efficient classification tool for predicting unknown records. The proposed approach for converting crisp decision tree to fuzzy decision trees could handle the situation successfully. The present approach is used to construct a multi-split fuzzy decision tree based classifier to predict the possible monthly/yearly subscribers for the OLS system. The performance of the present approach is found to be satisfactory as it could give a classification accuracy of **70.58%** on the test data as against **66.2%** given by crisp decision trees. The comparison of C4.5, GEE Technique [15] and Proposed Fuzzy Classifier is shown in Table 1.

Fuzzification of crisp decision tree requires additional time which results in considerable increase in the time required for model construction however the proposed approach will only require slightly more time for classification compared to crisp decision trees. Hence its overall performance on unknown data is promising.

Table 1

Algorithm	Training Error	Test Error
C4.5	90.02%	66.2%
GEE Technique	67.0%	61.2%
Proposed Fuzzy Classifier	90.02%	70.58%

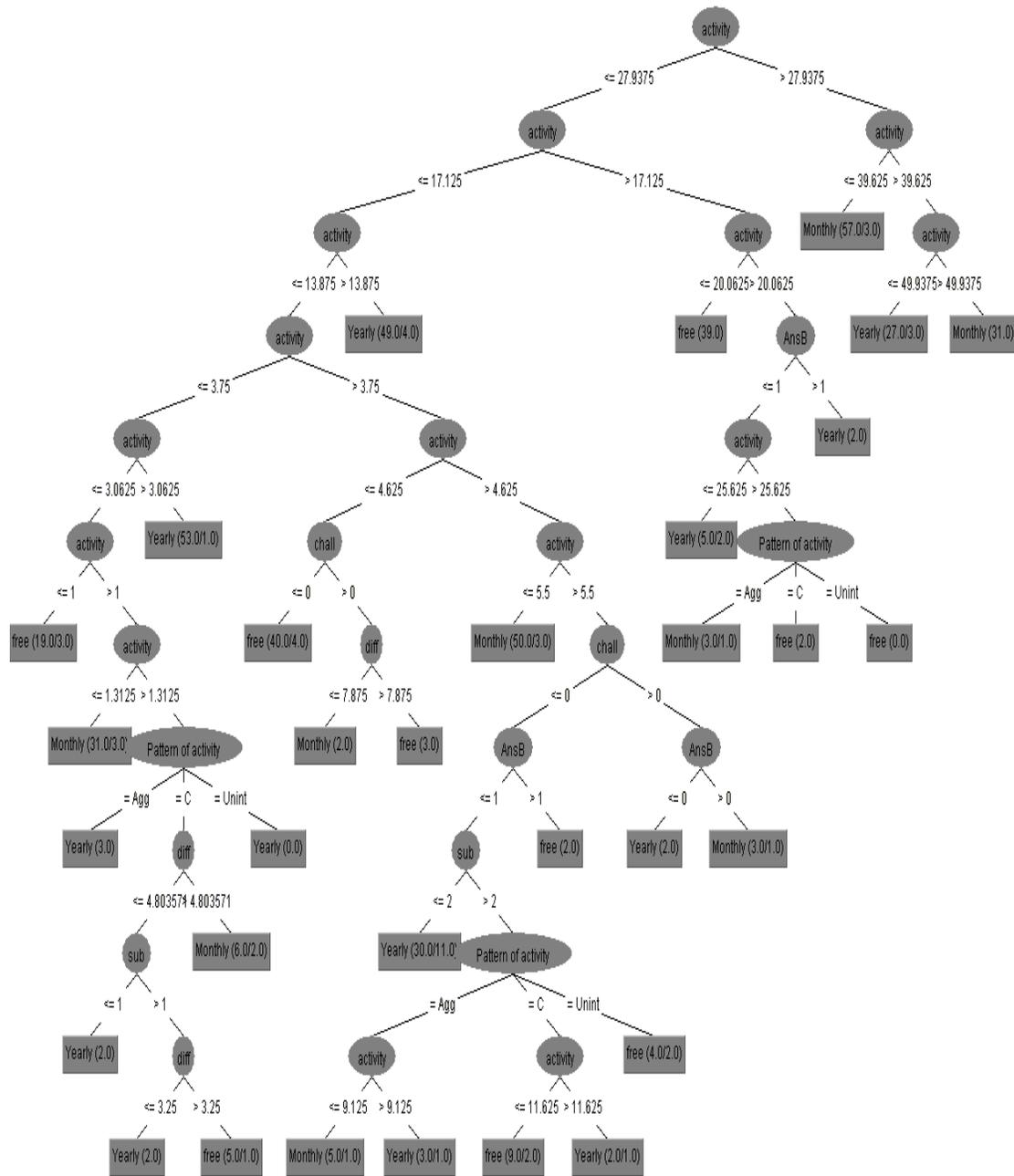


Figure 4: Crisp decision tree given by C4.5

REFERENCES

- [1].K.Crockett et al, On Constructing a fuzzy inference framework using crisp decision trees, *Fuzzy Sets and Systems* 157(2006) 2809- 2832.
- [2]. A. Bouchon- Meunier, M. Mamdani, Learning from imperfect data, in D. Dubois, H. Prade, R. Yager (Eds.), *Fuzzy Information Engineering: A Guided Tour of Applications*, Wiley, New York,1997,pp.139-148.
- [3] .J. Quinlan, *Induction of Decision Trees*, Machine Learning, vol.1, Kluwer Academic Press, Dordrecht, 1986 pp.81-106.
- [4] K. Crockett, On the optimization of T-norm parameters within Fuzzy Decision Trees, *IEEE* (2007)1-4244-1210-2.
- [5].Olaru, C.Wehenkel, A Complete fuzzy decision tree technique. *Fuzzy Sets and Systems*: 138, pp 221-254, 2003.
- [6].www.cramster.com
- [7].Janikow, C. *Fuzzy Information Processing NAFIPS'04*.volume 2, 27-30, pp877-881, 2004.
- [8].Yager, R. Uncertainty representation using fuzzy measures, *IEEE Transaction on Systems, Man and Cybernetics* 32, pp13-20,2002.
- [9].Zadeh L, *Fuzzy Sets. Information and Control* 8, 1965, pp338-353.
- [10].J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, Los Altos, CA, 1993.
- [11].J.R. Quinlan, Improved use of continuous attributes in C4.5, *J. Artificial Intelligence Res.*4 (1996)77-90
- [12] Umano M., Okamoto H., Hatono, I., Tamura, H., *Generation of Fuzzy Decision Trees by Fuzzy ID3 Algorithm and it's Application to Diagnosis by Gas in Oil, Japan -U.S.A Symposium*, pp. 1445-1450, 1994.
- [13] Sison L., Chong E.,*Fuzzy Modelling by Induction and Pruning of Decision Trees*, *IEEE Symposium on Intelligent Control U.S.A*, pp.166-171,1994.
- [14] G. Jaya Suma, Dr. M. Shashi, *Dynamic fuzzification for construction of decision tree. Orsicon2008 Dec 15th-17th 2008*.
- [15] G. Jaya Suma, Dr. M. Shashi A Generalized Multivariate Decision Tree for Prediction of Churner, *International Journal of Engineering and Technology*, Vol 1, pp 31-42, 2008.

On The Usage of Data Mining as a Descriptive and Predictive Tool for Cancer Management in Jordan: A Scenario

Asem Omari

Jarash Private University
Faculty of Science
Department of Computer Science
Jarash, Jordan
asem_omari@yahoo.com

Issa M. Hweidi

Jordan University of Science and Technology
Faculty of Nursing
Adult Health Nursing Department
Irbid, Jordan
Hweidi@just.edu.jo

Abstract

Data mining, as a part of the knowledge discovery process, aims to extract interesting information from large data sets. One of the useful applications of data mining is the health care field, particularly cancer management, since cancer is considered a challenging and national resources-drain health care problem. The purpose of this paper is to provide a scenario on how to improve and support decision-making process for cancer management in Jordan by applying different data mining techniques on the cancer information system in Jordan. It also provides valuable information about predicted distribution and segmentation of cancer in Jordan, which may be linked to possible risk factors. From the extracted patterns of applying different data mining techniques, the information need to be considered in the decision-making process can be identified and recognized as well, which yields to knowledge based decisions. Consequently, identifying health risk behaviors among target group of clients and adopting interventional and preventive measures can be initiated in order to decrease cancer incidence and prevalence and ultimately the health care costs.

Keywords: Data Mining, Cancer Management.

1. Introduction

The problem of cancer in Jordan is becoming increasingly a national problem as it is the second leading cause of death after heart diseases [10]. Furthermore, it is a major cause of morbidity among Jordanian population. Large amount of data about cancer patients are collected in the national Jordan Cancer Registry (JCR) in Jordan. Effective technologies need to be used to participate in preventing or

at least reducing new cancer cases. The Knowledge Discovery from Databases (KDD) is one of those leading technologies. According to [5], "Knowledge discovery in databases (KDD) is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data."

Data mining is the most important step in the KDD process. It extracts interesting patterns from large amounts of data [5]. It provides tools for automated learning from historical data and developing models to predict future trends and behaviors. Data mining has two main models; predictive and descriptive models. Predictive data mining model tries to predict unknown values or future trends or behaviors depending on other variables or historical values presented in the mined database. For example, an immunization dose against some virus may be given to a particular patient not because he is infected from that virus, but because his illness profile is very similar to another group of patients who were infected with the same virus.

Descriptive data mining model tries to extract useful patterns that can describe the mined data and explore its properties. For example, in a hospital database, it may be found that a large number of patients, who are infected by disease A, are also infected by disease B. The patterns extracted using data mining help organizations make better and knowledge based decisions. Data mining has many different applications in different fields of science [11]. In bioinformatics it can be used to analyze DNA sequences to find the relationship between some specific gene sequence and a specific disease. In other words, the change of the DNA sequence can give an indication of the high probability of the development of some disease.

Data mining is used also in medical decision support. Patient records include patient's demographic information, symptoms, blood measurements and laboratory test results. Mining those data can find the correlation between two dif-

ferent diseases, or the relationship between social and environmental issues and some diseases, or how a group of patients react to a specific drug. In this paper, we introduce a solution scenario on how to invest data mining techniques to support the decision making process in the field of cancer management in order to describe, predict, prevent, and control cancer in Jordan which will consequently participate in reducing cancer cases, preventing new cases to occur, and help in implementing new medical, or social strategies to come with better solutions and decisions to fight this national problem in Jordan.

This paper is structured as follows: An overview about cancer in Jordan is introduced in section 2. Related work is presented in section 3. In section 4, we give an overview about the knowledge discovery process, and discuss its phases. Then, in section 5, we will see how the extracted patterns using different Data Mining techniques can be used to improve the decision making process for cancer management. In section 6, we will see how to invest the extracted interesting patterns in the decision making process. Finally, in section 7, we summarize our paper and present the planned future work.

2. Cancer in Jordan

According to King Hussein Cancer Center (*KHCC*) [8], Cancer happens when cells that are not normal grow and spread very fast. Normal body cells grow and divide and know to stop growing. Over time, they also die. Unlike these normal cells, cancer cells just continue to grow and divide out of control and don't die. Cancer cells usually group or clump together to form tumors. A growing tumor becomes a lump of cancer cells that can destroy the normal cells around the tumor and damage the body's healthy tissues. Sometimes cancer breaks away from the original tumor and travel through bloodstream and lymphatic system to other parts of the body, where they keep growing and form new tumors.

The Jordan Cancer Registry (JCR) annual report describes the cancer incidence in Jordan and provides valuable statistics about the situation of cancer among the Jordanian population. JCR collects data from the pathology and hematology laboratories, in addition to the hospitals distributed all over the country from all health sectors [10]. According to the JCR report, The most cancer incidence among Jordanians for both males and females are *Breast, Colo-rectal, Lymphoma, Lung, Leukemia, Myeloma, Urinary bladder, Thyroid, Brain & CNS, and Prostate*.

3. Literature Review

A lot of research have been done for cancer control, prediction, and management using statistical approaches as

well as different data mining techniques. The authors in [14] demonstrated the use of an association rule mining approach to discover associations between selected socioeconomic variables and the four most leading causes of cancer mortality in the United States. They found that health service areas with high rates of low education, high unemployment, and low paying jobs were found to correlate with higher rates of cancer mortality. The work in [15] presented the mining processes and results of discovering trends and patterns from a set of health and living habit questionnaire data. The task was to discover the primary factors of cancer patients with the questionnaires. These factors were helpful to control cancer and decrease cancer incidence.

The authors in [1] presented an analysis of the prediction of survivability rate of breast cancer patients using data mining techniques and compared the performance of different data mining algorithms. The work in [2] discussed several data mining algorithms and techniques that were developed at the University of Arizona Artificial Intelligence Lab. It proposed an architecture for medical knowledge information systems that will permit data mining across several medical information sources. The authors in [9] discussed the process of designing a prototype that can help in the management of Childhood Acute Lymphoblastic Leukemia (*ALL*). The research used the data mining tool, *Clementine*, to apply decision tree techniques which was feeded with data from real life cases taken from specialized cancer institutes as well as relevant medical details such as patient medical history and diagnosis in order to improve the disease management.

In western countries, data mining usage for cancer management has been implemented and reported as many and variable. In Jordan, data mining techniques has never been implemented or even reported for cancer management.

4. The Knowledge discovery in databases process

The Knowledge Discovery in Databases (KDD) process is interactive and iterative, involving a number of steps with many decisions made by the user [5]. Here we list the main steps of the knowledge discovery process :

- Understanding the application domain: The knowledge discovery process begins by understanding the application domain and the goals of the data mining process [7]. Previously unknown patterns that are useful and effective in the decision making process are expected to be gained.
- Data integration and gathering: In the data integration and gathering step the target data sets are gathered from different data sources for example from heterogeneous databases and data warehouses and combined

in a suitable manner [3]. At this step, the relevant data to the analysis process is targeted and retrieved from the data source(s).

- **Data Preprocessing:** Eliminating errors, ensuring consistency, solving the problem of missed and repeated data, and transforming the selected data to format that are appropriate for the data mining procedure, are the main tasks of this step [4].
- **Data mining:** Data mining is the most important step in the knowledge discovery process in which different techniques are applied to extract interesting patterns [3]. In this step, the appropriate data mining algorithm(s) or/ and method(s) are decided to be applied on the target data. In the next section we discuss briefly different data mining techniques.
- **Visualization:** To better understand and analyze the discovered knowledge, it is visually represented to the user using different visualization techniques [12].
- **Pattern evaluation:** In this step, based on predefined measures, all interesting patterns representing meaningful knowledge are identified. It is common to combine some of these steps together. For example, the data integration and gathering and the data preprocessing steps can be combined together and considered as one preprocessing step. Furthermore, most of those steps are often repeated iteratively to reach some satisfying level of the expected results. For example, in the data mining step the data analyst may repeat the data mining process by using another algorithm or method that could be more suitable for better and refined results of the knowledge discovery process. The data analyst can also jump within different steps. For example, if the data analyst is using some data mining algorithm and he want to use another algorithm that needs different data format, then he may go back to the data preprocessing step in order to convert the target data to the suitable format.

In the following section, we discuss the basic data mining tasks: Data characterization, association rule mining, sequential pattern mining, classification, and clustering.

5. Using data mining techniques to mine the cancer information system

In order to get valuable information about the behavior of both patients and cancers, several data mining techniques can be applied on that information system, where the result of those data mining techniques (i.e. patterns) are that kind

of information that is not available directly in the information system as records or fields and cannot be extracted directly by running some query or traditional statistical tool. This extracted information can give the data analyst a better view about the patients, cancers, the relationship between patients and cancers, and the behavior of both patient and cancers with respect to time or other environmental, social, or bibliographic attributes such as:

- Who are the frequent patients?
- What kind of cancers are they usually infected by?
- What kind of cancers occur together?
- If the patient is infected by cancer x may he be infected by cancer y directly or within a period of time?
- What kind of effects has some bibliographic attribute on the development of a certain cancer?

The following patterns can be a result of one of the data mining techniques applied to the cancer information system.

5.1. Data characterization

To have a successful data mining process, it is always a good strategy to understand the target data (i.e. the dataset to be mined). This can be achieved by summarizing the target data and present it in a high conceptual level by using aggregate functions such as *Sum* and *Average*, using the Online Analytical Processing (*OLAP*) technique to explore the data with respect to different aspects and conceptual levels, or using data visualization to explore the data and display distributions of values in the dataset. Data characterization is the summarization of the general characteristics or features of a target class of data [6] such as:

- Describe and summarize the characteristics of patients who react positively to the diagnoses process within one year. The result could be a general profile saying that they are 20-40 years old, and have religious attitude.

5.2. Association rule mining

Association rule mining is the discovery of association rules showing attribute values that occur frequently together in a given set of data [6]. According to [7], an association rule is an expression of the form $X \Rightarrow Y$, where X and Y are sets of items and have no items in common. This rule means that given a database of transactions D where each transaction $T \in D$ is a set of items. $X \Rightarrow Y$ denotes that whenever a transaction T contains X then there is a probability that it contains Y , too. The rule $X \Rightarrow Y$ holds in the

transactions set T with confidence c if $c\%$ of transactions in T that contain X also contain Y . The rule has support s in T if $s\%$ of the transactions in T contains both X and Y .

Association Rule Mining is finding all Association Rules that are greater than or equal a user-specified minimum support (*minsup*), and minimum confidence (*minconf*). In general, the process of extracting interesting Association Rules consists of two major steps. The first step is finding all itemsets that satisfy minimum support (known as *Frequent-Itemset* generation). The second step is generating all Association Rules that satisfy minimum confidence using itemsets generated in the first step. The following rules can be a result of applying association rule mining:

- If a patient record R contains *Kidney* cancer there is a 70% chance that it contains *Bone* cancer as well, and 4% of all records contain both.
- 80% of the patients who are infected by *Breast* cancer is also infected by *Lymphoma* cancer and 8% of all patients are infected by both *Breast* and *Lymphoma* cancers.

5.3. Sequential pattern mining

Sequential pattern mining [13] is the mining for frequently occurring patterns with respect to time for example:

1. Patients who are infected by *Lung* cancer and are *Singles* seem to be infected by *Leukemia* cancer within 9 months.

5.4. Classification

Classification is the process of finding a set of models or functions that describe and distinguish data classes or concepts where the models derived based on a set of training data (i.e. data objects whose class label is known) [6]. The following pattern can be a result of using classification data mining technique:

- Classifying patients with respect to their age into three classes. The first class contains patients who are less than 20 years of age. The second class contains patients who are between 20 and 40 years old. The third class contains patients who are more than 40 years old. After analyzing the characteristics of each class the result could be that patients who are less than 20 years old are usually infected by *Lymphoma*, *Leukemia*, and *Brain* cancers. While patients between 20 and 40 years old are usually infected by *Prostate*, *Stomach*, and *Kidney* cancers. Patients who are more than 40 years old are usually infected by *Liver*, and *Thyroid* cancers.

5.5. Clustering

In clustering data objects have no class label. The objects are clustered or grouped based on the principle that objects in one class have high similarity to one another but are very dissimilar to objects in other clusters [7]. In clustering data objects have no class label. That means when we start clustering we do not know what the resulted clusters will be, or by which attribute the data will be clustered. For that reason, clustering is also called unsupervised learning. Before running any clustering algorithm, the data analyst removes any irrelevant meaningless attributes. The following rule is an example of interesting patterns using a clustering approach:

- Patients who are living in Amman are mostly in risk of infection of collection A of cancers which contains *Urinary Bladder*, *Lymphnodes*, *Brain*, *Lung*, and *Prostate*. On the other hand, female patients are mostly in risk of infection of a subset of collection B of cancers which includes *Breast*, *Ovary*, *Thyroid*, and *Hodgkin Lymphoma*.

6. Investment of interesting patterns

The extracted patterns from applying data mining techniques on the cancer information system database give the data analyst an overview about the behavior of the patients, the relation between patients and cancers, what kinds of cancers are usually occur together, and about cancer occurrence habits with respect to different bibliographic attributes such as customer age, or address. Thus, the decision maker can come with some decisions that will improve the treatment strategies, improving preventive medicine techniques, reducing cancer cases, preventing new cases to occur, and help in implementing new medical, or social strategies to come with better solutions and decisions to fight this national problem in Jordan.

Data mining as an advanced computational technology can has a significant impact on health care field in term of its application. Using data mining as a descriptive and predictive technique in health care systems can identify health risk behaviors and has a significant economic impact. Investing this technology in a limited-resources health care system as in Jordan, will enhance setting well-structured and designed plans to deal effectively with such risky behaviors, particularly cancer health risk behaviors. Furthermore, it helps characterise cancer patient behaviour to predict office visits, identify successful medical therapies for cancer patients, and predicts the effectiveness of surgical procedures or medical tests used in cancer management are of great value of applying data mining techniques.

The primary health care system that is mainly concerned

with health promotion and disease prevention is the principal beneficiary, where various health care strategies can be adopted to decrease cancer incidence and prevalence in a limited resources Middle Eastern country like Jordan.

7. Summary and future plans

In this paper we provided a solution scenario on how to improve and support the decision making process for cancer management in Jordan, by applying different data mining techniques such as data characterization, association rule mining, sequential pattern mining, classification, and clustering on the contents of the cancer information system in Jordan. From the extracted patterns we can recognize the information need to be considered in the decision making process which yields to knowledge based decisions.

As a future work, the plan is to use this approach to mine the real cancer data available in the Jordan Cancer Registry (JCR). On one hand, the mining process could be very general by mining the whole cancer information system to get interesting patterns. On the other hand, the cancer data can be filtered with respect to some attributes to get the target data. For example, by retrieving the patients records that contain *Breast Cancer*, to mine for interesting patterns that describe and predict the relationships between this kind of cancer and patients bibliographic information.

References

- [1] E. G. Abdelghani Bellaachia. Predicting Breast Cancer Survivability Using Data Mining Techniques. *Scientific Data Mining workshop in SIAM Conference on Data Mining*, pages 1–4, 2006.
- [2] Andrea Houston, Hsinchun Chen, Susan Hubbard, Bruce Schatz, Tobun Ng, Robin Sewell and Kristin Tolle. Medical Data Mining on the Internet: Research on a Cancer Information System. *Artificial Intelligence Review*, 13:437–466, 2000.
- [3] Asem Omari. *Data Mining for Retail Website Design and Enhanced Marketing*. VDM Publishing, Saarbrücken, Germany, 2008.
- [4] R. Cooley, B. Mobasher, and J. Srivastava. Data Preparation for Mining World Wide Web Browsing Patterns. *Knowledge and Information Systems*, 1(1):5–32, 1999.
- [5] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34, 1996.
- [6] J. Han and M. Kamber. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, San Francisco, 2001.
- [7] Ian H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers, San Francisco, 2005.
- [8] King Hussein Cancer Center. Cancer Information. <http://www.khcc.jo/cancerinformation.asp>, 2009.
- [9] N. Labib and M. Malek. Data Mining for Cancer Management in Egypt Case Study: Childhood Acute Lymphoblastic Leukemia . In *Proceedings of the World Academy of Science, Engineering, and Technology*, 2005.
- [10] Ministry of Health. Cancer incidence in Jordan. Technical Report number 11, Jordan Cancer Registry, 2006.
- [11] B. J. Read. Data Mining and Science: Knowledge Discovery in Science as Opposed to Business. In *Proceedings of the 12th ERCIM Workshop on Database Research*, Amsterdam, 1999.
- [12] Tom Soukup and Ian Davidson. *Visual Data Mining: Techniques and Tools for Data Visualization and Mining*. John Wiley & Sons, first edition, 2002.
- [13] J. Velásquez, H. Yasuda, and T. Aoki. Combining the Web Content and Usage Mining to Understand the Visitor Behavior in a Web Site. In *Proc. 3rd IEEE International Conference on Data Mining (ICDM 2003)*, pages 669–672. IEEE Computer Society Press, 2003.
- [14] S. Vinnakota and N. S. Lam. Socioeconomic inequality of cancer mortality in the United States: a spatial data mining approach. In *International Journal of Health Geographics*, volume 5. BioMed, 2006.
- [15] X. Zhang and T. Narita. Discovering the Primary Factors of Cancer from Health and Living Habit Questionnaires. In *Discovery Science DS99*, pages 371–372. Springer Verlag, 1999.

SESSION

PRIVACY PRESERVING DATA MINING

Chair(s)

Dr. Philippe Lenca

P-Sensitive *K*-Anonymity for Social Networks

Roy Ford, Traian Marius Truta, and Alina Campan

Abstract — The proliferation of social networks, where individuals share private information, has caused, in the last few years, a growth in the volume of sensitive data being stored in these networks. As users subscribe to more services and connect more with their friends, families, and colleagues, the desire to both protect the privacy of the network users and the temptation to extract, analyze, and use this information from the networks have increased. Previous research has looked at anonymizing social network graphs to ensure their *k*-anonymity in order to protect their nodes against identity disclosure. In this paper we introduce an extension to this *k*-anonymity model that adds the ability to protect against attribute disclosure. This new model has similar privacy features with the existing *p*-sensitive *k*-anonymity model for microdata. We also present a new algorithm for enforcing *p*-sensitive *k*-anonymity on social network data based on a greedy clustering approach. To our knowledge, no previous research has been done to deal with preventing against disclosing attribute information that is associated to social networks nodes.

Keywords: privacy, social networks, *k*-anonymity, clustering, greedy algorithm.

I. INTRODUCTION

The use of social network sites on the Internet, such as Facebook or MySpace, continues to grow at an exponential rate. The opening of Facebook to non-college membership caused a 500% growth in enrollment in one year [14]. In 2005, before Facebook became a public network, Gross and Acquisti analyzed the profiles of Carnegie Mellon University students and identified privacy implications in the data being stored in this social network [7]. The main privacy concerns reported by Gross and Acquisti were the potential for stalking and re-identification of users based on demographics or images of faces and the possibility of identity theft [7].

Obviously, there is a need to protect the privacy of individuals in social networks. Since social networking has become mainstream only in the last few years, the research in social networks privacy is also very recent, and many questions are still to be answered. Only a few researchers have explored this integrative field of privacy in social networks from a computing perspective.

R. Ford (e-mail: fordrl@nku.edu), T. M. Truta (phone: +1-859-572-7551; fax: +1-859-572-5398, e-mail: trutat1@nku.edu), and A. Campan (e-mail: campana1@nku.edu) are with the Department of Computer Science, Northern Kentucky University, Nunn Drive, Highland Heights, KY 41099, USA.

A. Related Work

Most of the existing work had focused on protecting the nodes' identities in a social network [4], [8], [12], [22]. There is a strong similarity between ensuring this type of privacy for social network nodes and preventing against identity disclosure in flat microdata [10]. Therefore, *k*-anonymity, the most popular model that guarantees identity protection in microdata [16], [18], has been extended from its primary form to also work for social network data [4]. With that end in view, the *k*-anonymity model for social networks had to additionally address the anonymization of network's structural information, which itself carries a disclosure "potential". Other researchers have proposed solutions for protecting the confidential links between nodes. Two nodes in a social network may have multiple connections, and some of them represent confidential relationships. Solutions to this link disclosure problem have been analyzed in [9], [21]. Less related to this paper, other contributions in the privacy in social networks field include: active and passive attacks [1], random perturbation [20], and access control / encryption protocols [5], [6]. A good survey of the state of the art in social networks' privacy can be found in [23].

B. Contributions

To our knowledge, this is the first work that extends the existing results on identity protection in social networks [4], [8], [12], [22] to also guard against the disclosure of sensitive information/attributes associated to network's nodes. An equivalent model for flat microdata would be one that guards against attribute disclosure [10].

This paper's contributions are: introducing a new privacy model for social network data entitled *p*-sensitive *k*-anonymity which combines the existing *k*-anonymity model for social networks [4] and the *p*-sensitive *k*-anonymity model for microdata [19], integrating existing algorithms for the *p*-sensitive *k*-anonymity for microdata and the *k*-anonymity for social networks to generate a *p*-sensitive *k*-anonymous social network, and performing experiments that prove the validity of the proposed model and algorithm.

II. *P*-SENSITIVE *K*-ANONYMOUS SOCIAL NETWORKS

We consider a social network to be a simple undirected graph $G = (\mathcal{N}, \mathcal{E})$, where \mathcal{N} is the set of nodes and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of edges. Each node represents an individual user of the social network; each edge represents a

relationship or connection between two users in the network.

All nodes in \mathcal{G} are described by a set of attributes. These attributes can be classified as follows

- I_1, I_2, \dots, I_m are *identifier* attributes such as *Name* and *SSN*.
- Q_1, Q_2, \dots, Q_q are *quasi-identifier* attributes such as *ZipCode* and *Age*. They may be known from other public datasets and could be potentially used to violate individuals' privacy.
- S_1, S_2, \dots, S_r are *sensitive* attributes such as *Disease*. These attributes' values must be protected against disclosure.

There are two related aspects in anonymizing a social network modeled as described above. Both the data associated to the social network's nodes, \mathcal{N} , (identifier, quasi-identifier and sensitive attributes) and the structural information the network carries about the nodes' relationships, \mathcal{E} , have to be properly masked. The resulting masked network data has to protect the nodes against: identity disclosure (i.e. determining who exactly is the individual owning the node) and attribute disclosure (i.e. finding out sensitive data about an individual, but without identity disclosure).

The process of anonymizing the nodes' attributes consists of removing the identifier attributes from the nodes information, ensuring that the quasi-identifier information is at least k -anonymous [16], [18], and ensuring that the sensitive attributes are at least p -sensitive [19]. P -sensitive k -anonymity property for nodes' data can be obtained by generalizing the quasi-identifier information, either with hierarchy-free generalization [11] for numeric data, or predefined hierarchies [13] for categorical data. The nodes' data generalization we envision is performed based on a partitioning of the node set \mathcal{N} into distinct *clusters*. The nodes' data generalization is performed at the cluster-level: each cluster will have identical quasi-identifier values for all nodes, a minimum size of k , and at least p distinct values for each sensitive attribute. The network's structural information (edges) is also masked starting from the established partitioning of \mathcal{N} into clusters. Basically, the detailed connectivity information of the individual nodes in a cluster is replaced with a summary of intra-connectivity and inter-connectivity information of the cluster as a whole. So, the essential task in anonymizing a social network is partitioning the node set \mathcal{N} – how we conduct this step and the reasoning behind it will be explained in the next section.

The goal of the anonymization process is not only to produce a masked p -sensitive k -anonymous social network (formally defined next), but also to create a good-quality masked social network. The quality of an anonymized social network is given by the amount of information that it preserves from the original unmasked network: lower information loss means higher quality of the anonymous

network. As we are dealing with graph data, the measures we use for quantifying information loss need to be sensitive to both the change in the quasi-identifier attributes and the change in the structural information that occurs due to edge anonymization [4]. To measure the information lost from nodes' quasi-identifier attributes generalization we use the generalized information loss, a measure defined in [2]. For assessing the structural information loss in the edge-anonymization process, we use the measure introduced in [4].

Definition 1 (generalization information loss): Let cl be a cluster and $QI = \{N_1, N_2, \dots, N_s, C_1, C_2, \dots, C_t\}$ the set of numerical and categorical quasi-identifier attributes. The **generalization information loss** caused by generalizing quasi-identifier attributes of the cl nodes is:

$$GIL(cl) = |cl| \cdot \left(\sum_{j=1}^s \frac{\text{size}(\text{gen}(cl)[N_j])}{\text{size}\left(\min_{X \in \mathcal{N}}(X[N_j]), \max_{X \in \mathcal{N}}(X[N_j])\right)} + \sum_{j=1}^t \frac{\text{height}(\Lambda(\text{gen}(cl)[C_j]))}{\text{height}(\mathcal{H}_{C_j})} \right).$$

where:

- cl 's generalization information, denoted by $\text{gen}(cl)$, is the "node" having as value for each quasi-identifier attribute, numerical or categorical, the most specific common generalized value for all that attribute values from cl nodes (see a formal definition in [3]);
- $|cl|$ denotes the cluster cl 's cardinality;
- $\text{size}([i_1, i_2])$ is the size of the interval $[i_1, i_2]$, i.e. $(i_2 - i_1)$;
- $N_i, i = 1..s$ are numerical quasi-identifier attributes;
- $C_i, i = 1..t$ are categorical quasi-identifier attributes
- $\Lambda(w), w \in \mathcal{H}_{C_j}$ is the subhierarchy of the C_j 's predefined value hierarchy (\mathcal{H}_{C_j}) rooted in w ;
- $\text{height}(\mathcal{H}_{C_j})$ denotes the height of the tree hierarchy \mathcal{H}_{C_j} .

To be able to compare this measure with the structural information loss, we normalize it to the range $[0, 1]$. This is shown in the Definition 2. Detailed justification of Definitions 1 and 2 can be found in [4].

Definition 2 (normalized generalization information loss): The **normalized generalization information loss** obtained when masking the graph \mathcal{G} based on the partition $\mathcal{S} = \{cl_1, cl_2, \dots, cl_v\}$, denoted by $NGIL(\mathcal{G}, \mathcal{S})$, is:

$$NGIL(\mathcal{G}, \mathcal{S}) = \frac{\sum_{j=1}^v GIL(cl_j)}{n \cdot (s + t)}.$$

where:

- n is the number of nodes for the graph \mathcal{G} ;
- $(s + t)$ is the number of quasi-identifier attributes.

Structural information loss quantifies the probability of error when trying to reconstruct the structure of the initial social network from its masked version. There are two

components for the structural information loss: the *intra-cluster structural loss* and the *inter-cluster structural loss*. These components occur due to an edge anonymization process [4]. In the anonymized graph, the cluster cl will be generalized to (collapsed into) a node, and the structural information we attach to it is the pair of values $(|cl|, |\mathcal{E}_{cl}|)$, where $|cl|$ represents the cardinality of the set cl . This information permits assessing some structural features about this region of the network that will be helpful in some applications. From the privacy standpoint, an original node within such a cluster is indistinguishable from the other nodes of the cluster. This intra-cluster edge generalization causes an intra-cluster information loss. In a similar way, given any two clusters cl_1 and cl_2 , let \mathcal{E}_{cl_1,cl_2} be the set of edges having one end in each of the two clusters ($e \in \mathcal{E}_{cl_1,cl_2}$ iff $e \in \mathcal{E}$ and $e \in cl_1 \times cl_2$). In the anonymized graph, this set of inter-cluster edges will be generalized to (collapsed into) a single edge and the structural information released for it is the value $|\mathcal{E}_{cl_1,cl_2}|$. This process is called inter-cluster edge generalization and induces inter-cluster information loss [4]. Based on this structural generalization method, a structural information loss (*SIL*) measure and a corresponding normalized structural information loss (*NSIL*) measure are derived. Due to the lack of space, for a complete definition we refer the reader to [4].

Given a partition of nodes for a social network \mathcal{G} , we are able to create an anonymized graph by the generalization techniques explained above.

Definition 3 (masked social network): Given an initial social network, modeled as a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, and a partition $\mathcal{S} = \{cl_1, cl_2, \dots, cl_v\}$ of the nodes set \mathcal{N} , $\bigcup_{j=1}^v cl_j = \mathcal{N}$,

$cl_i \cap cl_j = \emptyset$; $i, j = 1..v$, $i \neq j$; the corresponding **masked**

social network \mathcal{MG} is defined as $\mathcal{MG} = (\mathcal{MN}, \mathcal{ME})$, where:

- $\mathcal{MN} = \{Cl_1, Cl_2, \dots, Cl_v\}$, Cl_i is a node corresponding to the cluster $cl_j \in \mathcal{S}$ and is described by the “tuple” $gen(cl_j)$, the intra-cluster generalization pair $(|cl_j|, |\mathcal{E}_{cl_j}|)$, and the projection on all sensitive attributes of the nodes from cl_j ;
- $\mathcal{ME} \subseteq \mathcal{MN} \times \mathcal{MN}$; $(Cl_i, Cl_j) \in \mathcal{ME}$ iff $Cl_i, Cl_j \in \mathcal{MN}$ and $\exists X \in cl_j, Y \in cl_j$, such that $(X, Y) \in \mathcal{E}$. Each generalized edge $(Cl_i, Cl_j) \in \mathcal{ME}$ is labeled with the inter-cluster generalization value $|\mathcal{E}_{cl_i,cl_j}|$.

By construction, all nodes from a cluster cl collapsed into the generalized (masked) node Cl are indistinguishable from each other.

To have the k -anonymity property for a masked social network, we need to add one extra condition to Definition 3, namely that each cluster from the initial partition is of size at least k . The formal definition of a masked social network that is k -anonymous is presented below.

Definition 4 (k -anonymous masked social network): A masked social network $\mathcal{MG} = (\mathcal{MN}, \mathcal{ME})$, where $\mathcal{MN} = \{Cl_1, Cl_2, \dots, Cl_v\}$, and $Cl_j = [gen(cl_j), (|cl_j|, |\mathcal{E}_{cl_j}|)]$, $j = 1, \dots, v$ is k -anonymous iff $|cl_j| \geq k$ for all $j = 1, \dots, v$.

Now we have all the tools to introduce the p -sensitive k -anonymous masked social network that combines the above definition with the p -sensitive k -anonymity property for microdata [19].

Definition 5 (p -sensitive k -anonymous masked social network): A masked social network $\mathcal{MG} = (\mathcal{MN}, \mathcal{ME})$, where $\mathcal{MN} = \{Cl_1, Cl_2, \dots, Cl_v\}$, is p -sensitive k -anonymous if it is k -anonymous and the number of distinct values for each sensitive attribute is at least p within each $Cl_j, j = 1..v$.

III. SANGREEA_PK ALGORITHM

The *SaN GreeA_PK* algorithm builds on the work done by Campan et. al. in the areas of k -anonymity in social networks [4] and p -sensitive k -anonymity for microdata [3]. By combining the two algorithms presented in these papers, we developed the *SaN GreeA_PK* algorithm, which is able to perform p -sensitive k -anonymization for social networks.

The algorithm functions by taking the nodes of the social network and grouping them in a way that ensures p -sensitiveness of the formed clusters. The clusters formed will also have cardinality greater than k . Of course, some preconditions have to be respected for a social network to be amenable to p -sensitive k -anonymity, for given p and k values. For example, the social network must have at least p unique values for each of its nodes' sensitive attributes, and at least k nodes.

The cluster formation process is performed in a greedy manner. Each cluster is started from an initial seed node and fed with one other node at a time, until it becomes p -sensitive and k -anonymous. The node to be included in the currently developed cluster is the result of a greedy selection based on the values of the sensitive attributes and the levels of information loss (both through structural and attribute generalization) introduced in the summarization of the quasi-identifiable information. The functions that guide the selection process are: the *diversity* between the cluster being formed and the new node [3], the *NGIL*, and a function called *structural distance* that aims to limit the *SIL* (obviously, (*N*)*SIL* cannot be used as long as a complete partitioning of \mathcal{N} is not known) [4].

The diversity between a cluster and a new node helps achieve the desired level of sensitivity in each cluster. We introduce next this measure. Let $X^i, i=1..n$, be all the nodes from the social network. We denote an node label information as $X^i = \{k_1^i, k_2^i, \dots, k_q^i, s_1^i, s_2^i, \dots, s_r^i\}$, where k^i s are the values for the quasi-identifier attributes and s^i s are the values for the confidential attributes.

Definition 6 (diversity of two tuples): The *diversity of two tuples*, X^i and X^j w.r.t. the sensitive attributes is given by:

$$\text{diversity}(X^i, X^j) = \sum_{l=1}^r w_l \cdot \delta(s_l^i, s_l^j), \text{ where}$$

$$\delta(s_l^i, s_l^j) = \begin{cases} 1, & \text{if } s_l^i \neq s_l^j \\ 0, & \text{if } s_l^i = s_l^j \end{cases} \text{ and } \sum_{l=1}^r w_l = 1 \text{ are the weights of the sensitive attributes.}$$

The data owner can choose different criteria to define this weights vector. One good selection of the weight values is to initialize them as inversely proportional to the number of distinct sensitive attribute values in the original dataset. Along this paper we use this choice for the weights in all the experiments.

Definition 7 (diversity between a tuple and a cluster): The *diversity between a tuple X^i and a cluster cl* is given

$$\text{by } \text{diversity}(X^i, cl) = \sum_{l=1}^r w_l \cdot \rho(s_l^i, cl), \text{ where:}$$

$$\rho(s_l^i, cl) = \begin{cases} 1, & \text{if } s_l^i \text{ does not exist between the } S_l \text{ values in } cl \\ 0, & \text{if } s_l^i \text{ exists between the } S_l \text{ values in } cl \end{cases} \text{ and}$$

$$\sum_{l=1}^r w_l = 1 \text{ have the same meaning as in Definition 6.}$$

The justification that stands behind the selection of the structural distance for guiding cluster formation is presented in [4]. The formal definition of the structural distance measure follows. Assuming that the nodes in \mathcal{N} have an order, $\mathcal{N} = \{X^1, X^2, \dots, X^n\}$, we represent the neighborhood of each node X^i as an n -dimensional boolean vector $B_i = (b_1^i, b_2^i, \dots, b_n^i)$, where $b_j^i = 1$ if there is an edge $(X^i, X^j) \in \mathcal{E}$, and 0 otherwise, $\forall j = 1, n; j \neq i$. We consider the value b_i^i to be *undefined*, and therefore not equal with 0 or 1.

Definition 8 (structural distance between two nodes): The *structural distance between two nodes* (X^i and X^j) described by their associated n -dimensional boolean vectors B_i and B_j is:

$$\text{sdist}(X^i, X^j) = \frac{|\{\ell \mid \ell = 1..n \wedge \ell \neq i, j; b_\ell^i \neq b_\ell^j\}|}{n-2}.$$

Definition 9 (structural distance between a node and a cluster): The *structural distance between a node X and a cluster cl* is defined as the average distance between X and every node from cl :

$$\text{sdist}(X, cl) = \frac{\sum_{X^j \in cl} \text{dist}(X, X^j)}{|cl|}.$$

The following documents the new *SaNGreeA_PK* algorithm, which combines the *SaNGreeA* algorithm for

anonymizing social networks [4] with the *GreedyPKClustering* algorithm that anonymizes microdata to conform to p -sensitivity k -anonymity [3].

Algorithm **SaNGreeA_PK** is

Input: $G = (\mathcal{N}, \mathcal{E})$ - a social network
 k - as in k -anonymity
 p - as in p -sensitivity
 α, β - user-defined weight parameters;
allow controlling the balancing
between **NGIL** and **NSIL**

Output: $S = \{cl_1, cl_2, \dots, cl_v\}; \bigcup_{j=1}^v cl_j = \mathcal{N};$

$$cl_i \cap cl_j = \emptyset, i, j = 1..v, i \neq j;$$

$|cl_j| \geq k, j = 1..v$ - a set of clusters that ensures p -sensitive k -anonymity for $\mathcal{MG} = (\mathcal{MN}, \mathcal{ME})$ so that a cost measure is optimized;

$S = \emptyset;$

$i = 1;$

r_{seed} = a randomly selected node from $\mathcal{N};$

Repeat

$$r_{seed} = \arg \max_{r \in \mathcal{N}} (\text{diversity}(r_{seed}, r));$$

$$cl_i = \{r_{seed}\};$$

$$\mathcal{N} = \mathcal{N} - \{r_{seed}\};$$

Repeat

// make cl_i p -sensitive; for that, find

// the set of most diverse nodes w.r.t. cl_i

$$\text{div}cl = \arg \max_{r \in \mathcal{N}} (\text{diversity}(r, cl_i));$$

$$X' = \arg \min_{X \in \text{div}cl} (\alpha * \text{NGIL}(G_1, S_1) + \beta * \text{sdist}(X, cl_i));$$

// G_1 : subgraph induced by $cl_i \cup \{X'\}$ in G

// S_1 : partition with 1 cluster $cl_i \cup \{X'\}$

$$cl_i = cl_i \cup \{X'\};$$

$$\mathcal{N} = \mathcal{N} - \{X'\};$$

Until (cl_i is p -sensitive) or ($\mathcal{N} = \emptyset$);

If ($|cl_i| < k$) and ($\mathcal{N} \neq \emptyset$) then

Repeat

// add nodes until cl_i has k nodes

$$X' = \arg \min_{X \in \mathcal{N}} (\alpha * \text{NGIL}(G_1, S_1) + \beta * \text{sdist}(X, cl_i));$$

$$cl_i = cl_i \cup \{X'\};$$

$$\mathcal{N} = \mathcal{N} - \{X'\};$$

Until ($|cl_i| \geq k$) or ($\mathcal{N} = \emptyset$);

End If;

If ($|cl_i| \geq k$ and cl_i is p -sensitive) then

$$S = S \cup \{cl_i\};$$

$i++;$

Else

// this only happens to last cluster

DisperseCluster (S, cl_i);

End If;

Until $\mathcal{N} = \emptyset$;

End **SaNGreeA_PK**;

Function **DisperseCluster** (S, cl)

$S = S - cl;$

For every $r \in cl$ do

$$cl_u = \text{FindBestCluster}(r, S);$$

$$cl_u = cl_u \cup \{r\};$$

End For;

End **DisperseCluster**;

```

Function FindBestCluster( $r, S$ )
  bestCluster = null;
  infoloss =  $\infty$ ;
  For every  $cl_i \in S$  do
    If ( $\alpha \cdot \text{NGIL}(G_i, S_i) + \beta \cdot \text{sdist}(r, cl_i) < \text{infoloss}$ )
      then
        infoloss =  $\alpha \cdot \text{NGIL}(G_i, S_i) + \beta \cdot \text{dist}(r, cl_i)$ ;
        bestCluster =  $cl_i$ ;
      End If;
    End For;
  Return bestCluster;
End FindBestCluster;

```

To illustrate this algorithm, we give an example that shows how p -sensitive k -anonymity is achieved in a sample social network. Suppose the social network G_{ex} as shown in Figure 1 is given. The contents of the nodes are given in Table 1. The quasi-identifiers in this example are *age*, *zip* and *gender*, and the sensitive attribute is *illness*. The attribute *age* is numerical and subject to a hierarchy-free generalization. The value generalization hierarchies for the other two quasi-identifiers are given in Figure 2.

By running the *SanGreeA_PK* algorithm for this dataset with $k = 3$, $p = 2$, $\alpha = 0$, and $\beta = 1$, the masked social network $\mathcal{M}G_{ex1}$, shown in Figure 3, is generated. Due to the choice of α and β values, *SanGreeA_PK* guides the cluster formation to optimize the structural information loss and disregards the generalization information loss. When run with $k = 3$, $p = 2$, $\alpha = 1$, and $\beta = 0$, the masked social network $\mathcal{M}G_{ex2}$, shown in Figure 4, is generated. In this case, as $\beta = 0$, the generalization information loss is the only cost metric *SanGreeA_PK* tries to minimize in the cluster formation process.

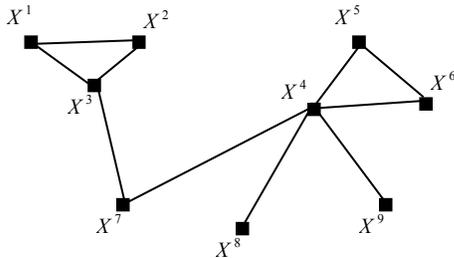


Fig. 1. The social network G_{ex} .

TABLE 1

THE NODES' QUASI-IDENTIFIER AND SENSITIVE ATTRIBUTES IN G_{ex}

Node	Age	Zip	Gender	Illness
X^1	25	41076	Male	Diabetes
X^2	25	41075	Male	Heart Disease
X^3	27	41076	Male	Diabetes
X^4	35	41099	Male	Colon Cancer
X^5	38	48201	Female	Breast Cancer
X^6	36	41075	Female	HIV
X^7	30	41099	Male	Diabetes
X^8	28	41099	Male	HIV
X^9	33	41075	Female	Colon Cancer

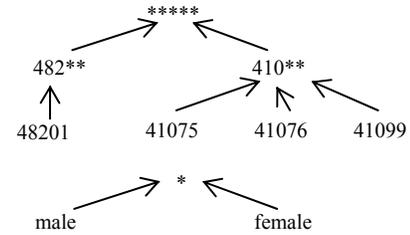


Fig. 2. The value generalization hierarchies for attributes *zip* and *person*

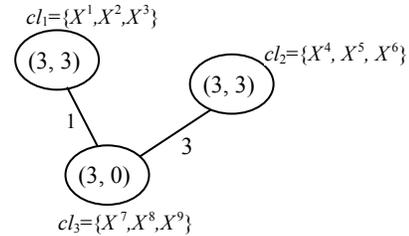


Fig. 3. The social network $\mathcal{M}G_{ex1}$.

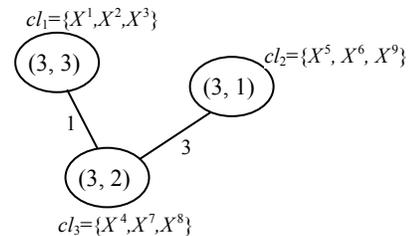


Fig. 4. The social network $\mathcal{M}G_{ex2}$.

When we analyze the resulting graph for both generalization and structural information loss, we produce the results shown in Table 2.

TABLE 2
THE GENERALIZATION AND STRUCTURAL INFORMATION LOSS FOR
THE SUMMARIZATIONS OF G_{ex}

$\mathcal{M}G$	<i>GIL</i>	<i>NGIL</i>	<i>SIL</i>	<i>NSIL</i>
$\mathcal{M}G_{ex1}$	14.30	0.53	5.77	0.32
$\mathcal{M}G_{ex2}$	7.73	0.28	8.44	0.46

IV. EXPERIMENTAL RESULTS

In this section we compare the *SanGreeA_PK* algorithm with the *SanGreeA* algorithm [4], over various combinations of k and p values, in terms of the generalization and structural information loss of the masked social networks they generate.

The two algorithms were implemented in Java. The tests were executed on a single CPU machine running at 2.53 GHz with 2GB of RAM and Windows XP Professional.

The algorithms were tested with a social network derived from the Enron e-mail dataset, an ex-employee

status report developed by Shetty et.al. [17], and the Adult dataset from the UC Irvine Machine Learning Repository [15]. The Adult dataset was necessary, as it provided quasi-identifier and sensitive attribute values for the nodes of our test social network. This type of information has been stripped out of the publicly available Enron e-mail dataset, to respect the privacy of the Enron employees. However, the Enron e-mail dataset provided the structural information for our test social network.

The nodes' data we formed contained the quasi-identifier attributes: *role* (coming from the ex-employee status report), *age*, *marital_status*, and *race* (coming from the Adult database). *Age* is the only numeric quasi-identifier; the other three quasi-identifiers are categorical. The heights of their predefined generalization hierarchies are 2 (for *role*), 2 (for *marital_status*), and 1 (for *race*). *Education* and *salary_range* are the sensitive attributes.

The attribute *role* is taken from the Enron ex-employee status report by matching the first and last name on the report with the e-mail database first and last names, resulting in 121 matching records. A Roles table was created that correlated role, education and salary ranges, as shown in Table 3.

TABLE 3
THE ROLES TABLE

Role	Education	Min Salary	Max Salary
Trader	Assoc-voc	40	110
Manager	Bachelors	40	110
Managing Director	Bachelors	70	150
CEO	Doctorate	90	200
President	Doctorate	90	200
N/A	HS-grad	30	60
Director	Masters	70	150
Vice President	Masters	90	200
In House Lawyer	Prof-school	40	110
Director of Trading	Prof-school	70	150
Employee	Some-college	30	60

From this table, the *education* attribute was used to randomly match and select a record from the Adult dataset; that particular record provided the values for the *marital_status*, *age*, and *race* attributes. A salary value was then randomly generated, using a uniform distribution, for each individual, within the range associated to the individual's Enron role. Exact salary values were then transformed into the following reporting ranges: \$25-50K, \$51-75K, \$76-100K, \$101-125K, \$126-150K, \$151-175K, and \$176-200K.

The edges of the graph were derived in the same way as in [17], with two users being considered related if they had exchanged 5 e-mails with each other. Once extracted, it was discovered that some nodes had dropped off the graph, as they did not meet the 5 e-mail connection rule with any of the other users. Finally, the edge information was

merged with the node information. Any isolated node and any edge for which one end had been already eliminated from the node set were eliminated. The resulting graph consists of 84 nodes and 191 edges.

The created test social network was anonymized with the *SaNGreeA* and *SaNGreeA_PK* algorithms, varying the value of k from 2 to 15 and the values of p from 2 to $\min(7, k)$. The experiments were run with two different (α, β) parameter values: (0, 1) and (1, 0). The pair (0, 1) guides the algorithm towards minimizing structural information loss, while (1, 0) reduces information loss due to the generalization of the quasi-identifier attributes. The normalized generalization information loss (*NGIL*) and the normalized structural information loss (*NSIL*) were computed for the resulting masked social networks. Figure 4 presents the resulting *NGIL* and *NSIL* values for $(\alpha, \beta) = (0, 1)$. Figure 5 depicts the same measures for the pair (1, 0).

Looking at the results, we can see that in general there is a correlation between an increase in the values of k and p and an increase in structural and generalization information loss values. The experiments also show that increasing k values (for a fixed p) are reflected in a greater information loss (both *SIL* and *GIL*) than when increasing p values for the same k .

As expected, α and β parameters values controlled the trade-off between *SIL* and *GIL*. For the same k and p values *SIL* is lower when $\alpha = 0$ then when $\alpha = 1$. Similarly, *GIL* is lower when $\beta = 0$ then when $\beta = 1$.

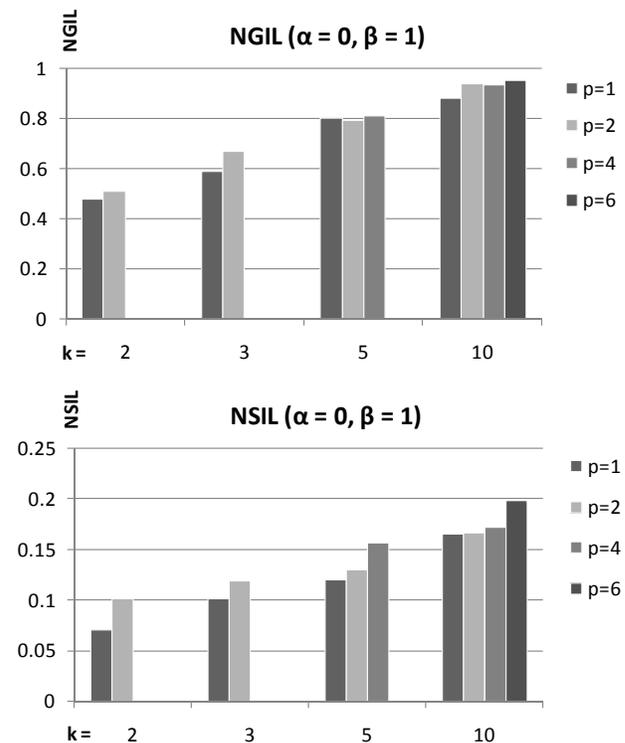


Fig. 4. The *NGIL* and *SGIL* values for the test social network with $\alpha = 0, \beta = 1$.

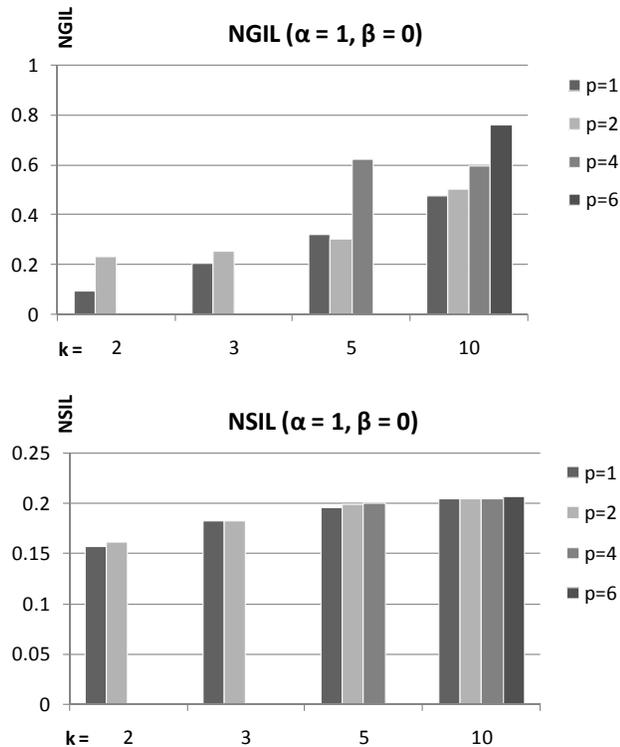


Fig. 5. The *NGIL* and *SGIL* values for the test social network with $\alpha = 1, \beta = 0$.

V. CONCLUSIONS AND FUTURE WORK

In this paper we extended the existing results on identity protection in social networks to also guard against the disclosure of sensitive information/attributes associated to network's nodes. To achieve this extension we introduced a new privacy model for social network data entitled *p*-sensitive *k*-anonymity. We also integrated existing algorithms for the *p*-sensitive *k*-anonymity for microdata and the *k*-anonymity for social networks into a new algorithm entitled *SaNGreeA_PK*. Our experiments showed that the new algorithm generates *p*-sensitive *k*-anonymous social networks with their corresponding information loss is similar to the existing *SaNGreeA* *k*-anonymity algorithm with only a modest increase in structural and generalization information loss. The new proposed algorithm can also be user-balanced towards preserving more the structural information of the network or the nodes' attribute values.

We consider two possible directions to extend this work:

- Analyze, using social networks from different areas, the utility of the anonymized social network. This may lead to the development of more practical data utility / information loss measures.
- Formally study how the greedy criteria can be improved based on the properties of the social network data and the selected *p* and *k* values.

REFERENCES

- [1] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore Art Thou R3579X? Anonymized Social Networks, Hidden Patterns, and Structural Stenography," In *International World Wide Web Conference (WWW)*, 2007, pp. 181 – 190.
- [2] J.W. Byun, A. Kamra, E. Bertino, and N. Li, "Efficient *k*-Anonymization using Clustering Techniques," In *International Conference on Database Systems for Advanced Applications (DASFAA)*, 2007, pp. 188 – 200.
- [3] A. Campan, T.M. Truta, J. Miller, and R. Sinca, "A Clustering Approach for Achieving Data Privacy," In *International Conference on Data Mining (DMIN)*, 2007, pp. 321 – 327.
- [4] A. Campan and T.M. Truta, "A Clustering Approach for Data and Structural Anonymity in Social Networks," In *Privacy, Security, and Trust in KDD Workshop (PinKDD)*, 2008.
- [5] B. Carminati, E. Ferrari, and A. Perego, "Private Relationships in Social Networks," In *Private Data Management Workshop (PDM)*, 2007, pp. 163 – 171.
- [6] J. Domingo-Ferrer, A. Viejo, F. Sebe, and U. Gonzalez-Nicolas, "Privacy Homomorphism for Social Networks with Private Relationships," In *Computer Networks*, Vol. 52, Issue 15, 2008, pp. 3007 – 3016.
- [7] R. Gross and A. Acquisti, "Information Revelation and Privacy in Online Social Networks (The Facebook Case)," In *Workshop on Privacy in the Electronic Society (WPES)*, 2005, pp. 71 – 80.
- [8] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis, "Resisting Structural Re-identification in Anonymized Social Networks," In *Very Large Data Base Conference (VLDB)*, 2008, pp. 102 – 114.
- [9] A. Korolova, R. Motwani, S. U. Nabar, and Y. Xu, "Link Privacy in Social Networks," In *International Conference on Data Engineering (ICDE)*, 2008, pp. 1355 – 1357.
- [10] D. Lambert, "Measures of Disclosure Risk and Harm," In *Journal of Official Statistics*, Vol. 9, 1993, pp. 313 – 331.
- [11] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional *K*-Anonymity," In *IEEE International Conference on Data Engineering (ICDE)*, 2006, pp. 25.
- [12] K. Liu and E. Terzi, "Towards Identity Anonymization on Graphs," In *ACM SIGMOD International Conference on Management of Data*, 2008, pp. 93 – 106.
- [13] M. Lunacek, D. Whitley, and I. Ray, "A Crossover Operator for the *k*-Anonymity Problem," In *Genetic and Evolutionary Computation Conference (GECCO)*, 2006, pp. 1713 – 1720.
- [14] A. McCard and K. Anderson, "Focus on Facebook: Who Are We Anyway," In *Anthropology News*, Vol. 49, No. 3, 2008, pp. 10 – 12.
- [15] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz, "UCI Repository of Machine Learning Databases", available at: www.ics.uci.edu/~mllearn/MLRepository.html, 1998.
- [16] P. Samarati, "Protecting Respondents Identities in Microdata Release," In *IEEE Transactions on Knowledge and Data Engineering*, Vol. 13, No. 6, 2001, pp. 1010 – 1027.
- [17] J. Shetty, and J. Adibi, "The Enron Email Dataset Database Schema and Brief Statistical Report," available at: www.isi.edu/~adibi/Enron/Enron.htm.
- [18] L. Sweeney, "K-Anonymity: A Model for Protecting Privacy," In *International Journal on Uncertainty, Fuzziness, and Knowledge-based Systems*, Vol. 10, No. 5, 2002, pp. 557 – 570.
- [19] T. M. Truta and V. Bindu, "Privacy Protection: P-Sensitive K-Anonymity Property," In *Privacy Data Management Workshop (PDM)*, 2006, pp. 94.
- [20] X. Ying and X. Wu, "Randomizing Social Networks: A Spectrum Preserving Approach," In *SIAM International Conference on Data Mining (SDM)*, 2008, pp. 739 – 750.
- [21] E. Zheleva and L. Getoor, "Preserving the Privacy of Sensitive Relationships in Graph Data," In *Privacy, Security, and Trust in KDD (PinKDD), LNCS*, Vol. 4890, 2008, pp. 153 – 171.
- [22] B. Zhou and J. Pei, "Preserving Privacy in Social Networks against Neighborhood Attacks," In *IEEE International Conference on Data Engineering (ICDE)*, 2008, pp. 506 – 515.
- [23] B. Zhou, J. Pei, and W.S. Luk, "A Brief Survey on Anonymization Techniques for Privacy Preserving Publishing of Social Network Data," In *SIGKDD Explorations*, Vol. 10, No. 2, 2008, pp. 12 – 22.

APHID: A Practical Architecture for High-Performance, Privacy-Preserving Data Mining

Jimmy Secretan, Anna Koufakou, Michael Georgiopoulos

Abstract—While the emerging field of privacy preserving data mining (PPDM) will enable many new data mining applications, it suffers from several practical difficulties. PPDM algorithms are difficult to develop and computationally intensive to execute. Developers need convenient abstractions to reduce the costs of engineering PPDM applications. The individual parties involved in the data mining process need a way to bring high-performance, parallel computers to bear on the computationally intensive parts of the PPDM tasks. This paper discusses APHID (Architecture for Private and High-performance Integrated Data mining), a practical software architecture for developing and executing large-scale PPDM applications. At one level, the system supports simplified use of cluster and grid resources, and at another level, the system abstracts communication for easy PPDM algorithm development. This paper offers an analysis of the challenges in developing PPDM algorithms with existing frameworks, and motivates the design of a new infrastructure based on these challenges.

Index Terms—Privacy-Preserving Data Mining, Distributed Data Mining, Cluster Computing, MapReduce

I. INTRODUCTION

Modern organizations manage an unprecedented amount of data, which can be mined to generate valuable knowledge, using several available data mining techniques. While data mining is useful within an organization, it can yield further benefits with the combined data of multiple organizations. And, in fact, many organizations are interested in collaboratively mining their data. This sharing of data, however, creates many potential privacy problems. Many organizations, such as health organizations, have restrictions on data sharing. Businesses may be apprehensive to share trade secrets despite the value of cooperative data mining. At the same time, privacy concerns for individuals are rapidly gaining attention. Instead of dispensing entirely with cooperative data mining, research has instead focused on Privacy Preserving Data Mining (PPDM), which uses various techniques, statistical, cryptographic and others, to facilitate cooperative data mining while protecting the privacy of the organizations or individuals involved.

However, PPDM research is still in its infancy, and few practical systems are currently in place. Even if organizations currently have the legal infrastructure in place for sharing data, there is a lack of developmental support for PPDM systems. Organizations attempting to implement PPDM systems would face a lack of available toolkits, libraries, middleware

and architectures that are ready for deployment. The costs involved are potentially high, because of the lack of familiarity with PPDM technology. In addition, because complex computation is often required, high performance and parallel computing technologies are necessary for efficient operation, adding yet another level of complexity to development. The purpose of this research is to provide an architecture and development environment that will allow organizations to easily develop and execute PPDM software. By borrowing from familiar parallel paradigms, the architecture aims to ease the introduction of PPDM technology into the existing database infrastructure. Furthermore, the system seamlessly integrates high performance computing technologies, to ensure an efficient data mining process.

Because of extensive communication over relatively slow wide area networks, and because of the large computational requirements of cryptographic and other privacy-oriented technologies, resource requirements for PPDM algorithms can be intense. One study [1] describes taking *29 hours* to build a 408-node decision tree from a 1728 item training set. While there is much research that discusses available algorithms and techniques in PPDM, few studies focus on high-performance computational architectures that support them. Therefore, this research presents a development environment and runtime system specifically geared toward PPDM.

The contributions of this research are: (1) middleware for managing the execution of PPDM algorithms across multiple organizations, (2) the integration of high performance and parallel computing middleware into the PPDM execution environment, and (3) a simple development framework for PPDM software. First in this paper, a brief background on PPDM and high performance computing middleware is given. Existing frameworks are analyzed in terms for their strengths and weaknesses. Then, the model of development used in the system is delineated and justified. Next, the design of the PPDM runtime environment is discussed, followed by the details of the implementation. A PPDM example of the Naïve Bayes classifier is presented, with an associated design and sample code for the system.

II. PRIVACY PRESERVING DATA MINING

Privacy Preserving Data Mining (PPDM) concentrates on how to coherently combine and mine databases to preserve the privacy of the individual parties' data. PPDM shares a common line of research with Distributed Data Mining (DDM). DDM analyzes the problem of coherently mining data at multiple sites, while considering processing power, network transfer, format compatibility and many other issues

Jimmy Secretan, Anna Koufakou and Michael Georgiopoulos are with the Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816, USA (phone: 407-882-2016; fax: 407-823-5835; emails: jsecretan@ucf.edu, akoufako@mail.ucf.edu and michaelg@mail.ucf.edu).

encountered in the course of coherently mining disparate data sets. PPDM adds a privacy dimension to these challenges. PPDM methods may address individual privacy (i.e. the privacy of specific customers or patients), or collective privacy, the privacy of information about an organization's records overall (i.e. summary statistics, etc) [2].

First, to support PPDM, there are data perturbation methods, which obfuscate the original data with random perturbations in a way that the original distributions of the data can be easily recovered. Another, similar area involves using signal processing techniques to make approximations of the data distributions, which more effectively preserve privacy than access to the raw data. A different approach leverages Secure Multi-party Computation (SMC) to compute the data mining functions in a way that is more exact and private, albeit at the expense of computational efficiency. Because of these potential advantages, PPDM through SMC is the focus of the proposed architecture, although much of the architecture could be applied to other PPDM techniques.

SMC has as one of its pillars Yao's work to solve the Millionaire's Problem [3]. The Millionaire's Problem is as follows: two millionaires wish to find who has more money, but neither wants to disclose his/her individual amount. Yao proved that there was a secure way to solve this problem by representing it as a circuit, sharing random portions of the outputs. Later it was proved in [4], that any function could be securely computed using this kind of arrangement. However, using Yao circuits is typically inefficient. The problem must be represented as a circuit, which may be large, especially for complex algorithms. Yao circuits are typically only practical for small sub-problems within the PPDM process. This gives rise to more specific solutions based on cryptographic primitives such as secure sums, secure set unions, and secure scalar products [5].

Many machine learning algorithms have been recast into privacy preserving versions, including decision trees [6], the Naive Bayes classifier [7], k-Means Clustering [8], Support Vector Machines [9], k-NN [10], and Association Rule Mining [11], just to name a few. APHID was developed to support the implementation of these and future algorithms

III. RELATED WORK

To develop a practical architecture for PPDM, we must first observe what technologies are available to support the process. In this section, we examine those technologies. First, the field of high-performance data mining will provide the necessary infrastructure and frameworks for the computationally intensive portion of the PPDM process. Next an analysis of existing DDM/PPDM middleware and web-services approaches to data mining will show what principles are worth keeping in a framework, and which are lacking.

A. High Performance Data Mining on Clusters and Grids

Systems that provide convenient abstractions for simple development within a parallel environment have received a great deal of interest in recent years. One of the most popular, MapReduce [12] is a simplified parallel paradigm for large

scale, data intensive parallel computing jobs. By constraining the parallel programming model to only the *map* function and the *reduce* function, the MapReduce infrastructure can simplify parallel programming. MapReduce versions of many machine learning algorithms have been developed including k-means, Logistic Regression, Naïve Bayes, linear Support Vector Machines, and many others [13].

Concepts within grid computing hope to make the use of computational resources, even those across organizational and administrative boundaries, as easy as drawing resources from the power grid. These systems include Networks of Workstations (NOW) architectures, which take advantage of idle machines to lower system costs. However, developing the software to support these architectures can be challenging, as they are often less reliable, less available, and have fewer resources than their dedicated cluster counterparts. In [14] a system for data mining in NOWs is developed, built on a simple primitive called *Distributed DOALL*. *Distributed DOALL* can be applied to loops that have no loop carried dependencies or conflicts, loops which are frequently encountered in data mining.

Because these technologies are oriented toward trusted local environments, they cannot support the PPDM process *per se*. However, they are capable of supporting computationally intensive sub-tasks which are found in PPDM.

B. Distributed Data Mining

Often, separate organizations or multiple sites of a single organization want to collaboratively mine databases which are in different geographic locations. Distributed Data Mining (DDM) research develops techniques and architectures to mine databases distributed across the Internet, while working within the constraints of limited bandwidth and computing.

A system called the Knowledge Grid (K-Grid), is a comprehensive architecture and tool suite for DDM [15], [16]. The core layer of K-Grid is implemented on top of Globus [17] services. K-Grid follows a *collection-of-services* approach: that is, the functions of the middleware are available as numerous services, which the application employs. The core layer is responsible for managing the metadata about data sources, algorithms and mining tools, as well matching the requirements of the data mining tools with the necessary grid resources. The high level layer of the K-grid software is responsible for orchestrating the execution algorithms and the production of models.

One of the most comprehensive DDM systems is the DataMiningGrid [18] software. It provides functionality for tasks such as data manipulation, resource brokering, and parameter sweeps. It was developed to encourage grid transparency, and the adaptation of current code to be grid enabled, through a service oriented architecture (SOA). The authors emphasize that extensibility is important to DataMiningGrid, and that developers should be able to add new components without adversely affecting the existing large components and implementations in the system.

Systems like K-Grid and DataMiningGrid provide excellent frameworks for unifying the data mining resources

of several organizations. However, they are not specifically designed to ease the complexity of developing distributed algorithms, but mostly for devising multi-stage distributed data mining processes. Furthermore, they offer no additional support for PPDM algorithms.

While DDM frameworks cannot be immediately adapted to the needs of PPDM, they still share much in the way of concerns and challenges. Performance, scalability, portability and cost of development are concerns for both disciplines. Therefore, we can emulate some of the techniques of DDM systems to apply to a PPDM framework.

The authors of [19] advocate having the most flexible architecture possible to support DDM. They mention that new algorithms should be included easily and that the system should integrate relational and data mining operators, as well as interoperability and resource management mechanisms. They argue that DDM architectures should be built not to support a specific kind of data mining paradigm (classification, ARM, clustering, etc), but should instead offer a broad base of support, much like an operating system.

The system described in [20] emphasizes scalability and portability. The system uses Java for the language, RMI for the intercommunication, XML for much of the storage, and JDBC for database connections. While this is put together into a flexible and efficient framework, the fact that a language choice is imposed may limit a company's ability to adapt current infrastructure to DDM environments, and therefore may hinder adoption.

SOAs are gaining popularity in DDM. In this paradigm, data sets and algorithms can be viewed as independently hosted services, called whenever they are needed. The system in [21] uses an execution framework in conjunction with a registry of algorithms and databases to complete a large-scale data mining task, by matching tasks to be executed to available services. SOAs decouple DDM systems, by allowing various parts of the applications to be hosted in different environments.

C. Architectures to Support PPDM

Systems to support an SMC-based PPDM process have begun to appear in the literature. TRIMF [22] proposes a runtime environment to support privacy preserving data access and mining. Built on top of a service oriented architecture and communicating over JXTA peer-to-peer technology [23], TRIUMF aims to provide an ensemble of related services for PPDM. TRIMF also supports fine-grained access control where each party can specify which data is accessible and to whom. While TRIUMF can enable efficient PPDM processes, and scale to many parties, it does not suggest a framework with which to implement PPDM algorithms.

In [24] the authors suggest a hierarchical structure combining P2P and grid concepts in order to efficiently support PPDM. Peers within virtual organizations (VO) communicate locally, and then use super-peers to communicate among the VOs. While an architecture resembling this has tremendous potential for facilitating large-scale PPDM, it is not clear

exactly how a system like this would operate, and what kind of programming model it would use.

The system described in [25] suggests that PPDM specific services be offered as services built on the K-Grid architecture [15], [16]. While the authors do not provide details on how this would be implemented, or communication framework developers would use, we do adapt the approach of providing PPDM services in our system.

Domain specific languages for SMC have the potential to free the user from the details of the execution and implementation. Fairplay [26] is a domain specific language for secure computation. Fairplay generates secure circuits in a Secure Hardware Description Language (SHDL) and then executes those circuits.

While domain specific languages can potentially ease the development of PPDM algorithms, they also have drawbacks. The new domain specific systems present a problematic learning curve to developers who are trained in the use of standard languages. In addition, systems that automatically generate SMC algorithms, are also implicitly parallelizing the computation; automated parallelization can only find mechanical ways of distributed the computation and does not make decisions to refactor the overall computation in more efficient ways. The use of specialized languages can inhibit developers from leveraging existing code for PPDM projects. Finally, the SMC languages surveyed do not offer the ability to leverage high-performance computing resources available to organizations. Therefore, their scaling can be limited for large data mining applications.

IV. APHID

Reviewing the literature and available software to support PPDM, it is clear that there is a significant barrier to developing PPDM applications. The first of these impediments is that there are no standardized libraries to support PPDM. Secondly, organizations would need a middleware framework to support PPDM, which is not sufficiently provided in current systems [27], [15], [18]. Even with the availability of such frameworks, simple development environments are lacking; it is especially difficult to integrate the PPDM level of mining with the use of local high-performance computing resources (e.g. grid, clusters and specialized hardware). APHID (Architecture for Private and High-performance Integrated Data mining) seeks to overcome these limitations. The design is influenced by several desiderata, which have been explicitly identified in the literature or found lacking in other systems. The system must have:

- Low development cost [20]
- A runtime environment for executing algorithms (as in [27], [15], [18], [22], [25])
- The capability to leverage high-performance computing resources whenever possible (as in [12], [14] and others)
- Flexibility to support an array of PPDM algorithms (emphasized in [19])
- Support many popular languages (addressing the emphasis for portability in [20])

- Simple abstractions to mitigate the complexity of PPDM development

To support low development cost and language independence, DDM/PPDM functions are provided as a collection of web services (as suggested in [15], [18], [22], [25]), which can be called by the application program. To begin with, web services libraries are available for almost every popular language in use, so the services can be implemented in any language or platform, and consumed by a different language or platform. Providing a set of frequently used services reduces development effort and hence cost, for implementing a new algorithm. In charge of these services is a set of master services, the Main Execution Layer (MEL), discussed in section IV-C. MEL orchestrates the execution of the PPDM algorithm, providing the necessary runtime environment.

APHID is explicitly built on a two-tier system of PPDM, which differentiates it from other systems. On the first tier, different organizations (also called parties in the PPDM context) communicate with each other, typically using secure, privacy preserving communications. The second tier includes grids and clusters within a particular party. Treating these tiers distinctly helps the developer to manage the complexities inherent in each level (see figure 1 for an illustration).

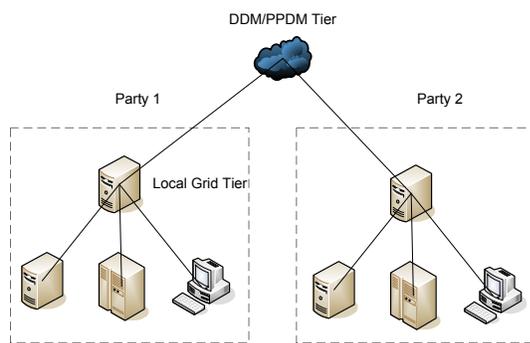


Fig. 1. A two tier PPDM architecture.

The interface to the local high performance machines is also provided as a set of web services for the individual functions in the algorithm. Therefore, an algorithm can be developed once and shared among all of the parties, with the developers at each individual party providing only what is necessary to interface with the party's database and high performance machines. These services support the requirement of leveraging HPC resources.

Figure 2 shows the stack of systems comprising a typical APHID installation within a single party. Organizational data to be mined is frequently stored in a relational database server. Because a relational database manager is typically insufficient for flexible data mining, and because these servers are often intimately involved in core business processes, this data is converted and transferred to a high-performance distributed file system (e.g. HDFS [28], or grid-based storage). This synchronization should be done periodically and at off peak times, before it is needed for a data mining process.

The PPDM process begins with a request from a client, typically as part of a larger application, for the output of a specific PPDM algorithm (e.g. a classified test point, a classifier model, a set of clusters, or a set of association rules). The algorithms are available by unique services representing the algorithm, a partitioning (vertical, horizontal, or arbitrary) and a specific implementation.

While the MEL is responsible for initiating the PPDM process, it will frequently need to use SMC-PPDM primitives (e.g. secure sum, secure scalar products) and perform compute and memory intensive operations on training data. The PPDM services layer and the High-Performance Computing (HPC) respectively support these needs. The PPDM services layer acts as a gateway to external parties. The HPC services layer is a generic interface that interacts with a pluggable set of cluster and grid runtime systems (e.g. MapReduce) to perform the local mining of the database which will become part of the larger PPDM algorithm. It will store and access training databases, and submit compute-intensive jobs through the appropriate channels. Having these broad collections of service-based functions available meets APHID's requirements for flexibility.

Before describing the functions of each layer in detail, a notation of analysis must first be established. Then, the development model around which APHID is structured is described. Finally, each of the three layers is described in detail, aided by an example of the code that would be found at each layer. The code is for the horizontally-partitioned Naïve Bayes classifier for categorical attributes [7].

A. Notation

To aid in our analysis, we first establish a notation. During a PPDM operation there are K parties involved, numbered $P^1 \dots P^k \dots P^K$. In the case of horizontal partitioning, the D -dimensional training data, \mathbf{X} , are divided among the parties, in some way, such that each party P^k owns several D -dimensional instances of the training data, with that set designated as \mathbf{X}^k and individual points labeled X_r^k .

B. Development Model

Before focusing more closely on each layer of APHID, a development model must first be established to provide a simple yet powerful abstraction for PPDM development. The cornerstones of APHID development are a program style similar to many HPC frameworks, and policy-attached shared variables, which mitigate complexity and cost.

1) *Program Structure*: In order to bridge computations on a grid or cluster with DDM/PPDM computations, a simplified interface is needed. Programming hundreds or thousands of machines of a cluster, typically on local networks, along with remote DDM/PPDM sites, typically connected on the Internet, has the potential to significantly confuse a developer. To simplify development, at the cluster/grid level, parallel development environments like MapReduce are used. At the DDM/PPDM level, an Single Program Multiple Data (SPMD) style is used. SPMD is the same programming style used in implementations of the Message Passing Interface

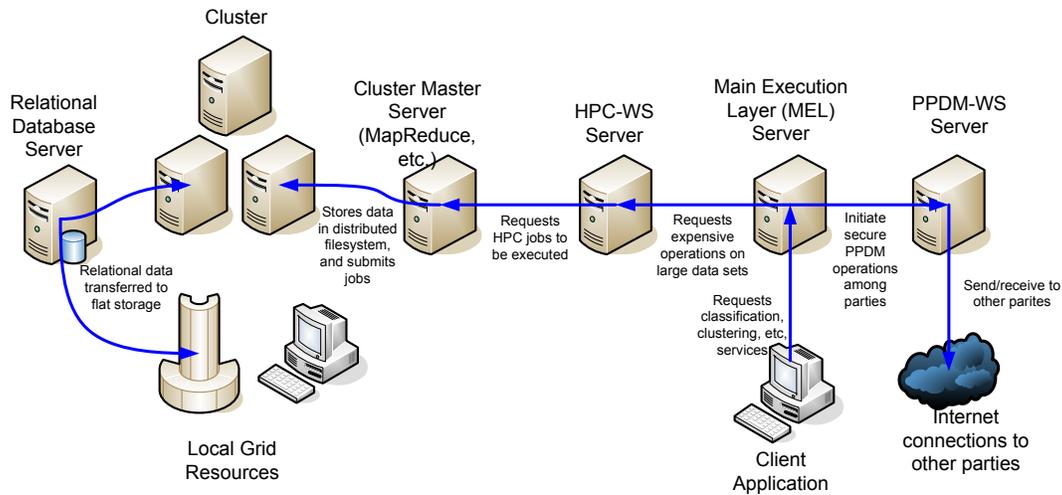


Fig. 2. The APHID system stack.

(MPI) [29], which is a popular development environment for distributed programming. The SPMD style is appropriate because all parties should be able to examine the operations involved in a PPDM algorithm. For each PPDM algorithm, there should exist one copy of the code that all parties can examine, thereby ensuring security.

2) *Shared Variables*: One technique APHID employs to simplify PPDM application development is the use of shared variables. These variables work similarly to those of traditional shared memory systems, with the exception that they have a particular *policy* attached. A policy determines how and by whom the value may be accessed.

The available policies are shown in table I. The first policy Intra-Party (IP) creates an intra-party shared variable only, which is simply a handle that allows the data to be passed when needed from machine to machine in the stack. The second policy, Fully Shared (FS), creates variables for which the unmodified value can be passed among parties. Finally the Secret Shared (SS) policy creates variables for which disparate shares are split among several parties. One example of this kind of sharing is the result of a shared secure scalar product, as in [30]. At the end of this operation, the two participating parties each have shares of the final scalar product value, which are each indistinguishable from random but whose sum is the full result of the operation.

TABLE I
TYPES OF VARIABLE SHARING POLICIES.

Policy	Description
Intra-Party (IP)	Only shared within a party, among layers of the PPDM stack.
Fully Shared (FS)	Represents a variable with shared read and/or write access between at least two parties.
Secret Shared (SS)	Represents a variable where independent shares are given to two or more parties, which combined yield the final result.

The shared variables with policies afford several advan-

tages. First, it makes the sharing and broadcasting of values among parties relatively transparent. An example is given in figure 3. All examples are given in language neutral pseudo-code to reflect that they should be implementable in any popular language. Line 1 declares a shared variable, and line 2 attaches a policy: in this case, the V_{shared} is fully shared between party P_1 and party P_2 . In lines 3–4, executed by party P_1 , a value of 1 is written to V_{shared} . When in lines 3–4, the value of V_{shared} is read by P_2 , P_2 automatically requests and caches that value.

```

1 float  $V_{shared}$ ;
2 setPolicy( $V_{shared}, Policy_{FS}(P_1, P_2)$ );
3 if  $P == P_1$  then
4   |  $V_{shared} = 1$ ;
5 else if  $P == P_2$  then
6   |  $R = 1 + V_{shared}$ ;
7 else if  $P == P_3$  then
8   |  $R = 2V_{shared}$ ;

```

Fig. 3. Example shared variable usage

One of the primary advantages behind shared variables with policies is that they offer automatic checking for authorization. Figure 3 also gives an example of this scenario. Suppose code is accidentally written such that it tries to access a variable locked onto another party (lines 7–8). Upon trying to retrieve this value, the runtime will throw a security exception, and the execution will typically be halted.

C. Main Execution Layer

The Main Execution Layer (MEL) is itself a collection of services. These are the high level services that comprise the full data mining algorithms themselves (e.g. Naïve Bayes, k-NN, SVM, etc.) which are then easily integrated into higher-level applications. The MEL also consists of the

processes that are responsible for directing the execution of the DDM/PPDM algorithm.

Input: Points in $\mathbf{X} = X_r, \forall r = 1 \dots n$
Output: Probabilities p_{yz} of an instance with class y and attribute value z .

```

1 Vector< float >  $\mathbf{c}^k$ ;
2 setPolicy( $\mathbf{c}^k, Policy_{IP}$ );
3  $\mathbf{c}^k = \text{categoryAttributeCounts}(\mathbf{X}^k)$ ;
4 Vector< float >  $\mathbf{c}$ ;
5 setPolicy( $\mathbf{c}, Policy_{FS}(P^1)$ );
6  $\mathbf{c} = \text{secureSum}(P^*, \mathbf{c}^k)$ ;
7 Vector< float >  $\mathbf{n}^k$ ;
8 setPolicy( $\mathbf{n}^k, Policy_{IP}$ );
9  $\mathbf{n}^k = \text{categoryCounts}(\mathbf{X}^k)$ ;
10 Vector< float >  $\mathbf{n}$ ;
11 setPolicy( $\mathbf{n}, Policy_{FS}(P_1)$ );
12  $\mathbf{n} = \text{secureSum}(P^*, \mathbf{n}^k)$ ;
13 foreach  $(y, z)$  do
14    $p_{yz} = c_{yz}/n_y$ ;

```

Fig. 4. Service to implement PPDM Naïve Bayes classifier [7]

The algorithm in figure 4 represents the portion of the Naïve Bayes algorithm which the MEL is responsible for executing. In lines 1–2 of the algorithm, a variable is created, only accessible to the local PPDM stack, which stores a vector of categorical/attribute pairs and how frequently they occur. The service *categoryAttributeCounts* one line 3 is called from HPC services. This value is then passed to PPDM services (line 6), where is added to the final value \mathbf{c} by secure sum, through a variable which can only be accessed by P^1 . In lines 7–12, similar actions are taken to obtain the overall counts of points in each category, which are again summed and given to P^1 . Finally, the output probabilities p_{yz} are determined using the calculated values.

D. High-Performance Computing Services

For interfacing with databases, and for resource intensive computing conducted during the PPDM process, the High-Performance Computing web services (HPC-WS) provide a generic interface to this functionality. The HPC layer can be adapted by each party to interface with their specific HPC installation, which can include clusters, grids and specialized hardware.

The algorithm given in figure 5 represents a portion of the Naïve Bayes algorithm which executes in the HPC layer. This code is in the form of a MapReduce program, which would interface with a cluster with the training database \mathbf{X} distributed in cluster storage. The *map* procedure in lines 1–4 take each training point as the value, and its index r as the key. The map phase simply counts (y, z) pairs of attributes and class labels for each training point. In the *reduce* phase (lines 6–8), these are summed together into a hashtable responsible for keeping the counts of (y, z) pairs. A vector of (y, z) pairs and their associated counts are returned to the calling application within the MEL.

Input: Points in $\mathbf{X} = X_r, \forall r = 1 \dots n$
Output: Vector of (y, z) pairs and associated counts

```

1 Hashtable  $H$ ;
2 map  $(k_1 = r, v_1 = X_r)$  begin
3   foreach  $z \in X_r$  do
4      $\lfloor \text{collect}((y, z), 1)$ ;
5 end
6 reduce  $(k_2 = (y, z), \mathbf{v}_2)$  begin
7    $H(y, z) += \sum \mathbf{v}_2$ ;
8 end

```

Fig. 5. MapReduce pseudo-code for *categoryAttributeCounts*

E. PPDM Services

The PPDM Services (PPDM-WS) layer provides the primitive DDM/PPDM operations (e.g. secure sum), as well as the send, receive and peer-finding functions on which those operations are built. Developing this set of services for PPDM is efficient, because most popular SMC-based PPDM algorithms utilize a small set of SMC operations. By providing a toolkit of frequently used operations, as suggested in [5], developers can easily implement numerous PPDM algorithms. Table II lists algorithms which utilize popular SMC operations.

TABLE II
EXAMPLES OF OPERATIONS AT THE PPDM LEVEL.

Operation	Reference
Secure Sum [31]	[9], [32]
Secure Scalar Product [30]	[33], [34], [32]
Yao Circuits [3]	[33], [35], [34]
Oblivious Transfer [36]	[7], [6]

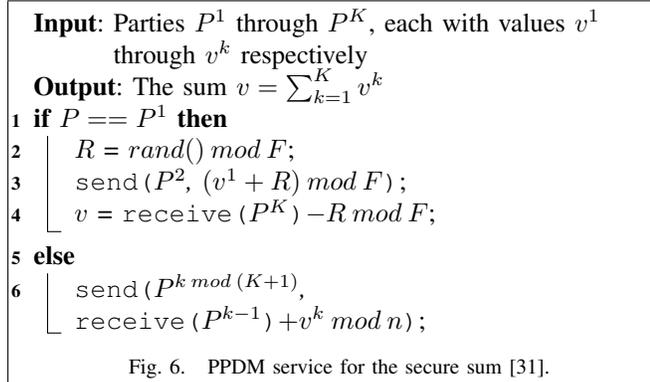
Figure 6 contains the algorithm for secure sum [31], implemented in the PPDM services layer to support the Naïve Bayes algorithm. To simplify the pseudo-code, it is assumed that the result should end up on party P^1 . P^1 starts by creating a random number (line 2) adding it to its value v^1 and sending it along (line 3). Meanwhile, parties P^2 to P^K receive what their previous neighbor is sending, add it to their respective value, and send it along (lines 5–6). Finally, P^1 receives the sum from P^K , subtracts the random number, and obtains the final sum v .

V. IMPLEMENTATION

The APHID prototype is implemented in Java, using Apache Axis2 [37] to handle web services communication. The MEL, PPDM-WS and HPC-WS layers are implemented as Axis2 modules, within a Tomcat [38] application container. MapReduce functionality is provided by Hadoop [28].

VI. CONCLUSIONS

Through analyzing the shortcomings of available PPDM runtime architectures and development systems, we designed a system capable of overcoming those challenges. As the field of PPDM continues to evolve, yielding new algorithms,



SMC techniques, and support for high-performance computing, APHID will continue to evolve as well.

ACKNOWLEDGMENT

This work was supported in part by a National Science Foundation Graduate Research Fellowship as well as NSF grants: 0341601, 0647018, 0717674, 0717680, 0647120, 0525429, 0806931 and 0837332.

REFERENCES

- [1] J. Vaidya and C. Clifton, "Privacy-preserving data mining: Why, how, and when," *IEEE Security and Privacy*, vol. 2, no. 6, pp. 19–27, 2004.
- [2] S. R. M. Oliveira and O. R. Zaane, "Toward standardization in privacy-preserving data mining," in *In Proc. of the 3rd Workshop on Data Mining Standards (DM-SSP 2004), in conjunction with KDD 2004*, 2004, pp. 7–17.
- [3] A. Yao, "How to generate and exchange secrets," in *Proc. 27th Annual Symposium on Foundations of Computer Science*, 1986, pp. 162–167.
- [4] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*. New York, NY, USA: ACM Press, 1987, pp. 218–229.
- [5] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *SIGKDD Explorations*, vol. 4, no. 2, pp. 28–34, 2003.
- [6] B. Pinkas, "Cryptographic techniques for privacy-preserving data mining," *SIGKDD Explor. Newsl.*, vol. 4, no. 2, pp. 12–19, 2002.
- [7] M. Kantarcoglu and J. Vaidya, "Privacy preserving naive bayes classifier for horizontally partitioned data," in *IEEE ICDM Workshop on Privacy Preserving Data Mining*, Melbourne, FL, November 2003, pp. 3–9.
- [8] J. Vaidya and C. Clifton, "Privacy-Preserving K-Means Clustering over Vertically Partitioned Data," in *The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, August 2003. [Online]. Available: <http://www.cs.purdue.edu/homes/jsvaidya/pub-papers/vaidya-kmeans.pdf>
- [9] H. Yu, J. Vaidya, and X. Jiang, "Privacy-preserving svm classification on vertically partitioned data," in *Pan-Asia Conference on Knowledge Discover and Data Mining (PAKDD)*, Singapore, 2006, pp. 647–656.
- [10] J. Zhan, L. Change, and S. Matwin, "Privacy preserving k-nearest neighbor classification," *International Journal of Network Security*, vol. 1, no. 1, 2005.
- [11] J. Vaidya and C. Clifton, "Privacy preserving association rule mining in vertically partitioned data," in *In The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2002.
- [12] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in *In Proceedings of OSDI'04: Sixth Symposium on Operating System Design and Implementation*, December 2004.
- [13] C.-T. Chu, S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun, "Map-reduce for machine learning on multicore," in *In the Proceedings of NIPS 19*, 2006.
- [14] S. Parthasarathy and R. Subramonian, "Facilitating data mining on a network of workstations," in *Advances in Distributed and Parallel Knowledge Discovery*. AAAI Press, 2000.
- [15] M. Cannataro, A. Congiusta, A. Pugliese, D. Talia, and P. Trunfio, "Distributed data mining on grids: services, tools, and applications," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 34, no. 6, pp. 2451–2465, 2004.
- [16] M. Cannataro, A. Congiusta, D. Talia, and P. Trunfio, "A data mining toolset for distributed high-performance platforms," in *Proceedings of Data Mining 2002*, W. I. Press, Ed., Bologna, Italy, 2002.
- [17] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *International Journal of Supercomputer Applications*, vol. 11, no. 2, pp. 115–128.
- [18] V. Stankovski, M. Swain, V. Kravtsov, T. Niessen, D. Wegener, J. Kindermann, and W. Dubitzky, "Grid-enabling data mining applications with datamininggrid: An architectural perspective," *Future Generation Computer Systems*, 2007.
- [19] J. M. P. na and E. Menasalvas, "Towards flexibility in a distributed data mining framework," in *Proc. DMKD Workshop*, 2001.
- [20] M. Z. Ashrafi, D. Taniar, and K. Smith, "A data mining architecture for distributed environments," in *Second International Workshop on Innovative Internet Computing Systems*, June 2002, pp. 27–38.
- [21] A. Kumar, M. Kantardzic, P. Ramaswamy, and P. Sadeghian, "An extensible service oriented distributed data mining framework," in *Proc. 2004 International Conference on Machine Learning and Applications*, December, 2004, pp. 256–263.
- [22] W. Ahmad and A. Khokhar, "Triumpf: A trusted middleware for fault-tolerant secure collaborative computing," University of Illinois at Chicago Tech Report, Tech. Rep. TR-MSL0786, August 2006.
- [23] "Jxta technology," <http://www.sun.com/software/jxta/>, 2008.
- [24] J. Wang, C. Xu, H. Shen, , and Y. Pan, "Hierarchical infrastructure for large-scale distributed privacy-preserving data mining," in *In Proc. of the 5th International Conference on Computer Science*, May 2005, pp. 1020–1023.
- [25] W. Ahmad and A. Khokhar, "Towards secure and privacy preserving data mining over computational grids," in *In Proceedings of the NSF International Workshop on Frontiers of Information Technology*, December 2003.
- [26] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay - a secure two-party computation system," in *Proc. of the USENIX Security Symposium*, 2004, pp. 287–302.
- [27] S. Bailey, R. Grossman, H. Sivakumar, and A. Turinsky, "Papyrus: A system for data mining over local and wide area clusters and super-clusters," in *1999 ACM/IEEE conference on Supercomputing*. Portland, OR: ACM Press, 1999.
- [28] "Welcome to hadoop!" <http://lucene.apache.org/hadoop/>, 2007.
- [29] "Mpi forum," <http://www.mpi-forum.org/>, 2008.
- [30] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikäinen, "On private scalar product computation for privacy-preserving data mining," in *Information Security and Cryptology - ICISC 2004, 7th International Conference, Seoul, Korea, December 2-3, 2004, Revised Selected Papers*, ser. Lecture Notes in Computer Science, C. Park and S. Chee, Eds., vol. 3506. Springer, 2004, pp. 104–120.
- [31] B. Schneier, *Applied Cryptography*, 2nd ed. John Wiley & Sons, 1995.
- [32] J. Secretan, M. Georgiopoulos, and J. Castro, "A privacy preserving probabilistic neural network for horizontally partitioned databases," Aug. 2007.
- [33] G. Jagannathan, K. Pillaipakkamnatt, and R. Wright, "A new privacy-preserving distributed k-clustering algorithm," in *Proceedings of the 2006 SIAM International Conference on Data Mining (SDM)*, 2006.
- [34] G. Jagannathan and R. N. Wright, "Privacy-preserving distributed k-means clustering over arbitrarily partitioned data," in *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. New York, NY, USA: ACM Press, 2005, pp. 593–599.
- [35] Y. Lindell and B. Pinkas, "Privacy preserving data mining," *Journal of Cryptology*, vol. 15, no. 3, 2002.
- [36] M. Naor and B. Pinkas, "Oblivious transfer and polynomial evaluation," in *STOC '99: Proceedings of the thirty-first annual ACM symposium on Theory of computing*. New York, NY, USA: ACM Press, 1999, pp. 245–254.
- [37] "Apache axis2," <http://ws.apache.org/axis2/>, 2008.
- [38] "Apache tomcat," <http://tomcat.apache.org/>, 2009.

High Speed Data Perturbation Methods for Privacy Preserving Data Mining

Mohammad Ali Kadampur, Somayajulu D.V.L.N.
Department of Computer Science & Engineering
National Institute of Technology, Warangal A.P.India-506004.

Abstract – Privacy preserving data mining is an activity of knowledge extraction in which data analysis is done while preserving the privacy of the data proper. Data perturbation is one of the well accepted methods to preserve privacy in security related data mining applications. In this paper signal processing methods have been explored for data perturbation. It is observed that the time complexity of these approaches show considerable improvement over the existing methods and are suitable for perturbing large data sets. A generalized algorithm inviting a signal processing method is presented and implementations are compared under a set of privacy preserving metrics for different signal processing methods. The paper concludes by reporting experimental results on real-life data sets under different signal processing methods.

Key words—Privacy preserving data mining, data perturbation, signal processing, wavelet transforms.

1 Introduction

Data perturbation is a technique in which the original data is altered in such a way that the modified data appears totally different from the original data but yet preserves the essential summary statistics of the original data to produce valid data mining results on par with the original data. If X is the original data then

$$Z = AXB + C \quad (1)$$

represents the perturbed data where A is a record-transforming mask, B is an attribute-transforming mask and C is a displacing mask (noise). A perturbation algorithm α is then a randomized algorithm that, given a dataset X creates a dataset Z having the same number of rows and columns. The perturbation algorithm is public. However, the actual random

numbers used by it are hidden. If $U\{a_1, a_2, a_3, \dots, a_N\}$ is a user analysis set where $a_1, a_2, a_3, \dots, a_N$, are analysis parameters such as mean, sum, regression, clusters, association rules, classification accuracy etc, then we need an algorithm α that gives a Z such that analysis on $U(X) \equiv U(Z)$.

Many methods [2][8][10][13] do exist to perturb the data. However these methods are incomplete as they do not address all the parameters in the U set, and U set itself is always an incremental set!. Apart from having their specific drawbacks these perturbation methods have high time complexity and are not suitable for large data sets. In this paper we are looking at some novel approaches for data perturbation which have improved time complexity and provide high speed data perturbation suitable for large datasets. We have explored application of Fast Fourier Transformations and Discrete Wavelet Transformations (DWT) to perturb the data. It is found that the time complexity is in the range $O(n \log n)$ for FFT to $O(n)$ for DWT. We have tested on the standard data sets and reporting our observations. However we do not make any claim of the proposed methods being generic, addressing the complete U set.

2 Motivation

Motivation for this work comes from the observations that we make from some of the signal transformations. In communication engineering there are several signal transformation methods [5]. These transformation methods are used to convert signals from t domain to s domain and visa versa.

$$f(t) \longleftrightarrow F(s)$$

These transformations from one domain to another domain offer greater convenience of analysis and computation of signals. It is to be noted that the information content in the signals is not lost irrespective of the domain that the signal is in. Consider the following equation (2), and its two representations shown in the Figure-1 and Figure-2,

$$g(x) = 20x^2(1-x)^4 \cos 12\pi x \quad (2)$$

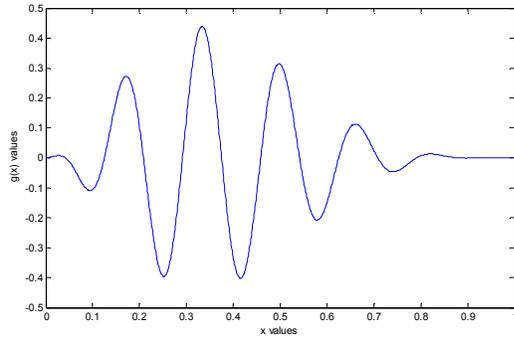


Figure-1: Original Signal g(x).

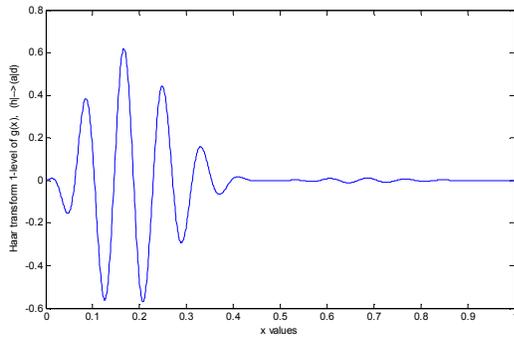


Figure-2: Level-1 Haar transformed Signal of g(x).

Now instead of presenting data from figure-1 we can as well present the data from figure-2 and still expect equally the same results of analysis for example the pattern of fall and rise in the signal, number of positive peaks and negative peaks etc. Observations such as this from wavelets motivate us to use them for data perturbation.

3 Wavelet Transform

The wavelet transform is a synthesis of ideas that emerged over many years from different domains, such as mathematical physics and signal processing. Practically, a wavelet Transform WT is a separate convolution of the signal in question with a family of functions obtained from a basic one by shifts and dilations.

$$W_{\psi}(\tau, a) f = \int 1/\sqrt{a} \psi'((t - \tau) / a) f(t) dt \quad (3)$$

Wavelets are nothing but realization of a decomposition of a function with respect to the representation of the affine group $x \rightarrow ax+b$ [5]. A wavelet transform converts data from original domain to a wavelet domain by expanding the row data in an orthogonal basis generated by dilation and translation of mother wavelet.[19].

Haar Wavelet transform

Haar wavelets are the most commonly used orthogonal wavelets in database and computer science because of their simplicity in understanding and fast way of computation. In

the process of Haar wavelet decomposition the length of each input data row is restricted to an integer power of 2. If this condition does not satisfy, each row can be extended by zero padding the row. The Haar wavelet has the following mother function[21].

$$\Psi_H(t) = \begin{cases} 1, & 0 < t < 0.5 \\ -1, & 0.5 < t < 1 \\ 0, & \text{otherwise} \end{cases}$$

In a dataset containing n attributes, each data record is defined as a vector $f = \{f_1, f_2, f_3, \dots, f_n\}$, located in the highest stage $R = \log_2(n)$. In the first step of Haar Wavelet decomposition, each data record is divided into an approximation coefficient and a wavelet coefficient part. In the next step, an approximation coefficient and a wavelet coefficient is computed from the approximation coefficient of the previous step. This process is repeated until a stopping criterion or the last (zero) stage.

Let $f = (f_1, f_2, f_3, \dots, f_N)$ is a discrete record, then it will be decomposed into approximation coefficients and detail coefficient in a recursive way. These coefficients correspond to low frequency and high frequency components respectively. For example, $(f \mapsto (a^1 | d^1))$ at level 1 decomposition. Where $a^1 = (a_1, a_2, a_3, \dots, a_{N/2})$ is the first trend signal which is formed by taking running averages of the first pair of values and $d^1 = (d_1, d_2, d_3, \dots, d_{N/2})$ is the first fluctuation signal. In general the trend subsignal and the fluctuation subsignal are computed using the formulae :

$$a_m = \frac{f_{2m-1} + f_{2m}}{\sqrt{2}} \quad (4)$$

and

$$d_m = \frac{f_{2m-1} - f_{2m}}{\sqrt{2}} \quad (5)$$

where $m = 1, 2, 3, \dots, N/2$.

If $f = (2, 4, 3, 6, 7, 12, 8, 3, 1, 5)$, Then the level-1 approximation coefficients (trend subsignal) and detailed coefficients (fluctuation subsignal) will be as below.

$$\begin{array}{cccccc} f: & 2, & 4, & 3, & 6, & 7, & 12, & 8, & 3, & 1, & 5 \\ & \swarrow & \searrow \\ a^1: & 6/\sqrt{2} & 9/\sqrt{2} & 19/\sqrt{2} & 11/\sqrt{2} & 6/\sqrt{2} & & & & & \end{array}$$

$$\begin{array}{cccccc} f: & 2, & 4, & 3, & 6, & 7, & 12, & 8, & 3, & 1, & 5 \\ & \swarrow & \searrow \\ d^1: & -2/\sqrt{2} & -3/\sqrt{2} & -5/\sqrt{2} & 5/\sqrt{2} & -4/\sqrt{2} & & & & & \end{array}$$

The Haar transform is performed in several stages, or levels till the stopping criterion is met. The first level Haar is defined by

$$f_H^1 \mapsto (a^1 | d^1) \quad (6)$$

similarly, multilevel Haar are obtained by recursive decomposition of the trend signal. e.g. $f_H^2 \mapsto (a^2 | d^2 | d^1)$ with a^2, d^2 being computed from a^1 .

In general any DWT decomposition proceeds to down sample the original matrix into i^{th} level approximation co-efficients and detail co-efficients at horizontal, vertical and diagonal dimensions of the matrix. This scheme is shown in figure-1. A_i contains i^{th} level approximation co-efficients, D_i^H , D_i^V , D_i^D contain horizontal, Vertical and diagonal coefficients respectively..

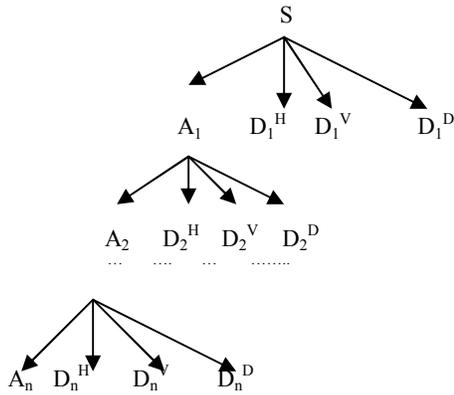


Figure-1 : The DWT decomposition schema.

A_1, A_2, \dots, A_n indicate approximation co-efficients in the wavelet decomposition and $D_1^H \dots D_n^H$ and $D_1^V \dots D_n^V$ indicate the difference components of decomposition in horizontal and vertical directions respectively. The 2-dimensional DWT initially uses high pass filter and then low pass filter on the entire data matrix of size $m \times n$. Then the process continues in a recursive manner down sampling the given input. The maximum number of decomposition levels L of a data matrix of size $m \times n$ is $L = \lceil \log_2 \min(m, n) \rceil$. It is to be noted that such transformations follow conservation of energy rule where energy ξ_f is defined as

$$\xi_f = f_1^2 + f_2^2 + f_3^2 + f_4^2 + \dots + f_N^2. \tag{7}$$

As we progress obtaining multiple levels of transformations, It will have compression effect on the input data. This fact that the transformed signal retains some of the properties of the original signal is very inspiring.

3 Related Work

In the past decade, there have been a large number of privacy preserving data mining literatures. Broadly this literature can be divided into two main categories

- (i) Methods that modify data mining algorithms so that mining proceeds in a secured way in distributed environment[6][11].
- (ii) Methods that modify the data proper values in the data set without affecting the summary statistics, classes, patterns, associations and cluster properties of the original data. This paper contributes to the literature of the later category.

Additive Noise :

Masking by uncorrelated noise addition : The vector of observations r_j for the j^{th} attribute of the original dataset R_j is replaced by a vector

$$r'_j = r_j + \epsilon_j$$

Where ϵ_j is a vector of normally distributed errors drawn from a random variable $\epsilon_j \sim N(0, \sigma_{\epsilon_j}^2)$, such that $Cov(\epsilon_t, \epsilon_l) = 0$ for all $t \neq l$. This method does not preserve variances nor correlations.

Masking by correlated noise addition : Correlated noise addition also preserves averages and additionally allows preservation of correlation co-efficients. The difference with the above method is that covariance matrix of the errors is now proportional to the covariance matrix of the original data. i.e. $\epsilon \sim N(0, \Sigma_\epsilon)$, where $\Sigma_\epsilon = \alpha \Sigma$.

Masking by noise addition and linear transformation : In [23] this method is proposed. It ensures by having additional transformations that the sample co-variance matrix of the masked attributes is an unbiased estimator for the covariance matrix of the original attributes.

Masking by noise addition and non-linear transformation : In [24] this method is proposed. The advantages of this method are that it can be applied to discrete attributes and that univariate distributions are preserved. However application of this method is time consuming and requires expert knowledge on the data set and the algorithm.

Multiplicative perturbation[22]: Multiplicative perturbation methods attempt to transform the data by keeping a good balance between privacy guarantee and data utility. These methods while transforming the data preserve the task and model specific information for mining. The major challenge here is selecting a good transformation that provides satisfactory level of privacy guarantee. There are three major transformation methods that have been investigated

- (i) Rotation transformation
- (ii) Projection transformation
- (iii) Geometric transformation

Rotation transformation [25] : This category of transformation includes all orthonormal perturbations. A rotation transformation is defined as following $G(X)$:

$$G(X) = RX \tag{8}$$

The matrix $R_{d \times d}$ is an orthonormal matrix. Which has the property $R^T R = R R^T = I$. The key feature of rotation transformation is that it preserves the Euclidean distance of multidimensional points during the transformation.

Projection transformation [26] : This transformation is based on the Johnson-Lindenstraus Lemma[11].Which states *Lemma: For any $0 < \epsilon < 1$ and any integer n , let k be a positive integer such that*

$$k \geq \frac{4 \ln n}{\epsilon^2/2 - \epsilon^3/3}$$

Then for any set S of n data points in d dimensional space \mathbb{R}^d , there is a map $f: \mathbb{R}^d \rightarrow \mathbb{R}^k$ such that , for all $x \in S$, $(1-\epsilon) \|x-x\|^2 \leq \|f(x)-f(x)\|^2 \leq (1+\epsilon) \|x-x\|^2$ where $\|\cdot\|$ denotes vector 2 norm.

Basically projection transformation is based on projecting a set of data points from a high-dimensional space to a randomly chosen lower-dimensional subspace. If P_{kxd} is projection matrix then the transformation would be

$$G(X)=PX. \quad (9)$$

Geometric transformation : This transformation is an enhancement to rotation transformation. The definition of geometric transformation is given by a function $G(X)$:

$$G(X) = RX + \Psi + \Delta \quad (10)$$

Ψ is a translation matrix , such that $\Psi = [t_1, t_2, t_3, \dots, t_{dxn}]$ i.e $\Psi_{dxn} = t_{dx1} \mathbf{1}_{N \times 1}^T$. Where $\mathbf{1}_{N \times 1}^T$ is the vector of N '1's. Δ_{dxn} is random noise(Gaussian noise $N(0, \sigma^2)$)matrix where each element is Independently and Identically Distributed (iid) variable ϵ_{ij} . Geometric transformation does not change distance as for any pair of points x and y .

$$\|(x+t)-(y+t)\| = \|x-y\| \quad (11)$$

By adding an appropriate level of noise Δ distance based reconstruction attacks can also be prevented.[28].

In [8][13] data anonymization strategies are proposed that suppress the confidential identifiers such as social security number, driver's license numbers or credit card numbers. However the intruder can still attack the data set by exploring relationships in other attributes known as "quasi identifiers".

Data randomization methods have been proposed in [2][9][13] in which a noise is added from the known distribution of the data so as to modify the original data. This method preserves data utilities such as patterns, association rules etc. However *Karguta et al* [13] showed that such modified data can be easily attacked by re-computing distributions and removing the noise.

Matrix decomposition and factorization methods have been proposed in [18]. Original data matrix D of size $n \times m$ is decomposed into three matrices U , Σ and V^T .

$$D=U\Sigma V^T \quad (12)$$

Where U is $n \times n$ orthonormal matrix, $\Sigma = \text{diag}[\sigma_1, \sigma_2, \sigma_3, \dots, \sigma_s]$ is a diagonal matrix of $n \times m$ order whose non-negative diagonal entries are in descending order. $s = \min(n, m)$. V^T is an orthonormal matrix of order $m \times m$. Due to the arrangement of singular values in matrix Σ , this method is also known as Singular Value Decomposition (SVD) method. SVD [18] method is found to be very effective in providing high level of data privacy preservation and better data utility. However run time complexity of $O(n^3)$ increases the computational cost of SVD method .

NNMF: Non Negative Matrix Factorization method [20], This method works on data set with non negative values, decomposing it into two non negative sets meeting the defined objective function.

Given the non negative matrix $A \in \mathbb{R}^{n \times m}$ with $A(i,j) \geq 0$ and a pre-specified positive integer $k \min(n, m)$, NNMF, finds two non negative matrices $W \in \mathbb{R}^{n \times k}$ with $W(i,j) \geq 0$ and $H \in \mathbb{R}^{k \times m}$ with $H(i,j) \geq 0$ so that $A = WH$ that minimizes the objective function

$$f(W, H) = \frac{1}{2} \|A - WH\|_F^2 \quad (13)$$

NNMF is shown to have better data utility and risk of disclosure properties over other decomposition methods.

Association rule hiding[27] : This is class of perturbation techniques developed to hide the confidential data from the 'frequent item set mining' or the association rule mining. The other class of methods that can be similarly included in this category are classification rule hiding, clustering model hiding[17], sequence hiding and so on and so forth.

4 The Proposed Approach

We are proposing to use Signal Transformations, Wavelet Transforms in particular, for data perturbation. As Wavelet transformations follow a recursive decomposition (divide and conquer), the given matrix D of order $m \times n$ is adjusted to have rows and columns to be of integer powers of 2 by a method of scaling up using zero padding. There is a Dominant Component corresponding to the zero frequency and this DC is moved to the center of the transformed matrix. Then forward ST is done on the newly DC centered matrix. Filtering (Low/High pass) is applied to get a filtered matrix D^p . Inverse ST is applied on D^p to get the distorted matrix D^p_d . By selecting the upper left $m \times n$ elements of D^p_d we get the finally required distorted version of D , i.e D' . D' preserves the properties of D and it is safe to publish.

4.1 Algorithm : Algorithm for Data Perturbation using Signal Transformation (ST)

Input: Matrix D of size M x N

Output : Perturbed Matrix Q of size M x N

Begin.

Step 0: Read the size of the matrix D.

Step 1: If $M \neq 2^k$ and $N \neq 2^p$, Zero pad the matrix D to the nearest integer which is power of 2. Let the zero padded matrix be D^{\sim} with the new size $M^{\sim} \times N^{\sim}$.

Step 2: Do dominant component shifting (DC shifting). To do this multiply each entry (x,y) of D^{\sim} by $e^{j\pi(x+y)}$ and get the new matrix D^{\wedge} of size $M^{\sim} \times N^{\sim}$.

Step 3: Do forward signal transformation on D^{\wedge} to get transformed matrix T.

Step 4: Do filtering (Either low pass or high pass) to get filtered matrix T^{\sim} .

Step 5: Obtain the inverse signal transformation (IST) on T^{\sim} and obtain a new matrix Q^{\sim} .

Step 6: Pick only the left side of Q^{\sim} for size of M x N. Let this new matrix be Q.

Step 7: Q has preserved the properties of D and is the perturbed version of the original matrix

Step 8: Submit matrix Q.

End.

Algorithm 4.1, uses any signal transformation (ST) at its core for data distortion. We have conducted experiments by substituting FFT and variants of Discrete Wavelet Transforms (DWT) in place of ST.

DWT decomposes the original data into approximation coefficients and detail coefficients and then suppresses the high frequency detail coefficients to achieve data distortion. The inverse signal transformation (IST) step sets back the dimensions of the transformed dataset to the original data set. The basis functions in the wavelets have been changed from single basis to multi basis and the effect on data distortion is observed.

5 Implementation

We have used C++ as programming language to implement the algorithms. MATLAB was used in the initial stages to test, confirm the steps in the algorithm. We have used in-place

version of FFT [5] algorithm for implementation. It has a run time complexity of $O(N \log N)$. m-files of MATLAB have been explored and the code is re-used at different integration stages of the software. The proposed algorithm has provision for substituting any Signal Transformation method (ST) during data perturbation experiments. Different thresh hold values can be substituted by designing different filters with required cut off values.

6 Experiments and Results

When a simple matrix such as D is distorted using the software developed we got matrix such as D'

$$D = \begin{pmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{pmatrix} \quad D' = \begin{pmatrix} 0.0018 & 0.0034 & 0.0061 \\ 0.0031 & 0.0074 & 0.0080 \\ 0.0057 & 0.0145 & 0.0066 \end{pmatrix}$$

D' is the distorted matrix of D. D' produces the same or nearly same mining results as does the matrix D. and D' is safe because when D' is submitted for public domain or the second party for knowledge extraction, the data proper values are kept secret.

6.1 Metrics

Once the data matrix is distorted we can measure the distortion by evaluating several defined metrics. The simplest among them all is the value difference VD defined as below, With X as undistorted matrix and X' as distorted matrix.

$$VD = \|X - X'\|_F / \|X\|_F \quad (15)$$

Accuracy: It is a statistical measure that quantifies how well a classifier identifies or excludes a condition. It indicates the percentage of correctly classified samples (true positives + true negatives) from a total pool of samples.

$$Accuracy = \frac{\text{Number of samples correctly classified}}{\text{Total Number of Samples}} \quad (16)$$

RP : After distortion the relative order of the value of data elements also changes. Metric RP denotes the average change of order for all attributes and is defined as

$$RP = \frac{(\sum_{i=1}^m \sum_{j=1}^n |Ord_j^i - \overline{Ord}_j^i|)}{m*n} \quad (17)$$

Ord_j^i is order of the i^{th} attribute in j^{th} row of original data and \overline{Ord}_j^i is order of the i^{th} attribute in j^{th} row of the perturbed data.

RK : It represents the percentage of elements that keep their orders of value in each column after perturbation.

$$RK = \frac{(\sum_{i=1}^m \sum_{j=1}^n RK^i_j)}{m*n} \quad (18)$$

Where,

$$RK^i_j = \begin{cases} 1, & \text{if } \text{Ord}_j^i = \overline{\text{Ord}_j^i} \\ 0, & \text{otherwise.} \end{cases}$$

Lower values for RK are expected for better privacy.

6.2 Dataset

Census income database from UCI archives is used for testing the implementation. This data set contains weighted census data extracted from the 1994 and 1995 current population surveys conducted by the U.S. Census Bureau. The data contains demographic and employment related variables. The data has the following characteristics.

Number of attributes = 40 ; Number of instances= 199523;
Duplicate or conflicting instances : 46716

TABLE 1: COMPARISON OF HIGH SPEED DATA PERTURBATION METHODS

Data	RP	CP	CK	RK	(SVM) Accuracy	Run-Time(S)
Original	---	---	---	---	86.49	---
FFT	936.5346	5.3208	0.0000	0.0100	85.42	3.63
Haar	928.6754	5.3609	0.0008	0.0087	83.98	1.98
Dubechies4	932.8765	5.4217	0.0004	0.0101	86.12	1.38
Symlet	930.1278	5.3367	0.0006	0.0091	85.87	1.46
Biorthogonal	934.6754	5.2889	0.0011	0.0104	85.54	1.42

CP : It defines the change of order of average value of the attributes.

$$CP = \frac{(\sum_{i=1}^m | \text{OrdAV}_i - \overline{\text{OrdAV}_i} |)}{m} \quad (19)$$

where OrdAV_i is the ascending order of average value of attribute i of the original data set. and $\overline{\text{OrdAV}_i}$ is ascending order of average value of attribute i of the perturbed data set.

CK : It measures the percentage of attributes that keep the orders of their average value after perturbation.

$$CK = \frac{(\sum_{i=1}^m Ck^i)}{m} \quad (20)$$

Where,

$$Ck^i = \begin{cases} 1, & \text{if } \text{OrdAV}_i = \overline{\text{OrdAV}_i} \\ 0, & \text{otherwise.} \end{cases}$$

It is to note that after data perturbation there will be value difference in the original set and the perturbed set. and these metrics [3], CP, CK, RP, RK in one or the other way indicate the extent of change in the relative order of the values.

Higher values for RP, CP and lower values for RK, CK indicate better privacy.

7 Conclusion and Future Work

In this paper we have presented the application of signal transformation methods for data perturbation. We have compared performance of Fast Fourier transforms and a few variants of wavelet transforms. It is found that the computational time used by FFT and DWT methods is very small and that makes them the high speed data perturbation methods. The classification analysis results of the perturbed data using wavelets and FFT methods are as good as that of the original one. With respect to the complexity of the run time the wavelet transforms have a complexity of $O(n)$ and FFT is of $O(n \log n)$ complexity.

Our experiments show that signal transformation methods carry the promise of faster methods of data perturbation suitable for large data sets. The simulation results and comparative study are to be substantiated by theoretical proofs in the future work.

References

- [1] Shuting Xu and Shuhua Lai , "Fast Fourier transform based data perturbation method for privacy protection" In the proceedings of IEEE Conference on Intelligence and Security Informatics, New Brunswick New Jersey, May 2007
- [2] R. Agarwal and R. Srikant, "Privacy preserving data mining" In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas Texas, May, 2000.
- [3] Shuting Xu, Jun Zhang, Dianwei Han and Jie Wang "Data distortion for privacy protection in terrorist analysis system" Technical report No 432-05, Department of Computer Science, University of Kentucky, Lexington, KY 2005.

- [4] Howard L. Resnikoff and Raymond O. Wells, Jr. "Wavelet Analysis-The Scalable Structure of Information" First Indian edition, Springer(India),2004,ISBN:81-8128-226-4.
- [5] M. V. Altaisky, "Wavelets Theory Applications Implementation" First edition, Universities Press(India),2005,ISBN: 81-7371-503-3.
- [6] Y. Lindel and B. Pinkas "Privacy Preserving Data Mining" In the Journal of Cryptography, vol. 15, no. 3, pp. 177-206, 2002
- [7] P. Samarati, "Protecting Respondent's Privacy in Microdata release," IEEE transactions on Knowledge and Data Engineering., Vol. 13, pp. 1010-1027, 2001.
- [8] L.Sweeney, "k-Anonymity: A Model for Protecting Privacy," International Journal on Uncertainty,Fuzziness and Knowledge-Based Systems, vol. 110, no. 5, pp.557-570, 2002.
- [9] D. Agrawal and C. Aggarwal, "On the Design and Quantification of Privacy Presrving Data Mining Algorithms," In the proceedings of 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS), 2001.
- [10] V. S. Verykios, E. Bertion, I.N. Fovino, L.P. Provenza, Y. Saygin and Y. Theodoridis, "State-of-the-Art in Privacy Preserving Data Mining," ACM SIGMOD Record, vol. 33, no 1, 2004.
- [11] J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data," In the proceedings of ACM Conference on Knowledge Discovery and Data Mining, 2002.
- [12] G. J. Kowalski and M.T. Maybury, "Information Storage and Retrieval Systems Theory and implementation" 1st edition Kluwer academic Publishers,1997.
- [13] Charu C. Agarwal, Yu Philip S., "Privacy Preserving Data Mining Models algorithms," eds, Springer, vol. 34, 2008,ISBN: 978-0-387-70991-8.
- [14] David Kuncicky "MATLAB programming" eds, Pearson Prentis Hall, 2003, ISBN:013035127X
- [15] Alfio Quarteroni & Fausto Saleri.djvu "Scientific Computing with Matlab", eds, Springer Verlag, vol.10, 2003, ISBN: 3540443630.
- [16] Terran Lane and Ronny Kohavi, UCI Census-income data ([http://archive.ics.uci.edu/ml/datasets/Census-Income+\(KDD\)](http://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD))) United States Census Beuro, Department of Commerce.
- [17] <http://www.spss.com/clementine/>
- [18] S. Xu, J.Zhang, D. Han, J.Wang, "A Singular Value Decomposition Based Data distortion Strategy for Privacy protection" Knowledge and Information Systems (KAIS) journal, vol. 10. No. 3, pp.383-397, 2006.
- [19] L.I.Tao,LI. Qi, Z.Shenghuo and O.Mitsunori, " A survey on wavelet applications in Data Mining" , SIGKDD, 2002,pp. 49-68.
- [20] Jie Wang, Weijun Zhong, Jun Zhang., "NNMF based Factorization techniques for High accuracy Privacy protection on Non Negative valued Datasets" .ICDMW06,0-7695-2702-7/06\$20.00© 2006 IEEE.
- [21] Burrus,C.S.,Gopinath, R.A.,and Guo,H.,"Introduction to wavelets and Wavelet transforms, A Primer", Prentice Hall, Englewood Cliffs, NJ,1997.
- [22] Charu C Aggarwal and Philip S. Yu "Privacy Preserving Data Mining : Models and Algorithms" Springer 2008;ISBN:978-0-387-70991-8.
- [23] J.J.Kim. "A method for limiting disclosure in microdata based on random noise and transformation. In the Proceedings of the section on Survey Research Methods", Pages 303-308, Alexandria VA,1986.American Statistical Association.
- [24] R.Brand. "Microdata protection through Noise addition. In J. Domingo Ferrer,editor,Inference Control in Statistical Databases", volume 2316 of LNCS pages 97-116,Berlin Heidleberg, 2002 Springer
- [25] Sadun,L. Applied Linear algebra:the Decoupling principle.Prentice Hall, 2001.
- [26] Liu,K., Kargupta, H., AND Ryan,J. "Random projection based multiplicative data perturbation for privacy preserving distributed data mining." IEEE Transactions on Knowledge and Data Engineering(TKDE) 18,1(January 2006),92-106.
- [27] M..Attallah, E.Bertino,A Elmagarmid, M.Ibrahim, and V.S.Verykios. "Disclosure limitation of sensitive rules". In proceedings of the 1999 IEEE Knowledge and Data Engineering Exchange Workshop (KDEX'99) pages 45-52,1999.
- [28] Chen, K., AND Liu, L. "Towards attack resilient geometric data perturbation." SIAM Data Mining Conference (2007).

SESSION

TEXT AND WEB MINING

Chair(s)

Dr. Yanjun Li

Personality Based Latent Friendship Mining

Fan Wang, Yuan Hong, Wenbin Zhang and Gagan Agrawal

Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA

Abstract—A web log, or simply a blog is a diary format communication channel on the web. With the massive growth of the global blogging infrastructure, there is a need for meta-services on blogs and bloggers. In this paper, we propose a novel approach for detecting latent friends among bloggers based on the subtle writing styles information revealed by their blog postings. The premise of our approach is that personality is an essential element for bringing people together mentally and enabling mutual friendship. Furthermore, our approach is based on the belief that people's writing styles or patterns, to a large extent, can reveal their personalities.

Based on the above two premises, as in many current systems, we present an approach that takes advantage of features extracted from people's blog postings that are likely to reflect the underlying personality of the bloggers, and then use this information to detect latent friendships. This is in contrast to the many current systems that use blog document content and topics as the main predictors of latent friendship. Our experiments show that most personality related features utilized in our approach are effective and our approach outperforms the approach in which only the information from document content is used.

1. Introduction

Blog, which is short for *web log*, is a diary format communication channel on the web. With the massive growth of the global blogging infrastructure, there is a need for meta-services on blogs and bloggers, which offer services such as searching for blog postings or finding potential friends.

One of the most important functionality of blogs is to provide a kind of platform to exchange and share information with the blogger's friends. Today's life requires more cooperations among people, especially with respect to their careers, businesses, or personal affairs. As a result, friends and friends of friends are often the best and the most precious resources and treasures a person want to explore and keep. Therefore, it is highly desirable that a *latent friendship detection system* could be designed and implemented. In view of this, there have been many recent efforts on blog mining and latent friends mining. The existing research, however, uses the content of the blog postings only, particularly the topics, words, and phrases used in the blog writings [1], [2], [3], [4], [5], [6], [7], [8]. None of these approaches consider the underlying writing patterns presented in blogger's articles.

We believe that the underlying *writing patterns* are useful in identifying, to a large extent, people's characteristics,

personalities, hobbies, attitudes, and emotional mindsets. The writing patterns here not only refer to content of a blogger's postings, but also the *writing style*. We consider the writing style a subtle but potentially useful piece of information capturing the profile of a writer. A blogger may have written many blogs articles on various topics, but if one reads through them, they will find some unchanging patterns underlying the postings. These can reflect the personality profile of the writer, just as people's handwriting can show their characteristics. Research in psychology also suggests a strong relationship between people's personality and their writing styles [9]. People who use exclamation marks, emotional icons, and font color changes or attach pictures and images frequently in their blog articles are **unlikely** to share the same personality or characteristics with the people who always use jargon and scientific words. Similarly, people who like to post long articles frequently will probably not be friends of the people who only post two or three articles in a year.

Based on the subtle but potentially useful writing pattern information in blogs, the method presented in this paper predicts the probability that two given bloggers can become friends. We assume that a writer's profile or personality can be reflected by the writing habits or posting styles and people with similar profile (personality) are more likely to be willing to become friends. The main disadvantage of the existing latent friend mining systems, which are based only on document content similarity, is the underlying assumption that friends are likely to be interested in similar topics. But, we argue that this assumption is restrictive. In real life, many close friends do not share the same opinion on a certain topic or may not have much in common. But, they are attractive to each other mentally. Our belief is that people's personality profile is a better indicator of their potential of becoming friends.

Our method is conducted in three steps. First, 20 carefully selected features related to blogger's personality and topics of interests are extracted from the blogs. Second, for each blogger, a profile vector is built based on the features extracted above. For each pair of bloggers, a personality similarity vector is constructed. Finally, a trained logistic regression model is used to predict the probability of two bloggers being friends. If the probability value is greater than a pre-specified threshold, our method suggests that these two bloggers are likely to be willing to become friends.

Many existing services, such as eHarmony¹ and Match², can match the users according to their personality profiles. These services, however, require that the users explicitly answer a long and tedious list of questions. This can be a time-consuming step, and, furthermore, may make users feel uncomfortable. Our method avoids detailed and explicit questions on personality, and automatically matches users' personality profiles based on carefully selected features from their blog postings. As a result, our method can be easily embedded in any current blogging service providers, such as MSN and Facebook, to enhance the functionalities of these web-sites. By incorporating our techniques, web-sites such as MSN or Facebook can notify a user about another user with similar personality, and establish a communication channel between them.

The rest of the paper is organized as follows. We formulate our problem in Section 2. In Section 3, we introduce the feature set and the mathematical model used in our approach. We evaluate our method in Section 4. We compare our work with other efforts in Section 5, and conclude in Section 6.

2. Problem Formulation

We have a collection of blogs $B = \{b_1, b_2, \dots, b_n\}$. Each blog b_i corresponds to a specific blogger w_i and all the bloggers form a collection of bloggers $W = \{w_1, w_2, \dots, w_n\}$. All the bloggers and their blogs are crawled from public accessible blogosphere. Each blog b_i is composed of a list of posted articles $P_i = \{p_1^i, p_2^i, \dots, p_m^i\}$. Each posted article p_j^i is a four-tuple which contains the following elements $\{C_j^i, T_j^i, Time_j^i, R_j^i\}$. C_j^i is the plain text content of the posted article p_j^i . T_j^i is the set of HTML tags used in the posted article. $Time_j^i$ is the time and date on which the article is posted. R_j^i contains all the information about the comments on the posting p_j^i from the blogger's friends for other bloggers. We assume that each blog b_i is written by a single person w_i who has a user profile ρ_i . We don't have the access to or don't know the user profile ρ_i of person w_i . We propose a function Ψ which is a global reduction function to obtain the user personality profile ρ_i based on the user's posted articles p_j^i , which is denoted as

$$\rho_i = \Psi_{j=1\dots m}(p_j^i)$$

We also propose a similarity function $\Phi(\rho_i, \rho_j)$ to compute a similarity vector θ between two user profiles, ρ_i and ρ_j . Finally, the similarity vector θ is processed by a prediction function Υ , and the probability of the two writers w_i and w_j being friends or willing to become friends is estimated. The prediction function is defined as follows

$$\sigma = \Upsilon(\theta) = \Upsilon(\Phi(\rho_i, \rho_j))$$

If the probability value σ is greater than a threshold β , we call the two writers w_i and w_j to be *friends*.

¹<http://www.eHarmony.com>

²<http://www.Match.com>

3. Feature Set, Profile Vector and Modeling

In this section, we will describe our approach in detail. First, we will introduce the personality related features we selected from the posted articles of each blogger. Secondly, we will explain the personality profile vector ρ_i for a blogger w_i , and the similarity vector for a pair of bloggers w_i and w_j . Finally, a logistic regression model is described to predict the probability of two writers being friends.

3.1 Feature Set

Friendship detection problem can be considered as a classification problem with two classes which are potential friends or not friends. In our system, the feature vector contains 20 carefully selected personality related features.

Average Posting Time (C1): The average time a blogger updates his/her blog. This feature captures the time that a blogger usually posts his/her articles. Some people would like to update their blog in the morning, while other people may like to update their blogs in the evening.

Average Number of Comments (C2): The average number of comments a blogger receives per article. A high number on this feature would suggest the blogger has many friends and is active in social communication, or the blogger's topic is interesting and attractive.

Average Number of Images (C3): The average number of images attached in the blogger's articles. People with different personalities choose different methods to express their ideas or feelings. Some people prefer words and others prefer pictures or music.

Average Number of Special Symbols (C4): The average number of special symbols used in the blogger's articles. Special symbols contains emotional icons, such as :) and :-D, and any special characters, such as \$, & and *. People like to use emotional icons or special symbols are more modern and with younger mentality than the people who seldom use them.

Average Number of Font Changes (C5): The average number of font changes made in the blogger's articles. This feature reflects the blogger's personality, writing habit and mentality.

Average Number of Punctuation (C6): The average number of punctuation used in the blogger's articles. We only focus on the punctuation which may reflect the mood or the mentality of the blogger, such as exclamation mark, colon and ellipsis.

Average Number of Words (C7): The average number of words in the blogger's articles. The bloggers who write long articles may have different personalities from the bloggers who only write articles with one or two sentences.

Average Number of Different Words (C8): The average number of distinct English words in the blogger's articles. This feature reflects the blogger's literacy, vocabulary and probably his/her education level.

Average Lexical Density (C9): The average lexical density of all the posted articles of a blogger. Lexical density [10] is the measurement of the proportion of the content words over the total words. It is designed to show how easy or difficult a text is to read. The lexical density D of a text is computed as follows $D = \frac{N_{dw}}{N_{tw}}$, where N_{dw} denotes the number of different words and N_{tw} denotes the total number of words. A higher lexical density suggests the writer has a relatively large vocabulary and is more likely to be educated.

Average Readability (C10): Readability is defined as reading ease, and it is resulted from a writing style. For each posted article, the readability is computed by the *Gunning Fog Index* [11]. The resulting number of *Gunning Fog Index* is an indication of the number of years of formal education that a person requires in order to understand the text on the first reading. The Gunning Fox Index readability score of a text is computed as $0.4(\frac{N_{tw}}{N_s} + 100(\frac{N_{cw}}{N_{tw}}))$, where N_{tw} denotes the total number of words, N_s denotes the number of sentences and N_{cw} denotes the number of complex words. Complex words refer to the words with three or more syllables, not including proper nouns (for example, Djibouti), compound words, or common suffixes such as -es, -ed, or -ing as a syllable, or familiar jargons. This feature has the same intention as the feature of lexical density.

Average Number of Sentences (C11): The average number of sentences in all the articles of a blogger.

Average Number of Characters (C12): The average number of English characters (letters) used in all the articles of a blogger.

Average Number of Hyperlinks (C13): The average number of hyperlinks in the articles of a blogger. This feature shows whether the writer likes to link his/her page to other pages or not. This feature also reflects a person's writing style and communication style.

Average Number of Adjectives (C14): The average number of adjectives used in all the articles of a blogger. The types of words a person like to use reveal the personality of the person. A person like to use many adjectives are likely to be emotional sensitive.

Average Number of Adverbs (C15): The average number of adverbs used in all the articles of a blogger. This feature is selected with the same purpose of feature C14.

Average Number of Nouns (C16): The average number of nouns used in all the articles of a blogger. This feature is selected with the same purpose of feature C14.

Average Number of Verbs (C17): The average number of verbs used in all the articles of a blogger. This feature is selected with the same purpose of feature C14.

Maximal Sentence Length (C18): The length of the longest sentence among all the sentences in all the articles of a blogger. This feature also reflects the readability of the articles.

Posting Frequency (C19): The average frequency of a blogger updates his/her blog. This feature reflects the posting

style of a blogger. If a blogger updates his/her blog very frequently, it may suggests that the blogger likes to share his/her opinions or experiences with others. If a blogger updates blog infrequently, it may suggests that the blogger is very busy or reluctant to share his/her experiences.

Plain Text (C20): The plain text of all the posted articles of a blogger. All stop words are removed and all the words are stemmed. The blog text reflects the topics and the content that a blogger is interested in. People with different personalities may prefer different types of topics.

All the above 20 features can be obtained by simple word processing techniques and lexical analysis. Although they cannot directly reflect the personality of a blogger, they can be obtained very easily from blogger's articles and to a large extent show some dimensions of people's personality.

3.2 Personality Profile Vector and Similarity Vector

In this section, we will introduce the personality profile function Ψ and similarity function Φ mentioned in Section 2.

3.2.1 Profile Vector and Ψ Function

Based on the features as described in Section 3.1, each blogger w_i has a profile vector with 20 elements as $\rho_i = \{f_1, f_2, \dots, f_{20}\}$. This profile vector is computed from all the articles p_j^i , $j = 1 \dots m$, written by the blogger w_i using a global reduction function Ψ as mentioned in Section 2. The function Ψ merges all the articles written by the blogger w_i into a large single document, denoted as $PG_i = \{CG_i, TG_i, TimeG_i, RG_i\}$, where $CG_i = \bigcup_{j=1..m} C_j^i$, $TG_i = \bigcup_{j=1..m} T_j^i$, $RG_i = \bigcup_{j=1..m} R_j^i$, $TimeG_i = \bigcup_{j=1..m} Time_j^i$. One point needs to be addressed here is that during the merging, all duplicates are kept. In the profile vector $\rho_i = \{f_1, f_2, \dots, f_{20}\}$, the first 19 features are numerical values, and the last feature is the stemmed stop-word removed plain text of all the posted articles of w_i .

3.2.2 Similarity Vector and Φ Function

Friendship is predicted based on the similarity between a pair of bloggers/writers. For two bloggers, w_i and w_j , a similarity vector is computed based on their profile vectors ρ_i and ρ_j . The similarity function Φ is illustrated in detail as follows.

The similarity vector θ is a vector with 20 elements which is denoted as $\theta = \{s_1, s_2, \dots, s_{20}\}$. Each element s_k of θ corresponds to the similarity between features f_k^i and f_k^j in the profile vector ρ_i and ρ_j respectively. For example, $s_1 = \Phi(f_1^i, f_1^j)$. For the first 19 features, which are numerical values, the similarity function is defined as

$$s_k = \Phi(f_k^i, f_k^j) = |f_k^i - f_k^j|, 1 \leq k \leq 19$$

In other words, for the first 19 features, the similarity function returns the absolute difference between the corresponding feature from each profile vector. The last feature of a profile vector is the plain text, and the similarity function returns the document similarity, which is denoted as

$$s_{20} = \Phi(f_{20}^i, f_{20}^j) = \text{DocSim}(f_{20}^i, f_{20}^j)$$

We use the cosine similarity function to compute the similarity between two documents. The TFIDF values for all the terms in the two documents f_{20}^i and f_{20}^j are computed and used in the cosine similarity computation. As a result,

$$\Phi(f_{20}^i, f_{20}^j) = \frac{\sum_k n_{ik} n_{jk}}{\sqrt{\sum_k n_{ik}^2 \times \sum_k n_{jk}^2}}$$

where n_{ik} is the TFIDF value of term k in blogger w_i 's blog plain text (the last element in the profile vector).

Based on the above definition of the Φ function, the similarity vector θ is computed.

3.3 Mathematical Model

Given a similarity vector θ between a pair of bloggers w_i and w_j , we predict the probability of these two bloggers to be friends using logistic regression as used widely in psychological analysis.

The logistic function is defined as $Y = \frac{e^A}{1+e^A}$, where the variable A is defined as:

$$A = \lambda_0 + \lambda_1 X_1 + \lambda_2 X_2 + \dots + \lambda_{20} X_{20}$$

Here the X_i is the elements in the similarity vector θ and λ_i is the corresponding regression coefficient of feature X_i . The output Y of the logistic regression function is a real number ranging from 0 and 1, which can be considered as the probability of the input similarity vector being classified as class 0 (not potential friends) or class 1 (potential friends).

In order to use logistic regression, we need a training data set. The training data set is obtained and labeled as follows. We collect the posted articles of some randomly selected bloggers, while the articles of these bloggers are crawled, we also extract the bloggers' friends information from the friends list of the blogger's main page. In this way, for each blogger we crawled, we have a list of true friends which are identified by the blogger himself/herself. The assumption here is that the bloggers will not randomly add other people as their friends. The similarity vectors of each pair of bloggers crawled as above are computed. If for two bloggers, they are true friends as indicated in the friends list of any of them, their similarity vector is labeled as 1 (potential friends), otherwise, if they are not added as friends in either's friends list, their similarity vector is labeled as 0.5 (unknown). The reason we don't label any similarity vector as 0 (not potential friends) is that we cannot be sure that two people who are not in each other's friends list may not likely to be friends in the future.

Our logistic regression model is trained using the data obtained as above. One important issue in logistic regression is model selection. In other words, since not all the predictors (features) in the model are useful, we need to select a best subset of features. There are mainly two methods for model selection. The first method selects the features with a large p-value which indicates a greater statistical contribution, and the second method is a step-wise feature selection method. In our experiments, we consider model selection problem and three logistic regression models are built, which are *full-feature model*, *significant-feature model* and *stepwise-feature model*. The performance and comparison of the above three regression models are shown in detail in Section 4.

4. Evaluation

In this section, we describe the experiments we conducted to evaluate our techniques. Our evaluation results show that all of our logistic regression models achieve good performance in terms of precision and recall. Furthermore, including the writing style features can significantly improve the performance of potential friendship detection.

4.1 Data Collection and Data Partition

We randomly selected bloggers from MSN live space using the search engine provided by MSN. For all the randomly selected bloggers, we first downloaded all their posted articles in the period from the year of 2005 to 2007 and filtered out the articles written in non-English languages. For the purpose of feature extraction, all the HTML tags in blog articles were extracted and meanwhile word stemming and stop-word removal were performed. Nouns, verbs, adjectives and adverbs are extracted using an existing part-of-speech tagger, TreeTagger [12]. Second, their friend lists were read and extracted. Third, based on the friend information we obtained in the previous step, all the posted articles of the friends of the randomly selected bloggers were also downloaded. In this way, we crawled a collection of MSN bloggers and also established the blogger-friends relations which was used in training data set labeling and testing data set validation.

There are totally 200 bloggers selected as our data and the amount of pure text data (excluding the embedded picture or music files in posted articles) for analyzing is 200MB.

Now we need to form our training and testing data set. Because the focus of our technique is to detect potential friendship, we need to pair the 200 bloggers selected in the data collection step. All possible permutations of the 200 bloggers are used to form blogger pairs, which yields $\frac{200 \times 199}{2}$ blogger pairs. These pairs are our entire data set. All the pairs are labeled automatically. For each blogger pair, if either of the two is a friend in the other's friend list, this pair is label as 1, indicating a friend pair, otherwise the pair is labeled as 0.5, indicating an unknown class. The training and testing data is split according to 80%-20% ratio, which

Table 1: Full Feature Regression Model

Predictor	λ Coefficients	P-value
Constant	-1.28819	0.001
C1	-0.0530238	0.108
C2	0.0335932	0.000 *
C3	-0.0471635	0.201
C4	0.994459	0.001 *
C5	-0.0164398	0.015 *
C6	0.0024974	0.946
C7	0.0043979	0.195
C8	-0.0002100	0.973
C9	-2.51700	0.053 *
C10	0.146851	0.000 *
C11	-0.0319387	0.162
C12	-0.0002312	0.631
C13	0.291657	0.000 *
C14	0.192159	0.001 *
C15	0.115692	0.140
C16	-0.0697147	0.000 *
C17	0.0007009	0.985
C18	-0.0002413	0.830
C19	-0.0024932	0.593
C20	8.18092	0.000 *

yields a training data set of size 15920 and a testing data set of size 3980.

4.2 Regression Model Selection

All the training data is used in regression model selection. We use the MINITAB software package to build the logistic regression models.

Full-Feature Model: The *full-feature model* is built using all the 20 features introduced in Section 3.1. The regression result is shown in Table 1. The first column in the table shows the 20 features used in the full-feature model, the second column is the regression coefficients λ_i , and the third column shows the p-value of the feature. A small p-value indicates higher statistical significance of the feature in the model. The features highlighted in red and the star signs in Table 1 are the features with significant statistical contribution to the model, while other features don't have significant contribution to the model.

Significant-Feature Model: In the *full-feature model*, only 9 of 20 features have a small p-value which indicates that we are able to construct a simpler model without losing much of the accuracy. In the *significant-feature model*, we only use the 9 features which are statistically significant (highlighted as in Table 1). The regression result of the *significant-feature model* is shown in Table 2.

Stepwise-Feature Model: We want to build a regression model which is simple (doesn't have too many features) and also fits the training data pretty well. To achieve this aim, we consider *stepwise-feature model*. The *stepwise-feature model* is constructed as follows. Initially, the model only contains one feature which gives the largest statistical significance.

Table 2: Significant Feature Regression Model

Predictor	λ Coefficients	P-value
Constant	-1.40997	0.000
C2	0.0337845	0.000
C4	0.853149	0.002
C5	-0.0129985	0.008
C9	-1.39296	0.127
C10	0.128089	0.000
C13	0.239424	0.000
C14	0.245883	0.000
C16	-0.0712249	0.000
C20	7.90527	0.000

Table 3: Stepwise Feature Regression Model

Predictor	λ Coefficients	P-value
Constant	-1.30565	0.000
C2	0.0292445	0.000
C4	0.782094	0.002
C7	0.0020670	0.002
C9	-2.09831	0.029
C13	0.228333	0.000
C20	5.06195	0.000

After the initialization, at each step, a new feature which yields the greatest statistical significance and increases the goodness of the model will be added to the old model to construct a new model and then all the features in the current model are checked to see whether some feature can be dropped without impacting the goodness of the model. This procedure will be continued until the algorithm converges (no new features would be added to the model and no features can be dropped as well) [13]. The *stepwise-feature model* is shown in Table 3. We observe that this model only contains 6 features which is smaller than the number of features in the *significant-feature model*. Among the 6 features, 5 of them are also in the *significant-feature model* and the feature C7 is newly added.

4.3 Meaning of Significant Features

The features appears in at least two regression models from the above three models are listed as below.

- 1) C2: Average Number of Comments
- 2) C4: Average Number of Special Symbols
- 3) C5: Average Number of Font Changes
- 4) C7: Average Number of Words
- 5) C9: Average Lexical Density
- 6) C10: Average Readability
- 7) C13: Average Number of Hyperlinks
- 8) C14: Average Number of Adjectives
- 9) C16: Average Number of Nouns
- 10) C20: Document Content Similarity

From the above list we can know that firstly, document content similarity is an important feature in determining the similarity between bloggers. Because similar document content suggests similar topics of interests. This is also illustrated by many existing work. Secondly, many other

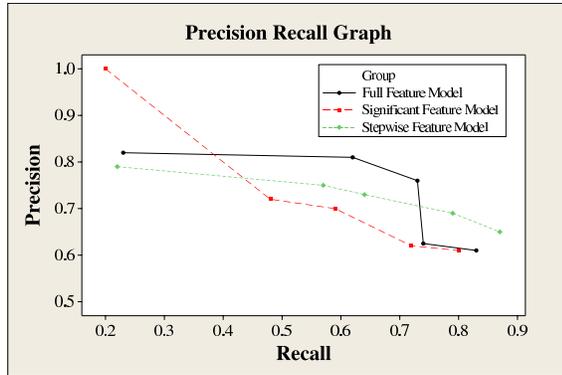


Figure 1: Precision-Recall Graph of Three Logistic Regression Models

personality related features also play important role in predicting potential friendship. We can see that the features related with blogger's writing styles (C7, C9, C10, C14 and C16), writing habit (C4, C5 and C13) and blogging style (C2) have statistical significance in these models.

4.4 Model Comparison

In this section, we further compare the three models using the testing data. We also compare the *full-feature model* with a model which only includes the feature C20, document content similarity. By this comparison, we want to show that adding personality related features can significantly improve the prediction precision and recall.

Different from most of the existing friendship mining systems which use case study as their evaluation, the metric we used in this experiment is precision and recall. To the best of our knowledge, we are the first to use precision and recall, the standard IR performance metrics, to evaluate a friendship mining system. Precision here means that among all the testing data (blogger pairs) which are predicted as potential friends, what percentage of these pairs are indeed labeled as potential friends. Recall means that what percentage of true friends are predicted by the model. In the problem of potential friend detection, recall is more important than precision. Because even if we mistakenly detect two persons who are not likely to be friends as friends, it won't do much harm. What matters is whether we can discover all possible pairs of friends. As a result, in this experiment, we favor the model which can give a high precision without losing the recall.

The precision-recall graph for the three models in Section 4.2 is shown in Figure 1. From Figure 1, we observe that first, with low recall, the *significant-feature model* gives the highest precision. With the increase of recall, the performance of the *significant-feature model* degrades. Second, the *full-feature model* has high precision (above 80%) consistently when the recall is lower than 75%. When

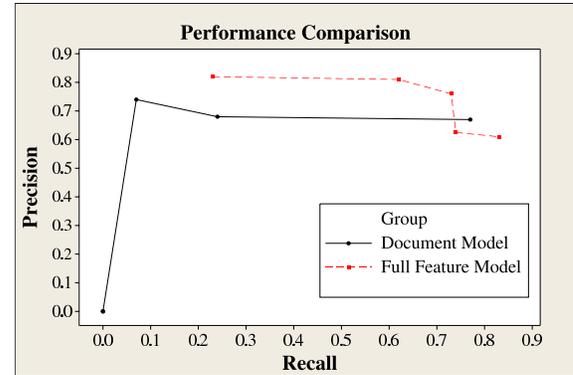


Figure 2: Precision-Recall Comparison of Full Feature Model and Model with Only Document Similarity

the recall is greater than 75%, the performance of the *full-feature model* degraded a little bit, but still has relatively high precision (above 65%). Thirdly, for the *stepwise-feature model*, when the recall is below 75%, the precision is consistently lower than the *full-feature model*. When the recall is higher than 75%, although the precision is the highest among the three, the advantage is not significant.

From the above analysis, we can conclude that overall the *full-feature model* gives the best precision with relatively high recall which is desired by the problem setting.

We also compare the precision and recall of the full-feature model (the approach with personality related features included) with a model with the document content similarity feature only (the approach which only considers blog article content). Since our approach incorporate supplemental information related with blogger's personality or writing style, we are expecting that our approach could outperform the document content only approach. The result is shown in Figure 2. We observe that when the recall is below 75%, the precision of the full-feature model is consistently higher than the model using document similarity only. When the recall is higher than 75%, the two models have comparable precisions (around 60%). From this figure, we can conclude that the use of personality related features significantly improves the precision and recall of the *full-feature model* and makes the *full-feature model* outperforms the model in which only document content information is considered.

In summary, among the three regression models, the full-feature model achieves the best performance, with a precision about 80% when the recall is up to 75%. The models with personality related features achieves higher precision and recall than the model with only document content information considered. This can show that our techniques can improve the performance of detection of potential friends over existing systems.

5. Related Work

We now compare our work with existing work on a number of topics related to blog information mining.

Opinion Mining: There has been great research efforts on opinion mining from blogs [3], [4], [5], [6], [7], [8]. In these system, only the pure text content of the blog articles are used. Two tasks are usually performed in these systems. The first one is to compute document similarity based on vector space model. The second task is to determine semantic orientation of sentences or passages based on supervised learning approach or ontology information, such as WordNet. The underlying assumption of the existing work is that all the blog articles should be related with a specific topic or an object. But in our problem, we don't have this constraint. Our focus is the personality of the bloggers which can be reflected from the their writing styles or patterns. It is highly likely that two articles about totally different topics are written by the same blogger.

Friendship Mining and User Profile Clustering: Much research has been conducted with an aim to discover the relationships among people. Li *et al* [1] propose a system which clusters blog articles by utilizing author/reader comments. Shen *et al* [2] propose a latent friend mining system based on the content and topics of blogger's articles. Their approach only considers document content information and assumes that friends must share the same topic of interests. In our method, besides the document content information, we also considers many other features which are related with the blogger's personality. Xie *et al* [14] propose a web user clustering system using the access log of users. The web page access patterns of web users are mined and users with similar patterns are considered to be in the same cluster. This cluster method is also based on similar interests which is different from our personality based method. Mishne *et al* [15] conduct several experiments on mood classification in blog posts. They select features from the blog posts and a mood classifier is trained. Most of the features they selected are related with the document content information, but with some features, such as special symbols, related with blogger's personality. In our work, we explore many new personality related features.

6. Conclusion

In this paper, we have presented the design, implementation, and evaluation of a novel approach for *latent friendship mining*. This approach is based on mining bloggers's personalities. Unlike all existing latent friendship mining systems, which assume that friends must share the same topics of interest, our approach is built on a what we consider a more realistic assumption, which is that friends are likely to share similar personalities. In our implementation, 20 carefully selected features are mined to capture a blogger's personality and topics of interest. Trained logistic regression models

are used to predict the probability of a pair of bloggers being friends, based on a *personality similarity vector*. Our experiments show that the full-feature regression model achieves the best performance with a high precision (80%) at a relatively high recall (75%). Furthermore, our method outperforms the method that only uses the document content information. Thus, we show that incorporating personality related features improves the overall performance of a latent friendship mining system.

Acknowledgements

This work was supported by NSF grants 0541058, 0619041, and 0833101. We also thank Dr. Hui Fang for her valuable suggestions on this work.

References

- [1] B. Li, S. Xu, and J. Zhang, "Enhancing clustering blog documents by utilizing author/reader comments," in *Proceedings of the 45th annual southeast regional conference*, 2007, pp. 94–99.
- [2] D. Shen, J.-T. Sun, Q. Yang, and Z. Chen, "Latent friends mining from blog data," in *Proceedings of the Sixth International Conference on Data Mining*, 2006, pp. 552–561.
- [3] Y. Chen, F. S. Tsai, and K. L. Chan, "Blog search and mining in the business domain," in *Proceedings of the 2007 international workshop on Domain driven data mining*, 2007, pp. 55–60.
- [4] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide Web*, 2003, pp. 519–528.
- [5] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," pp. 231–240, 2008.
- [6] M. Glance, M. Hurst, K. Nigam, N. siegler, R. Stockton, and T. Tomokiyoi, "Deriving marketing intelligence from online discussion," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2005, pp. 419–428.
- [7] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 168–177.
- [8] S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima, "Mining product reputations on the web," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 341–349.
- [9] S. D. Gladis, *Write Type: Personality Types and Write Types*. HRD Press, 1994.
- [10] J. Ure, "Lexical density and register differentiation," pp. 443–452, 1971.
- [11] R. Gunning, *The technique of clear writing*. McGraw-Hill, 1952.
- [12] H. Schmid, "Probabilistic part-of-speech tagging using decision trees," in *Proceedings of International Conference on New Methods in Language Processing*, 1994, pp. 25–36.
- [13] M. H. Kutner, *Applied linear regression models*. McGraw-Hill/Irwin, 2004.
- [14] Y. Xie and V. V. Phoha, "Web user clustering from access log using belief function," in *Proceedings of the 1st international conference on Knowledge capture*, 2001, pp. 202–208.
- [15] G. Mishne, "Experiments with mood classification in blog posts," in *Style2005 - 1st Workshop on Stylistic Analysis of Text for Information Access*, at SIGIR 2005, 2005.

Improved k -NN Algorithm for Text Classification

Muhammed Miah

Department of Computer Science and Engineering
University of Texas at Arlington, TX, USA

Abstract - Over the last twenty years, text classification has become one of the key techniques for organizing electronic information such as text and web documents. The k -Nearest Neighbor (k -NN) algorithm is a very well known and popular algorithm for text classification. The k -NN algorithm determines the classification of new document by the class of its k -nearest neighbor. In this paper we propose an improved k -NN algorithm with a built-in technique to skip a document from training corpus without looking inside the document if it is not important, which improves the performance of the algorithm. It also has an improved decision rule to identify class from k -nearest neighbor to improve the accuracy by avoiding bias of dominating class with large number of documents. We conduct experiments on benchmark text classification datasets. The new and improved k -NN algorithm is suitable for other applications as well.

Keywords: data mining, k -nearest neighbor, text classification or categorization, text mining.

1 Introduction

During the last twenty years the number of text documents in digital form has grown enormously in size. As a consequence, it is of great practical importance to be able to automatically organize and classify documents. Research into text classification aim to partition unstructured sets of documents into groups that describe the contents of the documents. There are two main variants of text classification: text clustering and text categorization. The former is concerned with finding a latent group structure in the set of documents, while the latter (also known as text classification) can be seen as the task of identifying the appropriate class for a new document according to a group structure that is known in advance. In this paper we work on text categorization or classification where we try to improve the k -NN algorithm by improving the factors that affect the algorithm both in terms of performance and accuracy.

Text classification is the process of grouping texts into one or more predefined classes based on their content. The goal of text classification is the automatic assignment of documents to a fixed number of semantic classes. Each document can be in multiple, exactly one, or no class at all.

Document classification appears in many applications, including e-mail filtering, mail routing, spam filtering, news monitoring, selective dissemination of information to information consumers, automated indexing of scientific articles, automated population of hierarchical catalogues of

Web resources, identification of document genre, survey coding and so on. Automated text classification is attractive because manually organizing text document bases can be too expensive, or unfeasible given the time constraints of the application or the number of documents.

A number of statistical classification and machine learning techniques have been applied to text classification, including regression models, Bayesian classifiers, decision trees, nearest neighbor classifiers, neural networks, and support vector machines. As mentioned earlier, in this paper we focus on improving the k -NN algorithm which is a special case of nearest neighbor classifier. The k -NN algorithm has shown to be very effective in text classification as well as in other domains.

k -NN algorithm classifies a test document based on its k -nearest neighbor. The training examples can be considered as vectors in a multidimensional feature space. The space is partitioned into regions by locations and labels of the training samples. A point in the space is assigned to a class in which most of the training points belong to that class within the k nearest training samples. Usually Euclidean distance or Cosine similarity is used. During the classification phase, the test sample (whose class needs to be identified) is also represented as a vector in the feature space. Distances or similarities from the test vector to all training vectors are computed and k nearest training samples is selected. There are a number of ways to classify the test vector to a specific class. The classical k -NN algorithm determines the class with the majority voters from its k -nearest neighbors. Fig 1 provides an example [32] of how k -NN algorithm identifies a new sample to a class using multidimensional feature space from its k -nearest neighbor training samples. The test sample (circle in the middle) should be classified either to the first class of squares or to the second class of triangles. If $k = 3$ it is classified to the second class because there are 2 triangles and only 1 square inside the inner circle. If $k = 5$ it is classified to first class (3 squares vs. 2 triangles inside the outer circle).

k -NN is a case-based learning method, which examines all the training data for classification. The algorithm is not very well suited in some applications such as dynamic web mining for a large repository as it is a lazy learning method. This is because it has to examine the whole training data for classification as well as look at every keyword in a specific document. k -NN is a simple but effective method for classification. This motivates us to improve the algorithm in terms of efficiency and accuracy avoiding any bias from dominating classes.

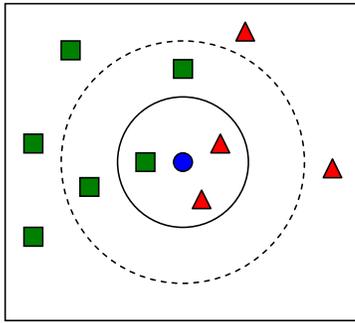


Fig. 1. Example of k -NN algorithm

There are three major factors that affect k -NN algorithm which are: 1) the similarity or distance measure used to find the k -nearest neighbor, 2) k , which is the number of nearest neighbor, and 3) the decision rule to identify a class for the test document from k -nearest neighbor. In this paper we try to improve the performance of the algorithm while improving the accuracy as well. We use the Cosine similarity measure which is a widely used and popular similarity measure in text classification. Like the classical k -NN algorithm, we do not consider all documents in the training corpus as well as try to avoid examining all keywords in a document to detect k -nearest neighbor fast. In fact, we propose a solution that may not need to go inside of each training document and also does not require examining all the keywords in a document, which makes the algorithm efficient. There is another problem in k -NN algorithm when a dataset is dominated by one or few classes that means when one or few classes contain most of the documents in the corpus and others have very few documents. In this situation, simply applying classical decision rule of k -NN algorithm which is selecting the class with most number of documents in k -nearest neighbor may mislead the result. To avoid this problem we use an improved decision rule to identify the right class from k -nearest neighbor for accurate classification.

In this paper we focus and conduct experiments for the improved k -NN algorithm in text classification domain, but this can be used in other applications as well such as e-mail filtering, mail routing, spam filtering, news monitoring, selective dissemination of information to information consumers, automated indexing of scientific articles, automated population of hierarchical catalogues of Web resources, identification of document genre, authorship attribution, survey coding and so on.

Main Contributions The main contribution in this paper may be summarized as follows:

1. We propose and develop an improved k -NN algorithm which is faster than classical k -NN algorithm while improving the accuracy.
2. We use a k -nearest neighbor buffer technique for which the algorithm can check very quickly whether it needs to go inside a training document to look at the keywords and skip the documents which are not important.
3. We impose improved decision rule to identify a class from k -nearest neighbor by giving more weight to the classes with higher number of training documents within

the k -nearest neighbor space and at the same time penalize the documents that are far from the test document to avoid the biasness by the class with large number of documents in training corpus.

4. We conduct experiments on three benchmark text classification datasets with different k values and evaluate the results.

The rest of the report is organized as follows. In section 2 we discuss some of the related works. Section 3 and 4 describe the algorithms and datasets used for the experiments respectively. In section 5 we discuss the data cleaning and preprocessing steps. Section 6 presents the results of the experiments. Section 7 is a short conclusion and then we provide the references in Section 8.

2 Related Work

Researchers have proposed a variety of algorithms for text classification purposes [1, 11, 13, 18, 21, 25, 26, 29, 30]. These are mainly on new algorithms and experiments were done how they perform compared to other algorithms. Some of them also compare existing algorithms [13, 28, 29].

As we mentioned before, there are many classification techniques have been proposed such as Naïve Bayesian (NB) classifiers [14], Decision Tree Classifiers [4], Decision Rules [16], Regression methods [29], Neural Network [22], k -NN classifiers [19, 29], Support Vector Machine (SVM) [10, 11], Rocchio classifiers [9, 20], etc. In applications with very large number of documents, efficiency is a vital issue.

Removing stop list (“a”, “the”, and so on) is also common in text classification as it contains meaningless words and can reduce the dimensionality of data. The practice of using a stop list is common and not normally considered for testing in the text mining process [1, 2, 13, 24, 26].

Stemming, a transformation method, converts words to their root form, is also a popular method for text classification. It reduces the number of distinct or unique words in a document by combining all forms of a word into one root form, which improve the efficiency. However, the reduction in unique words can affect the results of the overall process. Many studies report that the use of stemming will help categorization in some situations, but hurts it in other situations [5, 7, 12, 24]. Many researchers use stemming to improve efficiency.

Feature selection is also used in text classification [17, 21, 31] which is mainly use to reduce the dimensionality of data and improve scalability and efficiency.

[8] introduced very nice techniques to find approximate nearest neighbor which are also useful in text classification, but in this paper we focus on exact nearest neighbor, not approximate nearest neighbor.

The most related work to our paper would be text categorization using Weight Adjusted k -Nearest Neighbor (WAK -NN) text classification [6]. WAK -NN uses a weighted Term Frequency (TF) of keywords for similarity measure. But it did not consider avoiding looking inside the training documents which are not important. Also WA -KNN has an

additional computational cost for weight adjusted step of $O(cn^2)$ where c is the number of iterations in the weight adjustment step and n is the number of data points [6]. The decision rule used in *WA-KNN* is different than our paper. We give more preference to the classes with more number of documents to maintain the originality of k -nearest neighbor concept and at the same time penalize the documents which are far from the test document. *WAK-NN* just adds the similarity for each class on k -nearest neighbor and selects the one with the highest value.

3 Algorithms

As we mentioned earlier, in this work we intend to improve the k -Nearest Neighbor (k -NN) algorithm. We propose two improved versions of classical k -NN algorithms and compare with the classical k -NN algorithm. The brief fundamentals of the algorithms are given below:

3.1 Classical k -Nearest Neighbor Algorithm (k -NN)

The k -Nearest Neighbor algorithm (k -NN) is a method for classifying objects based on closest training examples in the feature space. k -NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification.

An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors. k is a positive integer, typically small. If $k = 1$, then the object is simply assigned to the class of its nearest neighbor. In binary (two class) classification problems, it is helpful to choose k to be an odd number as this avoids tied votes.

k -NN classifier is an instance-based learning algorithm that is based on a distance or similarity function for pairs of observations, such as the Euclidean distance or Cosine similarity measure. The cosine similarity is commonly used in Information Retrieval [23] and we used as the basic similarity measure in our algorithms.

The training corpus can be considered as an inverted list kind of narrow table (which is created during preprocessing step) with three columns {keyword, document, TF} where each row represents a keyword belongs to a specific document with corresponding Term Frequency (TF). Term Frequency (TF) is the frequency of the term or keyword in that document which is actually the number of times the keyword present in the document divided by the total number of keywords present in that document. We use TF in our algorithms and normalize within the document so that sum of all TFs in a document is equal to 1. Inverse Document Frequency (IDF) or combination of TF and IDF (TFIDF) also can be used. TFIDF is very popular in Information Retrieval domain. But in text classification, it was shown that TFIDF is not always very effective [3, 27].

The cosine similarity between two documents can be expressed as follows:

$$\text{Sim}(Q, D_i) = \frac{\sum_j w_{Q,j} w_{i,j}}{\sqrt{\sum_j w_{Q,j}^2} \sqrt{\sum_i w_{i,j}^2}}$$

where Q is a query or test document, D is a training document relevant to Q and w are term frequencies of keywords. We refer $\text{Sim}(Q, D_i)$ as v in rest of the paper.

The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct. A good k can be selected by various heuristic techniques, for example, cross-validation. The special case where the class is predicted to be the class of the closest training sample (i.e. when $k = 1$) is called the nearest neighbor algorithm.

3.2 Improved k -Nearest Neighbor Algorithm (k -NN_I)

This is an improved version of k -NN algorithm. We use Cosine similarity measure to find k -nearest neighbor. We find the similarity between test and training documents using term frequencies and considering only the keywords present in both documents. To make the algorithm more efficient, we try to avoid examining the keywords in a document which are not important. That means we do not want to consider each and every keyword in a document, we only consider the important keywords for similarity measure. By important keywords here we mean the keywords which are present in both the test document and training document for which we need to measure the similarity. For example, if a test document Q has keywords $\{w_1, w_2, w_3\}$ and training document D_i has keywords $\{w_1, w_3, w_4\}$, then the cosine similarity value between Q and D_i would be calculated based on keywords w_1 and w_3 only. We also maintain a top- k buffer/array which contains the nearest k neighbors based on higher similarity value at any time to avoid considering the training documents which are not important for the classification of a specific test document. So the algorithm does not need to consider each document as well as each keyword inside the document which improves the performance significantly. The term frequencies of all keywords for each document of training corpus are calculated in preprocessing step. When a new test document comes as input, the algorithm calculates the term frequencies of all of its keywords. It also initializes a buffer/array of size k with empty value.

For the first k training documents the similarity measure goes into the top- k buffer/array which contains the document name or id and the similarity value found for each of the k document. After that for each training document it checks whether the sum of the term frequencies for all keywords in the document is greater than the minimum similarity value in top- k buffer or not. If it is greater then it looks inside the document, otherwise skip that document. The document can be skipped as the term frequency of a keyword is multiplied

with the term frequency of the same keyword in test document, and each term frequency has a value less than 1, so the sum of the multiplication never be greater than a value v , when simply the sum of the term frequencies is not greater than v .

```

D = {D1, D2, ..., Dn} is the set of training documents
Q is the new test document need to classify.
TF is the term frequency of a term in a document which is
preprocessed for training documents.
v is the cosine similarity value between Q and Di; (i = 1, 2, ..., n)

Algorithm: k-NN_I
Step1: Calculate the TFs of all keywords in Q
Step2: Initialize an empty buffer/array of size k, A[]
/* contains document id/name and similarity value with
the highest k cosine similarity, we call it top-k buffer */
Step3: int count = 0 /* Initialize a counter variable */
Step4: For each document Di in D
    If (count <= k)
        Calculate v between Q and Di
        /* considers only keywords present in both */
        Add document name and v into A[]
        count++;
    Else
        If (Sum(TFs) in Di > Min(v) in A[])
            Calculate cosine similarity value v between
            Q and Di
            /* considers only keywords present in both */
            If (new v > Min(v) in A[])
                Remove document with Min(v) from A[];
                Add document Di and new v into A[];
            /* Else Skip the document */
Step5: Return the class with highest # of documents in A[]

Algorithm: k-NN_I_DR
Step1: Perform steps 1-4 of algorithm k-NN_I
Step2: Calculate the Euclidean distance for each document in A[]
Step3: For each class present in A[], Calculate
    Cj = number of documents in A[] of class Cj *
    Sum(Euclidean distances of all documents in A[] of
    Class Cj)
Step4: Return the class with the highest Cj value

```

Fig. 2. Summary of *k-NN_I* and *k-NN_I_DR* algorithms.

For example, let consider a training document D_i , a test document Q , and a top- k ($k = 2$) buffer with similarity values of $A[1.5, 2.7]$. Here the current minimum similarity value in the buffer is 1.5. Assume D_i contains three keywords w_1 , w_2 , and w_3 with term frequencies of 0.2, 0.5, and 0.3 respectively. The summation of the term frequencies in D_i is 1, which is less than the minimum value 1.5 in the top- k buffer. In this situation the algorithm will skip D_i and will not look inside D_i for examining its keywords as the summation 1 is less than the minimum value 1.5 in the buffer.

This is true as we can see in the following example. Assume Q has two keywords w_1 and w_2 with term frequencies of 0.3 and 0.7 respectively. If we calculate the similarity measure of D_i and Q , the result would be $(0.2*0.3 + 0.5*0.7 =$

0.41) which is less than 1.5 and never can be in the top- k buffer. Remember, to calculate the similarity we only consider the common keywords both in D_i and Q . If the summation of all term frequencies in D_i is greater than the minimum value in the buffer, it may or may not go into the buffer. That's why we need to calculate the cosine similarity to check whether D_i goes into the top- k buffer or not. This is the step where the algorithm might avoid looking into the keywords of all documents that improve the performance. Fig 2 summarizes the steps in the algorithm to identify the class for a query or test document.

Maintaining the top- k buffer/array with highest k similarity values, and avoiding looking inside of each and every training document, the algorithm improves its performance. The algorithm follows the traditional k -NN decision rule to identify class for a test document by the majority votes from its k -nearest neighbor.

3.3 Improved k-Nearest Neighbor Algorithm with Improved Decision Rule (*k-NN_I_DR*)

Class with large number of documents in a corpus can bias the classification. Class with highest number of documents in k nearest neighbor may mislead the result if simply the test document is classified based on tradition k -NN decision rule (i.e., test document belongs to the class with highest number of documents in k -nearest neighbor). This is because it may happen that most of the time too many documents from the same large class will appear in the k -nearest neighbor list even they are far from the query point or test document while other documents from another class is very close but number is less.

So we optimize *k-NN_I* to avoid this biasness and improve the accuracy. We give more weight (preference) to the class with higher number of documents in top- k buffer (k -nearest neighbor) and penalize the documents which are far from the test document. Combination of the above two can avoid the biasness of large class. After we get the final top- k buffer, we calculate the actual Euclidean distance from the test document for each document in top- k buffer. Divide the summation of all Euclidean distances in top- k -buffer with the summation of all distances in a class within the buffer and multiply the new value with the number of documents in that class in the buffer. Then return the class with the highest value. This eliminates the biasness or domination of a large class. Fig 2 summarizes the steps in the algorithm to identify class for a query or test document.

4 Datasets

In this paper we use three widely used benchmark datasets for text classification, which are: 20_Newsgrpup, 4_Universities and Reuters-21578 datasets. We intend to work on different types of dataset, so we choose one dataset (20_Newsgrroup) which contains text documents, another dataset (4_Universities) which contains web pages, and the

last one (Reuters-21578) delimited by SGML tags. The descriptions of the datasets are given below:

4.1 20_Newsgrupup dataset

It is a well known data set for text classification. The dataset is a collection of 20,000 messages, collected from UseNet postings over a period of several months in 1993. Data are divided almost evenly among 20 different UseNet discussion groups or classes which are:

alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.windows.x, misc.forsale, comp.sys.mac.hardware, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.misc, talk.religion.misc.

20_Newsgrupos dataset is taken from

<http://people.csail.mit.edu/jrennie/20Newsgrupos/> where data is sorted by date into training (60%) and test (40%) sets, does not include cross-posts (duplicates) and does not include newsgroup-identifying headers (Xref, Newsgrupos, Path, Followup-To, and Date).

4.2 4_Universities dataset

The dataset contains WWW-pages collected from computer science departments of various universities in January 1997. The dataset was collected by the World Wide Knowledge Base (Web->Kb) project of the CMU text learning group. The data were collected mainly from major 4 universities: Cornell, Texas, Washington, and Wisconsin, and that's why it is called 4_Universities dataset. The data collected from other universities are grouped as miscellaneous.

We take the data from the source

<http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/webkb-data.gtar.gz>. The data are classified into seven different classes which are: Student, Faculty, Staff, Department, Course, Project, and Other.

The total number of web pages in corpus is 8278. Originally it was 8282, but 4 web pages were removed because of unrecognized symbols and characters in the pages. The number of web pages for each of the seven classes is: student(1640), faculty(1124), staff(136), department(182), course(930), project(504), other(3762).

The class other is a collection of pages that were not deemed the "main page" representing an instance of the previous six classes. (For example, a particular faculty member may be represented by home page, a publications list, a vitae and several research interests pages. Only the faculty member's home page was placed in the faculty class. The publications list, vitae and research interests pages were all placed in the other category.)

The data set contains pages from the universities are: Cornell (866), Texas (825), Washington (1205), and Wisconsin (1263). 4119 miscellaneous pages collected from other universities.

4.3 Reuters-21578 dataset

The dataset contains 21578 Reuters news documents from 1987. They were labeled manually by Reuters personnel. Labels belong to 5 different category classes, such as 'people', 'places' and 'topics'. The total number of categories is 672, but many of them occur only very rarely. Some documents belong to many different categories, others to only one, and some have no category. Over the past decade, there have been many efforts to clean the database up, and improve it for use in scientific research. We take the data from the source

<http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>. In order to divide the corpus into training and test sets, we use the modified Apte (ModApte) split [15]. With this split the training and test sets contain 9603 and 3299 documents respectively.

5 Data Cleaning and Preprocessing

After collecting the data from sources, we did some cleaning and preprocessing on all the datasets.

5.1 Data Cleaning

We clean the datasets and convert them into bags of words (text documents). After cleaning the datasets, they look like bag of words or terms or keywords. The brief overview of the cleaning process on the datasets is given below:

20_Newsgrupup dataset The datasets originally are text documents. We remove all the existing headers such as "From", "Subject", "Organization", and "Lines". The statistics of the dataset after cleaning are as follows: total number of documents is 18846, number of training documents is 11314 and number of test documents is 7532.

4_Universities dataset The dataset are originally web pages in html format, and after cleaning we converted it into text document. As like other dataset, here we also remove all the existing headers such as "MIME-Version", "Server", "Date", "Content-Type", "Content-Length", and "Last-Modified", and so on. The source of dataset suggested taking data from any one university to make as test data and not the part of data from all universities. So, we select data from Wisconsin as test data and data from other universities as training data. The statistics of the dataset after cleaning are as follows: total number of documents is 8278, number of training documents is 7015, and number of test documents is 1263.

Reuters-21578 dataset The data format is divided in 22 files of about 1000 documents delimited by SGML tags. We remove all tags, headers, etc. and convert them into text files of simple bag of words as like other datasets described above. The statistics of the dataset after cleaning: total number of documents is 12362, number of Training documents is 9063, and number of test documents is 3299.

For all the datasets, we remove the stop words but no stemming has been done. We also remove all the special

characters which are meaningless on text classification such as “>”, “<”, “,””, “.””, and so on.

5.2 Data Preprocessing

We do some preprocessing for the training corpus of all the datasets in advance and stored the data in database tables. We create inverted list like tables to calculate some variables in preprocessing step which are used in the algorithms such as: total number of documents in corpus, total number of documents per class, total number of distinct terms in corpus, total number of terms per class, term frequencies of each keyword for each class, term frequencies of each keyword for each document, and so on.

6 Experiments

In this section we measure (a) the accuracy (percent of test documents correctly classified), and (b) the performance (total running time to classify all the test documents in datasets) and evaluate the results.

System Configuration: We used a PC with Intel Xeon 2.00-GHZ, 3.25GB of RAM and 465GB HDD for our experiments. We implemented all algorithms in C#, and used Microsoft SQL Server 2005 RDBMS to store and access preprocessed data. We connected to the RDBMS through ADO. Text documents were used in normal .txt format.

Figs 3 and 4 respectively display the performance and accuracy of the algorithms on 4_Universities dataset. Similarly, Figs 5 and 6 display the performance and accuracy of the algorithms on 20_Newsgroup dataset, Figs 7 and 8

display the performance and accuracy of the algorithms on Reuters-21578 dataset.

As we can see in Fig 4 that accuracy does not have much effect on k -value for different algorithms on 4_Universities dataset. There is a very little improvement on the accuracy of the algorithm k -NN_I_DR when k value increases from 50 to 100, but this is not a significant change. Also there is no effect on accuracy of using improved decision rule particularly on this dataset. This is because the dataset is not bias by any single large class or few classes. But in Figs 6 and 8 we can see that the accuracy increases with increasing k value for all the algorithms. Also algorithm k -NN_I_DR provides more accuracy than other two algorithms. This is because there is some biasness of a single or few classes in the dataset even originally the documents were almost evenly distributed among all classes. Also increasing k -value increases the accuracy.

In terms of performance, we can see in Fig 3, 5 and 7 that algorithm k -NN_I outperforms algorithm k -NN for all the datasets as it can avoid looking inside some of the training documents. As we can see in Fig 5 and 7, for datasets with larger documents the algorithm k -NN_I improve the performance significantly. Algorithm k -NN_I_DR takes little more time to execute than algorithm k -NN_I because it does not just take the majority votes from its k -nearest neighbor to identify the class for the test document. It has to calculate the actual Euclidean distance for the k -nearest neighbor documents to impose the improved decision rule which can avoid any bias from large dominating class. With little more execution time the algorithm k -NN_I_DR can provide better accuracy in this kind of situation than algorithm k -NN_I.

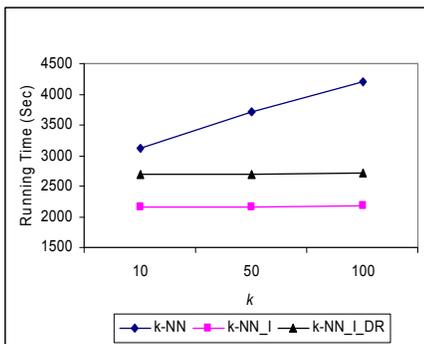


Fig. 3. Performance comparison on 4_Universities dataset with varying k value.

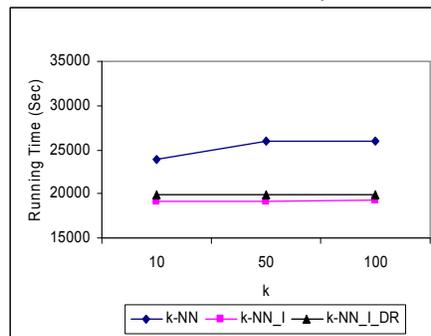


Fig. 5. Performance comparison on 20_Newsgroup dataset with varying k value.

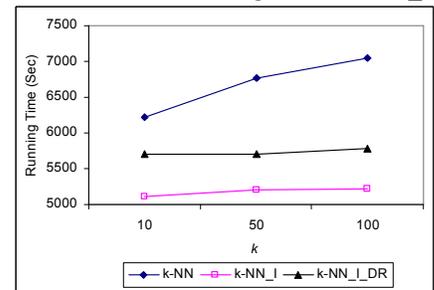


Fig. 7. Performance comparison on Reuters-21578 dataset with varying k value.

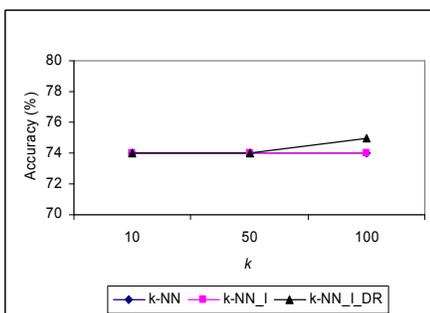


Fig. 4. Accuracy comparison on 4_Universities dataset with varying k value.

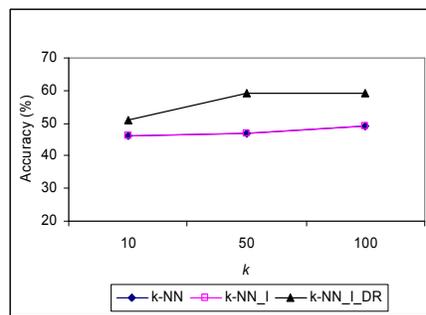


Fig. 6. Accuracy comparison on 20_Newsgroup dataset with varying k value.

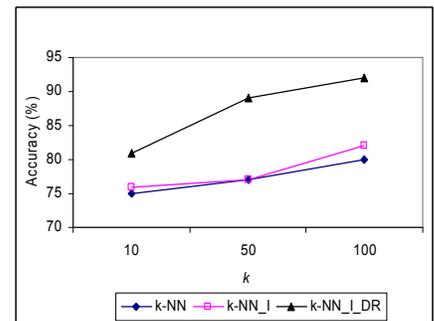


Fig. 8. Accuracy comparison on Reuters-21578 dataset with varying k value

Based on the discussion and experiments, it is clear that algorithm k -NN_I is faster than k -NN and k -NN_I_DR, and algorithm k -NN_I_DR is more accurate than the other two. So we can conclude that algorithm k -NN_I would be a better choice when the dataset is not bias by any single or few large classes that means there is no effect on the number of documents in a class that can affect the result of k -nearest neighbor. But when the dataset is bias by any large class, then definitely algorithm k -NN_I_DR would be the better choice. With little compromise on performance it can avoid such biasness and produce more accurate results. The accuracy measures might not seem very high from the experiments, this is because we do not use any indexing or stemming to improve the accuracy. We just want to compare the improved techniques with the classical k -NN technique. Use of indexing and stemming would improve the relative performance for the proposed techniques as well as for the classical k -NN technique.

7 Conclusion

In this paper our goal was to improve the classical k -NN algorithm in terms of performance and accuracy by considering couple of major factors that affect the algorithm. We imposed a top- k buffer technique that can skip looking inside each and every training document and also each and every keyword in a document, which improves the performance of the algorithm. We also proposed an improved decision rule to identify a class from k -nearest neighbor space by maintaining the classical k -NN property (majority votes) with penalizing the training samples which are far from the test sample, which can avoid any biasness from large dominating class in a dataset and improves the accuracy. This seems similar to weight adjusted k -NN approach but our technique is different as we discussed in related work section. We conduct experiments on three benchmark text classification datasets and evaluate the results.

In this work, our goal was not to compare our improved techniques with other state of the art text classification algorithms, but to improve the classical k -NN algorithm overcoming the factors that weaken the algorithm. In future we look forward to compare our proposed techniques with other text classification algorithm such as NB, SVM, Decision Tree, WA-KNN, etc. and improve the algorithms if needed such that they can outperform the existing algorithms both in terms of performance and accuracy.

8 References

- [1] Apte, C., Damerau, F., and Weiss, S., "Automated Learning of Decision Rules for Text Categorization." ACM Transactions of Information Systems 12.3 (1994): 233-251.
- [2] Blake, C., and Wanda Pratt. "Better Rules, Fewer Features: A Semantic Approach to Selecting Features from Text." ICDM 2001: 59-66.
- [3] Boley, D., Gini, M., Gross, R., Han, E. H., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., and More, J., "Partitioning Based Clustering for Web Document Categorization", Decision Support Systems (1999).
- [4] Cohen, W. W and Singer, Y., "Context-Sensitive Learning Methods for Text Categorization", ACM Trans. Inform. Syst. (1999): 141-173.
- [5] Dave, K., Steve Lawrence, and David Pennock. "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews." Proceedings of World Wide Web (2003): 519-528.
- [6] Han, H., Karypis, G., Kumar, V., "Text Categorization Using Weight Adjusted k -Nearest Neighbor Classification", PAKDD 2001: 53-65
- [7] Harman, D. "How Effective Is Suffixing?" Journal of the American Society for Information Science 42.1 (1991): 7-15.
- [8] Piotr Indyk, Rajeev Motwani: Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. STOC 1998: 604-613
- [9] Joachims, T., "A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization", Proceedings of ICML-97, 143-151.
- [10] Joachims, T., "A Statistical Learning Model for Text Classification for Support Vector Machines", 24th ACM International Conference on Research and Development in Information Retrieval (SIGIR) 2001.
- [11] Joachims, T., "Text Categorization with Support Vector Machines: Learning with Many Relevant Features", Proceedings of 10th European Conference on Machine Learning, Chemnitz, Germany (1998): 137-142
- [12] Kao, A., and Steve Poteet. "Report on KDD Conference 2004 Panel Discussion Can Natural Language Processing Help Text Mining?" ACM SIGKDD Explorations Newsletter 6.2 (2004): 132-133.
- [13] Lewis, D., Ringuette, M., "A Comparison of Two Learning Algorithms for Text Categorization." In Proc. of the Third Annual Symposium on Document Analysis and Information Retrieval (1994): 81-93
- [14] Lewis, D.D, "Naïve (Bayes) at forty: The independent assumption in information retrieval". Proceedings of ECML-98, 10th European Conference on Machine Learning (1998): 4-15.
- [15] Lewis, D. D.: Reuters-21578 Document Corpus V1.0. <http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>
- [16] Li, H., Yamanishi, K., "Text Classification Using ESC-based Stochastic Decision Lists", CIKM-99, 8th ACM International Conference on Information and Knowledge Management (1999): 122-130.
- [17] Li, Z., Sun, M., "Scalable Term Selection for Text Categorization", Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 774-782, Prague, June 2007.
- [18] McCallum, A., and Kamal Nigam. "A Comparison of Event Models for Naive Bayes Text Classification." Proceeding of AAAI/ICML-98 Workshop on Learning for Text Categorization (1998): 41-48.
- [19] Mitchell, T. M., "Machine Learning", McGraw Hill, NY (1996).
- [20] Rocchio, Jr. J. J., "Relevance Feedback in Information Retrieval. The SMART Retrieval System: Experiments in Automatic Document Processing", Editor: Gerard Salton, Prentice-Hall, Inc, Englewood Cliffs, New Jersey (1971).
- [21] Rogati, M., Yang, Y., "High-Performing Feature Selection for Text Classification", CIKM 2002, 659-661.
- [22] Ruiz, M. E. and Srinivashan, P., "Hierarchical Neural Networks for Text Categorization", SIGIR-99 (1999), 281-282.
- [23] G. Salton, "Automatic Text Processing", The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, 1989.
- [24] Waegel, D., and April Kontostathis. "TextMOLE: Text Mining Operations Library and Environment." Proceedings of the 37th SIGCSE technical symposium on Computer science education (2006): 553-557.
- [25] Weiss, S., Chidanand Apte, Fred Damerau, David Johnson, Frank Oles, Thilo Goetz, and Thomas Hampp. "Maximizing Text-Mining Performance." IEEE Intelligent Systems 14.4 (1999): 63-69.
- [26] Yang, Y. "An Evaluation of Statistical Approaches to MEDLINE Indexing." Proceedings of AMIA-96, Fall Symposium of the American Medical Informatics Association (1996): 358-362.
- [27] Yang, Y., and Chute, C. G., "An Example-Based mapping Method for Text Categorization and Retrieval", SIGIR-94 (1994).
- [28] Yang, Y., and Jan Pedersen. "A Comparative Study on Feature Selection in Text Categorization." ICML 1997: 412-420.
- [29] Yang, Y., Liu, X. "A re-examination of text categorization methods." Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1999): 42-49.
- [30] Zaiane, O., and Maria-Luiza Antonie. "Classifying Text Documents by Associating Terms with Text Categories." Proceedings of the thirteenth Australasian conference on Database technologies 5 (2002): 215-222.
- [31] Zheng, Z., Wu, X., Srihari, S., "Feature Selection for Text Categorization on Imbalanced Data", SIGKDD Explorations, 2004.
- [32] http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm.

SVM fuzzy hierarchical document categorization and retrieval method

T. Guernine¹, K. Zeroual²

Département d'Informatique, Université de Sherbrooke, Sherbrooke, Québec, Canada

Abstract—*Document categorization is useful for information retrieval. The focus of this paper is to investigate an automatic fuzzy hierarchical categorization structure and to propose a new retrieval model based on fuzzy logic in order to find documents that satisfy user's query. The fuzzy hierarchical method proposed is based on Support Vector Machine. Due to the explosive growth of the number of documents available on the web, we introduce an SVM at each level to separate large number of classes. Our method consists in building dynamically a fuzzy hierarchical structure from the training data. The fuzzy hierarchical structure extracts the fuzzy relationships between distinct classes. Our experimental results improve high accuracy in the 20 Newsgroups collection compared with standard document categorization techniques.*

Keywords: SVM, Document categorization, Hierarchical classification, Fuzzy classification.

1. Introduction

Solving multi-class text problem with high accuracy is a challenging problem because there is an important increasing processing of textual data in databases. Furthermore the manual document categorization is not able to keep up with the growth of textual data. Even if there are many automatic methods proposed to treat document categorization problem, few handle the huge volume of text to be categorized. An automatic text categorization consists in gathering the most similar documents in the same class based on its textual content. Until now, categorization of documents remains among the primary worry in the field of classification. In order to address document categorization problem, many machine learning methods and statistical techniques has been proposed : Decision trees (Weis et al.,1999), Nearest neighbor classifiers (Xie & Beni, 1991), Bayesian models (Lewis & Ringuette, 1994; Koller & Sahami, 1997), regression models (Schutze, Hull & Pedersen, 1995) and Support Vector Machine (Joachims, 1998; Yang & Liu, 1999; Sebastiani, 2002). SVM becomes in the last years a very important operational tool for text categorization [4], [38]. Recent works proved that Support Vector Machine has been successfully applied to this task [30], [31].

In this work, we adapt our methods [10], [39] to address document categorization problem and propose a new method to classify and find similar documents in a database.

Due to the large number of classes in text, we use hierarchical structure to reduce the number of classes at each level and decompose the originally problem into simple binary problems to ease discrimination between classes. The binary problems can then be solved easier with high accuracy.

The basic goal behind application of fuzzy hierarchical classification is to extract automatically the fuzzy relationships between documents at each node. In our work, we use also equivalence classes to gather similar documents into single class. We get a direct hierarchy of classes that improved search and classification of new documents.

The remainder of the paper is organized as follows. In section 2, we provide an overview of related works. In section 3, we give a brief review of SVM. In section 4, we describe notion of the fuzzy hierarchical classification. In section 5, we describe document categorization problem. In section 6, we describe the process of our document categorization method as well as document search method. In section 7, we present our experimental results. Our future research works are presented in section 8.

2. Related Works

The most important issue in multi-class problem is the existence of confusion classes [43]. The hierarchical structure is one of the techniques used to solve the confusion classes. This work addresses document categorization and retrieval problem. The document categorization problem based on SVM is mainly related to hierarchical multi-class pattern recognition problems. Most of recent works used hierarchical structure to address document categorization problem [11], [41], [40], [38], [44]. In this section we are interested in to hierarchical structure and use of SVM techniques that allow us to construct a hierarchy.

Joachims [11] identifies the advantage of using SVM in document categorization problem. High performance is obtained by SVM compared with four standard learning methods : Bayes, Rocchio, C4.5 and K-NN.

Joachims [41] propose a TF-IDF classifier in probabilistic way, that is given by :

$$P(d \setminus c_j) = \sum_{w \in (d \cap c_j)} \frac{P(w \setminus c_j) \cdot P(c_j)}{\sum_{c \in C} P(w \setminus c_j) \cdot P(c)} \cdot P(w \setminus d) \quad (1)$$

Where c and c_j represent categories, $P(d \setminus c_j)$ represents the probability that document d belongs to category c_j and

$P(w \setminus d)$ represents the probability that feature w belongs to document d .

Peng et al [40] proposed a new classification algorithm based on a hierarchical structure. The algorithm consists of the following stages : (i) generating category information tree (ii) hierarchical feature propagation (iii) feature selection of category information and (iv) single path traversal. The proposed hierarchical classification system allows adding new categories as required, organizing the web pages into a tree structure, and classifying web pages by searching through only one path of the tree structure.

In [38], a hierarchically SVM classification method is proposed to accomplish document categorization. Hao et al [38], subdivide the original problem into sub-problems to facilitate discrimination between the huge number of categories. They based their work on the results of Support Vector clustering method to obtain a hierarchy structure. The results obtained in [38] show high performance compared with non-hierarchical SVM classifiers.

Wang et al [44], propose a text categorization system to solve multi-class problem. The system proposed contains two modules : (i) Processing module and (ii) Classifying module. In [44], the fuzzy set theory is introduced in the classifying module. They proposed a OOA-FSVM classifier to implement a multi-class classification system. The empirical results obtained by the proposed system show that OOA-FSVM performs better than OOA-SVM.

3. Support Vector Machine

In this section we give a brief review of Support Vector Machine. We present respectively binary and multi-class classification.

3.1 Binary classification

Generally, SVM classifiers are designed to solve binary classification problem [5]. SVM consists in minimizing the empirical classification error and finding optimal hyperplane with large margin. The basic idea is how to find the optimal hyperplane with large margin [6], [7]. Suppose a data set $(x_i, y_i) : (i = 1, \dots, n)$, where x_i corresponds to the attribute set for the i^{th} element. Let $y_i \in \{-1, +1\}$. The optimal hyperplane can be found by minimizing the margin ω in equation 2 :

$$(P) \begin{cases} \text{Min} \frac{1}{2} \|\omega\|^2 \\ y_i(wx_i + b) \geq 1 : i = 1, \dots, n : \forall x \in R^n \end{cases} \quad (2)$$

Where w and b are parameters of the model. It is therefore to find the parameters w and b .

The solution of optimization problem is given by Lagrangian:

$$L_p = \frac{1}{2} \|\omega\|^2 - \sum_{i=1}^n \alpha_i [y_i(wx_i + b) - 1] \quad (3)$$

Where α_i are called the Lagrange multiplier.

We can simplify the problem given by equation 3 as follows :

$$L_D = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i x_j \quad (4)$$

In several cases, linear solution could not solve the problem. In this situation, non linear separator is required :

$$f(x) \begin{cases} wx_i + b \geq (1 - \xi_i) & \text{if } y_i = 1 \\ wx_i + b \leq (1 - \xi_i) & \text{if } y_i = -1 \\ \xi_i > 0, \forall i \end{cases} \quad (5)$$

The objective function will change as follows :

$$f(w) = \frac{1}{2} \|\omega\|^2 + \zeta \sum_{i=1}^n (\xi_i)^\kappa \quad (6)$$

where ζ and κ are specified by the user, representing the penalty of misclassification.

The Lagrangian is written as follows :

$$L_p = \frac{1}{2} \|\omega\|^2 + \zeta \sum_{i=1}^n (\xi_i) - \sum_{i=1}^n \alpha_i [y_i(wx_i + b) - 1 + \xi_i] - \sum_{i=1}^n \mu_i \xi_i \quad (7)$$

We can however, simplify the problem given by 7 as follows :

$$L_D = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i x_j \quad (8)$$

3.2 Multi-class classification

Even if SVM is basically designed to solve binary classification, it is also used to solve multi-class problems. Several methods has been proposed to handle multi-class problems. The following methods are developed to deal with multi-class problems using SVM binary classifiers at each node :

- 1) **One-against-one** : The one-against-one method requires one classifier SVM_{ij} for each pair of classes (i, j) . It builds $n(n-1)/2$ classifiers for n -class classification problem. During the test phase, the test set is evaluated by all SVM_{ij} . The final decision is obtained by vote.
- 2) **One-against-all** : The one-against-all method is simple and efficient. It requires n classifiers SVM_i : $i = 1 \dots n$, for n -class classification problem. During the test phase, the test set is evaluated by the SVM_i : $i = 1 \dots n$. SVM_i which shows highest decision value is chosen.
- 3) **Directed Acyclic Graph SVM** : The DAGSVM method constructs also $n(n-1)/2$ classifiers SVM_{ij} . During the test phases, it creates a list of all candidates classes. At each test, the class that obtained negative score is eliminated from the list.

4. Fuzzy Hierarchical classification

In this section, we describe in detail concepts used to develop our method. We start by explaining the fuzzy relation. We describe Min-Max transitivity and transitive closure of a fuzzy relation. Then, we explain hierarchy and direct hierarchy notions.

4.1 Fuzzy relation

The fuzzy relation is defined by its membership function as follows :

1) Symmetry

$$\forall x, y \in K, \mu_R(x, y) = \mu_R(y, x) \quad (9)$$

2) Anti-Symmetry

$$\forall x, y \in K : x \neq y : \mu_R(x, y) \neq \mu_R(y, x) \quad (10)$$

3) Reflexive

$$\forall x \in K : \mu_R(x, x) = 1 \quad (11)$$

4) Transitivity Min-Max

$$\forall x, y, z \in K : \mu_R(x, z) \leq \text{Min}_y[\text{Max}(\mu_R(x, y), \mu_R(y, z))] \quad (12)$$

- Min-Max Transitivity

The concept of transitivity is used to find shorter links between documents. The link obtained between documents must be the shortest compared with all indirect links. The Min-Max transitivity relation is defined by :

$$\forall x, y, z, \mu_R(x, z) \leq \text{Min}_y[\text{Max}(\mu_R(x, y), \mu_R(y, z))].$$

- Transitive closure of a fuzzy relation R

Transitive closure is a relation τ obtained by the composition of R relations and its membership function. It is defined as follows :

$$\mu_\tau(x, z) \leq \bigwedge_y [(\mu_\tau(x, y) \vee \mu_\tau(y, z))] \quad (13)$$

We compute the transitive closure Min-Max given by the equation 13 until we obtain transitive closure Γ equals to $\Gamma = R^{\kappa-1} = R^\kappa$ at κ levels. This equality assures the existence of a hierarchy [10]. This relation gives the transitive distance Min-Max which locates the level of each document.

4.2 Hierarchy

Let H subset of K . H forms a hierarchy on K if only if the two following conditions hold :

1) Every singleton A belongs to H :

- a) $\text{Card}(A) = 1$ and $A \subset K \Rightarrow A \in H$;
- b) A singleton is a class with a unique element.

2) $\forall A, B \in H, A \cap B \neq \phi \Rightarrow (A \subset B) \vee (B \subset A)$.

4.3 Directed Hierarchy

We obtain a directed hierarchy H on set K if there exist function $\phi : H \rightarrow R^+$ (see figure 1) :

1) For every singleton, $\phi = 0$, i.e :

- $(A \in H \text{ and } \text{Card}(A) = 1) \Rightarrow \phi(A) = 0$

2) Consider that A and B are elements of a hierarchy H . If $B \subset A$ then $\phi(A)$ is greater than $\phi(B)$, i.e :

- $\forall A, B \in H, B \subset A \text{ and } A \neq B \Rightarrow \phi(A) > \phi(B)$

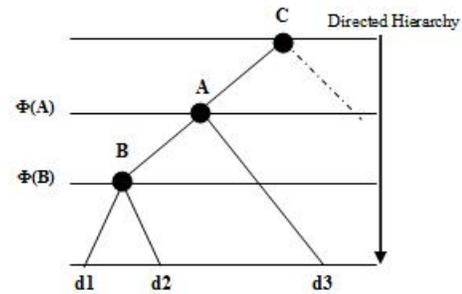


Fig. 1: Concept of directed hierarchy

5. Document categorization problem

The rapid growth of textual data in databases needs an automatic method for structuring documents, simplifying their indexing and finding interesting information in the Net [11]. However, manual methods for processing large data become difficult. Indeed, the manual methods are slow and costly. To accomplish its tasks, document categorization starts by preprocessing data step which is usually used in Information Retrieval [12], [13]. It consists in transforming characters of text into vectoriel representation. In this section we give some concepts related to document categorization problem and which are necessary to develop our method.

5.1 Tf-Idf representation

Tf-Idf (Term frequency-Inverse document frequency) is used usually in Information Retrieval (IR) domain. According to Zipf's law [14] :

- 1) A frequent term in a document is more informative.
- 2) A frequent term in all the corpus is less discriminant :
(Stop-Words : the, a, an).

The Tf-Idf is given as follows :

$$Tf - Idf = Tf * Idf = Tf * \log \frac{|D|}{df_i} \quad (14)$$

Where $|D|$ represents the number of documents and df_i represents the number of documents that contain the term i .

5.2 Terms selection

Terms selection is an important step in text categorization. It consists in selecting terms that appear in documents with threshold of appearance fixed by the user. To accomplish this task, many statistical techniques have been used : mutual information [15], [16], statistical hypothesis test χ^2 [17] and other methods using the frequency of appearance [18].

5.3 Latent Semantic Indexing

Latent Semantic Indexing (LSI) consists in extracting relationships between terms and documents [19]. The output of LSI is an occurrence matrix where rows represent documents and columns represent terms. Each element of this matrix represents the term frequency Tf in a document. Applying LSI consists also in eliminating unusual words from the document that hinders the decision making task [25].

5.4 Similarity measure

The cosine is the similarity metric usually used in document categorization problem. Similarity between two documents d_i and d_j is the cosine of the angle formed by their vectors \vec{d}_i and \vec{d}_j [20]. More the angle decreases more the similarity increases [21]. The cosine measure is expressed as follows :

$$\cos(\theta) = \frac{\vec{d}_i \cdot \vec{d}_j}{\|\vec{d}_i\| \cdot \|\vec{d}_j\|} \quad (15)$$

6. Document classification and retrieval using SVM fuzzy hierarchical method

In this section we first present our SVM fuzzy hierarchical method for text categorization. Second, we present our SVM fuzzy hierarchical method for retrieving documents. The detail of our method is given in [42].

6.1 SVM fuzzy hierarchical text categorization

The proposed method consists of three steps : Data preprocessing step to reduce the large number of features (ii) Fuzzy hierarchical classification and (iii) A node SVM classification.

The first step of our proposed method consists in removing stop words that are unnecessary in discriminating between classes. Removing stop words allows reducing feature space. We applied Tf-Idf representation in order to select words frequency. We selected only terms who appear at least three times. Then, we apply Latent Semantic Indexing on the documents obtained by Tf-Idf representation. In this step we obtained a similarity matrix, where rows represent documents and column represent terms.

In the second step and in order to measure similarity between documents, we build a similarity matrix of documents. Similarity measure is computed using equation 15. Then, we compute the transitive closure from the similarity matrix by equation 13. This technique assures us to find a fuzzy

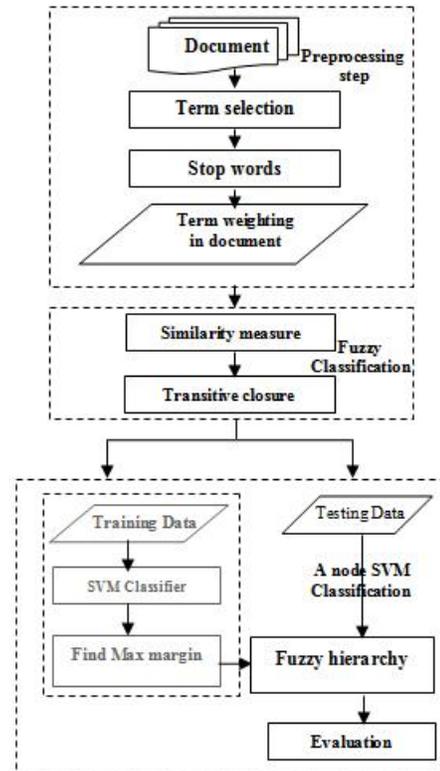


Fig. 2: Overview of our document categorization process.

short distance between the elements.

The third step of our method consists in associating a SVM to each node of the hierarchy. From the similarity matrix obtained in the second step we trained SVM as follows : We consider documents that have the shortest distance as positive and the rest as negative. We iterate on this process at each node from the hierarchy until leaves. Our method has the advantage to use only values from the similarity matrix for the SVM training rather than using values randomly. Similarity matrix assures a priori separate classes in the hierarchy. Let $D = \{d_1, d_2, d_3\}$ be a set of documents. Let Γ_{ij} be the distance between document i and document j obtained from the similarity matrix : $(i, j = 1, 2, 3)$. Suppose we have $\Gamma_{23} < \Gamma_{12} < \Gamma_{13}$, the SVM classifier learns at each node of the hierarchy that $\{d_2, d_3\}$ belongs to positive class : $SVM_{l+} = \{d_2, d_3\}$ and $\{d_1\}$ belongs to negative class : $SVM_{l-} = \{d_1\}$. We iterate on each level until reaching leaves containing only documents.

6.2 SVM fuzzy hierarchical document retrieval method

In this section we present our retrieval method that consists in :

- Having a hierarchy;
- Building the fuzzy subset $F_{\lambda_i}^i$;

- Computing the logical combination of the query;
- Computing the fuzzy entropy.

Let λ_i be a threshold value belonging to interval $[0, 1]$. Document d_j belongs to a class C of the hierarchy would have value μ where $\mu \geq \lambda_i$. Let D be a set of documents and $F = \{F_1, F_2, \dots, F_n\}$ a set of features. Let $d \in D$ where d has at least one feature F_i with $\mu : \mu \in [0, 1]$ value. We can state $\mu_{F_i}(d) = \mu$. Let F_i be a fuzzy subset of F .

	d1	d2		dn
Fi	μ_{i1}	μ_{i2}	...	μ_{in}

Fig. 3: Feature fuzzy representation

From F , we build the subsets $F_{\lambda_i}^i$ given by their membership functions as follows :

$$\mu_{F_{\lambda_i}}(x) \begin{cases} \mu_{F_{\lambda_i}}(x) < \lambda_i \Rightarrow \mu_{F_{\lambda_i}}(x) = 0 \\ \mu_{F_{\lambda_i}}(x) \geq \lambda_i \Rightarrow \mu_{F_{\lambda_i}}(x) = 1 \end{cases} \quad (16)$$

where λ_i are threshold values of features F_i , fixed by the user.

For example, we would like to find a document that satisfies query : $Q = (F1 \wedge F2) \vee F3$. To analyze Q , we transform it as follows : $Q = (F_{\lambda_1}^1 \cup F_{\lambda_2}^2) \cap F_{\lambda_3}^3$. We obtain a subset of documents that satisfy Q . To choose the best document from this subset, we compute the entropy of each document. This entropy expresses the average amount of ambiguity to decide which document is better than the others. The fuzzy entropy is stated as follows :

$$E = \sum_{i=1}^N [(\mu_{d_j}(x_i) \cdot \ln(\mu_{d_j}(x_i))) + (1 - \mu_{d_j}(x_i)) \cdot \ln(1 - \mu_{d_j}(x_i))] \quad (17)$$

From the subset obtained, the document related to the lowest entropy is chosen as the document that satisfies query Q . The advantage of this method is that the number of documents that are considered in the comparison process with the query is reduced.

7. Experimental results

7.1 Data

In our experiments, we compare the performance of our method with SVM that uses polynomial and RBF kernels, naive Bayes classifier, Rochio, C4.5 and K-NN algorithms. The empirical evaluation is done on 20 Newsgroups collection. The 20 Newsgroups collection is a popular collection. It contains 20000 documents, partitioned in 20 classes. In our work, we used **20NG** collection which contains for each class 100 documents. We choose only documents that belong to more than a class in the same time. About 1200 documents are used for training and 800 are used for testing. We give detail of **20NG** in table 1.

Table 1: Problem detail.

Problem	
Data	20 Newsgroups
Collection	20NG
# documents	2000
# Class	20
# Training documents	1200
# Testing documents	800
# Term	17766
# phrase	5874

7.2 Results

The goal of these experiments is to evaluate the performance of our proposed method versus basic SVM, naive Bayes classifier, the Rochio algorithm, C4.5 and K-NN. In our works, we used three standard criterions that are widely used in document categorization field [37] : *Precision*, *Recall* and *F-Measure* :

$$Precision = \frac{TP}{(TP) + (FP)} \quad (18)$$

$$Recall = \frac{TP}{(TP) + (FN)} \quad (19)$$

$$F - Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (20)$$

Where :

- TP (True Positive) : is the number of documents correctly assigned to the class;
- FP (False Positive) : is the number of documents incorrectly assigned to the class;
- FN (False Negative) : is the number of documents incorrectly removed from the class.

First, we evaluate our method using polynomial and RBF kernels. Table 2 shows the results given by our method. On **20NG** collection, our method performs better using RBF kernel ($\gamma = 0.7$). In the second experiment, we tested the performance of our method with the basic SVM. For the basic SVM, we tested different kernel functions : Polynomial ($d=2,3,\dots,8$) and RBF ($\gamma = 0.1, 0.2, \dots, 1$). We choose only results where SVM performs well. The results are given in table 3.

Table 4 shows our results versus famous standard document categorization techniques : Naive Bayes, Rochio, C4.5 and K-NN. For Rochio algorithm, high performance is obtained with $\beta = 0.5$. For K-NN algorithm, high performance is achieved with $K = 35$. In all cases, our method proved high performance.

Table 2: Detailed test results for different classes of **20NG** with our method

Classes	Performance results					
	$SVM_{Poly:d=5}$			$SVM_{RB:\gamma=0.7}$		
	Recall	Precision	F_1	Recall	Precision	F_1
Class1	92,5	89,9	91,18	97	97,7	97,35
Class2	93,6	91,4	92,49	95	96,6	95,79
Class3	96,5	93,7	95,08	100	100	100
Class4	84,5	82,9	83,69	91,2	92	91,6
Class5	89,3	88,6	88,95	90	89	89,5
Class6	96	94,9	95,45	100	99	99,5
Class7	93	94,6	93,79	96	94,7	95,34
Class8	84	82,3	83,14	82,4	86,2	84,26
Class9	88,3	87	87,65	90	91,2	90,60
Class10	89,4	87,8	88,59	88,7	87,9	88,30
Class11	94,4	90	92,15	94	93	93,50
Class12	92	88	89,96	94,2	93	93,60
Class13	88,6	85	86,76	86,8	85,7	86,25
Class14	72,5	69,1	70,76	79	77,4	78,19
Class15	96,5	94,8	95,64	97	96,7	96,84
Class16	93,2	91,7	92,44	93,7	94	93,85
Class17	78,4	72,4	75,28	80,4	79,6	80
Class18	74,5	77,8	76,11	81	84,3	82,62
Class19	95,4	94	94,69	100	98	98,99
Class20	89,9	88	88,94	89,8	89,4	89,60
Microavg	89,132	87,20	88,14	91,31	91,27	91,29

Table 3: Our method versus basic SVM classifier

	Performance results								
	$SVM_{Poly:d=4}$			$SVM_{RB:\gamma=0.6}$			Our Method		
	Recall	Precision	F_1	Recall	Precision	F_1	Recall	Precision	F_1
Microavg	86,7,5	85,9	86,29	85,4	85,6	85,5	91.31	91.27	91.29

Table 4: Performance comparison between methods.

	Performance results				
	Bayes	Rochio	C4.5	K-NN	Our Method
Recall	72,4	78	83,4	76,4	91,31
Precision	74,7	81,2	85	79,8	91,27
F-Measure	73,53	79,57	84,19	78,06	91,29

8. Conclusions

In this paper, we developed a new SVM fuzzy hierarchical classification method for document categorization and retrieval problem. The retrieval document method consists in finding documents that satisfy queries in databases. The retrieval method is fast in finding the pertinent document. Indeed, we tested our method in **20NG** collection to evaluate its performance. In this work, our method was compared with many different techniques : Basic SVM, Naive Bayes, Rcochio and K-NN algorithm. Our method achieves high performance compared with these techniques on 20 News-groups.

Our method takes its advantage from three concepts : (i)

Latent Semantic Indexing to reduce features dimension (ii) closure transitive which consists in finding shortest link between two documents and (iii) SVM classifier to subdivide original problem into binary problems classification. Our future works consists in introducing fuzzy SVM at each level of the hierarchy to obtain more precise classification. Moreover, we will create a new dynamic Kernel in order to categorize automatically documents.

References

- [1] S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, "A novel ultrathin elevated channel low-temperature poly-Si TFT," *IEEE Electron Device Lett.*, vol. 20, pp. 569–571, Nov. 1999.
- [2] A. Lehman and P. Bouvet, "Evaluation, rectification et pertinence du résumé automatique de texte pour une utilisation en réseau", International Society of Knowledge Organization, pp. 111–125. 2001.
- [3] F. Bellomi and M. Cristani, "Supervised document classification based upon domain-specific term taxonomies", *International J. Metadata, Semantics and Ontologies*, pp.37–46. 2006.
- [4] Th. Joachims, "Learning to classify text using Support Vector Machines : Methods, Theory and Algorithms", Kluwer Academic. University of Rome Tor Vergata. 2002.
- [5] V. Vapnik, "Statistical Learning Theory", John Wiley, Sons. 1998.
- [6] P. N. Tan, M. Steinbach and V.Kumar, "Introduction to Data Mining", Addison-Wesley. 2006.

- [7] S. Tuffery, "Data Mining et statistique decisionnelle. L'intelligence des données", Technip. Paris. 2007.
- [8] L. A. Zadeh, "Fuzzy Sets : Information and Control", vol. 8, pp. 338–353, June. 1965.
- [9] L. A. Zadeh, "Fuzzy Logic. IEEE Transaction on Knowledge and Data Engineering". 1989.
- [10] J. M. Nkoghe, K. Zeroual and W.Shengrui, "Specification Reuse : A Fuzzy Approach", International Joint Conference on Artificial Intelligence. 1995.
- [11] Th. Joachims, "Text Categorization with Support Vector Machines : Learning with many relevant features", Proceedings of the 10th European conference on machine learning. 1998.
- [12] G. Salton and C. Buckley, "Term-Weighting approaches in automatic text retrieval", Department of Computer Science, Cornell University, Ithaca, NY 14853, USA. 1988.
- [13] Th. Joachims, "Transductive Inference for text classification using Support Vector Machine", In proceedings of 16th International Conference on Machine learning, pp. 200–209, San Francisco, USA. 1999.
- [14] http://fr.wikipedia.org/wiki/georgekingsley_Zipf.
- [15] I. Moulinier, "Feature selection : a useful preprocessing step", Proceedings of BCSIRSG-97, the 19th Annual Colloquium of the British Computer Society Information Retrieval Specialist Group, Electronic Workshops in Computing, Aberdeen, UK. 1997.
- [16] S. Dumais, J. Platt, D. Heckerman and M. Sahami, "Inductive learning algorithms and representations for text categorization", Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management, ACM Press, pp. 148–155. New York, US. 1998.
- [17] H. Schutze, D. A. Hull and J. O. Pedersen, "A comparison of classifiers and document representations for the routing problem", Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval, pp. 229–237. ACM Press, New York, US. 1995.
- [18] Y. Yang and J.O. Pedersen, "A comparative study on feature selection in text categorization", Proceedings of ICML-97, 14th International Conference on Machine Learning, pp. 412–420. San Francisco, US. 1997.
- [19] S. Deerwester, S. Dumais, T. Landauer, G. Furnas and R. Harshman, "Indexing by latent semantic analysis", Journal of the American Society of Information science, Vol 416, pp. 391–407. 1998.
- [20] G. Salton, "Automatic Text Processing : the Transformation, Analysis and Retrieval of Information by Computer", Addison-Wesley, New York. 1988.
- [21] G. Salton and M.J.McGill, "Introduction to Modern Information Retrieval", McGraw-Hill, New York, USA. 1983.
- [22] <http://www.ga.gov.au/an/meta/sgmlwhy.html>.
- [23] D. Manning, P.Raghavan and H.Schutze, "An Introduction to Information Retrieval", Cambridge University, England. 2008.
- [24] R. Simon, "Catégorisation automatique de textes et cooccurrence de mots provenant de documents non étiquetés", Université Laval. 2005.
- [25] V. Mitra, C. J. Wang and S. Banerjee, "A Neuro-SVM Model for Text Classification using Latent Semantic Indexing", Proceedings of International Joint Conference on Neural Networks, IJCNN, Montreal, Canada. 2005.
- [26] T. Y. Wang and H.M.Chiang, "Fuzzy support vector machine for multi-class text categorization", Information Processing and Management, Vol 43, pp. 914-929. 2007.
- [27] D. Koller and M.Sahami, "Hierarchically classifying documents using very few words", Proceedings of the 14th International Conference on Machine Learning (ML-97), pp. 170-178. Nashville, Tennessee, July. 1997.
- [28] , D. D. Lewis, "Naive (Bayes) at forty : The independence assumption in information retrieval", Proceedings of the 10th European Conference on Machine Learning (ECML'98), pp. 415. 1998.
- [29] S. Dumais and H.Chen, "Hierarchical Classification of Web Content", Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 256-263. Athens, Greece. 2000.
- [30] H. Lodhi, C.Saunders, J.S.Taylor, N.Cristianini and C.Watkins, "Text Classification using String Kernels. Journal of Machine Learning Research", pp. 419-444. 2002.
- [31] T. Y. Liu, Y.Yang, H.Wan, H.J.Zeng, Z.Chen and W.Y.Ma, "Support Vector Machines Classification with a Very Large-scale Taxonomy", SIGKDD Explorations, Vol 7, pp. 36-43. 2005.
- [32] M. T. Valdivia, L.A.Lopez and M.G.Vega, "The learning vector quantization algorithm applied to automatic text classification tasks", Neural Networks, Vol 20, pp. 748-756. 2007.
- [33] S. M. Weiss, C. Apte, F. J. Damerau, D. E. Johnson, F. J. Oles and H. Goetz. Maximizing text mining performance", IEEE Intelligent Systems, Vol 14, pp. 2-8. 1999.
- [34] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering", IEEE Transactions Pattern analysis and Machine Intelligence, Vol 13, pp. 841-847. 1991.
- [35] D. D. Lewis and M.Ringuette, "A comparison of two learning algorithms for text categorization", In Third annual symposium on document analysis and information retrieval (SDAIR'94), pp. 81-93. 1994.
- [36] Aes and Eikvil, "Text categorization : A survey", Report No 941, Norwegian Computing Center. ISBN 82-539-0425-8. 1999.
- [37] F. Sebastiani, "Machine Learning in Automated Text Categorization", ACM Computing Surveys, Vol 34, pp. 1-47. 2002.
- [38] P.Yi Hao and J-H.Chiang and Y-K.Tu, "Hierarchically SVM classification based on support vector clustering method and its application to document categorization", Expert Systems with applications, Vol 33, pp. 627-635. 2007.
- [39] T. Guernine and K. Zeroual, "SVM fuzzy hierarchical classification method for multi-class problems", The 2009 IEEE International Symposium on Mining and Web.
- [40] X.PENG and B.CHOI, "Automatic Web Page Classification in a Dynamic and Hierarchical Way", IEEE International Conference on Data Mining. pp. 386-393. 2002.
- [41] Th. Joachims, "A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization", In International Conference on Machine Learning (ICML), 1997.
- [42] T. Guernine and K. Zeroual, "A new fuzzy hierarchical classification based on SVM for text categorization", International Conference on Image Analysis and Recognition, Halifax, Canada. 2009.
- [43] F. Schwenker, "Hierarchical Support Vector Machines for Multi-Class Pattern Recognition", Fourth International Conference on knowledge-Based Intelligent Engineering Systems, Allied Technologies, Brighton, UK. 2000.
- [44] T. Y. Wang and H. M. Chiang, "Fuzzy support vector machine for multi-class text categorization", Information Processing and Management, Vol 43, pp. 914-929. 2007.

Evaluation of Structured and Un-Structured document Classification Techniques

Saifullah Azim and Dr. Sohail Asghar

Department of Computer Science

Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad, Pakistan

Email: saifullahazim@yahoo.com, Sohail.Asghar@SZABIST-isb.edu.pk

Abstract: Data mining is the process of analyzing data from different dimensions and summarizing it into useful information. The explosive growth of data in different sectors has highlighted the importance of data mining tools to transform the data into information which was not previously possible due to technological limitations. It is commonly used in a wide range of applications, such as marketing, fraud detection, production, multimedia, streaming and World Wide Web data and scientific discovery. Data mining can be applied to data sets of any size. The phenomenon of explosive growth of documents publishing in both structured and unstructured documents has raised the problem of classifying the documents into their predefined categories. Librarians and subject experts who are usually employed to perform classification task by following the predefined classification system are now looking for automated solution to come out from deadlock situation. Automated Classification can be done through supervised and unsupervised machine learning techniques whereas some of the hybrid techniques are also adopted. In supervised classification technique, a classifier is built from training data and then used for predictive purpose. Alternatively, unsupervised technique utilizes unlabeled data for classification and there is no way to incorporate the user experiences and knowledge about data in this technique. In this paper we broadly review the document classification problem in context of the supervised learning technique. Inherent characteristics of features from structure document and their impact on classification accuracy have been compared with the feature selected from unstructured documents. Similarly the impact of different methodology and model adopted have been critically reviewed in terms of accuracy achieved. The study will act as a landmark to thoroughly understand the document classification problem. It will also elaborate the influence of single or multi word terms on classification performance and impact of different measures adopted on representation of

common and rare classes. In addition, it will also help to understand strength and weakness of different methodologies and models in discussion.

1. Introduction

Classification is a very well known process to organize the items into predefined categories for the purpose to locate the things quickly and easily. Librarians or subject experts are usually hired to perform the classification activity by practicing the widely used classification systems (DDC, LCC, BC..). Due to explosive growth in published material and high cost involved in hiring subject experts has diverted the scientists' attention toward challenge task to automate classification process. In our study, we initially consider both structure and unstructured documents being used as an input for classification but our focus would be on structured documents in scientific domain because structure documents have an added advantage due to their physical and logical formation. Supervised approach has been explored as classification methodology because of its closeness to previous practice. Similarly terms have been explored as performance indicator to enhance the classification results.

Even though hybrid techniques are recently in discussion but due to inherent nature of supervised technique and its closeness to traditional learning process through background knowledge, got intuitive attention to further explore the existing technique toward solution enhancement, instead of reinventing the wheel. On the other hand, unsupervised technique (clustering) which has no concern with previous knowledge itself restricts its suitability as an alternate to problem due to its dependency on user provided parameters.

In supervised approach, previously classified data as a source of knowledge is carefully

divided into training and test data sets. Training data set is initially used as a source to train the classifier. Whereas, test dataset is applied to check the learning level achieved by the classifier during training phase. Feature extraction (single or compound terms) from training or test documents (structure or unstructured) is a common step. Different scoring methodologies are adopted initially for better feature selection and later on to enhance the machine learning process to improve classification result through equal representation of large and rare classes lie in training data set.

2. Literature Review

Bag-of-words approach which previously got equal importance in structured and unstructured document classification, gradually lose its fame in scientific domain due to inherent feature properties and association lies among them. A pattern-based supervised document classification approach was introduced based on association between frequent keyword combinations [1]. Filtration methodology was adopted to remove the infrequent patterns and was represented in the form of regular expression. Patterns extracted from documents were treated as feature to construct the feature vector table and as input to classification algorithm. Ghanem *et al.* [1] refused to adopt Traditional Natural Language Processing (NLP) approach because of its lot of preprocessing requirements. Pattern based methodology have an added advantage in term of reduce dimensionality however no such solution was proposed to classify the document having no pattern.

Zhang *et al.* [2] introduced an inductive Genetic Programming (GP) technique where structural content (For example Title, Abstract) and citation-based information was combined to enrich the document representation for better classification in predefined categories [2]. GP inductive learning technique follows the biological inheritance principal. Classification result obtained from the GP framework were equally good as traditional content-based Support Vector Machine (SVM) when applied to K-Nearest Neighbor (KNN) algorithm, but this time user provided keywords and term extraction methodology were totally ignored.

Automatic text classification, which was initially considered as leading problem, gradually diverges

into multiple issues. Ranking and top scoring methodology previously adopted for feature selection was failed to achieve better classification for small classes due to loss of useful predictive feature during term filtration process. Forman [3] proposed an algorithm (SpreadFx[Round-Robin, IG]) based on round-robin scheduling technique to avoid the pitfall using multi-class SVM by selecting the top 500 features. The algorithm proposed was with an added advantage of using different ranking algorithms for different classes based on prior knowledge for particular classes that could get benefit from certain known methods. Through uniform distribution, enhancement in representation for smaller classes was achieved at a cost of slight decrease for some easy classes. Dramatic improvement in F-measure contributed for better classification but at the stage of preprocessing for feature extraction, normalization of features itself were totally ignored by omitting the process of stemming and stopword which may also affects the performance due to increase in the dimensionality.

Nenadic *et al.* [4] analyze the performance of the Support Vector Machine (SVM) classifier in context with the different type of selected textual unit (words, lemmas and stems, extracted terms). Experimental results performed on Medline abstract shows that the domain-specific terms achieve better performance as compared to the bag-of-words approach. Similarly, classes with large number of training entities have shown better classification results as compared to under-represented or rare classes. Nenadic point of view that the newly arrived compound terms in medical domain are useful source of knowledge indirectly support the Forman [3] idea to equally represent the small classes which are lost during filtration process. Newly arrived domain specific terms not only needed to update in its appropriate class for future classification but also requires for the purpose to identify the new enhancements in specific field.

Building single classifier for multiple domains has its own limitation in term of knowledge building, computing power and to upgrade the model with fresh knowledge [5]. Fu *et al.* [5] proposed a distributed multi-agent architecture to solve the supervised text classification problem by inspiring the distributed computing model of internet. Administration agent is the in-charge for distributing the documents and domain-agents are responsible for conducting the actual classification task. Each agent is responsible to classify content

from its specific domain. For each class, extracted features are the top ranked terms from the corresponding training documents based on TF*IDF term weights. Each document is represented as a document vector. Cosine similarity score is calculated between a document vector and a class vector. If the similarity score exceeds a pre-defined threshold, the document is considered as a member of this class. When an agent is unable to identify any or all of the classes for a document, the agent seeks help from other agents based on the agent coordination strategy. Micro-averaging F score was found higher than the macro averaging F score which is more influenced by classification performance on common classes. Distributed approach reduced the computing power requirements but filtration methodology adopt has decrease the classification performance in rare classes.

As per Khor *et al.* [6], Conference papers are well structured documents and have an added advantage over un-structured documents due to its physical and logical organization. Realizing the fact that user provided keywords are very small in number but are considered sufficient for document classification. Compound terms were broken into single terms by considering that single terms are sufficient for classification. Normalize extracted terms and user expertise was incorporated for constructing the Bayesian Network (BN) for classification. Approach adopt have its own added advantage in term of less text preprocessing. Dependencies between the topics were represented through shared keywords but considering only user provided keyword have its own drawback in term of shortage of terms to achieve better representation of the document.

Flat models do not achieve satisfactory classification results because it ignores the taxonomy information and treat content categorization as a multi-class problem [7]. Whereas, hierarchical models outperform flat models in training efficiency, classification efficiency, and classification accuracy. Almost all hierarchical methods rely on certain predefined content taxonomies. Content taxonomies are usually created for ease of content management or access, so that semantically similar categories are grouped into a parent category. A subject expert or librarian is employed to organize the category labels into a hierarchy. Such taxonomy is often generated independent of data (documents). Hence, there may exist some inconsistency between the given

taxonomy and data, leading to poor classification performance. Likewise, semantically similar categories may not be similar in lexical terms. Most content categorization algorithms are statistical algorithms based on the occurrences of lexical terms in contents. Hence, a semantically sound hierarchy does not necessarily lead to the intended categorization result. Similarly, even for the same set of categories, there could be different semantically sound taxonomies. Semantics cannot guarantee a unique taxonomy. Various users might use different taxonomies to organize the categories.

The stagnant nature of taxonomy and the dynamic change of semantics reflected in data motivate the data-driven adaptation of a given taxonomy in hierarchical classification.

Tang *et al.* [7], reviewed the related work on hierarchical content classification and taxonomy generation and proposed that a semantically sound taxonomy can be used as a base for a divide-and-conquer strategy. A classifier is built independently at each internal node of the hierarchy using all the documents of the subcategories of this category, and a document is labeled using these classifiers to greedily select sub-branches until we reach a leaf node, or certain constraints are satisfied. Feature selection is often performed at each node before constructing a classifier. One advantage of the hierarchy-based approach is its efficiency in training and testing, especially for a very large taxonomy. Hierarchical models make it easy to modify and expand taxonomy, like add one sub-category, delete one category, merge several into one. For each modification, it is not necessary to update the classifiers of all the nodes since the classifiers are built independently. The only thing need is to update a small portion of the classifiers. Therefore, the hierarchical approach is preferred when facing a large taxonomy.

Clustering approach was explored as an alternative to solve the hierarchical classification problem because of its capability to generate a new taxonomy, but due to its limitation for user provided parameters as to specify the maximum depth of the taxonomy or the number of children nodes under each parent category, show that clustering-based hierarchy does not perform as well as the semantics-based hierarchy [7].

The semantics based taxonomy is a form of prior knowledge and provides valuable information to narrow down gradually to find efficiently

reliable hierarchies with good classification performance. Therefore, instead of discarding the given hierarchy information, Tang L *et al.* [7], modify the given hierarchy gradually to generate a data-driven hierarchy, so that classification improvement can be achieved. In order to change a hierarchy to another admissible hierarchy, three elementary operations which include Promote (roll up one node to upper level), Demote (push down one node to its sibling) and Merge (merge two sibling nodes to form a super node) were introduced with condition that the set of leaf nodes remain unchanged. The proposed algorithm traverses the hierarchy using a top-down approach and search for better hierarchies. A hierarchical model was built based on training data by selecting various numbers of features at each node. Hierarchy was adjusted by applying the proposed algorithm and a classifier was built for each of the nodes in the hierarchy. A naive Bayes classifier was used. The greedy search algorithm was used for classifying a document. The average result of macro-averaged recall and F-measure with a 10-fold cross validation shows significant classification difference exists between adjusted and non-adjusted hierarchical models, especially when the number of features selected in each internal node is relatively small. The difference diminishes as more features are selected.

Li *et al.* [8] emphasized on inherent associations lie in textual structures (word, phrase, clause, sentence, paragraph....) of text data and proposed substructure associative classification for better classification. High quality substructure association classification rules (ACR:: A--->C) were extracted based on bottom-up strategy to build a classifier. Compound terms were thought to convey more specific and accurate semantic information than individual terms. At preprocessing stage, stop words were removed and stemmed surface words. Sentences were split based on simple punctuation search strategy like “.” and “?”. For the parts without punctuation each line was considered as a sentence. Feature selection is performed through a set of cascaded associative classifiers rather than by some statistical metrics like information gain. Proposed Adoptive Associative Classifier delivered improved results for small class as compared to the result achieved from Support Vector Machine (SVM).

Even though the Bag of Word (BOW) approach is simple and commonly used but has its own limitations. It breaks the multi-word expressions,

maps synonymous words into different components, and treats polysemous as one single component. Wang *et al.* [9] proposed a document enrichment methodology to overcome the shortages of the BOW approach by embedding background knowledge extracted from Wikipedia into a semantic kernel, which is then used to enrich the representation of documents. The new generated vectors after enrichment process are less sparse, with non-zero entries not only for terms included in the original document, but also for terms semantically related to those present in the document. The enriched representation brings documents which are semantically related closer to each other, and therefore it facilitates the categorization of documents based on their content.

Enrichment of document representation from Wikipedia information source was previously explored by Gabrilovich *et al.* in 2006 where they first build an auxiliary text classifier that can match documents with the most relevant articles of Wikipedia, and then enhance the BOW representation with new features which were the concepts (mainly the titles) represented by the relevant Wikipedia articles. However, the processing effort was very high, because each document needs to be scanned many times. Furthermore, the feature generation procedure inevitably brings a lot of noise, since a specific text fragment contained in an article may not be relevant for its discrimination. In other words, word sense disambiguation processing was still not achieved at satisfactory level. Similarly, Milne *et al.* build a professional, domain-specific thesaurus of agriculture from Wikipedia in 2006 which take little advantage of the rich relations within Wikipedia articles.

Approach adopted by Wang *et al.* [9] relies on a general thesaurus, which supports the processing of documents concerning a variety of topics. Semantic kernel developed was able to keep multi-word concepts unbroken. It captures the semantic closeness of synonyms, and performs word sense disambiguation for polysemous terms. But Building single kernel for multiple domains has its own limitation in terms of knowledge building, computing power, Storage requirements and to upgrade the model with fresh knowledge. Similarly, threshold for enrichment process was not discussed to manage the dimensionality of the document.

3. Critical Evaluation

Carefully selected Structure [1,2,3,4,6,9], unstructured [3,5,7,8,9] and in combination [3,9] of the documents used as an input for training and test data sets are picked for review purpose. In our review section, we will initially discuss the papers where structure contents part (for example title, abstract, keyword) have been discovered for mean of feature extraction for onward classification purpose and then we will discuss the feature extraction methodologies in context of inherent characteristics hidden in pattern and association for classification enhancement. Later on we will discuss the classification models and their strong and weak points in term of accuracy achieved.

Ghanem *et al.* [1] reviewed the performance of the Bag-of-Word approach with their proposed pattern based methodology. Association between the words was captured in the form of patterns and presented as regular expression. TF-IDF was used as scoring methodology. Classification accuracy achieved over B-O-W approach was disappointing (60%) whereas results achieved through pattern based approach over the same SVM classifier were impressive (80%). Similarly, Zhang *et al.* [2] experienced classification of structure documents in scientific domain by discovering the “best similarity tree” in each class and reported enhancement in classification results. Meanwhile Forman [3] highlight the pitfall of ranking and top scoring methodology where filtration process was actually losing the useful predictive features. Forman [3] proposed algorithm with Round-Robin scheduling technique, contributed for uniform distribution and for equal representation of common and rare classes which were previously ignored or discarded during filtration process. Dramatic improvement in Macro and Micro F-measure contributed for better classification in both large and small classes.

Nenadic *et al.* [4] analyze the impact of selected textual unit (words, lemmas and stems, extracted terms) on classification performance in context of the domain specific selected features. Experimental results performed on Medline abstract shows that domain-specific terms achieve better performance as compared to the single words (B-O-W) approach when applied to SVM classifier. Furthermore, Nenadic point of view that the newly arrived compound terms in

medical domain are useful source of knowledge and needed to updated in its appropriate class for future classification actually supported the Forman [3] idea for equal representation of rare classes. Fu *et al.* [5] presented the same overall decrease in classification performance by experience the Forman [3] reported dilemma raised due to top ranking filtration methodology with distributed computational model.

Khor *et al.* [6] adopt a Bayesian Network (BN) based approach to draw a network sketch to obtain association lies between the user-provided keywords extracted from conference papers. Even though the average result (83.75 %) of the proposed BN classification performance was considerable but reported dropping predictive accuracy in some of the topics due to learning capability of BN. Meanwhile Tang *et al.* [7] reviewed the performance of the hierarchical model which has a better training efficiency, classification efficiency, and classification accuracy due to its divide-and-conquer strategy. Independently build classifier for each class give an opportunity to represent the small classes which are previously dominated by large classes due to their global scoring methodology.

Similarly, Li *et al.* [8] reported better classification accuracy achieved in rear classes by extracting the substructure association rules lie in textual structure (word, phrase, clause, sentence, paragraph.....). Recently, Wang *et al.* [9] proposed a document enrichment methodology to overcome the shortages of the BOW approach by embedding background knowledge extracted from Wikipedia. During text preprocessing phase, extracted terms and their candidate concept in Wikipedia are compared and single terms are replaced with their conceptual compound terms. Wang's proposed methodology has achieved higher micro and macro precision values on all data sets. Best result achieve are reported for the OHSUMED and 20NG data sets.

Structure documents like scientific articles and conference paper have an added advantage over unstructured documents like web documents and news articles due to its physical and logical organization of the documents [6]. Zhang *et al.* [2] combined the structural content (For example Title, Abstract) and citation-based information to achieve better classification in predefined categories. But on the other hand, Khor *et al.* [6] only extracted the user provided keyword of

conference paper for classification task. Whereas, Wang *et al.* [9] combined the extracted feature from structured content (titles and/or abstracts from MEDLINE database) part with the features of external information source (Wikipedia) to enrich the document representation. The enrichment process actually fills the missing information and brings documents closer to each other which are semantically related, and therefore it facilitates the classification accuracy.

Similarly, the data sets [1,3,4] explored as an input for training and testing of classification model in scientific domain are mainly either the abstract or in combination with the title information. Even though, smallest part of the structure documents was considered for feature extraction but interestingly the results achieved are equally good or even better as presented with unstructured documents. Similarly, Nenadic *et al.* [4] claimed that Domain-specific terms (medical domain) have achieved better performance as compared to the bag-of-words approach.

Macro-averaged F-measure emphasizes the smaller classes, while micro-averaged F-measure is more influenced by common classes. The higher gain in micro-average score in combination with dropped in macro-averaged F-measure, actually presents the domination of the larger or common classes over rare class [3]. Forman [3] proposed algorithm with uniform distribution methodology dramatically improve the Macro average F measure at a cost of slight decrease in micro-averaged which enhanced the classification accuracy in small class. The uniform distribution approach [3] contributed as an alternate solution for ranking and filtration problem lie with B-O-W approach. Similarly, the suggestion of using different ranking algorithms for different classes actually supported the highlighted dilemma lie with flat models [7]. The proposed approach also supported the Fu *et al.* [5] idea, to build multiple classifiers for different domains to overcome computational requirements and to update the classifier with new knowledge efficiently.

Nenadic *et al.* [4] evaluation in context of the different selected text feature and its performance over the SVM classifier shown that domain specific terms extracted through C-value (by generating candidate concept) method perform better in classification accuracy.

Through experimental evaluation, Nenadic *et al.* [4] explore that by combining single words and terms have no effect on classification accuracy so there is no need to preprocess text. In addition, they also experienced that adding more features may introduced noise that derogated the overall performance. Nenadic *et al.* [4] and Ghanem *et al.* [1] have commonly experienced B-O-W approach with discouraging classification performance and reported better classification results with domain specific terms or patterns extracted from the same medical domain. In contrast, Wang *et al.* [9] initially generated single words through B-O-W approach and then gradually generated multi-terms concept from single terms. Subsequently, passing through enrichment process of external source (Wikipedia thesaurus), Wang also achieved better classification results using same datasets as reported by Nenadic *et al.* [4] and Ghanem *et al.* [1]. Result Comparison of Ghanem *et al.* [1], Nenadic *et al.* [4] and Wang [9], shows that 80 to 84% accuracy is achieved by Ghanem, Nandec achieve better classification result for large class whereas Wang achieved equally good result in both large and rear class.

Fu *et al.* [5] proposed a distributed multi-agent architecture to solve the text classification problem. The proposed multi-agent strategy was observed close to Forman [3] solution to perform the cost sensitive classification for large number of classes. Classification results over multi-agent architecture were experienced using unstructured documents (newswire stories). Traditional TF-IDF approach was adopted with filtration methodology. Constantly higher micro-averaging F score then that of Macro-average showed that the rear classes are completely dominated by common classes. The phenomena observed was exactly the same as was previously in 'siren pitfall' [3]. Interestingly, the overall performance (in term of micro and micro average F score) of the proposed frame work was observed less then that of centralized classification system with same data set. Similarly, Li [8] have reported improved macro-averaged scores for same dataset by extracting substructure association rule. Whereas, Wang [9] have reported the higher values for both micro and macro-averaged scores by adopting document enrichment strategy through semantic kernel. Using unstructured documents with different data set, Tang [7] also reported the improvement in Macro-average F score by adjusting the

hierarchy structure to achieve semantically sound taxonomy

Khor *et al.* [6] experienced the classification of 400 conference papers into four topics through small number of user provided keywords. Author provided keywords are usually compound terms which are normally different from the topic of discussion [6]. Methodology adopt to split the compound terms into single term looks in reverse order of the approach adopt by Wang [9] to generate candidate terms from single terms. Slight drop in accuracy of the classification in two topic is reported due to learning capability of classifier(BN) but alternatively, the feature filtration methodology looks closer to previously reported approach with 'siren pitfall' [3]. Achieving acceptable classification accuracy with small number single terms can be compared with the reported phenomena of Tang [7] where this has been observed that increasing feature exceeding 2000 will no more contribute for classification enhancement. Means that, Forman [3] approach to apply threshold (top 500) for feature selection based on prior knowledge of particular classes is equally important. Similarly, Khor [6] approach to split the compound terms into single terms can me consider for classification where the number of provided terms and/or training data are comparably small in size.

In hierarchical models semantically similar categories are grouped into a parent category and are often developed independent of data (documents). A subject experts or librarians are usually employed to organize the category labels into a hierarchy. Inconsistency between the given taxonomy and data may lead to poor classification performance. Tang *et al.* [7], proposed the hierarchy modification to generate a data-driven hierarchy and reported enhancement in Macro Average F score. This shows that the classification performance in rare classes was improved through hierarchy modification. Fu [5] (multi-agent) and Tang [7] (hierarchical) models have resemblance based on dive-and-conquer strategy being adopt. But in contrary, Fu [5] reported loss in Macro average score after applying multi-agent framework. Interestingly, for the same data set through centralized classifier, Fu [5] has observed the higher Macro average F score, means that by combining different classes, classification in rare classes can be improved. The increase in macro average F score observed in Fu [5] centralized

classifier experiment actually support the Tang [7] idea to modify the hierarchy for better classification results.

Li *et al.* [8] reported classification improvement in rare classes by extracting substructure association rules lie in textual unit (term). Similarly, Ghanem *et al.* [1] extracted patterns to draw association lie between the words of the same document and reported classification enhancement (up to 84%). Comparably, Khor *et al.* [6], adopt a Bayesian Network (BN) based approach and draw association between topics by decomposing multi words concept into single keywords or by mean of shared keyword for better classification results.

4. Conclusions and Future work

Performance and accuracy of the classifier not only depends on the learning capability of the classifier but equally depend on the size of training data, format (single or multi term feature) of the extracted text feature, scoring methodology and threshold applied for extraction of the quality texture contents. Similarly, adding more textual contents does provide a guarantee to enhance classification performance.

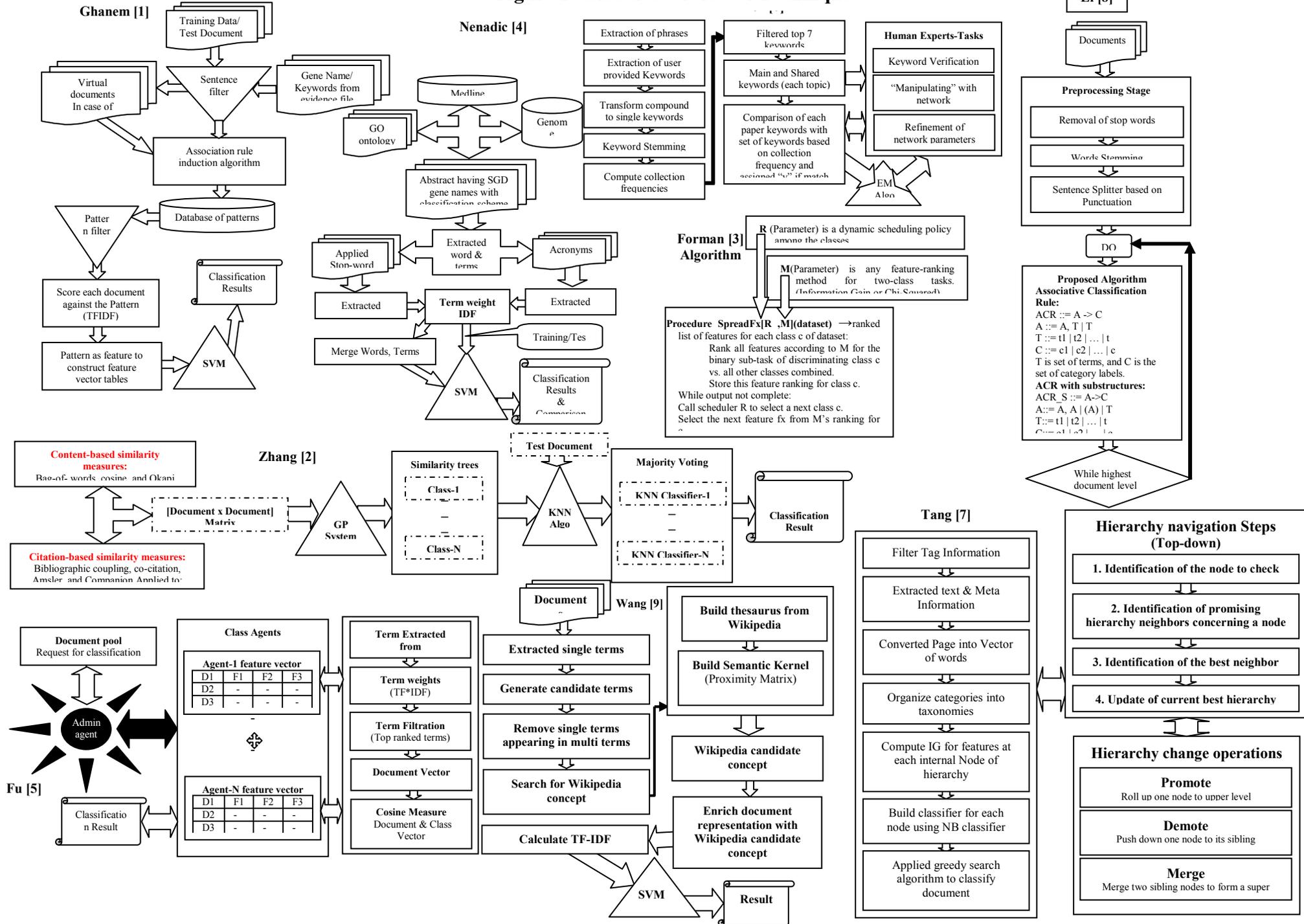
Our future plan is to propose a better approach for feature selection (in structure and unstructured documents) that can contribute classification accuracy in both data sets. Similarly, based on currently proposed models, we will also prose our own hybrid model that can enhance the overall classification performance.

Reference:

- [1] Ghanem M. M., Guo. Y, Lodhi H., Zhang Y., Automatic Scientific Text Classification Using Local Patterns: KDD CUP 2002 (Task 1), JCDL'05, June 7-11, 2005, Denver, Colorado, USA, ACM 1-58113-876-8/05/0006., 2002.
- [2] Zhang B., Andre M., Goncalves, Fan W., Chen Y, Fox Edward A., Calado P., Cristo M., Combining Structural and Citation-Based Evidence for Text Classification, ACM 1-58113-874-1/04/0011, 2004.
- [3] Forman G., A Pitfall and Solution in Multi-Class Feature Selection for Text Classification, Appearing in Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.
- [4] Nenadić G., Rice S., Spasić I., Ananiadou S, Stapley B., Selecting Text Features for Gene

- Name Classification: from Documents to Terms, 2004
- [5] Fu Y., Ke W., Mostafa J., Automated Text Classification Using a Multi- gent Framework, ACM 1-58113-876-8/05/0006, 2005.
 - [6] Khor K. C., Ting C. Y., A Bayesian Approach to Classify Conference Papers, MICAI 2006, LNAI 4293, pp. 1027 – 1036., 2006.
 - [7] Tang L., Zhang J., Liu H., Acclimatizing Taxonomic Semantics for Hierarchical Content Classification, ACM 1-59593-339-5/06/0008.2006
 - [8] Li B., Sugandh N., V E., Garcia, Ram A., Adapting Associative Classification to Text Categorization, ACM 978-1-59593-776-6/07/0008., 2007.
 - [9] Wang Pu, Domeniconi C., Building Semantic Kernels for Text Classification using Wikipedia, ACM 978-1-60558-193-4/08/08, 2008.

Figure 1: Global Overview of Techniques



Author.	Text classification		Experimental Results	Model
	Structured/ Un-Structured Doc.	Approach/Domain		
Ghanem [1]	Database of scientific papers from Medical Domain	Association lie between the words of the same document were extracted as pattern and presented in the form of regular expressions. Pattern were used as feature to construct vector tables and finally passed as an input to classifier algorithm with score of each document against the pattern.	1. KEYWORD-BASED CLASSIFIERS 60% accuracy on the training data 2. PATTERN-BASED CLASSIFIERS classifier based on 335 automatically extracted patterns provided accuracy in the range of 80% for the training data and following accuracy results on the evaluation data: Ranked-list: 84%, Yes/No curate paper: 58% Yes/No gene products: 59%	PATTERN-BASED CLASSIFIERS Identifying frequently co-occurring localized word patterns <ul style="list-style-type: none"> Patterns defined using regular expressions on words Build a database of patterns by scanning the training data set using a association rule induction algorithm Applied filter to reduce the size of pattern patterns as features to construct the feature vector tables Score each document against the patterns and pass the table as input to the classification algorithm.
Zhang [2]	Documents from ACM Digital Library and the ACM Computing Classification System	Citation-based information and structural content (e.g., title, abstract) were combined using Genetic Programming system to get the best similarity tree of different classes which is further used as an input to KNN classifier for improved classification into predefined categories.	Effectiveness of classification framework is demonstrated in two ways: 1) By comparing its performance against a majority voting of classifiers using the best baselines for each class; and 2) By comparing experimental results with results achieved through a content-based SVM Macro average F1 measure is used as comparison criteria. Results obtained shows that the performance of SVM is slightly worse than that of the majority GP.	Content-based and citation-based similarity measures were represented as document \times document matrices and served as input to GP system. The training process within GP tried to discover imilarity trees for each class. The overall classification framework is as follows: 1) Discover "best similarity tree" for each class 2) Apply the "best similarity tree" of each class on a set of testing documents to a kNN algorithm 3) Combine the output of each classifier through a simple majority voting.
Forman [3]	Research paper abstracts into various fields of computer science. Dataset extracted a set of abstracts and their associated classifications within the Cora topic categorization available by Whizbang.com. OHSUMED, Reuters data set, TREC	Feature selection algorithm have a very common flaw of ignoring the features of rare class is being highlighted. Proposed algorithm is based on Round-Robin scheduling for uniform distribution for equal presentation of rare classes with an added advantage of using different ranking algorithms for different classes.	Algorithm SpreadFx[Round-Robin, IG] makes over traditional IG for the 36-class Cora dataset performed 4-fold cross-validation on the dataset, using multi-class SVM and selecting the top 500 features. The result shows dramatic improvement in F-measure for each class. Especially for rare classes, and a slight decrease for some of the easy classes.	<ul style="list-style-type: none"> Include all words except very common words appearing in >25% of the documents, and rare words occurring in fewer than three documents No stemming and no stopword list applied Applied Algorithm SpreadFx[Round-Robin, IG] over traditional IG using multi-class SVM by selecting the top 500 features.
Nenadic [4]	Medline abstracts (NLM, 2003) with 52,845 abstracts, Saccharomyces Genome Database for gene names.	Analyze performance of Support Vector Machine (SVM) in context with the different type of selected textual unit (words, lemmas and stems, extracted terms).	Domain-specific terms achieve better performance as compared to the bag-of-words approach. classes with large number of training entities perform better classification as compared to under-represented or rear classes	<ul style="list-style-type: none"> Single words have been extracted as features in step-1 by applying stopword. Domain specific Terms have extracted using enhanced C-value method in step-2. Mapped acronyms with their expanded forms Combined word and term features to generate feature vector Applied to SVM for comparison of classification performance
Fu [5]	RCV1-v2 was chosen as data set for the experiments. It contains a corpus of more than 800,000 manually categorized newswire stories from Reuters, Ltd.	A multi agent framework is proposed based on distributed classification strategy. Administrative agent is in charge for document distribution. Classification agents from different classes are actually responsible for classification task.	In multi agent classification frame work, micro-averaging F score is always higher then the Macro-average, mean that, common class are more influenced by rare classes. Secondly the overall performance (in term of micro and micro average F score) of the presented multi agent frame work is less then the centralized classification system.	<ul style="list-style-type: none"> Administrative agent is responsible for coordination whereas classification agent perform classification task in their limited domain. Documents and classes are represented in Vector Space Model using TF*IDF term weights. the features are the top ranked terms from the corresponding training documents Cosine similarity score is calculated between a document vector and a class vector.
Khor [6]	Conference Papers (400) with 4 major topics.	A Bayesian Network (BN) based approach was adopted to classify the conference papers. Only user provided compound keywords are extracted and considered for classification purpose by decomposing into single keywords. Normalize extracted terms and user expertises are incorporated for classification task.	The experimental results of the proposed NB network has achieved 90% rated by human expert while obtaining an average of 83.75% that is higher than BN learned from training data (76.25%), and Naive Bayesian network(82.5%). However, a moderate drop of predictive accuracy was observed in two topics.	<ul style="list-style-type: none"> Extracted user provided keywords through proposed algorithm Decompose compound terms to single terms and normalize through stemming process. Compute the collection frequency of keywords and filter top 7 words. Main and shared keywords from each topic were verified by human experts. Comparison of each paper keywords with set of keywords based on collection frequency and assigned "y" if match otherwise "n" Applied network to Expectation Maximization algorithm to learn the parameters of nodes. Experts are involved to verify the parameters learned before evaluating the network.
Tang [7]	1800 web pages obtained from AOL.com with 8 categories: Basketball, Football, Politics, Economics, Movies, Music, Video Game, and Word Game.	Hierarchal models and their top-down methodology is exploited for better classification results. Semantically sound taxonomy has been explored for potential change in its structure for better classification results. A naive Bayes classifier was used to build classifier for each of the nodes in the hierarchy.	Macro-average F measure have been used for performance comparison. As per experimental results, Macro average F measure has been improved after applying the hierarchy-adjustment algorithm even with small number of features. Similarly this has also been observed that at one point (2000 feature) the increase in features will not further more affect the classification results toward improved	Proposed algorithm consists of multiple iterations to traverse the hierarchy using a top-down approach and search for better hierarchies. For each search step, following procedures have been performed: 1. Identification of the node to check. 2. Identification of promising hierarchy neighbors concerning a node. 3. Identification of the best neighbor. 4. Update of current best hierarchy. Algorithm will stop at a hierarchy which performs better than the original hierarchy.
Li [8]	Reuters-21578 dataset	Compound terms have been considered more useful and accurate source of information as compared to individual terms. Association lies between the terms have been explored as mean of classification by proposing association rule with substructure strategy for better classification results	Compared to kNN and SVM, HMY and Proposed strategy (ADPT_HM) achieve better macro-averaged scores and perform quite well on small categories, whereas SVM shown excellent generalization capacity on large categories. Among the 10 categories, SVM achieves the best performance for five large categories, where ADPT_HMY and HMY perform best for small categories respectively. For rare category SVM achieve the poorest performance.	<ul style="list-style-type: none"> Preprocessed the text by applying Stopword and stemming. Spilt the continuous text of document into sentences. Extract substructure association classification rules based on bottom-up strategy. Build instance-centric associative classifier for classification.
Wang [9]	OHSUMED test collection from MEDLINE medical information database, consisting of titles and/or abstracts Reuters-21578,20, Newsgroups (20NG) Movie Reviews (Movies)	Considering Wikipedia as one of the largest source of information, semantic Kernel is derived from Wikipedia thesaurus which is initially used as a source for concept matching for newly arrived document and later on used to enrichment the document representation for better classification.	Micro-averaged and the macro-averaged precision values are obtained for the three methods on four data sets. Proposed method provides higher micro and macro precision values on all data sets. The improvement with respect to the Baseline (BOW) is significant for all four data sets. The largest improvements with respect to Wiki-Enrich are obtained for the OHSUMED and 20NG data sets.	<ul style="list-style-type: none"> Extracted words using Bag-O-word approach Generated candidate terms and exclude single terms appearing in candidate terms only. Pick the multi-terms without stemming Identified candidate concepts using Wikipedia thesaurus to select synonyms, hyponyms, and associative concepts of the candidate ones. Enrich the vector representation of the corresponding document. Compute TF-IDF of the terms and used SVM classifier for classification

Towards Online Personalized Foreseeing System by New Approach through Web Usage Mining

Mehrdad Jalali¹, Norwati Mustapha², Md Nasir Sulaiman², Ali Mamat²

¹ Software Engineering Dept., Islamic Azad University, Mashhad branch, Iran

Ph.D. Candidate of Computer Science, University of Putra Malaysia

² Faculty of Computer Science and Information Technology,
University of Putra Malaysia (UPM), Selangor, Malaysia.

Abstract - Web usage mining has become the subject of exhaustive research, as its potential for web based personalized services, foreseeing user near future intentions, adaptive Web sites and customer profiling is recognized. A variety of the recommendation systems for online personalization through web usage mining have been proposed. However, the quality of the recommendations in the current systems to foresee users' future requests systems cannot still satisfy users in the particular huge web sites. To provide online foreseeing effectively, we develop a model for online foreseeing through web usage mining system and propose a novel approach for classifying user navigation patterns to foresee users' future intentions. The approach is based on the new graph partitioning algorithm to model user navigation patterns for the navigation patterns mining phase. Furthermore, longest common subsequence algorithm is utilized to classify current user activities to foresee user next movement. We have tested our proposed model on the two datasets. The results indicate that the approach can improve the quality of the system for both clustering and classifying phases.

Keywords: Web Usage Mining; Personalization Systems; Navigation Pattern Mining.

1 Introduction

Given the rate of growth of the Web, proliferation of e-commerce, Web services, and Web-based information systems, the volumes of clickstream and user data collected by Web-based organizations in their daily operations has reached huge proportions. Meanwhile, the substantial increase in the number of websites presents a challenging task for webmasters to organize the contents of the websites to cater to the needs of users. Modeling and analyzing web navigation behavior is helpful in understanding what information of online users demand. Following that, the analyzed results can be seen as knowledge to be used in intelligent online applications, refining web site maps, web based personalization system and improving searching accuracy when seeking information. Nevertheless, an online navigation behavior grows each passing day, and thus extracting information intelligently from it is a difficult issue. Web usage mining refers to the automatic discovery and analysis of patterns in clickstream and associated data collected or generated as a result of user interactions with

Web resources on one or more Web sites [1-3]. Web usage mining has been used effectively as an approach to automatic personalization and as a way to overcome deficiencies of traditional approaches such as collaborative filtering. The goal of personalization based on Web usage mining is to recommend a set of objects to the current (active) user, possibly consisting of links, ads, text, products, and so forth, tailored to the user's perceived preferences as determined by the matching usage patterns. This task is accomplished by matching the active user session with the usage patterns discovered through Web usage mining.

In this paper, to provide online foreseeing effectively, we develop a model for online foreseeing through web usage mining system and propose a novel approach for classifying user navigation patterns to foresee users' future intentions. The approach is based on the new graph partitioning to model user navigation patterns for the navigation patterns mining phase. Furthermore, longest common subsequence algorithm is utilized to classify current user activities to foresee user next movement.

The rest of this paper is organized as follows: In section 2, we review recent research advances in web usage mining. Section 3 describes the system design and section 4 focuses on the system evaluation. Results of experimental evaluations are reported in section 5. Finally, section 6 summarizes the paper and introduces future work.

2 Background and Related Work

Recently, several WUM systems have been proposed to foresee user's preference and their navigation behavior. In the following, we review some of the most significant WUM systems and architecture that can be compared with our system.

Analog [4] is one of the first WUM systems. It is structured according to an off-line and an online component. The off-line component builds session clusters by analyzing past users activity recorded in server log files. Then the online component builds active user sessions which are then classified according to the generated model. The classification allows to identify pages related to the ones in the active session and to return the requested page with a list of suggestions. The geometrical approach used for clustering is affected by several limitations, related to scalability and to

the effectiveness of the results found. Nevertheless, the architectural solution introduced was maintained in several other more recent projects.

Mobasher et al., present WebPersonalizer a system which provides dynamic recommendations, as a list of hypertext links, to users [5, 6]. The analysis is based on anonymous usage data combined with the structure formed by the hyperlinks of the site. Data mining techniques (i.e. clustering, association rules and sequential pattern discovery) are used in the preprocessing phase in order to obtain aggregate usage profiles. In this phase Web server logs are converted in clusters made up of sequences of visited pages, and cluster made up of set of pages with common usage characteristics. The online phase considers the active user session in order to find matches among the user's activities and the discovered usage profiles. Matching entries are then used to compute a set of recommendations which will be inserted into the last requested page as a list of hypertext links. WebPersonalizer is a good example of two-tier architecture for Personalization systems.

A partitioning graph theoretic approach is presented by Perkwitz and Etzioni [7], who have developed a system that helps in making Web sites adaptive, i.e., automatically improving their organization and presentation by mining usage logs. The core element of this system is a new clustering method, called cluster mining, which is implemented in the PageGather algorithm. PageGather receives user sessions as input, represented as sets of pages that have been visited. Using these data, the algorithm creates a graph, as signing pages to nodes. An edge is added between two nodes if the corresponding pages co-occur in more than a certain number of sessions. Clusters are defined either in terms of cliques, or connected components. Clusters defined as cliques prove to be more coherent, while connected component clusters are larger, but faster to compute and easier to find. A new index page is created from each cluster with hyperlinks to all the pages in the cluster. The main advantage of PageGather is that it creates overlapping clusters. Furthermore, in contrast to the other clustering methods, the clusters generated by this method group together characteristic features of the users directly. Thus, each cluster is a behavioral pattern, associating pages in a Web site. However, being a graph based algorithm, it is rather computationally expensive, especially in the case where cliques are computed.

Liu and Keselj [8] proposed the automatic classification of web user navigation patterns and proposed a novel approach to classifying user navigation patterns and predicting users' future requests. The approach is based on the combined mining of Web server logs and the contents of the retrieved web pages. They used character N-grams to represent the contents of web pages, and combined them with user navigation patterns by building user navigation profiles composed of a collection of N-grams. In this system they can incorporate their current off-line mining system into an on-

line web recommendation system to observe and calculate the degree of real users' satisfaction on the generated recommendations, which are derived from the predicted requests, by their system.

Baraglia and Palmerini proposed a WUM system called SUGGEST, that provide useful information to make easier the web user navigation and to optimize the web server performance [9, 10]. SUGGEST adopts a two levels architecture composed by an offline creation of historical knowledge and an online engine that understands user's behavior. As the requests arrive at this system module it incrementally updates a graph representation of the Web site based on the active user sessions and classifies the active session using a graph partitioning algorithm. Potential limitation of this architecture might be: a) the memory required to store Web server pages is quadratic in the number of pages. This might be a severe limitation in large sites made up of millions of pages; b) it does not permit us to manage Web sites made up of pages dynamically generated.

All of these works attempt to find architecture and algorithm to improve quality of the personalized recommendation, but the recommendations still do not meet satisfaction. In our work, we advance a model and propose novel approach to foresee user intention in the near future request.

3 System Design

In this paper, a model is proposed to foresee user's future requests. According to different functions, the model can be partitioned into two main phases; offline phase and online phase. In spite of these phases are separated in the model, the offline phase strongly affects to the online phase. Figure 1 illustrates the model of the system.

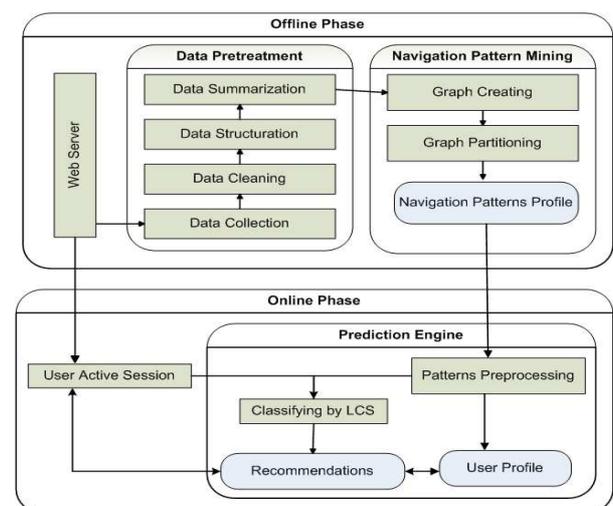


Figure 1: Model of the system

3.1 Offline phase of the model

Two main modules are consisted in this phase; Data pretreatment and navigation patterns mining. Data pretreatment module is done to extract web users navigation's sessions from the original web user logs file. The clustering algorithm based on graph partitioning is used for navigation patterns mining. Data pretreatment and navigation pattern mining are stated in the following.

3.1.1 Data Pretreatment

Generally, data pretreatment in a web usage mining system aims to reformat the original web user logs file to identify all web users navigation's sessions. The Web server usually registers all web user navigation as web user logs file. Due to different server setting parameters, there are many types of web logs, but typically the log files share the same basic information, such as: client IP address, request time, requested URL, HTTP status code, referrer, etc.

Moreover, several pretreatment tasks need to be done before performing navigation patterns mining algorithms on the web user logs file. In this work, these include data collection, data cleaning, data structuration and data summarization. Data collect from web server log file from several web servers in data collection task and irrelevant data eliminated from log file in the data cleaning phase. User and user's sessions identified in the data structuration task by utilizing some well-known algorithms. Consequently, data file transfers to a relational database and then applies data transformation and aggregated data computation for user sessions in the data summarization. These preprocessing tasks are the same for any web usage mining problem and are discussed in [11, 12].

3.1.2 Navigation Pattern Mining

In the proposed system, user navigation patterns are described as the common browsing characteristics among a group of users. Since many users may have common interests up to a point during their navigation, navigation patterns should capture the overlapping interests or the information needs of these users. In addition, navigation patterns should also be capable to distinguish among web pages based on their different significance to each pattern. After the data pretreatment step, we perform navigation pattern mining on the derived user access sessions. The representative user navigation pattern can be obtained by clustering algorithms. Clustering of the user navigation pattern aims to group sessions into clusters based on their common properties. Access sessions that obtained by the clustering process are the patterns of web user activities. These patterns will be used further to classifying current user activities in online phase of the system. In this study, the user navigation pattern is defined as follows:

Definition 1: A user navigation pattern np captures an

aggregate view of the behavior of a group of site users based on their common interests or information needs. As the results of session clustering, $NP = \langle np_1, np_2, \dots, np_k \rangle$ is used to represent the set of user navigation patterns, in which each np_i is a subset of P , the set of web pages [8].

In this study, a clustering model is used for navigation pattern mining. The model exploits graph partitioning algorithm by applying new method for creating undirected graph.

The clustering model is built to find collection of related pages at a web site, relying on the visit-coherence assumption. Visit-coherence assumption is defined in [13]. The pages that a user visits during one interaction with the site tend to be conceptually related. The process of the clustering are elaborated in the next.

1. *Compute the degree of connectivity between web pages and create an adjacency matrix.* For each pair of pages P1 and P2, we compute $W(a, b)$, a weight which is the degree of connectivity between web pages. Here a new measurement is proposed for approximating the degree of connectivity for each pair of web pages in a session. First, let us introduce two concepts related to this measure, "Time Connectivity" and "Frequency".

Time Connectivity, measures the degree of visit ordering for each two pages in a session. In (1) we compute this measure with this novel formula:

$$TC_{a,b} = \frac{\sum_{i=1}^N \frac{T_i}{T_{ab}} \times \frac{f_a(k)}{f_b(k)}}{\sum_{i=1}^N \frac{T_i}{T_{ab}}} \quad (1)$$

where T_i is time duration in i -th session, that containing both pages a and b , T_{ab} is the difference between requested time of page a and page b in the session. We consider $f(k)=k$ if web page appears in position k . However, a different form for f could be chosen. For instance, it is also possible to increase the importance of the position of each two pages in a session by taking $f(k) = k^2$. The formula is also normalized so all values for time connectivity are between 0 and 1.

"Frequency" measures the occurrence of two pages in each sessions. In (2) we compute this:

$$FC_{a,b} = \frac{N_{ab}}{\text{Max}\{N_a, N_b\}} \quad (2)$$

where N_{ab} is the number of sessions containing both page a and page b . N_a and N_b are the number of session containing only page a and page b . and this formula also has the values between 0 and 1.

In our system, "Time Connectivity" and "Frequency" are considered two strong indicators of the degree of connectivity for each pair of web pages. Therefore, in the weight measure we devised, "Time Connectivity" and "Frequency" are valued equally. We use the harmonic mean of "Time Connectivity" and "Frequency" to represent the connectivity of each two pages, shown as (3). We take this formula for weight of each edges in the undirected graph.

$$W_{a,b} = \frac{2 \times TC_{ab} \times FC_{ab}}{TC_{ab} + FC_{ab}} \quad (3)$$

2. Create an undirected graph corresponding to the adjacency matrix. The data structure can be used to store the weights is an adjacency matrix M where each entry M_{ab} contains the value W_{ab} computed according to (3). To limit the number of edge in such graph, elements of M_{ab} whose value is less than a threshold are small correlated and thus discarded. This threshold is named as *MinFreq* in this study.

3. Find the connected component in the graph based on graph search algorithm. The graph partitioning algorithms divide a graph into k disjoint partitions, such that the partitions are connected and there are few connections between the partitions. In this paper, graph partitioning algorithm is utilized to find groups of strongly correlated web pages by partitioning the graph according to its connected components. Depth-first search (DFS) is an algorithm for traversing or searching a graph. Starting from a vertex a Depth First Search (DFS) on the graph induced by M is applied to search for the connected component reachable from this vertex. Once the component has been found, the algorithm checks if there are any nodes not considered in the visit. If so, it means that a previously connected component has been split, and therefore, it needs to be identified. To do this the DFS is again applied by starting from one of the nodes that are not visited. In the worst case, when all the URLs are in the same cluster, the cost of this algorithm will be linear in the number of edges of the complete graph G .

We have to take into account of two main parameters while algorithm applies to the undirected graph. Minimum frequency and minimum cluster size are two parameters that affect extremely to mining of navigation patterns.

Minimum frequency is a parameter for filtering the weights which are below a constant value, named as *MinFreq* in this paper. The edges of the graph whose values are less than *MinFreq* are inadequately correlated and thus not considered by DFS graph search algorithm.

Moreover, all the connected components having number of nodes greater than a fix size are considered and the rest of components is not significant enough. In this study, the minimum cluster size named as *MinClusterSize*.

In this study, connected components that created based on graph partitioning algorithm are a set of navigation patterns.

The results of the algorithm show as $NP = \langle np1, np2, \dots, npk \rangle$, which NP is a set of navigation patterns. Meanwhile, NP can be considered as a set of clusters that further utilize in the online phase.

Figure 2 illustrates an example of the clustering process. There are a set of web pages in each session, and we consider each pages as graph vertices (figure 2(a)), an undirected graph is created based on degree of connectivity between web pages (figure 2(b)). We take *MinFreq*=2 and *MinClusterSize*=3, consequently the edges below than this value is eliminated before applying graph search algorithm

(figure 2(c)). In the last part, DFS algorithm applies to the undirected graph to find the connected components in the graph (figure 2(d)). The clusters C2 and C5 are eliminated seeing that *MinClusterSize* is below than three in these clusters. The results of navigation pattern mining (Clustering) are shown as follow.

C1=NP1= $\langle P1,P2,P5,P6,P9,P11 \rangle$

C3=NP3= $\langle P4,P8,P14 \rangle$

C4=NP4= $\langle P13,P15 \rangle$

The results of navigation pattern mining further will be used for online phase.

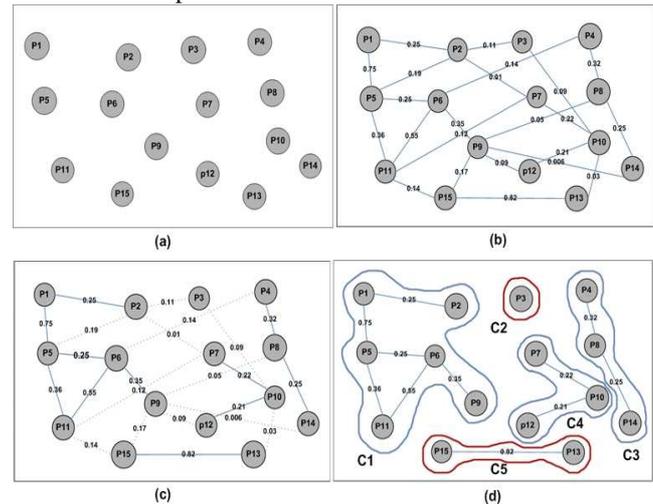


Figure 2: An example of clustering process

3.2 Online phase of the model

According to different part of the proposed system, navigation patterns are generated in offline phase of the system by utilizing graph partitioning algorithm. Online components in a majority of prediction engine are done in the online phase of the system to foresee user's future requests.

Classifying of user's current activities based on navigation patterns in the particular web site is the main objective of the current step. In addition, creating a list of recommendations web pages as prediction is another objective in online phase. In the following, we state the online components.

3.2.1 Prediction Engine

As the first objective, the prediction engine is used to classify user navigation patterns and foresees users' future requests. For this purpose, we propose a novel approach to classify current user activity. In order to classify user's active session, we look for the navigation pattern that includes the larger number of similar web pages in the session. Pattern search approaches can be utilized to find similar web pages between the current active session and navigation patterns. The longest common subsequence (LCS) algorithm is to find the longest subsequence common to all sequences in a set of sequences (often just two). In the next section, we describe

about the longest common Subsequence algorithm.

The second objective of this component is computing a recommendation set for the current session, consisting of links to pages that the user may want to visit based on similar usage patterns. The recommendation set essentially represents a "short-term" view of potentially useful links based on the user's navigational activity through the site. These recommended links are then added to the last page in the session accessed by the user before that page is sent to the user browser.

a) Longest Common Subsequence

The problem of comparing two sequences $\vec{\alpha}$ and $\vec{\beta}$ to determine their similarity is one of the fundamental problems in pattern matching. One of the basic forms of the problem is to determine the longest common subsequence (LCS) of $\vec{\alpha}$ and $\vec{\beta}$. The LCS string comparison metric measures the subsequence of maximal length common to both sequences [14].

Formally, given a sequence $\vec{\alpha} = \langle \alpha_1, \alpha_2, \dots, \alpha_n \rangle$, another sequence $\vec{\gamma} = \langle \gamma_1, \gamma_2, \dots, \gamma_n \rangle$ is a subsequence of $\vec{\alpha}$ if there exists a strictly increasing sequence $\langle j_1, j_2, \dots, j_n \rangle$ of indices of $\vec{\alpha}$ such that for all $i=1,2,\dots,l$, we have $\alpha_{j_i} = \gamma_i$. Given two sequence $\vec{\alpha}$ and $\vec{\beta}$, we say that $\vec{\gamma}$ is common subsequence of $\vec{\alpha}$ and $\vec{\beta}$ if $\vec{\gamma}$ is a subsequence of both $\vec{\alpha}$ and $\vec{\beta}$. We are interested in finding the maximum-length or longest common subsequence (LCS) given two paths or sequence of page-visits $\vec{\alpha} = \langle \alpha_1, \alpha_2, \dots, \alpha_n \rangle$, $\vec{\beta} = \langle \beta_1, \beta_2, \dots, \beta_m \rangle$.

The LCS has a well-studied optimal sub-structure property as given by the following:

Theorem 1: Let $\vec{\alpha} = \langle \alpha_1, \alpha_2, \dots, \alpha_n \rangle$ and $\vec{\beta} = \langle \beta_1, \beta_2, \dots, \beta_m \rangle$ be sequences, and let $\vec{\gamma} = \langle \gamma_1, \gamma_2, \dots, \gamma_n \rangle$ be any LCS of $\vec{\alpha}$ and $\vec{\beta}$.

If $\alpha_n = \beta_m$, then $\gamma_n = \alpha_n = \beta_m$ and $\vec{\gamma}_{l-1}$ is a LCS of $\vec{\alpha}_{n-1}$ and $\vec{\beta}_{m-1}$.

If $\alpha_n \neq \beta_m$, then $\gamma_n \neq \alpha_n$ implies $\vec{\gamma}$ is a LCS of $\vec{\alpha}_{n-1}$ and $\vec{\beta}$.

If $\alpha_n \neq \beta_m$, then $\gamma_n \neq \beta_m$ implies $\vec{\gamma}$ is a LCS of $\vec{\alpha}$ and $\vec{\beta}_{m-1}$.

where $\vec{\alpha}_{n-1} = \langle \alpha_1, \alpha_2, \dots, \alpha_{n-1} \rangle$, $\vec{\beta}_{m-1} = \langle \beta_1, \beta_2, \dots, \beta_{m-1} \rangle$ and $\vec{\gamma}_{l-1} = \langle \gamma_1, \gamma_2, \dots, \gamma_{n-1} \rangle$.

The proof of this theorem can be found in [15]. Efficient recursive algorithms to compute the LCS exist using this property of the LCS [16]. We shall not go into the details of the algorithms since a lot of literature already exists on the topic [16-18].

Definition 2: Let $s1$ and $s2$ be two sequences. $|LCS(s1, s2)|$ is the size of the longest common subsequence between $s1$ and $s2$. The degree of similarity between $s1$ and $s2$ is defined as (4):

$$\text{Sim}_{LCS} = \frac{2 \times |LCS(s1, s2)|}{|s1| + |s2|} \quad (4)$$

In the next we introduce recommendation algorithm through LCS.

b) Recommendation Algorithm with LCS

Pattern search algorithm can be utilized to find navigation patterns based on the current user activities to foresee and recommend user future's request. We apply a pattern search algorithm namely Longest Common Subsequences (LCS) in the recommendation part of the system. In this paper, there are several steps to create recommendations set based on the current user's session in the online phase of the system.

1. *Data Pretreatment for recommendation.* Preprocessing for both current active session and navigation patterns is done in the first step of the prediction engine. What the prediction engine performs in this step is preparing data for applying LCS algorithm with take into account the efficiency of the algorithm.

In this study, the current active session S is represented as a vector:

$$\vec{S} = \langle P1, P2, \dots, Pm \rangle$$

where $P_i = n$, and n is a unique numeric value that we assigned to each web pages in the offline phase. If user visits a web page, the system replaces it with a predefined unique numeric value.

Fix-size sliding window is utilized over the current active session to capture the current user's activities. We call this sliding window, the user's active session window. We consider the mean of web pages in each session of dataset as user's active session window.

Furthermore, in the proposed algorithm, web pages inside the user's active session window have to put in order according to the numeric values.

Here, we present each cluster that created in the offline phase as set of navigation patterns.

$$n\vec{p} = \langle n\vec{p}1, n\vec{p}2, \dots, n\vec{p}m \rangle$$

where $n\vec{p}i$, is a set of k web pages as a navigation pattern and show as below :

$$n\vec{p}i = \langle P1, P2, \dots, Pk \rangle$$

where $1 \leq i \leq n$, and pi is a web pages in a navigation pattern. Moreover, $n\vec{p}i$ have to put in order as same as user's active session window.

2. *User classifying based on LCS algorithm.* There are two sets, navigation patterns $n\vec{p}i$ and active session window \vec{S} as input of this step. Classifying algorithm attempts to find a navigation pattern (Cluster) by utilizing longest common subsequences algorithm. A navigation pattern with the highest degree of similarity is found according to the LCS algorithm to foresee next user's activities and create a recommendation set.

3. *Foresee user next's intention.* In this step of the algorithm, a set of web pages shows to the user as recommendation set. A recommendation engine attempts to show a set of web pages to the current user after the system finds a navigation pattern with the highest degree of similarity. Meanwhile, web pages in the recommendation set are ranked in terms of degree of connectivity between web pages in adjacency matrix M that created in the offline phase. Moreover, for improving the quality of recommendation, the recommendation engine shows only the web pages by highest degree of connectivity and the rest of web pages not consider in the recommendation set. The new recommendation set is created by next user's movement in the web site. In this case, after each user's activity, new user session window is created.

4 System Evaluation

A variety of techniques are used to measure the performance of the recommendation systems. Some experimentations have been done for characterize the quality of the recommendation. In order to evaluate effectiveness of the proposed system several tests should be conducted for both online phase and offline phase.

The system effectiveness is evaluated by using some parameters that we utilize them in offline phase and online phase. The quality of the clusters produced by navigation patterns mining module is evaluated in the offline phase by a parameter namely visit-coherence that introduced by [19]. They introduce a hypothesis that users behave coherently during their navigation, i.e. pages within the same session are in general conceptually related. In this study, we use this concept to obtain a measure of quality for the proposed system.

Definition 3: Visit-coherence measures the percentage of the web pages inside a user session which belong to the cluster representing the session considered.

Visit-coherence is utilized to evaluate the quality of the clusters (navigation pattern) produced by the offline phase. Furthermore, visit-coherence quantifies a session intrinsic coherence. As in the PageGather system [19], the basic assumption here is that the coherence hypotheses hold for every session.

To evaluate the visit-coherence, we split dataset up into two halves after the pretreatment phase. The clustering task applies on the first half and recommendation engine acts on the second half to create the recommendation and visit-coherence is evaluated based on this half. Moreover, second half of the dataset named as evaluation dataset. In this study, parameter β is defined to measure the number of web pages in the session i that belong to a navigation patterns (Cluster) found for that session.

$$\beta i = \frac{|\{p \in Si \mid p \in npi\}|}{Ni} \quad (5)$$

where p is a page, Si is i -th session, npi is the cluster representing i , and Ni is the number of pages in i -th session. The average value for β over all Ns sessions in the evaluation half of the dataset is shown as:

$$\alpha = \frac{\sum_{i=1}^{Ns} \beta i}{Ns} \quad (6)$$

where α is percentage of the visit-coherence that should be considered for the various range of the *MinFreq*.

Another well-know evaluation parameter that is considered in the clustering-based system is outlier. Clustering-based outlier's detection algorithm is utilized to find outlier.

Definition 4: The percentages of web pages that do not belong to any navigation pattern (cluster) therefore not contribute for the online phase.

The outliers are found according to difference values of the *MinFreq* by applying the clustering algorithm. A great deal of outliers web pages in the clustering phase point out that the method is not efficient.

Measuring the accuracy of the predictions and foreseeing in the recommendation system needs to characterize the quality of the results obtained. As we described evaluation method in the offline phase, to measure the quality of recommendation, we use second half of the dataset after the dataset divided into two halves; training set and evaluation set. Each navigational pattern npi (a session in the dataset) in the evaluation set is divided into two parts. The first n pageviews in npi are used for generating predictions, whereas, the remaining part of npi is used to evaluate the generated predictions. The active session window is the part of the user's navigational patterns used by the prediction engine in order to produce a prediction set. We call this part of the navigational pattern np the active session with respect to np , denoted by as_{np} . The prediction engine takes as_{np} and a recommendation threshold τ as inputs and produces a set of pageviews as a prediction list. Recommendation threshold τ is the *MinFreq*. We denote this prediction set by $P(as_{np}, \tau)$. The set of pageviews $P(as_{np}, \tau)$ can now be compared with the remaining $|np| - n$, pageviews in np . We denote this part of np by $eval_{np}$. Our comparison of these sets is based on 3 different metrics, namely, *accuracy*, *coverage* and *F1 measure*. The *accuracy* of prediction set is defined as:

$$Accuracy(P(as_{np}, \tau)) = \frac{|P(as_{np}, \tau) \cap eval_{np}|}{|P(as_{np}, \tau)|} \quad (7)$$

where, $|P(as_{np}, \tau) \cap eval_{np}|$ is number of web pages that these are common in the prediction list and evaluation set. *Accuracy* is Number of relevant web pages retrieved divide by the total number of web pages in recommendations set.

In the other hand, *accuracy* measures the degree to which the prediction engine produces accurate recommendations. Another evaluation parameter in the online phase is *coverage* that is defined as:

$$Coverage(P(as_{np}, \tau)) = \frac{|P(as_{np}, \tau) \cap eval_{np}|}{eval_{np}} \quad (8)$$

Coverage is number of relevant web pages retrieved divide by the total number of web pages that actually belong to the user sessions. In the other hand, coverage measures the ability of the prediction engine to produce all of the pageviews that are likely to be visited by the user.

The *F1 measure* attains its maximum value when both accuracy and coverage are maximized. Finally, for a given prediction threshold τ , the mean over all navigational pattern in the evaluation set is computed as the overall evaluation score for each measure.

$$F1 = \frac{2 \times Accuracy(P(as_{np}, \tau)) \times Coverage(P(as_{np}, \tau))}{Accuracy(P(as_{np}, \tau)) + Coverage(P(as_{np}, \tau))} \quad (9)$$

All parameters attempt to measure the quality of recommendation in the range of *MinFreq* between 0 and 1.

5 Experimental Evaluation

In order to evaluate the performance of the proposed system, two main experiments have been conducted. In the first experiment, clustering algorithm is utilized for navigation pattern mining. Here, several experiments have been done for evaluating the performance of the offline phase of the system. In the second experiment, prediction of the user's next request has been done by classifying algorithm based on longest common subsequence.

5.1 System Requirement for Experimental

This section provides a summary of datasets that the study use them for experimental evaluation and both the hardware and software requirement to run the proposed system.

All evaluation tests have run on a dual processor Intel® Core™ Duo CPU 2.4 GHz with 3.23 GBytes of RAM, operating system Windows XP. Our implementations have run on .Net framework 2 and VB.net and C#.net have used for coding the proposed system.

For our experiments, it is necessary to use such a dataset that allows us to analyze web log data. Our experiments have been conducted on DePaul University CTI logs file dataset (www.cs.depaul.edu) and MSNBC dataset.

5.2 Web-Log Pretreatment results

Well-known pretreatment algorithms have been done on the initial CTI dataset. The only cleaning step performed on these data was the removal of references to auxiliary files (e.g., image files). No other cleaning or preprocessing has been performed in the first phase. The data are in the original log format used by Microsoft IIS.

The length of the active session window is important to classify current active session in the proposed recommendation system. The average number of web pages

in a session can be used for considering the length of active session. As shown in Figure 3 the percentage of the sessions formed by a predefined number of pages quickly decreases when the minimum number of pages in a session increases. Moreover, for CTI and MSNBC datasets, the average length of a user session is about 3 pages. Since for this value we still have almost half of all the sessions, we choose this value as the minimum length for an active session to be classified.

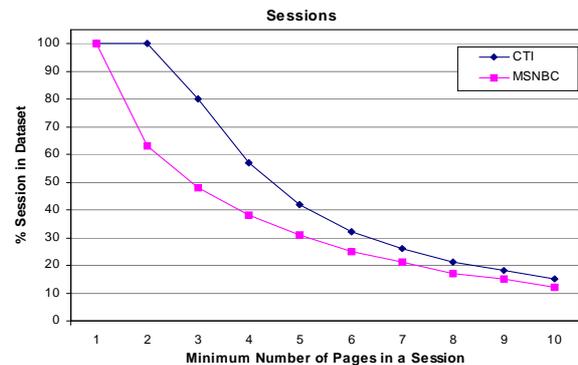


Figure 3: Minimum Number of pages in a session

5.3 Results of navigation patterns Mining

Following the methodology, we create a weighted undirected graph M based on new method to assign weight to the graph. The results of the graph partitioning algorithm on the graph M is a set of navigation pattern as clusters. The weights of the graph for each pair of web pages represent the degree of the connectivity for the web pages. DFS search algorithm has been applied to find the connected components as clusters.

To evaluate the clustering method, the results of the system are compared with method that proposed by [9]. We have done their methodology by own datasets, consequently, the results can be utilized to compare the methods.

Figure 4 plots the number of cluster based on the *MinFreq* for both previous work and proposed method. The Experiment runs on the CTI dataset.

In the proposed method, the number of clusters increases up to *MinFreq*=0.4. Obviously, the system obtains fewer connected components of the graph if we discard more edges of the graph according to the *MinFreq*. Moreover, higher values of the *MinFreq* (*MinFreq*=0.4) create graph with highly disconnected components, thus the clusters found are smaller than *MinClusterSize* and these clusters are discarded. Therefore, the total number of clusters found does not increase.

Figure 5 plots the number of clusters created based on previous work and proposed method for different values of the *MinFreq* for the MSNBC dataset. Similar reasoning can be used to describe behavior of the curves. Specification of the MSNBC dataset shows that the number of web pages

categories is about 20 categories and dataset consist of more than 120,000 session s. In this experiment, we run the clustering algorithm on the sessions that consist of only the categories of the web pages in the MSNBC web site. The details of the each session had not been captured in the dataset.

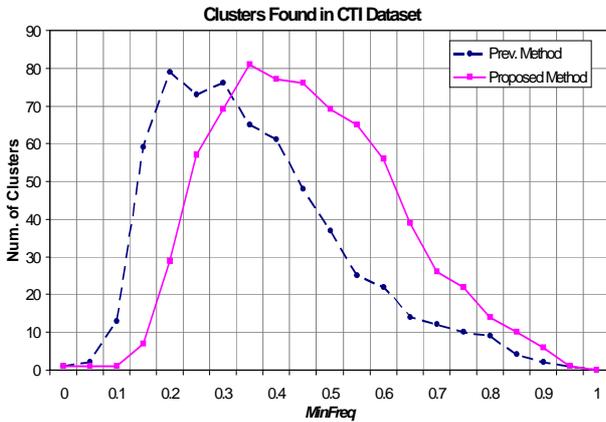


Figure 4: Number of cluster found for CTI dataset

The number of the clusters is not significance enough to evaluate the performance of the clustering. Outliers and visit-coherence are another two main parameters that we measure to evaluate the quality of the clustering.

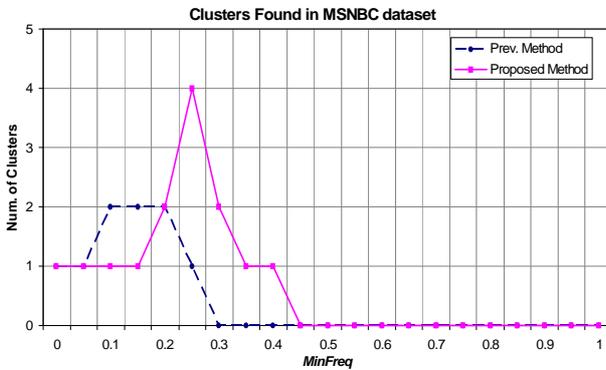


Figure 5: Number of cluster found for MSNBC dataset

Figure 6 shows the percentage of outlier for the CTI and MSNBC datasets. The percentage of outliers increases for higher values of the *MinFreq*. Moreover, higher values of the *MinFreq* create graph with highly disconnected components, thus the size of clusters is decreased in this case, therefore, the number of web pages that do not belong to any cluster increase for higher values of the *MinFreq*.

To measure the quality of clustering, visit-coherence is an evaluation parameter that can be utilized to describe the accuracy of the clustering.

To evaluate the visit coherence, we split the datasets obtained from the pretreatment phase into two halves; apply the clustering based on graph partitioning algorithm on one

half and measure the quality of the clusters based on the second half of the datasets. Moreover, clustering has done based on the previous work and we also apply the datasets on their method.

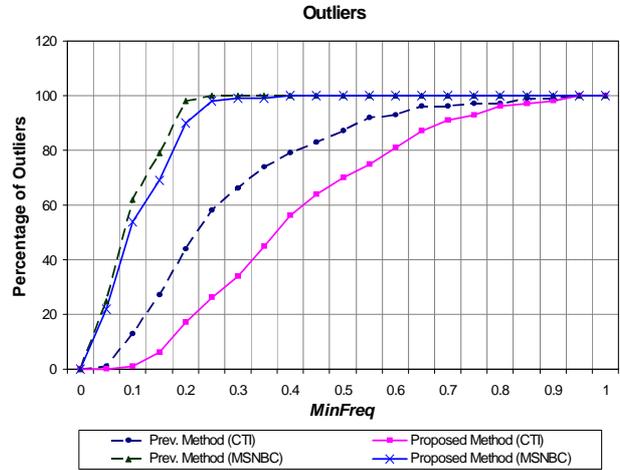


Figure 6: percentage of outliers

Figure 7 plots the visit-coherence for two datasets. Mostly, the percentage of visit-coherence in the proposed system is higher than previous work according to the difference values of the *MinFreq*. Moreover, for the higher values of the *MinFreq* the quality of the clustering to grow less in view of the fact that the system creates fewer number of the clusters for high values of the *MinFreq*. In addition, there are fewer numbers of web pages in connected components for the higher value of *MinFreq*. Furthermore, the clusters consist of a great deal of web pages for small values of the *MinFreq*, consequently, number of web pages that belong to the particular cluster increase in this case.

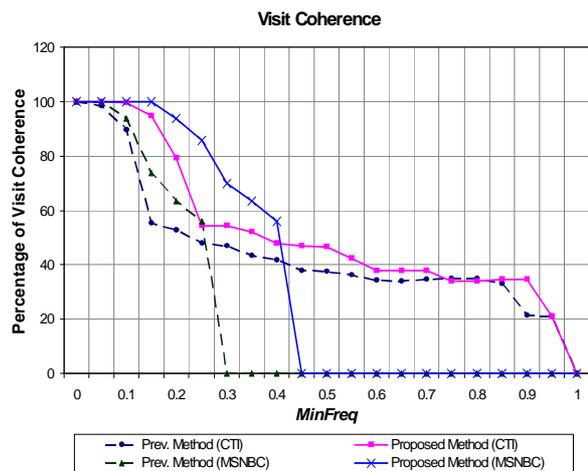


Figure 7: Visit coherence in two dataset

5.4 Evaluation of the recommendations

In this section, we measure the quality of the recommendation generated by prediction engine during the online phase. Sessions found in one half of the both datasets are submitted to the prediction engine to classify current user's activities and to generate recommendation. Subsequently, the overlapping between the generated prediction $P(as_{np}, \tau)$ and the session pages $eval_{np}$ is computed by using expression (8) that introduce in section 3. Finally, the percentage of all predictions figures out as quality of the foreseeing in the proposed recommendation system.

In this study, we evaluate the quality of the recommendations, for the navigation pattern created by proposed method and classifying method based on LCS algorithm versus both clustering and classifying methods by previous algorithms. Three parameters have been measured to verify the quality of the predictions; accuracy, coverage and F1.

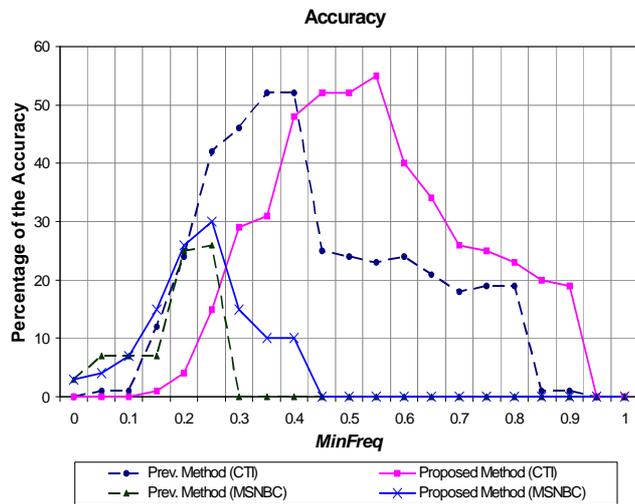


Figure 8: Accuracy of the recommendations

Figure 8 depicts the accuracy of the proposed system for *MinFreq* ranging from 0 to 1 for the CTI and MSNBC datasets. The percentage of the accuracy for the proposed system achieves the best results when we choose the value of the *MinFreq* to be around 0.55 for CTI dataset. The results illustrate the approach can improve the accuracy of the recommendation in a great deal of difference values of the *MinFreq* against the previous work.

Figure 9 depicts the coverage of the proposed system for *MinFreq* ranging from 0 to 1 for the two datasets. The coverage achieves high percentage for the lower values of the *MinFreq*. In this case, for the lower values of the *MinFreq* by classifying current user's activities, the prediction engine can find the clusters with large size that many web pages inside the cluster belong to the actual user's activities. Moreover, similar reasoning can be used for the

higher values of the *MinFreq* that decrease the percentage of the coverage. The results illustrate the approach can improve coverage of the recommendation system in comparison with previous work.

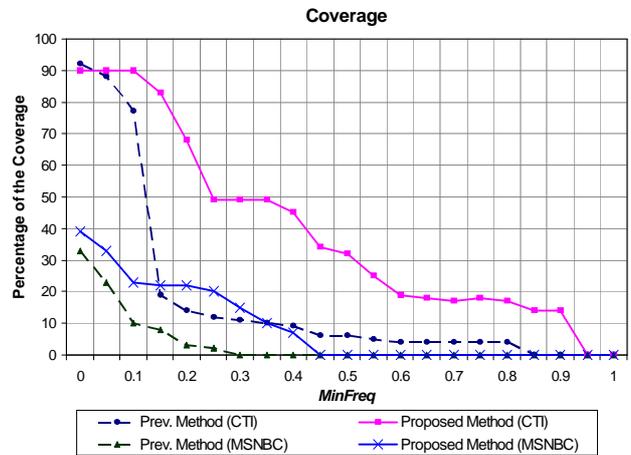


Figure 9: Coverage of the recommendations

Figure 10 depict the F1 measure for *MinFreq* ranging from 0 to 1 for the CTI and MSNBC datasets. In the most time, F1 achieves higher percentages for the proposed system than the previous work.

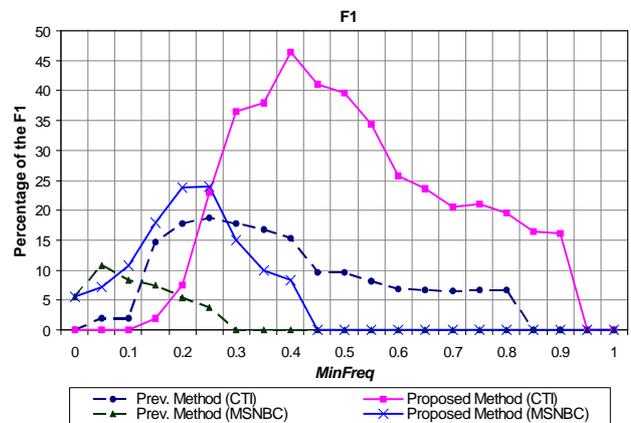


Figure 10: F1 measure of the recommendations

For all experiments, we run paired t-test with 95% confidence. Paired t-test is used to compare means on the same or related subject over time or in the differing circumstances. In the other hand, the t-test assesses whether the means of two groups are statistically different from each other. This analysis is appropriate whenever you want to compare the means of two groups. The assumption for using the paired t-test is that the observed data are from the same subject or from a matched subject and are drawn from a population with a normal distribution. In this study, a paired t-test is carried out to compare the experimental results for the F1 measure. The mean of F1 measure for the CTI dataset

is 19.5 for the proposed approach and 7.9 for the previous work. The two-tail p -value for the paired t-test is 0.0006 that achieves significant value (p -value \leq 0.05). Moreover, the mean of F1 measure for the MSNBC dataset is 6 for the proposed approach and 2 for the previous work. The two-tail p -value for the paired t-test is 0.009 that achieves significant value (p -value \leq 0.05). The implication of this result is that the proposed approach is more accurate than the previous work.

Based on experimental results, it indicates that our approach to classify current user's activities to foresee user's next request can improve the quality of the predictions in the web usage mining recommendation systems.

6 Conclusions and Future Work

In this paper, we advanced a web usage mining model and proposed a novel approach to classify the user navigation pattern for online foreseeing users' future intentions through mining of web server logs. We also utilized a graph partitioning algorithm to model user navigation patterns. In order to mining user navigation patterns, we established an undirected graph based on connectivity between each pair of the web pages. Moreover, we proposed a novel formula for assigning weights to edges of the undirected graph. To classify current user activities, we used longest common subsequences algorithm to foresee user near future movement. We used some evaluation methodologies that can evaluate the quality of the clusters found and quality of the recommendations. The experimental results show that our approach can improve the quality of clustering for user the navigation pattern and quality of recommendation for both CTI and MSNBC datasets.

There are some aspects in that can be improved in our system. For instance, we can take into account the semantic knowledge about underlying domain to improve the quality of the recommendation. In the other hand, integrating semantic web and web usage mining can achieve best recommendations in the dynamic huge web sites. Moreover, we would perfect the system and let it serves for actual users to the best of its abilities.

REFERENCES

- [1] R. Cooley, B. Mobasher, and J. Srivastava, "Web mining: information and pattern discovery on the World Wide Web," pp. 558-567, 1997.
- [2] J. Wang and I. NetLibrary, *Encyclopedia of Data Warehousing and Mining*: Idea group Reference, 2006.
- [3] J. Srivastava, R. Cooley, M. Deshpande, and P. N. Tan, "Web usage mining: discovery and applications of usage patterns from Web data," *ACM SIGKDD Explorations Newsletter*, vol. 1, pp. 12-23, 2000.
- [4] T. W. Yan, M. Jacobsen, H. Garcia-Molina, and U. Dayal, "From user access patterns to dynamic hypertext linking," *Computer Networks and ISDN Systems*, vol. 28, pp. 1007-1014, 1996.
- [5] B. Mobasher, R. Cooley, and J. Srivastava, "Automatic personalization based on Web usage mining," *Communications of the ACM*, vol. 43, pp. 142-151, 2000.
- [6] M. Nakagawa and B. Mobasher, "A hybrid web personalization model based on site connectivity," *The Fifth WEBKDD Workshop*, pp. 59-70, 2003.
- [7] M. Perkowitz and O. Etzioni, "Towards adaptive Web sites: Conceptual framework and case study," *Artificial Intelligence*, vol. 118, pp. 245-275, 2000.
- [8] H. Liu and V. Kešelj, "Combined mining of Web server logs and web contents for classifying user navigation patterns and predicting users' future requests," *Data & Knowledge Engineering*, vol. 61, pp. 304-330, 2007.
- [9] R. Baraglia and F. Silvestri, "Dynamic personalization of web sites without user intervention," *Communications of the ACM*, vol. 50, pp. 63-67, 2007.
- [10] R. Baraglia and F. Silvestri, "An Online Recommender System for Large Web Sites," *IEEE/WIC/ACM International Conference on Web*, pp.199-205, 2004.
- [11] R. Cooley, B. Mobasher, and J. Srivastava, "Data Preparation for Mining World Wide Web Browsing Patterns," *Knowledge and Information Systems*, vol. 1, pp. 5-32, 1999.
- [12] D. Tanasa and B. Trousse, "Advanced data preprocessing for intersites Web usage mining," *Intelligent Systems, IEEE*, vol. 19, pp. 59-65, 2004.
- [13] M. Perkowitz and O. Etzioni, "Adaptive Web sites," *Communications of the ACM*, vol. 43, pp. 152-158, 2000.
- [14] A. Apostolico, "String editing and longest common subsequences," *Handbook of Formal Languages*, vol. 2, pp. 361-398, 1997.
- [15] T. H. Gormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Introduction to Algorithms," *MIT Press*, vol. 44, pp. 97,138, 1990.
- [16] V. Dancik, *Expected Length of Longest Common Subsequences*: University of Warwick, 1994.
- [17] A. V. Aho, D. S. Hirschberg, and J. D. Ullman, "Bounds on the complexity of the longest common subsequence problem," 1974.
- [18] D. S. Hirschberg, "Algorithms for the Longest Common Subsequence Problem," *Journal of the ACM (JACM)*, vol. 24, pp. 664-675, 1977.
- [19] M. Perkowitz and O. Etzioni, "Adaptive Web Sites: Conceptual Cluster Mining," *International Joint Conference on Artificial Intelligence*, pp. 264-269, 1999.

Algebraic Algorithms to Solve Name Disambiguation Problem

Ingyu Lee¹, Byung-Won On², and Seong No Yoon³

¹Sorrell College of Business, Troy University, Troy, AL., USA

²Department of Computer Science, University of British Columbia, Vancouver, BC., Canada

³College of Business Administration, Savannah State University, Savannah, GA., USA

Abstract—When we are looking for information about a specific person, we type the name in search engines. Then, search engines return many web pages which include the given name strings. However, the resulting web pages are mixed with related information and unrelated information. This is called a name disambiguation problem. Current search engines provide links but do not distinguish related and unrelated web pages. In this paper, we described our algorithm to solve a name disambiguation problem using two linear algebraic approaches: Singular Valued Decomposition (SVD) and Nonnegative Matrix Factorization (NMF). Experiments show that using NMF algorithm yields the best performance in terms of precision, recall and Fmeasure, and applying SVD requires only 10% computation time compared to traditional K-means algorithm. Our solution with search engines will provide more precise search results than traditional search engine services.

Keywords: Data mining, Text mining, Clustering, SVD, NMF

1. Introduction

Internet allows people to work with or to get information about someone whom we never met before in real life. Search engines are used to find information about persons we are interested in. When we are looking for information about a specific person, we type the name string in search engines. Then, search engines return many web pages including the name strings. However, some web pages include the name string by coincidence but do not contain related information. Current search engines do not distinguish between web pages described in the latter. For example, if we are looking for *Tom Mitchell* from Carnegie Mellon University, then we type the name in Google search engine. Google search engine returns web pages including *Tom Mitchell*. There are 37 different *Tom Mitchell* within the first 100 returning web pages [1]. Only 57 documents hold information about *Tom Mitchell* from Carnegie Mellon University and all the other documents hold information for different *Tom Mitchell*. We call this a name disambiguation problem.

One approach to solve the problem is providing web pages as groups of related information. Since search engines return many related and unrelated web pages, users might be misled by bunch of unrelated web pages. To prevent this, search engines cluster related web pages and provide clustered information to users. The latter will reduce chances

that leading users to unrelated web pages. Therefore, clustering algorithms such as K-means, hierarchical clustering and graph partitioning have been used to solve a name disambiguation problem [2], [3], [4], [5].

At the same time, linear algebraic approaches have been used in information retrieval applications [6], [7]. Especially, Singular Value Decomposition and Nonnegative Matrix Factorization (NMF) are popularly used in text and data mining applications. When dimensions become large, applying SVD and NMF could significantly reduce dimension with a slight loss of original information. In this paper, we applied these two algorithms to solve a name disambiguation problem and compared performance with a regular K-means algorithm. Experiments show that SVD reduces clustering time and NMF improves performance. Our solution with search engines provide more precise search results than traditional search engine services with a slight overhead.

The rest of the paper consists of the followings. Linear algebra algorithmic backgrounds are described in Section 2. Characteristics of a name disambiguation problem and metrics we are using to measure performance are described in Section 3. Experimental results with sample data name set using Singular Value Decomposition (SVD) and Nonnegative Matrix Factorization (NMF) are described in Section 4. Related researches are described in Section 5. Re-ranking clustered results are described in Section 6. Concluding remarks and future plans are followed in Section 7.

2. Linear Algebraic Algorithms

Vector Space Model (VSM) has been used in information retrieval applications to represent text documents [6], [7]. Assume we have n documents corpus and we want to represent each document using m terminologies. Then, we make n textual documents into n document vectors d_1, d_2, \dots, d_n where each document vector has m terminologies. Therefore, the term-by-document matrix A is represented as

$$A_{m \times n} = [d_1 | d_2 | \dots | d_n] \quad (1)$$

where columns are document vectors and rows are terminologies as shown in Figure 1.

To retrieve relevant information from this term-document matrix, we create query vector q and find the document d which is the closest to a query q . Euclidean distance or cosine similarity are used to measure distances between a

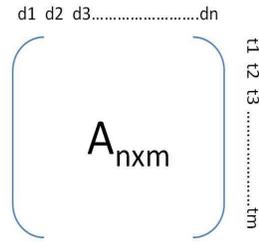


Fig. 1: Term-Document Matrix

query q and a document d defined as

$$Euclidean_Distance(d, q) = \sqrt{\sum_{k=1, \dots, m} (d_k - q_k)^2} \quad (2)$$

and

$$\cos \theta = \frac{q^T d}{\|q\|_2 \|d\|_2} \quad (3)$$

, respectively.

2.1 K-means Clustering

Assume we have a $m \times n$ term-document matrix A whose m rows represent terminologies and n columns represent documents as defined in the previous section. We wish to cluster n documents into k disjoint clusters C_1, \dots, C_k . Then, the objective function of K-means algorithm is defined as

$$\min \sum_{i=1, \dots, k} \sum_{d_j \in C_i} (d_j - m_i)^2 \quad (4)$$

where d_j is a document column set from A and m_i is a centroid of cluster C_i for $i = 1, \dots, k$.

K-means algorithm works the following way.

- 1) Choose random k cluster centers (centroid) $m_j, j = 1, \dots, k$ from column set of document vectors.
- 2) Assign each document column to the closest clusters $C_j, j = 1, \dots, k$ using Euclidean distance or cosine similarity as defined in the previous section.
- 3) Compute new centroid, $m_j, j = 1, \dots, k$ which are means of all points in C_j defined as

$$m_j = \frac{1}{|C_j|} \sum_{t=1, \dots, |C_j|} d_t \quad (5)$$

- 4) Repeat until changes in clusters C happen.

K-means algorithm is the most popular to cluster data sets because of its simplicity. In addition, the quality of K-means algorithm is known as good for general clustering problems. However, the convergence of the algorithm depends on initial centroid selection. If initial centroid is located far from the

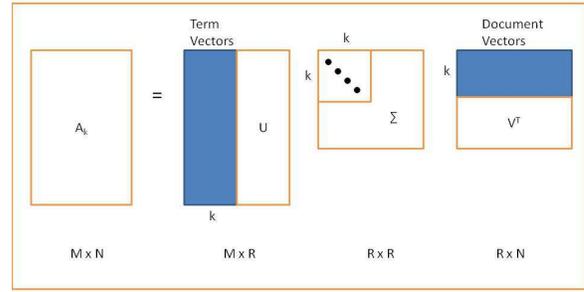


Fig. 2: Singular Value Decomposition

true solution set, then K-means requires lots of iteration to converge. Details of K-means and variants algorithms are found in [8], [9].

2.2 Singular Value Decomposition

Singular value decomposition (SVD) has been used to solve linear least squares problems, matrix rank estimation, and correlation analysis [6], [10], [11]. Given a $m \times n$ matrix A , where $m \geq n$, the singular value decomposition of matrix A is defined as

$$A = U \Sigma V^T \quad (6)$$

where U and V are orthogonal matrices which satisfy $U^T U = V^T V = I$, and Σ is a diagonal matrix whose diagonal elements are $diag(\sigma_1, \dots, \sigma_n)$.

If we have a term-document matrix A whose columns represent documents holding the terminology on each row, then matrix A is generally large and sparse in real applications. Applying K-means on this large and sparse matrix A requires lots of time and memory space. To overcome the latter, SVD has been used to represent documents with reduced number of terminologies which best represents the original matrix A .

Dimension reduced matrix A_k is noted as

$$A_k = U_k \Sigma_k V_k^T \quad (7)$$

where U_k and V_k are orthogonal matrices on reduced dimension, and Σ_k is a diagonal matrix with k singular values. In the reduced domain, V_k represents the projection of the documents into k terminology spaces. Figure 2 shows the outline result of SVD factorization. Matrix A is factorized into three matrices. Columns of the leftmost matrix U are basis vectors for terminology and columns of the rightmost matrix V represent documents with a combination of basis vectors. The diagonal matrix value represents scaling factors of each basis vector. Matrix V is a projection of our matrix A in terms of terminologies we selected as shown in Figure 2.

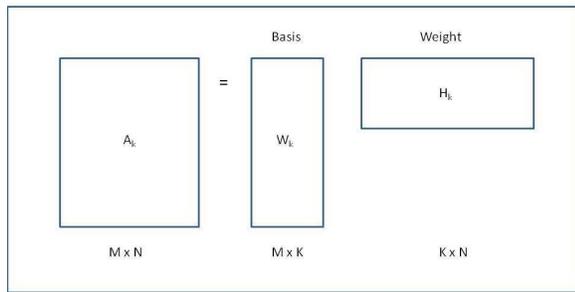


Fig. 3: Nonnegative Matrix Factorization

Using A_k in place of A improves performance by reducing dimension. A_k considers only essential components of term-by-document matrices. It also filters out the noise, and uses best rank- k approximation. The drawbacks are U_k and V_k are completely dense matrices which requires $(m+n)k$ memory space. Furthermore, truncation point k is hard to determine, and interpretation of basis vectors u_i and v_i is difficult due to mixed signs. Details of algorithms and characteristics are described in [6], [10], [11].

2.3 Nonnegative Matrix Factorization

Nonnegative Matrix Factorization (NMF) has been used in data mining applications [12], [13] since negative values in SVD matrices are hard to understand the meaning. Unlike SVD factorization, Non-negative Matrix Factorizations (NMF) uses low-rank approximation with nonnegative factors. The following shows the outline of NMF algorithm.

$$A_k = W_k H_k \quad (8)$$

where columns of W are the underlying basis vectors (i.e. each n column of A can be built from k columns of W) and columns of H give the weights associated with each basis vector such as

$$A_k e_1 = W_k H_{*1} = [w_1] h_{11} + [w_2] h_{21} + \dots + [w_k] h_{k1} \quad (9)$$

where w_i is a basis and h_{ji} is a contribution weight for each basis $w_j, j = 1, \dots, k$. The outline of the algorithm is shown in Figure 3.

Since w_i s are not orthogonal, basis vectors could overlap topics. Unlike SVD, W and H can also be restricted as sparse matrices. The value w_{ij} means the contribution of term j in basis vector w_i . The value h_{i1} shows how much doc_1 is pointing in the direction of topic vector w_i . Since W and H are not unique, there are several different algorithms to factorize into W and H .

Lee and Seung applied NMF algorithm to image processing applications in [14], [15]. Berry used NMF algorithms

```
function [W, H] = NMF(A)
% initialize nonnegative random matrix, W
W = abs(randn(m,k));
% initialize nonnegative random matrix, H
H = abs(randn(k,n));
for i=1 to maxiter do
% Solve for H
H = (W' * W + eye(k) .* lambda) \ (W' * A);
% Keep positive only
H = H .* (H >= 0);
% Solve for W
W^T = (H * H^T + eye(k) .* lambda) \ (H * A^T);
% Keep positive only
W = W .* (W >= 0);
end for
```

Fig. 4: Nonnegative Matrix Factorization

with sparse constraints on matrix H in his paper [12], [13], and Paatero and Tapper proposed an alternative least squares NMF algorithm which is known as faster and simpler in their paper [16]. NMF has great interpretability, and comparable performance to SVD. Furthermore, sparsity constraints during factorization also allow significant storage savings. It is also scalable and possibly faster than SVD in parallel environments. However, the factorization is not unique and depends on algorithms and parameters. When a new dimension is added, it is unable to reduce the size of the basis without recomputing NMF. Figure 4 shows an alternative least squares NMF algorithm we used in this paper. In the algorithm, λ is a sparse constraints factor for matrix H .

3. Problem Settings and Metrics

For the evaluation of our algorithms, we used name data set from Bekkerman [1]. It has twelve personal names and each name set has different number of categories. Table 1 and Figure 5 shows the statistics of our dataset. Table 1 shows that each name set has variants number of categories. For *Adam Cheyer* and *Leslie Park Kaebbling* has 2 categories but *Tom Mitchell* has 37 different categories. In Figure 5, each color represent different clusters and sizes of bar show the number of documents belong to each cluster. The naming data set is extremely skewed in terms of cluster size. For *Adam Cheyer* and *Leslie Park Kaebbling*, all documents except one document belong to the same class. Others such as *Andrew Ng* and *Voss*, one or two classes dominate the whole data set. This skewed characteristic makes it difficult to use traditional clustering algorithm.

The first step for experiments is building *term-document* matrix A from the name data set. We used TMG [17] software to generate the *term-document* matrix with normalization and stemming options. We also removed common words from the term lists. We used *term frequency* for

Name	Pages	Categories
Adam Cheyer	97	2
William Cohen	88	10
Steve Hardt	81	6
David Israel	92	19
Leslie Pack Kaebbling	89	2
Bill Mark	94	8
Andrew McCallum	94	16
Tom Mitchell	92	37
David Mulford	94	13
Andrew Ng	87	31
Fernando Pereira	88	19
Lynn Voss	89	26
Total	1085	187

Table 1: Dataset Statistics. Each name has different number of documents and different number of categories.

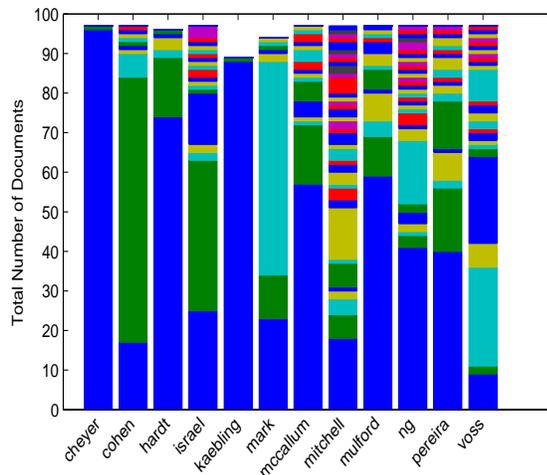


Fig. 5: Statistics of Problem Set: Name sets are extremely skewed and one class dominates the whole document set.

local term weighting and *inverse document frequency* for global term weighting, respectively. Each nonzero elements in $A(i, j)$ is defined as

$$A(i, j) = TF_{i,j} \times IDF_{i,j} \quad (10)$$

where TF is *term frequency* and IDF is *inverse document frequency*. We also normalized each column of A . After built *term-document* matrix A , we used Singular Value Decomposition (SVD) and Nonnegative Matrix Factorization (NMF) to reduce dimensions and project web documents on the reduced terminology space.

The first metric we considered is the accuracy of clustering algorithm. We defined the precision, recall and Fmeasure as

in equations

$$Precision_i = \frac{CorrectlyPredict_i}{Predict_i} \quad (11)$$

$$Recall_i = \frac{CorrectlyPredict_i}{Cluster_i} \quad (12)$$

$$Fmeasure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (13)$$

where $Predict_i$ is the results of clustering algorithm and $Cluster_i$ is a manually generated solution set. Precision is defined as the number of correctly clustered documents divided by the number of documents in the cluster. Recall is defined as the number of correctly clustered documents divided by the number documents in manually generated solution set. Fmeasure is an arithmetic mean of precision and recall.

Since the original name set is extremely skewed, the precision and recall could be high even quality of clustering results is not good. For example, *Cheyre*, one class has 95 documents out of 96 documents and the second class has only 1 document. The precision is 99% no matter which clustering algorithms are used. To distinguish performance between clustering algorithms, we also applied RAND index which has been used in statistics community to compare two clustering results [18]. Given a set of n documents $D = \{d_1, \dots, d_n\}$, we wish to compare two clusters: C and S . The resulting cluster of our algorithm is defined as $C = \{C_1, \dots, C_k\}$ and manually derived solution set is defined as cluster $S = \{S_1, \dots, S_k\}$. Then, RAND index is defined as

$$RAND = \frac{a + b}{a + b + c + d} \quad (14)$$

where a is the number of pairs of elements in D that are in the same set in C and the same set in S , b is the number of pairs of elements in D that are in different sets in C and in different sets in S , c is the number of pairs of elements in D that are in the same set in C and in different sets in S , and d is the number of pairs of elements in D that are in different sets in C and in different sets in S . Intuitively, $a + b$ is the number of agreements between C and S , and $c + d$ is the number of disagreements between C and S . RAND index has a value between 0 and 1 with 0 indicating that two data clusters do not agree on any pair of points and 1 indicating that the data clusters are exactly the same.

4. Experimental results

K-means algorithm use centroids and medoids. Centroid is a computed center as defined in the previous chapter and medoid is a member of cluster which is closest to the computed centroid. Using medoids is known as more efficient than using centroids [19]. Figure 6 shows the performance of two methods. We averaged results of 100 experiments for each name data set. The x-axis shows different metrics

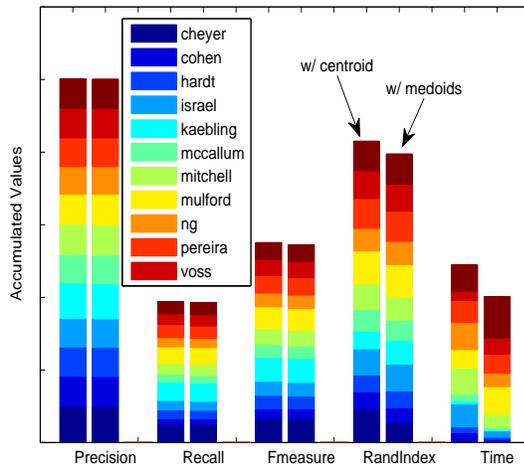


Fig. 6: Performance of two different approach: centroids vs. medoids.

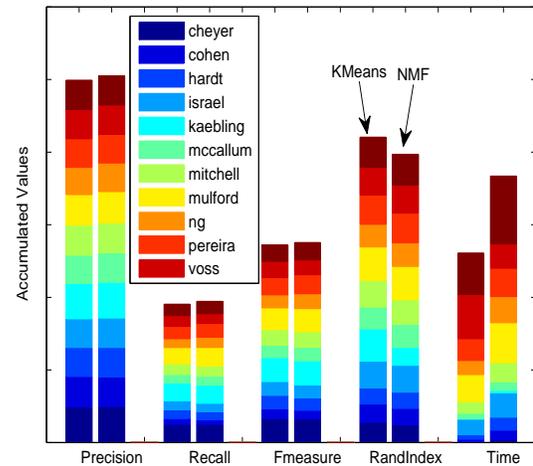


Fig. 8: Performance results of K-means and NMF.

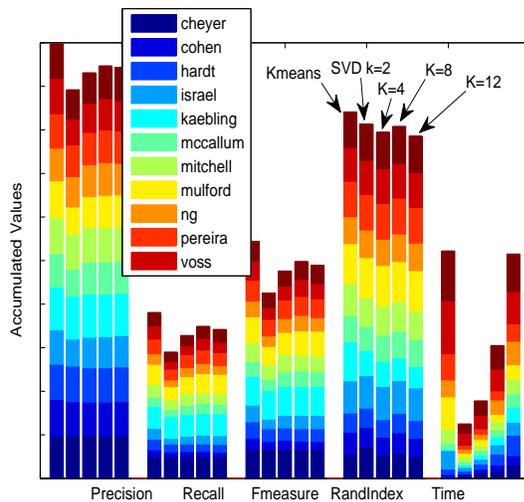


Fig. 7: Performance results of K-means after dimension reduction.

and the y-axis are accumulated value of all name sets. Two methods show similar performance in Fmeasure but using centroids shows a little bit better performance in Rand index. However, using medoids converges faster than using centroids. In this paper, we used medoids in our K-means clustering algorithm rather than using centroids since it shows similar Fmeasure with less computation time.

Figure 7 shows the performance comparison of K-means algorithm and K-means on reduced matrix when we use $k=2,4,8,12$. We experimented 100 times and computed averages for each name data set. We achieved 90% of performance in terms of Fmeasure and Rand index with

10% execution time when we use $k=2$. Bigger k values improve performance slightly better but require more time to converge. The experimental results show that only several principal terminologies are enough to define documents in name data sets.

To measure NMF clustering algorithm, we used an alternative updating algorithm as shown in Figure 4 with λ value 0.02. The factorization iteration stops when the difference between the previous norm of H and the current norm of H is smaller than 0.01. After factorizing the matrix A into WH , we searched the biggest component from column vector H_i . Then, we assign the document with the biggest component cluster. Figure 8 shows the results of comparing performance of two methods. Using NMF shows slightly better performance in terms of precision, recall and Fmeasure but K-means shows better performance in Rand index and time. Considering the size of testing data set is small and using sparse matrix format for term-document matrix, K-means shows faster performance is not surprising.

To compare SVD and NMF with the same k dimension number, we applied K-means algorithm after reducing dimension to $m \times k$ with SVD and compared the results with NMF algorithm. The results are shown in Figure 9. NMF shows slightly better performance in precision, recall, Fmeasure and Rand index. It also much faster than SVD to converge. In addition, considering parallelizing SVD is much harder than NMF, our experiments show that NMF is a better choice if we are using the same k value. If data sets are large in parallel environment, NMF is the better algorithm but SVD with lower k gives benefits in execution time.

Experiments show that extremely skewed name data sets are hard to cluster using regular clustering algorithm. However, SVD could help in reducing dimensions with a

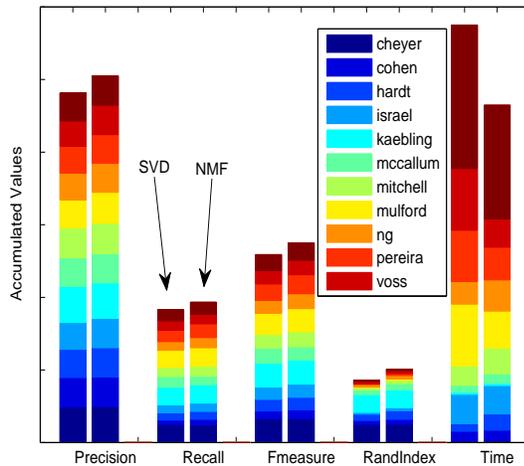


Fig. 9: Performance results SVD with K-means and NMF.

little loss of quality and NMF shows the best performance. Therefore, for time limited applications, applying SVD is helpful but NMF algorithm generates a better performance in general. Especially, considering parallelizing NMF is relatively easier than SVD, NMF is the algorithm in parallel environment.

5. Related Work

Bekkerman et. al. proposed two algorithms to disambiguate web appearances of people in a social network in their paper [3]. One is based on link structure of web pages, another using multi-way distributional clustering method. Their algorithms show more than 20% improvement in Fmeasure. Minkov et. al. used lazy graph walk algorithm to disambiguate names in email documents in their paper [4]. They provided a framework for email data, where content, social networks and a timeline to integrated in a structured graph. Banerjee et. al. proposed multi-way clustering on relation graphs in [2]. Different types of entities are simultaneously clustered based not only on their intrinsic attribute values, but also on multiple relations between entities. On and Lee used multi-level graph partitioning methods to provide a scalable name disambiguation solution in their paper [5].

In authors awareness, this is the first paper to provide an algebraic approach in a solving name disambiguation problem. In addition, we analyzed the name disambiguation problem characteristics and used Rand Index to measure the performance of extremely skewed clustering problem. Our experiment results show some promising results in choosing a proper algorithm based on the problem and computational environments.

6. Discussion

As the results of clustering, relevant web pages are clustered to the same group. For instance, there are four different people with the same name spellings, shown in Table 2. In the example, the total number of web pages related to “Tom Mitchell” is 13, where four web pages (i.e., A, B, C, and D), three ones (i.e., E, F, and G), five ones (i.e., H, I, J, X, and Y), and one web page are associated with professor at CMU, reporter at CNN, musician at Shady record company, and minister at Kansas city, respectively. Suppose that web pages are ranked by PageRank scores like the third column in Table 2. The current search result of Google is A, E, J, F, G, I, H, D, Y, B, X, C, and Z. These 13 web pages are clustered to four groups in terms of our clustering scheme, and furthermore we need to re-arrange the four groups in a certain order (like PageRank). Let us denote this process as *re-ranking* of clusters.

This is a considerably challenging issue in this paper. For instance, note the four web pages of “Tom Mitchell” at CMU. Web page A is firstly ranked by Google PageRank algorithm, while the rest pages are mostly located in bottom – B (10th), C (12th), and D (8th). In this case, it is hard to determine which position the cluster (labeled as CMU/Prof) should be rank among the four clusters (labeled as CMU/Prof, CNN/Rep, Shaddy/Mus, and Kansas/Min). To address this problem, we propose an approach based on the hypothesis that *re-using ranking information generated by Google is sufficiently effective*. For this, we consider four different methods as follows:

- Consider *relative ranking positions* of web pages per cluster which is defined as

$$ClusterRank_j = \frac{\sum_{i \in Cluster_j} PageRank_i / N_{doc}}{N_{cluster_j}} \quad (15)$$

where $Cluster_j$ is the number of pages in cluster j and N_{doc} is the total number of pages. For example, take a look at the cluster, named as “CMU/Prof”. We can compute the cluster ranking by

$$\begin{aligned} & \frac{PageRank(A)}{13} + \frac{PageRank(B)}{13} + \frac{PageRank(C)}{13} + \frac{PageRank(D)}{13} \\ &= \frac{\frac{1}{13} + \frac{10}{13} + \frac{12}{13} + \frac{8}{13}}{4} \\ &= \frac{(0.08+0.77+0.92+0.62)}{4} = \frac{2.39}{4} = 0.6 \end{aligned}$$

That is, $ClusterRank(CMU/Prof) = 0.6$. Finally, the clusters are re-ordered by the $ClusterRank()$ scores in the ascending order.

- Consider the *highest ranking position* of web pages per cluster which is defined as

$$ClusterRank_j = \min_{i \in Cluster_j} PageRank_i. \quad (16)$$

For instance, the cluster of “CMU/Prof” is ranked in the first position due to the ranking position of A. In other words, $ClusterRank(CMU/Prof) = 1$. Finally, the

Cluster Label	Web Page ID	PageRank
Tom Mitchell Professor CMU	A	1
	B	10
	C	12
	D	8
Tom Mitchell Reporter CNN	E	2
	F	4
	G	5
Tom Mitchell Musician Shady Records	H	7
	I	6
	J	3
	X	11
	Y	9
Tom Mitchell Minister Kansas City	Z	13

Table 2: An example of re-ranking in our context.

clusters are re-ordered by the $ClusterRank()$ scores in the ascending order.

- Consider the *Top-k ranking positions* of web pages per cluster which is defined as

$$ClusterRank_j = \frac{\sum_{TopK \in Cluster_j} PageRank_{topk} / N_{doc}}{K} \quad (17)$$

where *TopK* is the highest *k* rank values in cluster *j*. For instance, the cluster of “CMU/Prof” has top-2, $PageRank(A)$ and $PageRank(D)$. Then,

$$\begin{aligned} ClusterRank_j &= \frac{\frac{PageRank(A)}{13} + \frac{PageRank(D)}{13}}{2} \\ &= \frac{(0.08 + 0.62)}{2} = 0.35. \end{aligned}$$

Finally, the clusters are re-ordered by the $ClusterRank()$ scores in the ascending order.

- Consider the *median ranking position* of web pages per cluster which is defined as

$$ClusterRank_j = median\{Cluster_j\} \quad (18)$$

As an example, $ClusterRank(Shaby/Mus) = 7$ due to $PageRank(J) = 3$, $PageRank(I) = 6$, $PageRank(H) = 7$, $PageRank(Y) = 9$, and $PageRank(X) = 10$. Finally, the clusters are re-ordered by the $ClusterRank()$ scores in the ascending order.

7. Concluding Remarks

In this paper, we characterized and analyzed a name disambiguation problem. We also applied algebraic approaches to solve a name disambiguation problem using SVD and NMF algorithms. Using SVD showed drastic improvement in terms of execution time. It requires only 10% of execution time if we are using smaller number of *k*. Using NMF shows a little bit better performance in terms of precision, recall, Fmeasure and Rand Index. Considering that NMF is relatively easier to parallelize than other algorithms, NMF is

an algorithm to use in parallel environment with large name data sets.

Current implementation is based on a single machine and it could not handle huge number of data sets. In the future, we are considering to implement our algorithms, especially NMF algorithm, on a clustered system to handle large number of data sets. In addition, several variation metrics are needed to measure performance of extremely skewed clustering problem. We are trying to provide better quality metrics for a name disambiguation problem. We also proposed several re-ranking algorithms of clustered results. However, we are still working on the details of several re-ranking algorithms.

References

- [1] R. Bekkerman, “Name data set,” <http://www.cs.umass.edu/~ronb>.
- [2] A. Banerjee, S. Basu, and S. Merugu, “Multi-way clustering on relation graphs,” in *Proceedings of SIAM Data Mining 2007*, 2007.
- [3] R. Bekkerman and A. McCallum, “Disambiguating web appearances of people in a social network,” in *Proceedings of International World Wide Web Conference Committee*, 2005.
- [4] E. Monkov, W. Cohen, and A. Y. Ng., “Contextual search and name disambiguation in email using graphs,” in *Proceedings of SIGIR*, 2006.
- [5] B. On and D. Lee, “Scalable name disambiguation using multi-level graph partition,” in *Proceedings of SIAM Data Minings*, 2006.
- [6] M. Berry and M. Browne, *Understanding Search Engines*. SIAM, 2005.
- [7] M. Berry, S. Dumais, and G. O'Brien, “Using linear algebra for intelligent information retrieval,” *SIAM Review*, vol. 37, pp. 573 – 595, 1995.
- [8] J. Han, M. Kamber, and A. Tung, *Spatial clustering methods in data mining: A survey*. In *Geographic Data Mining and Knowledge Discovery*. Taylor and Francis, 2001.
- [9] J. MacQueen, “Some methods for classification and analysis of multi-variate observations,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.
- [10] G. Golub and C. V. Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: Johns Hopkins University Press, 1996.
- [11] M. Heath, *Scientific Computing: An Introductory Survey*. New York: McGraw Hill, 2002.
- [12] M. Berry, M. Browne, A. Langville, V. Pauca, and R. Plemmons, “Algorithms and applications for approximate nonnegative matrix factorization,” *Computational Statistics and Data Analysis*, vol. 52, pp. 155 – 173, 2007.
- [13] F. Shahnaz, M. Berry, V. Pauca, and R. Plemmons, “Document clustering using nonnegative matrix factorization,” *Information Processing and Management: an International Journal*, vol. 42, pp. 373–386, 2006.
- [14] D. Lee and H. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, pp. 788 – 791, 1999.
- [15] —, “Algorithms for non-negative matrix factorization,” in *NIPS*, 2000.
- [16] P. Paatero and U. Tapper, “A non-negative factor model with optimal utilization of error estimates of data values,” *Environmetrics*, vol. 5, pp. 111–126, 2006.
- [17] D. Zeimpekis and E. Gallopoulos, *TMG: A MATLAB toolbox for generating term document matrices from text collections*, 2006.
- [18] W. M. Rand, “Objective criteria for the evaluation of clustering methods,” *Journal of the American Statistical Association*, vol. 66, pp. 846–850, 1971.
- [19] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, 1990.

Improving Cohesiveness of Text Document Clusters

Gaurav Ruhela

Center for Data Engineering

International Institute of Information Technology (IIIT-H)

Hyderabad, INDIA

Abstract—Text document clustering plays an important role in the performance of information retrieval, search engines and text mining systems. Traditional clustering algorithm like *k*-means, bisecting *k*-means depend on the similarity value between document vector and cluster vector. So, assigning weight to the feature vector of a cluster is an important issue. More weight should be given to the representative and discriminative features of the cluster. Clustering is an unsupervised process, thus it is difficult to capture the possible distribution of documents. So, an intermediate step has to be introduced from which a clustering algorithm can find the discriminative features for the cluster. In this paper, we propose a method to identify discriminative feature set for each cluster. Using these feature sets, we refine a given set of clusters by incrementally moving documents between clusters. We present several experimental results to show that our approach produces more homogeneous and cohesive clusters.

Keywords: Cluster refinement, Discriminative features, Feature weighting.

1. Introduction

With the exponential growth in the world-wide web, the speed and the scale of information distribution has also increased. With the ever growing demand of on-line information, it has become important to provide efficient mechanisms to locate and present textual information effectively. Due to enormous collection, even a simple task of browsing through documents has become strenuous, and has become a very challenging task to find the truly relevant content.

Clustering is used to increase the performance of many information retrieval, search engines and text mining systems. Clustering is the partitioning of a data set into subsets, so that the data in each subset share some common trait. The main idea is to (i) extract unique content-bearing words from the set of documents and (ii) represent each cluster as a vector of certain weighted terms. In a text document, features can be terms, phrases or any other unit which can be used to represent its content. Most of the feature extraction algorithms follow “Bag of words” [1] model to capture feature vector of a document. Feature vector of a cluster is a vector of terms along with their weights, which shows the significance of a term to a cluster. Ideally, representative

(highly weighted) features of a cluster and representative feature of documents in that cluster should be identical. A feature should be a representative feature of a cluster if the feature is frequent in this cluster and at the same time not so frequent in other clusters. In other words, a representative feature of a cluster should also discriminate the cluster from other clusters. By capturing and weighting these discriminative features there is a scope to obtain quality clusters.

Unlike text categorization and text classification, clustering is not a supervised procedure. Text classification is a supervised learning process, which assigns pre-defined category labels to new documents based on the likelihood suggested by the training set. Availability of these pre-categorized data-sets is less than non-structured and non-categorized data-sets. This shows the need of unsupervised methods which can use and learn from the beneficial steps of supervised learning and return quality results.

State of art clustering algorithms extract and use the relationship between features and documents, but do not use classes as they are not available. In this paper, we focus on quality of clustering and putting aside the orthogonal problem of time complexity. We use bisecting *k*-means algorithm to cluster the documents. These clusters are not considered as the final result and are used as virtual clusters. These virtual clusters are now used to (i) capture the neighborhood information of documents, (ii) identifying and weighing discriminative features of each cluster, and (iii) refine clusters by incrementally moving documents from one cluster to other based on a new (proposed) similarity criteria between document and cluster vector (cluster mean).

The rest of this paper is organized as follows. In Section 2, related work is reviewed. Section 3, presents the preliminary steps needed to represent a document as a vector. In Section 4, the proposed cluster refinement scheme is explained. In section 5, reports the experimental results. Conclusion and future work is presented in section 7.

2. Related Work

Document clustering methods can be mainly categorized into two types: partitioning and hierarchical clustering. Both of these methods have been extensively studied by researchers and there has been lot of interesting work in this area. Clustering plays an important role in automated

indexing, for use in browsing collection of documents, genre identification of a document, spam filtering, word sense disambiguation, image segmentation, gene expression analysis by organizing large amount of documents into a small number of meaningful clusters [2][3] [4].

Hierarchical clustering methods are useful for various data mining tasks. Since hierarchical clustering is a greedy search algorithm, the merging decision made early in the agglomerative process are not necessarily the right ones. In [5], a solution is proposed to refine a clustering produced by the agglomerative hierarchical algorithm to correct the mistakes made early in the agglomerative process.

Scatter/Gather [6] is a well-known algorithm which has been proposed for a document browsing system based on clustering. It uses a hierarchical clustering algorithm to determine an initial clustering which is then refined using the k-means clustering algorithm. Many variants of the k-means algorithm have been proposed for the purpose of text clustering. The classical version of k-means uses euclidean distance, however this distance measure is inappropriate for its application to clustering a collection of text documents [7]. An effective measure of similarity between documents, and one that is often used in information retrieval, is cosine similarity, which uses cosine of angle between two document vectors.

K-means depends on the initial seed set and there is a possibility that it may not give globally optimal clustering solution. In [8], various clustering initialization methods are discussed for better separation in the output solution. Bisecting k-means algorithm outperforms basic k-means as well as agglomerative hierarchical clustering algorithms in terms of accuracy and efficiency [9] [3].

In [10], term extraction and weighting scheme is proposed which uses open web directory to improve clustering performance. Qualitative text document clustering without list of topics, ontology, and other lexical resources is still a challenging task. In this paper, without using any external knowledge repository, we calculate document's similarity with the discriminative feature vector of cluster and are able to get quality clusters by incrementally moving documents to the cluster which they belong to.

3. Preliminaries on Document Modeling

To represent the document in vector format, basic steps needed are explained in further sub-sections.

3.1 Pre-processing

Stop-words such as 'i', "the", "am", "and" etc. are removed using a stop-word list. All letters have been converted to lowercase, then the terms are reduced to their basic stem by applying a stemming algorithm. These steps remove non-informative words.

3.2 Document Representation

Let 'D' be the total number of documents in the data-set. For each document $d_j, 1 \leq j \leq D$, one can define feature vector of each document ' d_j ' in ' n ' dimensional term space as $\vec{d}_j = (t_1, t_2 \dots t_n)$. In more general form the bag of words representation of documents d_j is a vector of terms and their weights.

$$\vec{d}_j = (\langle t_1, w_1 \rangle, \langle t_2, w_2 \rangle \dots \langle t_n, w_n \rangle) \quad (1)$$

where w_n is weight of term t_n of document \vec{d}_j . Simplest model to weigh a term is the boolean model based on the set theoretic approaches. Weight of a term $t_n, w_n \in \{0, 1\}$, is assigned '0' or '1' based on the absence or presence of a term. This model ignores the information provided by the frequency of a term, whether a term appears only once or more number of times, same importance is given. TF-IDF and its various variants removes this disadvantage by assigning non-binary weights to the terms. Several weighting schemes like OKAPI [11] and LTU [12] are introduced which take use of document length as well.

3.3 Similarity Measure

We measure the similarity between two document vectors \vec{d}_1 and \vec{d}_2 by calculating cosine similarity,

$$Sim(\vec{d}_1, \vec{d}_2) = Cosine(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1 \cdot \vec{d}_2}{|\vec{d}_1| |\vec{d}_2|} \quad (2)$$

Where, ' \cdot ' indicates the vector dot product and $|\vec{d}_1|$ is the length of document vector \vec{d}_1 . Similarly, value of dot product between a document vector and cluster's mean is equivalent to the average similarity between the document and all the documents of the cluster which is being represented by the cluster mean [3].

$$Sim(\vec{d}_1, \vec{C}_k) = \frac{\vec{d}_1 \cdot \vec{C}_k}{|\vec{d}_1| |\vec{C}_k|} = \frac{1}{|S|} \sum_{k \in S} Cosine(\vec{d}_1, \vec{d}_k) \quad (3)$$

4. Proposed Cluster Refinement Scheme

In this section, basic idea is explained, then in further sub-sections, proposed approach for refinement of clusters is explained.

4.1 Basic Idea

The problem of finding clusters in data is challenging when clusters are of widely differing sizes, densities and shapes, and when the data contains large amounts of noise and outliers. The use of a shared nearest neighbor definition of similarity removes problems with varying density, while the use of core points handles problems with shape and size [13]. If this knowledge about distribution of documents is utilized there is scope to improve quality of clusters. An intermediate phase should be introduced from which clustering algorithm can learn about the neighborhood of

the documents. A cluster should have documents which are correlated to each other and at the same time documents should have some discriminative features of the cluster.

4.2 Description of Proposed Approach

We explain proposed approach in following steps: (i) Generation of virtual clusters, (ii) Finding core documents in virtual clusters, and (iii) Refinement of virtual clusters. We explain these steps one by one.

4.2.1 Generation of virtual clusters

Bisecting k-means [14] is used to cluster the given dataset. These clusters are considered as virtual clusters, which are used to (i) capture neighborhood information of documents, (ii) identifying and weighing discriminative features of each cluster, and (iii) refine clusters by incrementally moving documents from one cluster to other based on a new (proposed) similarity criteria between document and cluster vector (cluster mean). Number of virtual clusters is equal to the desired number of clusters, i.e. 'N'. So, VC_k is the k^{th} virtual cluster, where, $1 \leq k \leq N$. Mean of VC_k is denoted by $\overrightarrow{VM_k}$. Given a set, S, of documents in a virtual cluster VC_k , we define the mean of virtual cluster to be:

$$\overrightarrow{VM_k} = \frac{1}{|S|} \sum_{k=1}^S \vec{d}_k \quad (4)$$

which is obtained by averaging the weights of the various terms present in the documents of 'S'. Here, we use TF-IDF weighting scheme to weigh terms of document vector \vec{d}_k . Though we can use various weighting scheme, some of them are mentioned in Table 2.

4.2.2 Finding core documents in virtual cluster

Core documents for a VC_k ($CDoc_k$) are the documents whose similarity with the mean $\overrightarrow{VM_k}$ is greater than ' α '. The documents which are not close (less similarity value) to the mean are considered for the transfer to other cluster. As the clusters can be of different size and different density, global threshold for similarity should not be fixed. If a global threshold for similarity measure is fixed and if its value is low, then the number of core documents will be high and may contain the documents which should be in other clusters. If the threshold for similarity measure is chosen to be high, then non-core documents will be more and the membership of documents which should remain in this cluster, is unnecessarily checked. To overcome this problem of variable density of different clusters, similarity threshold for core documents should not be global. The information from virtual clusters can be used to locally set the threshold for similarity. So, threshold ' α ' is set to be the average pairwise similarity between all documents in the VC_k .

$$\alpha = \frac{1}{|S^2|} \sum_{x=1}^S \sum_{y=1}^S Sim(\vec{d}_x, \vec{d}_y) \quad (5)$$

Value of ' α ' will be more for a dense cluster as the documents will be in close proximity. In such a dense cluster, if a document is far from a mean, then it may not be the part of this cluster. So, only non-core documents will be checked for membership in other clusters. Value of ' α ' will be low for clusters with low density.

4.2.3 Refinement of virtual clusters

In this step, membership of a non-core document (documents whose $Sim(\vec{d}_k, \overrightarrow{VM_k})$ is less than ' α ') is checked. For a document \vec{d}_k , similarity with all clusters is calculated and \vec{d}_k is moved to the cluster to which \vec{d}_k has the maximum similarity value. The quality of this iterative movement process depends on the selection of similarity function. As cosine similarity is used as similarity function, weight assigned to the terms in \vec{d}_k and $\overrightarrow{VM_k}$ is crucial. Traditional clustering algorithms like k-means, bisecting k-means etc., which decide the membership of a document into a cluster based on similarity value with the cluster, uses the same weighting scheme for documents and cluster vector i.e. if TF-IDF is used for \vec{d}_k then TF-IDF will be used to weigh terms in cluster mean ($\overrightarrow{VM_k}$).

Here, TF-IDF is not used to weigh the mean of a cluster as it treats each feature equally for all clusters, and it does not take into account of fact that the weight of a feature in a document is not only related to the document, but also to the cluster that a document belongs to [15], [16][17]. A feature may have different importance for different clusters. In this paper, Refined-mean ($\overrightarrow{RM_k}$) is introduced to keep track of discriminative features of the VC_k . Feature vector of cluster, i.e. $\overrightarrow{RM_k}$, should be weighed in such a manner that the weight represents the importance of the feature for a cluster. A weighting scheme, "term frequency inter cluster frequency (TF-ICF)" is proposed to weigh the terms in $\overrightarrow{RM_k}$.

- *TF-ICF weighting scheme:* Once virtual clusters are formed, features are weighed according to their frequency in a cluster and taking into account the other clusters as well. This measure captures the importance of a term for a cluster. By this measure, features which occur in a cluster and are not so frequent in other clusters gets more weight.

$$W_{ik} = tf_{ik} * \log_2 \left(\frac{N+1}{n} \right) \quad (6)$$

Where,

W_{ik} = weight of term t_i in VC_k

tf_{ik} = frequency of term t_i in VC_k

N = number of clusters to be formed

n = number of cluster where t_i occurs atleast once.

For every term ' t_i ' of VC_k , weight ' W_{ik} ' is calculated which represents the contribution of term ' t_i ' in the

discriminative feature of VC_k . More the weight ' W_{ik} ', more is the importance of this term/feature for this cluster. Any document which has the same semantics as of this cluster (VC_k), should be moved to VC_k .

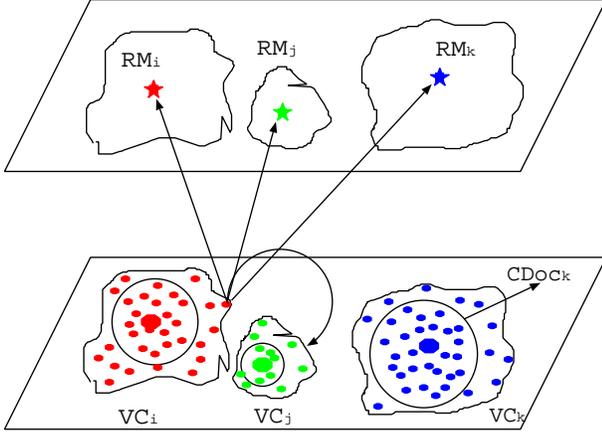


Fig. 1: Refining of Clusters

To decide the membership of non-core documents, similarity of document vector \vec{d}_d with the refined mean ($\overrightarrow{RM_k}$) of VC_k is calculated, for all 'k'. A document will be moved in VC_k if and only if

$$Sim(\vec{d}_d, \overrightarrow{RM_k}) > Sim(\vec{d}_d, \overrightarrow{RM_j}) \quad \forall j, j \neq k \quad (7)$$

Otherwise, document is moved to the cluster to which it has maximum similarity value. Removing a document from VC_i and assigning it to VC_j (Fig. 1) changes the mean of VC_i and VC_j . In other words, terms in $\overrightarrow{RM_i}$ and $\overrightarrow{RM_j}$ will change. So, TF-ICF weight for the terms of \vec{d}_d is calculated. As frequency of terms and number of different clusters that a term appears (" $t_{f_{ij}}$ " and 'n' respectively) may change for the terms in \vec{d}_d .

Refined means ($\overrightarrow{RM_k}$) of the cluster "from which" and "to which" a document is transferred have to be updated after every transfer. Due to these movements, neighborhood of the documents in cluster keeps on changing, so does the density of different features. More weight is given to the feature which is more frequent in a cluster and not so frequent in others.

After every movement, if there is any, check for small clusters is performed. Clusters having less than '5' documents are considered as small clusters. If a small cluster is found, its documents are assigned to some other clusters, using the same notion of similarity calculation. As the number of clusters has to be 'N', the largest cluster is bisected using bisecting k-means (this step follows the routine algorithm, same weighting scheme for both cluster vector and document vector). This process can be repeated for 'ITER' number of times. For Example: in Fig 1, document \vec{d}_d is moved from VC_i to VC_j because $Sim(\vec{d}_d, \overrightarrow{RM_j})$ is greater than $Sim(\vec{d}_d, \overrightarrow{RM_i})$ and $Sim(\vec{d}_d, \overrightarrow{RM_k})$. After this movement $\overrightarrow{RM_i}$ and $\overrightarrow{RM_j}$ will be generated again.

Table 1: Algorithm for Cluster Refinement

Input: Document Set D , ITER, 'N'

Output: 'N' Clusters

```

1: k = 0
2: GetCluster by bisecting k-means
3: Calculate tf-icf for terms in all virtual clusters
4: while (k != ITER)
5:   k++
6:   foreach  $VC_k$ ,  $k \in N$ 
7:      $CDoc_k = Find\_CoreDocs(k)$ 
8:     foreach document not in  $CDoc_k$ 
9:       Doc-feature-vector = Make-feature-vector(document)
10:      ClusterNo = Assign-clusterNo(Doc-feature-vector)
11:      if(ClusterNo != k)
12:        TF-ICF(Doc-feature-vector)
13:        if(Check-for-small-cluster())
14:          Assign-docs-to-other-clusters()
15:          TF-ICF(Documents-in- $VC_k$ )
16:          bisect-largest-cluster()
17:        endif
18:      endif
19:    end
20:  end

```

In short, after clustering the documents, feature vector of all the clusters are compared with document vector, and the document is moved to the cluster having most similar feature vector. After the movement feature vectors of cluster are updated. Feature vector of clusters are now more dynamic and can incorporate the changes occurred in the feature density.

Once a document is moved, it is not considered for transfer to other cluster, at least for this iteration. Same process is repeated for pre-defined number of iterations (ITER). Algorithm for cluster refinement is given in Table 1. Description of some important steps is as follows: Step 7: finds the core documents of cluster number 'k'. Step 9: Documents which are in cluster 'k', but are non-core, they are checked for their membership in other clusters. Step 11: checks whether document is moved to another cluster. Step 12: TF-ICF weight of the terms which are in this document is calculated. Step 13: checks whether all the clusters have atleast pre-defined number of documents. If not, then at Step 14, all the documents of small cluster are moved to other clusters. And again TF-ICF weight for these document terms has to be calculated at Step 15. Step 16: As number of clusters should be 'N', bisect k-means is used to bisect the largest cluster.

5. Experiments

To compare the performance of proposed approach with existing clustering algorithms, experiments are conducted on

WebData¹ data-set consisting of 314 web documents already classified into 10 categories. For evaluation of several refinement steps and cluster quality, purity value for clusters is registered.

Let the manually labeled clusters be

$$C = \{C_1, C_2, \dots, C_{10}\}$$

and clusters obtained be

$$C' = \{C'_1, C'_2, \dots, C'_{10}\}$$

Each resulting cluster C'_i from a partitioning C' of the overall document set D is treated as if it were the result of a query.

The precision of a cluster $C'_i \in C'$ for a given category $C_j \in C$ is given by

$$Precision(C'_i, C_j) = \frac{|C'_i \cap C_j|}{|C'_i|} \quad (8)$$

The overall value of purity is computed by taking the weighted average of maximal precision values:

$$Purity(C', C) = \sum_{C'_i \in C'} \frac{|C'_i|}{|D|} \max_{C_j \in C} Precision(C'_i, C_j) \quad (9)$$

Table 2: Term Weighting Schema. In this table tf indicates term frequency, N is the total number of documents in collection, n is the number of documents containing term t_i , df is the document frequency, dl is the document length, avg_dl is the average document length for a collection.

Name	Term Weight Schema
TF	tf
TF-IDF	$tf * \log(\frac{N}{n})$
OKAPI	$\frac{tf}{0.5 + 1.5 \frac{dl}{avg_dl}} \log(\frac{N - df + 0.5}{df + 0.5})$
LTU	$\frac{(\log(tf) + 1) * \log(\frac{N}{df})}{0.8 + 0.2 \frac{dl}{avg_dl}}$

In Table 3, “iPurity” is the initial purity value registered by traditional bisecting k-means, in which both document vector and cluster vector are weighted by same scheme. These values are used as baseline purity value of clusters. “rPurity” is the refined purity value, refinement values are shown

upto five iterations. It can be noted that purity values keeps on increasing with the number of iteration of refinement process.

In Fig 2., iteration number and average purity are the two axis. Iteration number ‘0’ means that the number of times refinement process used is ‘0’. In other words, results obtained from traditional bisecting k-means is considered as the final result. Similarly, iteration number ‘1’, ‘2’ upto ‘5’ shows the number of iterations of refinement process. A document vector can be weighted with several weighting schemes, resulting in different clusters. Refinement process has been tested to refine clusters obtained from four different weighting schemes (Table 2).

In Table 3, it can be noticed that for iteration number ‘0’, TF weighting scheme registered lowest average purity value. TF-IDF registered highest purity value followed by LTU and then OKAPI. These purity values are taken as baseline purity value. Significant improvement in purity values can be noted even after only ‘1’ iteration. 15.6% improvement on TF weighting scheme, 6.74%, 12.5%, 9.09% improvement is registered on weighting schemes TF-IDF, OKAPI and LTU respectively.

Table 3: Average Purity Values in different settings

Scheme	iPurity	rPurity after iteration				
		#1	#2	#3	#4	#5
TF	0.6004	0.7564	0.7686	0.7809	0.7820	0.7831
TF-IDF	0.7542	0.8216	0.8274	0.8285	0.8296	0.8296
OKAPI	0.6239	0.7489	0.7487	0.7948	0.7935	0.7935
LTU	0.7142	0.8051	0.8263	0.8294	0.8294	0.8294

Theoretical proof for convergence is not presented, but experimental results show that after ‘3’ iterations process seems to converge. Experimental result shows that even one run of the refinement process improves the cluster quality. With the number of iterations, percentage increase in purity value decreases.

6. Discussion

By traditional bisecting k-means algorithm, if TF-IDF is used as weighting scheme, both document vector and cluster mean would be weighed by TF-IDF method. If proposed refinement approach is used, in which, cluster mean is weighted by TF-ICF measure, significant improvement in the purity values of clusters can be noticed. Similarly for the weighing schemes in Table 2, it can be observed that refinement improves the clustering result no matter what weighting scheme is used to weight document vector. **NOTE:** TF-ICF is used to weight cluster mean, only when membership of a document has to be checked during refinement process. If we need to bisect any cluster (when all the document of a small cluster were allocated to other clusters and to maintain the number of clusters to be ‘N’), traditional weighting method

¹<http://pami.uwaterloo.ca/~hammouda/webdata/>

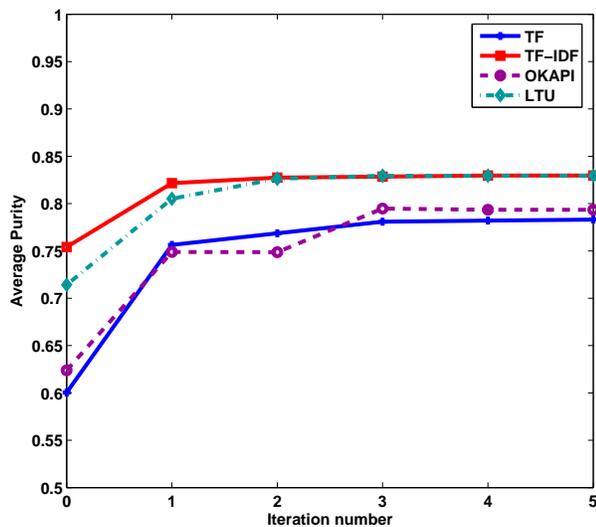


Fig. 2: Average Purity values after every iteration

is used to weigh both document vector as well as cluster mean.

7. Conclusion and Future Work

An approach for assigning weight to the terms representing the cluster is proposed. Using these weighted feature vectors, documents can be iteratively moved to the appropriate cluster. Performance evaluation exhibits clear superiority of the proposed method with its improved document clustering quality with significant factor. Furthermore, terms are weighed in such a way that the weight shows term's importance for a cluster. For a term, weight can be different for different cluster. Apart from quality clusters, discriminative terms for the clusters are also obtained.

As part of future work, detailed experiments by considering other types of data-sets are planned.

References

- [1] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, 1988.
- [2] F. Sebastiani, "Text categorization," in *Text Mining and its Applications to Intelligence, CRM and Knowledge Management*. WIT Press, 2005, pp. 109–129.
- [3] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *In KDD Workshop on Text Mining*, 2000.
- [4] E. M. Rasmussen, "Clustering algorithms," in *Information Retrieval: Data Structures & Algorithms*, 1992, pp. 419–442.
- [5] V. K. George Karypis, Eui-hong (Sam) Han, "Multilevel refinement for hierarchical clustering," in *University of Minnesota - Computer Science and Engineering Technical Report*, 1999.
- [6] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey, "Scatter/gather: a cluster-based approach to browsing large document collections," in *SIGIR '92: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 1992, pp. 318–329.
- [7] L. Zhu, J. Guan, and S. Zhou, "Cwc: A clustering-based feature weighting approach for text classification," in *MDAI '07: Proceedings of the 4th international conference on Modeling Decisions for Artificial Intelligence*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 204–215.
- [8] A. Strehl, E. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering," in *In Workshop on Artificial Intelligence for Web Search (AAAI 2000)*. AAAI, 2000, pp. 58–64.
- [9] B. C. M. Fung, K. Wang, and M. Ester, *The Encyclopedia of Data Warehousing and Mining*. Hershey, PA: Idea Group, August 2008, ch. Hierarchical Document Clustering, pp. 970–975.
- [10] G. Ruhela and P. K. Reddy, "Improving text document clustering by exploiting open web directory," in *Proceedings of the Twenty-First International Conference on Software Engineering and Knowledge Engineering (SEKE'09)*, Boston, USA, July 1-3, 2009.
- [11] J. Prager, E. Brown, A. Coden, and D. Radev, "Question-answering by predictive annotation," in *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 2000, pp. 184–191.
- [12] E. Gabrilovich and S. Markovitch, "Harnessing the expertise of 70,000 human editors: Knowledge-based feature generation for text categorization," *J. Mach. Learn. Res.*, vol. 8, pp. 2297–2345, 2007.
- [13] L. Ertöz, M. Steinbach, and V. Kumar, "Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data," in *SDM*, 2003.
- [14] S. M. Savaresi and D. L. Boley, "On the performance of bisecting k-means and pddp," in *Proceedings of the First SIAM International Conference on Data Mining (ICDM-2001)*, 2001, pp. 1–14.
- [15] P. Soucy and G. W. Mineau, "Beyond tfidf weighting for text categorization in the vector space model," in *In Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, 2005, pp. 1130–1135.
- [16] H. Xu and C. Li, "A novel term weighting scheme for automated text categorization," in *ISDA '07: Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 759–764.
- [17] M. Lan, C. L. Tan, and H. B. Low, "Proposing a new term weighting scheme for text categorization," Boston, 2006, pp. 763–768.

Impact of a New Attribute Extraction Algorithm on Web Page Classification

Göksel Biricik, Banu Diri, *Yildiz Technical University, Computer Engineering Department*

Abstract—This paper introduces a new algorithm for dimensionality reduction and its application on web page classification. A heterogeneous collection of web pages is used as the dataset. The textual contents of pages are the selected attributes for classification. Using the offered algorithm, high dimension of attributes- words extracted from the pages- are projected onto a new hyper plane having dimensions equal to the number of classes. Results show that processing times of classification algorithms dramatically decrease with the offered reduction algorithm. This is achieved by the fall of the number of attributes given to classifiers. Accuracies of the classification algorithms also increase compared to tests run without using the proposed reduction algorithm.

I. INTRODUCTION

THE rapid and exponential growth of web pages on the internet makes difficult to organize relevant pages together. Thus, people tend to organize data sources into categories depending on the content because it is easier to track between categories and reach the desired data. Dewey and Library of Congress Headings are such examples of classification systems used in libraries to achieve categorization. Web pages are not logically organized in the internet which makes the data access and retrieval difficult. When we look at the internet, we can find web page classification solutions in forms of web directories like Dmoz or Yahoo!. The problem is that these directories are formed by human effort- with the hands and minds of the editors. The web is in fact several times bigger than the coverage of these directories plus search engines. That is why automation of this categorization process is important. Classification algorithms may help editors to classify new coming web pages into existing categories. This is also known as web page classification. There are many solutions proposed to help solve this problem. Dumais and Chen [1] offers hierarchical classification using SVM's on LookSmart web directory. Choi and Peng [2] offers a dynamic and hierarchical classification scheme. Mladenic uses Yahoo web directory with a Naïve Bayes classifier [3]. Holden and Freitas classify their web dataset using ant-colony algorithm [4]. Many other researches can be found in literature about web page classification problem.

One of the models widely used in text classification is the

vector space model in which the documents are represented as vectors described by a set of identifiers, for example, words as terms. This model is also known as bag-of-words model. According to this model, every document acts as a bin containing its words. Thinking in the vector space, each term is a dimension for the document vectors. The nature of this representation causes a very high-dimensional and sparse feature space, which is a common problem to deal with when using bag-of-words model. There are two effective ways to overcome this dimensionality problem: Attribute selection and attribute extraction.

Attribute selection algorithms output a subset of the input attributes, results in a lower dimensional space. Instead of using all words as attributes, attribute selection algorithms evaluate attributes on a specific classifier to find the best subset of terms [5]. This results in reduced cost for classification and better classification accuracy. The most popular attribute selection algorithms include document frequency, chi statistic, information gain, term strength and mutual information [6]. Chi-square and correlation coefficient methods have been shown to produce better results than document frequency [7]. The lack of attribute selection algorithms is that the selection procedure is evaluated on a certain classifier. Hence, the produced subset may not be suitable for another classifier to improve its performance.

Attribute extraction algorithms simplify the amount of resource required to describe a large set of data. The high-dimensional and sparse structure of vector space model requires large amount of memory and computation power. The aim of attribute extraction is to combine terms to form a new description for the data with sufficient accuracy. Attribute extraction works by projecting the high-dimensional data into a new, lower-dimensional hyperspace. Mostly used techniques are principal components analysis (PCA), Isomap and Latent Semantic Analysis (LSA). Latent Semantic Indexing is based on LSA and it is the most commonly used algorithm in text mining tasks nowadays. Our algorithm also extracts attributes from the original vector space and projects them into a new hyperspace with dimensions equal to number of classes.

In section 2, we introduce our evaluation dataset and the preprocessing steps. Section 3 gives brief description about related attribute extraction algorithms and introduces the proposed method. In section 4 we discuss our experimental results. Section 5 addresses conclusions and feature work.

G.Biricik is with the Computer Engineering Department, Yildiz Technical University, Istanbul, 34349 Turkey (e-mail: goksel@ce.yildiz.edu.tr).

B.Diri is with the Computer Engineering Department, Yildiz Technical University, Istanbul, 34349 Turkey (e-mail: banu@ce.yildiz.edu.tr).

II. EVALUATION DATASET AND PREPROCESSING

A. Evaluation Dataset

Evaluation dataset is prepared from web pages under Dmoz [8] category World/Turkce. We work to develop an automated classifier for the Dmoz Turkish directory. We used the dataset that we prepared for our study because the attribute extraction algorithm we introduce in this paper is a part of our system. World/Turkce category included 28567 unique web pages in Turkish at the time we crawled. There are thirteen main categories in this set:

1. Shopping
2. News
3. Computers
4. Science
5. Regional
6. Recreation
7. Business
8. Reference
9. Arts
10. Games
11. Health
12. Sports
13. Society

Each category has several sub categories (total 758), which are not considered at this time.

B. Preprocessing

The dataset is crawled in raw HTML format. It is then parsed with HTML Parser¹ to fetch the content of web pages.

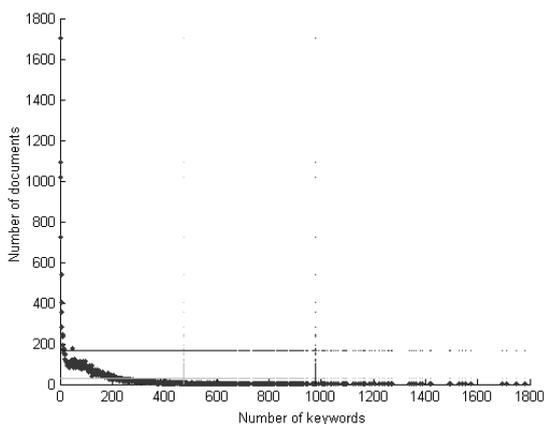


Fig.1. Distribution of terms and documents in the dataset. We can see that most of the dataset elements contain less than 200 keywords.

Term statistics are calculated to see the outlying documents. Fig.1 shows the distribution of terms and documents, in other words, the number of words documents have and the number of documents with a certain amount of terms. This plot shows us that some documents contain vast

amount of keywords, while some have very few.

A filter similar to Box-Plot is used to find the outliers in this distribution. The mean and standard deviations of the axes are calculated and a box is drawn on the distribution with the center (mean_x, mean_y) and boundaries (σ_x, σ_y). The area within the box is used as the input dataset; the samples outside this box are considered as outliers and cleared. The center of the box and the boundaries are also shown in Fig.1.

The dataset fetched out of Dmoz's World/Turkce category is in Turkish. Because Turkish is a suffix language, meanings of the words change with suffixes. To get rid of this effect, all of the terms are stemmed into their roots by using a Turkish stemmer tool named Zemberek².

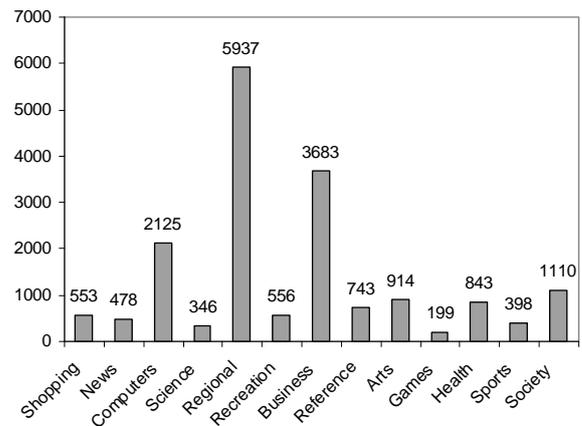


Fig.2. Distribution of web pages among classes after preprocessing step.

The final preprocessing step is to clean up the Turkish stop words. After this step, we have 17.885 documents (web pages) and 18.363 unique keywords. Looking from the bag-of-words view, we have a term-document matrix with 17.885 rows and 18.363 columns. Because any of the documents will contain only a tiny subset of our terms, our matrix is very sparse. The cells of the matrix contain tf-idf values of terms calculated by (1), (2) and (3), where n_i is the number of occurrences of the considered term in document d_j , $|D|$ is the total number of documents, $|\{d_j : t_i \in d_j\}|$ is the number of documents where term t_i appears..

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (1)$$

$$idf_i = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|} \quad (2)$$

$$tfidf_{i,j} = tf_{i,j} \times idf_i \quad (3)$$

¹ Available at <http://htmlparser.sourceforge.net/>

² Available at <http://code.google.com/p/zemberek/>

Our matrix leads us to a 18.363 dimensional vector space, which is hard to deal with because running classification algorithms on such a multidimensional space is very time and memory consuming. The distribution of web pages among 13 classes is also uneven, which causes another problem for classification algorithms. Number of web pages per class after preprocessing step is given in Fig.2.

III. ATTRIBUTE EXTRACTION

Attribute extraction algorithms simplify the amount of resource required to describe a large set of data. The aim of attribute extraction is to combine terms to form a new description for the data with sufficient accuracy. In this section, commonly used extraction algorithms -principal component analysis, PCA and latent semantic analysis, LSA- and the proposed method are introduced. Our algorithm projects attributes into a new hyperspace with dimensions equal to number of classes.

A. Related Research

1) *Principal Component Analysis*: PCA transforms correlate variables into a smaller number of correlated variables- principal components. Invented by Pearson in 1901 [9], it is generally used for exploratory data analysis.

PCA is used for attribute extraction by retaining the characteristics of the dataset that contribute most to its variance, by keeping lower order principals, which tend to have the most important aspects of data. This is accomplished by a projection into a new hyper plane using Eigen values and Eigen vectors.

PCA is a popular technique in pattern recognition, but its applications are not very common because it is not optimized for class separability [10]. It is widely used in image processing,

2) *Latent Semantic Analysis*: Patented in 1988, LSA is a technique that analyzes relationships between a document set and the terms that they contain by producing a set of concepts related to them [11]. As the name suggests, singular value decomposition breaks our matrix down into a set of smaller components.

LSA uses singular value decomposition (SVD) method to find the relationships between documents. Singular value decomposition breaks term-document matrix down into a set of smaller components. The algorithm alters one of these components (reduces the number of dimensions), and then recombines them into a matrix of the same shape as the original, so we can again use it as a lookup grid. The matrix we get back is an approximation of the term-document matrix we provided as input, and looks much different from the original.

LSI, based on LSA, is mostly used for web page retrieval and document clustering purposes. It is also used for document classification via voting, or information filtering. Many algorithms utilize LSI in order to improve performance by working in a less complex hyperspace.

B. Proposed Attribute Extraction Method

Our attribute extraction algorithm neither uses Eigen values, Eigen vectors, nor singular value decomposition. In this method, weights of terms and their probabilistic distribution over the classes are taken into account. We project term probabilities to classes, and sum up those probabilities to get the impact of each term to each class [15].

Assume we have I terms, J documents and K classes. Let $n_{i,j}$ be the number of occurrences of term t_i in document d_j and N_i be the total number of documents that contain t_i . We calculate the total number of occurrences of a term t_i in class c_k with (4). We calculate the weight of term t_i (in a document d_j) that affects class c_k with (5).

$$nc_{i,k} = \sum_j n_{i,j}, \quad d_j \in c_k \quad (4)$$

$$w_{i,k} = \log(nc_{i,k} + 1) \times \log\left(\frac{N}{N_i}\right) \quad (5)$$

We calculate the total effect of terms in a document over a class c_k with (6). At the end, I terms are projected onto number of classes; we have K new terms in hand for a document. Repeating this procedure for all documents gives us a reduced matrix with J rows (one row per document) and K columns (number of extracted features equals the number of classes). Finally, we normalize new terms by using (7).

$$Y_k = \sum_i w_{i,k}, \quad w_i \in d_j \quad (6)$$

$$newTerm_k = \frac{Y_k}{\sum_k Y_k} \quad (7)$$

We give a brief example to explain the algorithm comprehensibly. Assume we have 8 terms, 6 documents and 2 classes given in Fig.3. The corresponding term-document matrix is in Table 1. The cells of Table 1 correspond to term weights calculated with (4).

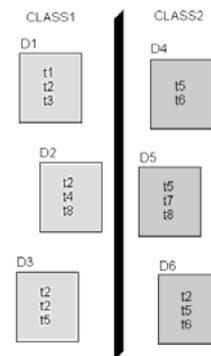


Fig.3. Sample dataset to explain the proposed extraction algorithm comprehensibly.

TABLE I
TERM-DOCUMENT MATRIX OF THE SAMPLE DATASET

	D1	D2	D3	D4	D5	D6
t1	1	0	0	0	0	0
t2	1	1	2	0	0	1
t3	1	0	0	0	0	0
t4	0	1	0	0	0	0
t5	0	0	1	1	1	1
t6	0	0	0	1	0	1
t7	0	0	0	0	1	0
t8	0	1	0	0	1	0

We calculate term weights over classes with (5). This produces I by K matrix, given in Table 2. After applying (6) and (7), we have the reduced J by K matrix with the extracted features of the processed documents. Result matrix is given in Table 3. This matrix is visualized in Fig.4. We can also read the extracted values as the membership probabilities of documents to classes, as seen on Fig.4. For example, the content of D4 tells us that it is 84% Class2 and 16% Class1. Thus, *the extraction method we propose may also be used as a stand-alone classifier*. When we feed a new document, the system decides which class this document belongs to by calculating (6) and (7) with the new document's content.

TABLE II
TERM WEIGHTS OVER CLASSES FOR THE SAMPLE DATASET

	C1	C2
t1	0.234	0
t2	0.123	0.053
t3	0.234	0
t4	0.234	0
t5	0.053	0.106
t6	0	0.228
t7	0	0.234
t8	0.144	0.144

TABLE III
TERM WEIGHTS OVER CLASSES FOR THE SAMPLE DATASET

	C1	C2
D1	0.918	0.082
D2	0.718	0.282
D3	0.585	0.415
D4	0.137	0.863
D5	0.289	0.711
D6	0.313	0.687

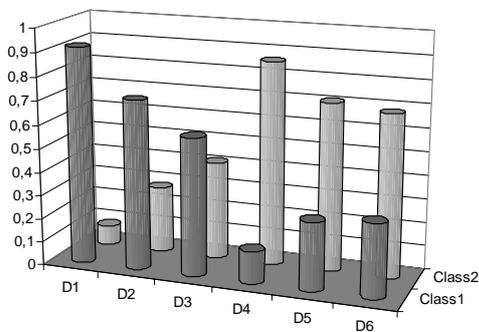


Fig.4. Visualization of Table III. Extracted values can be taken as the membership probabilities of documents to the classes.

IV. EXPERIMENTAL RESULTS

We prepare our evaluation dataset and test our attribute extraction algorithm's performance using 5 different classifiers. We use 10-fold cross-validation for testing.

We use Naïve Bayes as a simple probabilistic classifier, which is based on applying Bayes' theorem with strong independence assumptions [12]. We choose J48 Tree, an implementation of Quinlan's C4.5 decision tree algorithm [13] for a basic tree based classifier, and a random forest with 100 trees to construct a collection of decision trees with controlled variations [14]. We choose radial basis functions network for a function based classifier and K-nearest neighbor algorithm with K=10 to test instance-based classifiers.

We evaluate the classifiers' performance with their classification accuracy which is calculated by (8) and F-Measure by using (9). The classification performance results are given in Table 4. We can see that K-nearest neighbor classifier fails because our documents are unevenly distributed between classes. *Regional* and *Business* classes have much more instances than the other classes; this is the reason why an instance-based learner fails. RBF network also fails because the distribution of the attributes can not be easily modeled with Gaussian local models.

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{8}$$

$$F = \frac{2 \times TP}{(2 \times TP + FP + FN)} \tag{9}$$

TABLE IV
RESULTS FOR THE CLASSIFIERS EVALUATED ON OUR DATASET WITHOUT USING ATTRIBUTE EXTRACTION

	Classification Accuracy	F-Measure
Naïve Bayes	55.7%	0.591
J48 Tree (C4.5)	57.7%	0.572
Radial Basis Functions Network	32.4%	0.221
K-Nearest Neighbor (K=10)	31.8%	0.245
Random Forest (with 100 trees)	61.4%	0.583

TABLE IV
RESULTS FOR THE CLASSIFIERS EVALUATED ON OUR DATASET USING THE PROPOSED ATTRIBUTE EXTRACTION METHOD

	Classification Accuracy	F-Measure
Naïve Bayes	67.5%	0.675
J48 Tree (C4.5)	58.7%	0.589
Radial Basis Functions Network	70.6%	0.705
K-Nearest Neighbor (K=10)	74.3%	0.740
Random Forest (with 100 trees)	75.5%	0.753

We give the classification performance results of the classifiers using our proposed attribute extraction algorithm in Table 5. We can see that our attribute extraction algorithm increases the classification accuracy by 11.8% when using Naïve Bayes, 1.0% using J48, 38.2% using RBF network, 42.5% using K-nearest neighbor, and 14.1% using a random forest. Another outcome of our method is that it decreases classification times on both classifiers dramatically.

Fig.5 compares the classification accuracies and Fig.6 compares F-measures with and without using our extraction method. We can see that both classification accuracies and F-measures increase. Our method highly improves the performance of K-nearest neighbor classifier on an unevenly distributed dataset. The performance of RBF network is also boosted because the final attributes tend to have a Gaussian like distribution.

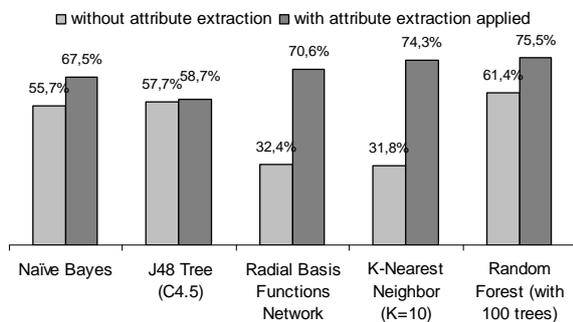


Fig.5. Comparison of classification accuracies with and without using the proposed attribute extraction algorithm.

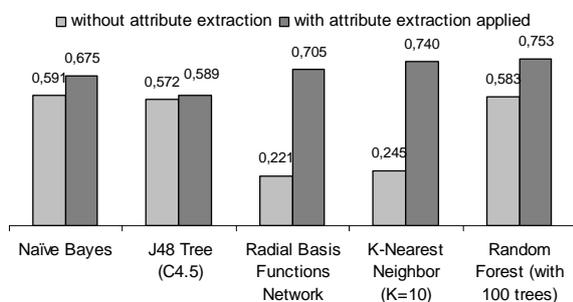


Fig.6. Comparison of F-measures of the classifiers with and without using the proposed attribute extraction algorithm.

V. CONCLUSION AND FUTURE WORK

We introduce a new algorithm for attribute extraction and its application on web page classification. We use a heterogeneous collection of web pages crawled from Dmoz's World/Turce as the dataset. Selected attributes for classification are the textual content of the web pages, in other words, preprocessed terms according to vector-space model. Using the offered algorithm, high dimension of attributes- terms extracted from the pages- are projected onto a new hyper plane having dimensions equal to the number of classes. Results show that processing times of

classification algorithms dramatically decrease with our reduction algorithm. This is achieved by the fall of the number of attributes given to classifiers. Accuracies of the classification algorithms also increase compared to tests run without using the proposed reduction algorithm, especially on the classifiers that fail on unevenly distributed datasets.

We plan to compare our attribute extraction method with other algorithms like LSA and PCA as a future work. As we see that our method can also act as a classifier, we program to test our method as a classifier with other classification algorithms. Furthermore, we have in view to improve our method to fit hierarchical classes.

REFERENCES

- [1] S.Dumais and H.Chen, "Hierarchical Classification of Web Content", In Proc. of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval, Athens, Greece, 2000, pp 256-63.
- [2] B.Choi, X.Peng, "Dynamic and Hierarchical Classification of Web Pages", Online Information Review, vol.28, no.2, pp.139-147, 2004.
- [3] D.Mladenic, "Turning Yahoo into an Automatic Web-Page Classifier", 13th European Conference on Artificial Intelligence Young Researcher Paper, 1998.
- [4] N.Holden, A.A.Freitas, "Web Page Classification with an Ant Colony Algorithm", in Parallel Problem Solving from Nature – PPSN VIII, vol.3242, Springer Berlin-Heidelberg, 2004, pp.1092-1102.
- [5] Y.Yiming, J.O.Pedersen, "A comparative study on feature selection in text categorization", in 14th International Conference on Machine Learning, 1997, pp.412-420.
- [6] J.Zhu, H. Wang and X.Zhang, "Discrimination-Based Feature Selection for Multinomial Naïve Bayes Text Classification," in 2006 Proc. ICCPOL2006, LNAI 4285, pp.149-156.
- [7] R.Jensen, Q.Shen, "Computational Intelligence and Feature Selection Rough and Fuzzy Approaches", IEEE-Wiley, New Jersey, 2008, pp.203-218.
- [8] Directory Mozilla Project, DMOZ, available at: <http://www.dmoz.org>.
- [9] K.Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space", Philosophical Magazine2(6):pp.559-572, 1901.
- [10] K.Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, London, 1990.
- [11] T.K. Landauer, S.T. Dumais, "Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge." Psychological Review, 1997, vol:104, no:2, pp. 211-240.
- [12] A. McCallum, K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification". In Proc. AAAI/ICML-98 Workshop on Learning for Text Categorization, 1998, pp. 41-48.
- [13] J.R. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.
- [14] L.Breiman, "Random Forests". Machine Learning, vol:45, no:1, pp.5-32, 2001.
- [15] Z.Kaban, B.Diri, "Recognizing Type and Author in Turkish Documents Using Artificial Immune Systems", 16th Signal Processing and its Applications Congress, Turkey, 2008.

On the Ranking of Text Documents from Large Corpora

Houssain Kettani¹ and Gregory B. Newby²

¹ECECS Department, Polytechnic University of Puerto Rico, San Juan, Puerto Rico, USA

²Arctic Region Supercomputing Center, University of Alaska, Fairbanks, Alaska, USA

Abstract - *Ranking text documents based on their relevance to a topic is of great importance in information retrieval. However, giving the increasingly available avalanche of digital documents, the size of collection pool from which these documents are drawn makes this task more challenging. In addition, current computing infrastructure is unable to deal with very large corpora directly. Thus, new algorithms are needed to seek parallel solutions and utilize more processing power to solve this problem. In this paper we propose a new algorithm that partitions a large collection of documents (a corpus) into smaller corpora that can each be handled by a single processor for the purpose of ranking. These multiple rankings are then merged together to provide a unified listing of all selected documents from the original large corpus.*

Keywords: Text documents ranking, large corpus, latent semantic indexing, algorithms.

1 Introduction

Searching large collections of documents is challenging for computational, practical and humanistic reasons. The great success of today's online search engines has allowed us to overlook many of these challenges, since results are, generally, satisfying. Yet, only a small fraction of available electronic data is available through such search engines [8, 12]. From a practical standpoint, we would like to make increased numbers of items of all types (Web pages and other documents) available for search. In some cases, this will require interacting directly with a search service on a specific site, rather than a centralized aggregation of content (as seen in Google & others). Such distributed search might offer the only method for overcoming security and accessibility concerns for access to specific data sets [5].

If practical considerations for distributed search were overcome, there are still computational and humanistic concerns remaining. Humanistically, consider that the role of search systems – more generally called information retrieval or IR systems – is to provide a ranked list of the most relevant documents. This list is created based on matching the human

expression of information need – the query – to a set of documents. The set of documents can be very large, as we see with today's Web search engines. The challenge here is that relevance is a difficult concept [10]. The list of relevant documents for a query is situational, and for a particular human information seeker the relevance of a given document will depend on what he or she is really looking for, as well as prior knowledge and expertise in the subject area. Expecting an IR system to make an effective ranked list of documents based on just a few query words is ambitious, indeed! Successful search engines today often present a variety of different results in the first set of 10 or 20 documents, so that different information seekers can quickly find high-quality documents of interest in the list. Below, we present latent semantic indexing (LSI) as one approach to better match the conceptual meaning of a query against the concepts found in a document, rather than being solely reliant on the (few) terms of a query, and the often ambiguous language used in queries and documents [2].

Computational challenges of search have, to some extent, been addressed through heroic engineering undertaken by database and Web search engine providers, among others. Being able to respond to a query in only a second or so is testament to the ability to rapidly produce results by parallelizing search across a single large corpus. In this paper, we consider the computational challenges brought about by distributed search and the merger of results from multiple collections, which has not yet been the subject of intense engineering efforts. We anticipate that the practical need to search across data sets, along with the humanistic need to get search results which are more closely targeted for individual information needs, will provide impetus to address such problems.

In this paper, we describe the LSI approach to information retrieval, and discuss some new approaches to efficiently build representations of queries, documents and document collections which attempt to overcome some of the limitations of ambiguity in human language. The basic LSI approach has been known, and demonstrated successfully, for years, but computational and practical challenges to its application have limited its use for large-scale Web search. By presenting effective algorithms for applying LSI across distributed data sets, we hope to provide a base for increasingly effective search across readily available Web-based content, as well as for content which is not currently available to general-purpose search engine harvesters.

This work was supported in part by the Arctic Region Supercomputing Center at the University of Alaska – Fairbanks, with additional support from the US Department of Defense High Performance Computing Modernization Program Office under contract G00003435 and others. This work was also supported in part by the United States Nuclear Regulatory Commission under grant number NRC-27-09-310.

2 Background

2.1 Vector space model

When addressing the issue of document ranking, it is much more convenient to reduce it to a mathematical problem. Thus, [9] introduced a vector space model (VSM) where keywords are the axis and documents are points in this multidimensional space. For example, if we have two documents \mathbf{d}_1 and \mathbf{d}_2 , the keywords are the set {cat, dog}, the word “cat” occurs five and seven times in \mathbf{d}_1 and \mathbf{d}_2 , respectively, and the word “dog” occurs two and three times in \mathbf{d}_1 and \mathbf{d}_2 , respectively, then \mathbf{d}_1 and \mathbf{d}_2 are represented by the points (5, 2) and (7, 3), respectively, and the query vector could be (0, 1), (1, 0), or (1, 1). A pictorial illustration of this vector space model is given in Figure 1.

The ranking of the documents is achieved by calculating the distance between the documents and the query. The closer a document to a query is, the higher its rank. Alternatively, since calculating distances is time consuming, we can use the angle θ_i between each document and the query as a measure for ranking. More precisely, we can use the cosine of this angle for the purpose of ranking. Thus, we can define,

$$\alpha_i = \cos \theta_i = \frac{\mathbf{q}^T \mathbf{d}_i}{\|\mathbf{q}\| \|\mathbf{d}_i\|}$$

Furthermore, if the vectors are normalized, this measure can be reduced to $\alpha_i = \mathbf{q}^T \mathbf{d}_i$. Thus, if all vectors are normalized, and since all the points lie in the nonnegative hypercube, minimizing the distance between \mathbf{d}_i and \mathbf{q} is equivalent to maximizing α_i , which is a simple dot product between the query and the corresponding document.

2.2 Latent semantic indexing

The vector space model is an excellent approach; however, it fails to address synonymy and polysemy of words in a language like “car” vs. “automobile,” or like “can” in “I

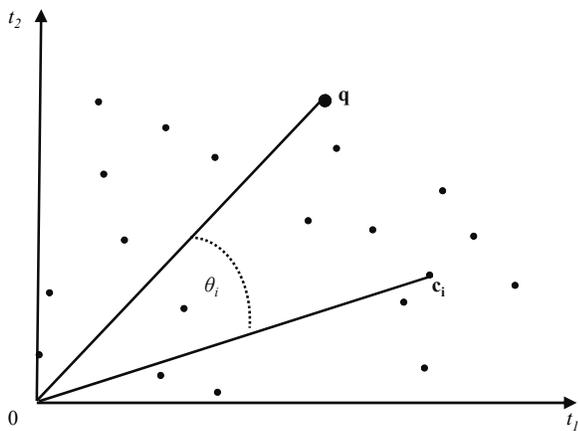


Figure 1. Pictorial illustration of the vector space model of document ranking, where $\{t_1, t_2\}$ is the set of keywords, \mathbf{q} is the query, other points are the documents, θ_i is the angle between a document and the query.

can” vs. “can” in “tuna can.” Thus, [2] introduces latent semantic indexing (LSI) algorithm as an add-on to the vector space model, which uses singular value decomposition (SVD) to reduce redundancy in data. Accordingly, [2] points out that a corpus can be presented as a matrix \mathbf{C} with non-negative integer entries representing the number of occurrence of terms in each document. The columns of \mathbf{C} represent documents and its rows represent the terms as follows:

$$\mathbf{C} = \begin{matrix} & \overbrace{\begin{matrix} c_{11} & c_{12} & \cdots & c_{1m} \\ c_{21} & \ddots & & \\ \vdots & & \ddots & \\ c_{n1} & c_{n2} & \cdots & c_{nm} \end{matrix}}^{m \text{ documents}} \\ \left. \begin{matrix} \\ \\ \\ \end{matrix} \right\} n \text{ terms} \end{matrix}$$

In our case, m is in order of millions, while n is in order of hundreds of thousands. Latent Semantic Indexing suggests first obtaining the corresponding singular value decomposition as follows:

$$\mathbf{C} = \mathbf{U}\mathbf{S}\mathbf{V}^T = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1m} \\ u_{21} & \ddots & & \\ \vdots & & \ddots & \\ u_{n1} & u_{n2} & \cdots & u_{nm} \end{bmatrix} \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_m \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1m} \\ v_{21} & \ddots & & \\ \vdots & & \ddots & \\ v_{m1} & v_{n2} & \cdots & v_{mm} \end{bmatrix}^T$$

Then, a rank reduction is obtained by zeroing out the singular values s_i such that $s_i \ll s_1 \forall i \geq k$, which introduces the following approximation of the term-by-document matrix \mathbf{C} :

$$\hat{\mathbf{C}} = \hat{\mathbf{U}}\hat{\mathbf{S}}\hat{\mathbf{V}}^T = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1k} \\ u_{21} & \ddots & & \\ \vdots & & \ddots & \\ u_{n1} & u_{n2} & \cdots & u_{nk} \end{bmatrix} \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_k \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1k} \\ v_{21} & \ddots & & \\ \vdots & & \ddots & \\ v_{m1} & v_{n2} & \cdots & v_{mk} \end{bmatrix}^T$$

Then, after normalizing the columns of $\hat{\mathbf{C}}$, the Vector Space Model can be applied.

2.3 Time complexity

The singular value decomposition of the $n \times m$ matrix \mathbf{C} requires $O(nm^2 + n^2m + n^3)$ multiplications [13]. The multiplication of an $n \times m$ matrix \mathbf{A} and an $m \times p$ matrix \mathbf{B} requires $O(nmp)$ [13]. Thus, obtaining the matrix $\hat{\mathbf{C}}$ requires $O(\max\{nk^2, nkm\}) = O(nkm)$, since $k < m$. On the other hand, obtaining each α_i requires $O(n)$, and therefore, obtaining the score vector requires $O(n^2)$, which is the time complexity of VSM. This can further be reduced to $O(n)$ through parallelization. Since LSI consists mainly of the aforementioned three parts, then it can readily be seen that SVD controls the complexity of LSI. Accordingly, the time complexity of LSI algorithm is in $O(nm^2 + n^2m + n^3)$.

2.4 Practical challenges

For a very large corpus, LSI is impossible to implement with current speed of processors and memory constraint. For example, the TREC Million Query Track [11], results in about 600,000 terms by about 25 million document matrix C . Storing this matrix is doable, since it is very sparse with about 99.99% of its elements are zero. However, applying SVD to it is even beyond the capabilities of today's large supercomputers. Thus, many researchers realized that this challenge can only be tackled with new high performance computing algorithms and infrastructure [6, 7]. This challenge is the prime motivation for this paper.

3 Proposed algorithm

Since one processor cannot perform LSI on the large corpus (due both to processing speed and the need for very large memory), we can divide the corpus to sub-corpus and send them to different processors, then rank the documents in each sub-corpus. We then merge the ranking obtained from each processor to obtain the result to the original document ranking problem. An itemized description of the proposed algorithm is as follows. We start by using VSM to represent the query and corpus by letting:

1. \mathbf{q} represent an $n \times 1$ query vector with unit norm, and
2. C represent an $n \times m$ term by document matrix.

We then apply LSI as follows:

3. $C = USV^T$ is SVD of C ,
4. Obtain \hat{S} by keeping the first k diagonal elements of S ,
5. Adjust U and V accordingly to get \hat{U} and \hat{V} , by keeping the first k columns of each,
6. Approximate C by $\hat{C} = \hat{U}\hat{S}\hat{V}$,
7. Normalize the columns of $\hat{C} = [\hat{c}_1, \hat{c}_2, \dots, \hat{c}_m]$,
8. Define $\alpha_i = \mathbf{q}^T \hat{c}_i$,
9. Let $\mathbf{a} = [\alpha_1, \alpha_2, \dots, \alpha_n]^T$ be the score vector,
10. The higher the score is, the higher the rank of the document.

However, for a very large in size matrix C , Step 3 and/or Step 6 cannot be handled with current computing infrastructure. Thus, suppose we have p processors at our disposal to use. Then,

11. Partition $C = [C_1, C_2, \dots, C_p]$,

12. Apply LSI to each sub-corpus C_i to obtain the matrix $\hat{C} = [\hat{C}_1, \hat{C}_2, \dots, \hat{C}_p]$, which is an approximation to \hat{C} .
13. For each C_i , calculate $\alpha_i = [\alpha_{i1}, \alpha_{i2}, \dots, \alpha_{i(m/p)}]^T = \mathbf{q}\hat{C}_i$,
14. Put $\hat{\mathbf{a}} = [\alpha_1, \alpha_2, \dots, \alpha_p]^T$, which is the estimated score vector.
15. Then, as before, the higher the score is, the higher the rank of the document.

4 Performance evaluation issues

4.1 LSI effect on ranking

The proposed algorithm stems from the fact that the order of documents in a corpus should not affect their ranking. Indeed, if SVM is applied alone, then the partitioning does not affect the ranking. However, due to LSI, we generally have $\hat{\hat{C}} \neq \hat{C} \neq C$. Thus, performance evaluation of the proposed algorithm will involve using test data to check the algorithm's robustness to the number of partitions.

4.2 How small should k be?

Singular value decomposition (SVD) produces a diagonal matrix containing singular values of the matrix C in descending order. As pointed earlier, in its attempt to approximate C , LSI zeroes out the singular values s_i such that $s_i \ll s_j \forall i \geq k$. But when is the condition " $s_i \ll s_j$ " satisfied? In his illustrative example, [2] picked the largest two singular values and zeroed out the other seven. So the authors suppressed all singular values less than 71% of the largest singular value. Others like [3], were more cautious and argued that the condition " $s_i \ll s_j$ " corresponds to s_i less than 10% of s_j . Thus, [3] suggests that " $s_i \ll s_j$ " \Leftrightarrow " $s_i < s_j/10$;" which we refer to here as "the 10% criterion."

4.3 The use of random matrices

To empirically evaluate the performance of the proposed algorithm, it is tempting to use random matrices to simulate the term-by-document matrix C . This randomness, however, makes singular values decrease very slowly even if we ensure that the matrix is very sparse. For example, to generate such sparse random matrix in MATLAB, we write the following in the command line " $C = \text{ceil}(r - \text{rand}(m, n))$," where $0 \leq r \leq 1$, which results in about $100r\%$ of the elements of C being nonzero. The slow decay of singular values will entail keeping most of them. For instance, if the 10% criterion is followed, we will need to keep 75% to 99% of the singular values. This in turn, defeats the rationale behind the use of LSI.

Nonetheless, the structure of human languages makes the content and formation of a text document far from being random. Luckily TREC Million Query Track [11] provides an

excellent, yet challenging, datasets for evaluation purpose. This dataset consists of millions of English documents and queries gathered from government domains. We constructed C from sub-collections of this dataset with thousands of terms and documents. More than 99% of the elements of C are zero. We observe an exponential decrease in the magnitude of the corresponding singular values. For example, following the 10% criterion, we keep only 0.15% to 0.69% of the singular values. Accordingly, the TREC dataset is used to evaluate the performance of the proposed method in this paper.

4.4 Time complexity

Following the discussion from Section 2.3, the time complexity for performing LSI on each partition C_i in Step 12 is obtained by substituting m/p for m . Accordingly, each C_i requires $O(nm^2/p^2 + n^2m/p + n^3)$. Since the p partitions are processed in parallel, the latter is the overall performance of the proposed algorithm. Thus, assuming that $m \gg n$, as is the case with TREC data, we can see that the proposed algorithm reduces the complexity of LSI by a factor of p^2 , where p is the number of processors. Furthermore, even if the partitions are processed in series, following similar analysis, the complexity of LSI is reduced by a factor of p (the complexity in this case is $O(nm^2/p + n^2m + pn^3)$). Note also that as the number of documents m grows large; LSI may not be handled with the current computing infrastructure. Consequently, for a very large m , the proposed method may be the only way to practically implement LSI.

5 Empirical analysis

5.1 Real data

To evaluate the performance of the proposed algorithm, we apply it to corpuses obtained from the TREC Million Query Track [11]. TREC is the Text REtrieval Conference, held annually at the National Institute for Standards and Technology (NIST) in Gaithersburg, Maryland. TREC is intended to provide a forum for comparison of information retrieval techniques across a variety of test conditions. Different tasks are evaluated as part of "tracks," which typically last for at least a few years before being updated, merged with other tracks, or discontinued. In the past, TREC focused on relatively structured textual data, such as abstracts, news articles and proceedings such as the Federal Register.

The growth of the Web, and interest in Web-based search engines, has resulted in numerous tracks that examined HTML, often harvested from the live Web. These harvests result in fixed test collections, with the largest holding over 20 million unique Web pages from over 6,000 unique hosts. By reusing a test collection for multiple years of TREC experiments, a set of judgments for the relevance of series of queries (known in TREC as topics) against the collection (known as a corpus) can help to build knowledge of the collection, and compare results from one year to other years.

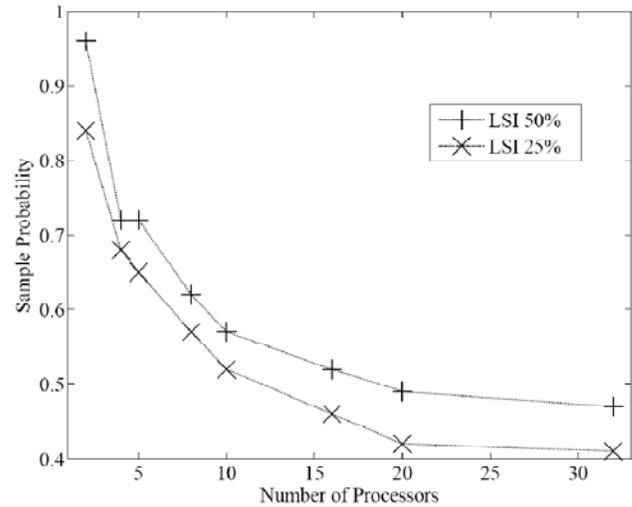


Figure 2. A plot of the sample probability vs. the number of processors using LSI with 25% and 50% of the eigenvalues.

At the Arctic Region Supercomputing Center (ARSC), we have participated in TREC's very large corpus track (VLC), million query track (MQ) and terabyte track (TB) in past years. These tracks have provided tens of thousands of unique TREC topics for evaluation against subsequently larger collections of Web-based documents.

In [4], we used an early approach to the divide and conquer method described in this paper. This followed past experiments based on techniques from [1] and other methods for merging results from separate systems. For the 2008 TREC million query track, our approach was to portray entire collections as vectors in an LSI space. For a given query, we determined the most similar collections, based on the cosine between the collection and query vectors. The fifty collections with the highest cosines were then searched, rather than the whole set of over 6,000 collections. (These collections were simply subsets of the entire 25 million documents in the TREC gov2 collection, in which Web pages from over 6,000 servers in the US .gov space were harvested.)

By applying LSI techniques at the collection level, rather than the document level, we were able to obtain greater efficiency by selecting only those collections deemed to be more highly related to a query. For practical purposes with distributed search, this is important since the incremental cost (in terms of latency and larger sets of documents to rank) for adding more collections can be avoided by only searching collections of greater interest.

5.2 Performance evaluation

From TREC data, we selected 320 documents and 100 queries that consist of at least one word. We then applied LSI algorithm and kept a certain percentage of the singular values. We simulated the case of the number of processors $p = 2, 4, 5, 8, 10, 16, 20$, and 32. Performance comparison is achieved

through the calculation of the *sample probability* that the top 10% of ranking without partitioning is contained in the top 20% of the ranking with partitioning. This was done for each query and then we took the average over all 100 queries. Ideally, we want this sample probability to be as close to one as possible. We run LSI using 25% and 50% of the total singular values at each stage, which was rounded to the next integer if needed using the ceiling function.

For example, with $p = 2$, and using LSI 25%, we compare the ranking obtained using LSI by keeping 80 singular values and the ranking obtained by using LSI on each partition keeping 25% of the singular values or 40 per each partition. The result of this analysis is depicted in Figure 2. Accordingly, with 25%, the average sample probability is 0.84, 0.68, 0.65, 0.57, 0.52, 0.46, 0.42, and 0.41 for $p = 2, 4, 5, 8, 10, 16, 20$, and 32, respectively. With 50%, we get 0.96, 0.72, 0.72, 0.62, 0.57, 0.52, 0.49, and 0.47. Thus, the average sample probability is inversely proportional with respect to the number of processors. This average, on the other hand, is directly proportional to the percentage of singular values used by LSI.

5.3 Remark

It is clear that if VSM is used without the add-on of LSI, then the sample probability will be one, no matter how many processors are used. The deteriorating performance of the proposed algorithm with respect to the number of processors puts LSI stability into question. This issue will be investigated in more detail in future research.

6 Conclusions and future direction

Motivated by TREC Million Query datasets, and the issue of ranking text documents drawn from a very large corpus, we have developed a divide-and-conquer algorithm implementation of the latent semantic indexing (LSI) algorithm to cope with the current speed of processors and memory constraint. The proposed algorithm reduces the time complexity of LSI by the square of the number of processors used in parallelizing the problem. We presented performance analysis of the proposed algorithm on different queries and corpuses gathered from TREC dataset to assess its performance and stability. This analysis brought the stability and robustness of LSI into question. That is ranking of documents is affected by the number of singular values retained. This should be investigated more in further research. Other study involves checking whether LSI needs to be part of the proposed algorithm and the possibility of involving other algorithms from the literature to replace LSI as a step in the proposed algorithm.

Acknowledgement

The authors wish to acknowledge constructive discussions with Chris Fallen, Kylie McCormick, and Miles Efron.

References

- [1] A. L. Calvé and J. Savoy, "Database merging strategy based on logistic regression." *Information Processing and Management*, 2000, 36(3):341-359.
- [2] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis," *Journal of the Society for Information Science*, 1990, 41(6):391-407.
- [3] M. Efron, "Eigenvalue-based model selection during latent semantic indexing," *Journal of the American Society for Information Science and Technology*, 2005, 56(9):969-988.
- [4] C. Fallen, G. B. Newby, and K. McCormick, "Distributed multisearch and resource selection for the TREC Million Query track." Gaithersburg: National Institute of Standards and Technology, 2008.
- [5] K. Gamiel, G. B. Newby, and N. Nassar, "Grid information retrieval requirements (GFD-I.027)." Lamont, Illinois: Open Grid Forum, 2003.
- [6] E. M. Garzón and I. García, "Solving eigenproblems on multicomputers: two different approaches," *International Journal of Computers and Applications*, 2004, 26(4):213-222.
- [7] M. R. Guarracino, F. Perla, and P. Zanetti, "A parallel block Lanczos algorithm and its implementation for the evaluation of some eigenvalues of large sparse symmetric matrices on multicomputers," *International Journal on Applied Mathematics and Computer Science*, 2006, 16(2):241-249.
- [8] S. Raghavan and H. Garcia-Molina, "Crawling the hidden web," *Proceedings of Very Large Databases (VLDB)*, 2001.
- [9] G. Salton, A. Wong, and C. S. Yang, "A Vector Space Model for Automatic Indexing," *Communications of the ACM*, 1975, 18(11):613-620.
- [10] L. Schamber, M. B. Eisenberg, and M. Nilan, "A re-examination of relevance: toward a dynamic, situational definition," *Information Processing and Management*, 1990, 26(6):755-776.
- [11] E. Voorhees, "Overview of TREC 2007," *Proceedings of the 16th Annual Text REtrieval Conference*. Gaithersburg: National Institute of Standards and Technology, 2007.
- [12] J. Xu, H. Chen, Z. Zhou, and J. Qin, "On the topology of the dark web of terrorist groups," *Proceedings of the IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2006.
- [13] G. Golub and A. van Loan, "Matrix Computations," Johns Hopkins University Press, 1996.

EmotiBlog: an Annotation Scheme for Emotion Detection and Analysis in Non-traditional Textual Genres

Ester Boldrini, Alexandra Balahur, Patricio Martínez-Barco, Andrés Montoyo

University of Alicante Department of Software and Computing Systems, E-03080 Alicante, Spain,
{eboldrini, abalahur, patricio, montoyo}@dlsi.ua.es

Abstract— *With the growth of the social Web, people have started to freely express their opinion on a whole variety of topics on forums, e-commerce sites or blogs. As a consequence, we need special methods to automatically extract opinions and emotions from these texts. Systems implementing such methods require training data from well annotated corpora.*

This paper presents EmotiBlog, an annotation scheme for labelling affective phenomena in non-traditional textual genres. We manually collected 300 blog posts in three different languages: Spanish, Italian and English on three topics of interest. We present the corpus, the annotation model we created, and we also discuss the results of the annotation and its corresponding statistical measures. Finally, we analyse the problems encountered during the annotation process and comment upon some of the conclusions we have reached while performing this research.

Keywords: Sentiment Analysis, Annotation Scheme, Web 2.0., Non-Traditional Textual Genres.

1. Introduction

Emotions are profoundly human and their mechanisms are fascinating and difficult to explain, as any human life-related topic. Research on emotion dates as far back as human civilization; each period giving a distinct explanation and interpretation to the diversity of sentiments. Moreover, each society used these clues for the definition of social norms, for the detection of anomalies, and even for the explanation of mythical or historical facts. Emotions were praised for their argumentative power, but also criticised as a “weakness” of the human being, that should ideally be rational. Modernity and the development of technology applied to different fields of science has proven that emotions are given by a compound of biological, neuronal transformations and are the root of all human reactions and decisions. Moreover, recent studies in Artificial Intelligence have shown that approximations of emotional response outperform the modelling of rational decision making. Furthermore, together with the development of technology and the growing access to information, we have witnessed the birth of a new type of society – that of the interaction and communication. In this new context, the role of emotion has become crucial. Facts determine emotion in people, who absorb facts and express the effect these facts have on them.

Other persons have access to these expressions of sentiments and facts, which they, in their turn, transform under the influence of their own emotional perception. Practically, access to information has also given way to access to emotional response to information, in the light of which information changes. Thus, people react to both facts and attitude on facts. Therefore, it becomes crucial to determine the manner in which emotion is expressed in texts, both traditional ones (newspaper/magazine articles), as well as non-traditional (blogs, reviews on the web). In recent years there has been growing interest in analysing how emotions are expressed in texts. Due to the large volume of emerging texts to which people have access, this task must be automated. Thus, emotion detection has become an essential task in the field of Natural Language Processing (NLP), where researchers are investigating data, methods and techniques to detect and classify emotion in texts. They are interested in capturing the manner in which humans express and detect emotion, in order to build tools and applications that automatically perform this task.

2. Related work

In recent years, there has been growing interest in the field of emotion detection as an NLP subtask. Different authors have addressed the necessities and possible methodologies from the linguistic, theoretical and practical points of view. Thus, the first step involved resided in building lexical resources of affect, such as WordNet Affect [1], SentiWordNet [2], Micro-WNOP [3] or “emotion triggers” [4]. All these lexicons contain single words, whose polarity and emotions are not necessarily the ones annotated within the resource in a larger context. Our work, therefore, concentrates on annotating larger text spans, in order to consider the undeniable influence of the context. The starting point of research in emotion is represented by [5], who centered the idea of subjectivity around that of private states, and set the benchmark for subjectivity analysis as the recognition of opinion-oriented language in order to distinguish it from objective language and giving a method to annotate a corpus depending on these two aspects – MPQA [6]. Subsequently, authors show that this initial discrimination is crucial for the sentiment task, as part of Opinion Information Retrieval (last three editions of the TREC Blog tracks competitions, the TAC 2008

competition), Information Extraction [7] and Question Answering [8] systems. Once this discrimination is done, or in the case of texts containing only or mostly subjective language (such as e-reviews), opinion mining becomes a polarity classification task. Our work takes into consideration this initial discrimination, but we also add a deeper level of emotion annotation. Since expressions of emotion are also highly related to opinions, related work also includes customer review classification at a document level, sentiment classification using unsupervised methods [9], Machine Learning techniques [10], scoring of features [11], using PMI, syntactic relations and other attributes with SVM [12], sentiment classification considering rating scales [10], supervised and unsupervised methods [13] and semisupervised learning [14]. Research in classification at a document level included sentiment classification of reviews [15], sentiment classification on customer feedback data [16], comparative experiments [17]. Other research has been conducted in analysing sentiment at a sentence level using bootstrapping techniques [7], considering gradable adjectives [18], semisupervised learning with the initial training set identified by some strong patterns and then applying NB or self-training [19] finding strength of opinions [20] sum up orientations of opinion words in a sentence (or within some word window) [21], [22], determining the semantic orientation of words and phrases [23], identifying opinion holders [24], comparative sentence and relation extraction and feature-based opinion mining and summarization [9]. Finally, fine grained, feature-based opinion summarization is defined in [25] and researched in [9] or [10]. All these approaches concentrate on finding and classifying the polarity of opinion words, which are mostly adjectives, without taking into account modifiers or the context in general. Our work, on the other hand, represents the first step towards achieving a contextual comprehension of the linguistic roots of emotion expression.

3. Motivation and contribution

Recent years have marked the beginning and expansion of the social Web, in which people freely express and respond to opinion on a whole variety of topics on forums, e-commerce sites or blogs. Moreover, at the time of taking a decision, more and more people search for information and opinions expressed on the Web on their matter of interest and base their final decision on the information found. While the growing volume of opinion information available on the Web allows for better and more informed decisions of the users, the quantity of data to be analyzed imposed the development of specialized NLP systems that automatically extract, classify and summarize the opinions available on the Web on different topics. Research in this field, of opinion mining (sentiment analysis), has proven the task to be very difficult, due to the high semantic variability of affective language. Different authors have addressed the problem of extracting and classifying opinion from different perspectives and at different levels, depending on a series of factors. These factors include: level of interest

(overall/specific), querying formula (“Nokia E65?”/“Why do people buy Nokia E65?”), type of text (review on forum/blog/dialogue/press article), and manner of expression of opinion - directly (using opinion statements, e.g. “I think this product is wonderful!”/“This is a bright initiative”), indirectly (using affect vocabulary, e.g. “I love the pictures this camera takes!”/“Personally, I am shocked one can propose such a law!”) or implicitly (using adjectives and evaluative expressions, e.g. “It’s light as a feather and fits right into my pocket!”). While determining the overall opinion on a movie is sufficient for taking the decision to watch it or not, when buying a product, people are interested in the individual opinions on the different product characteristics; when discussing a person, one judges and gives opinion on the person’s actions. Moreover, the approaches taken depend on the manner in which a user queries the data – whether it is a general formula such as “opinions on X” or a specific question “Why do people like X?” and the text source that needs to be queried. For example, querying reviews on e-commerce sites largely guarantees that opinions and attitudes expressed are directly related to the product in question. Retrieving opinion information in newspaper articles or blogs posts is more difficult, because it involves the detection of different discussion topics, the subjective phrases present and subsequently their classification according to polarity. Especially in the blog area, determining points of view expressed along a dialogue and the mixture of quotes and pastes from newspapers on a topic can, additionally, involve determining the persons present and whether or not the opinion expressed is on the required topic or on a point previously made by another speaker. As such, this difficult NLP problem requires the use of specialized data for system training and tuning, gathered, annotated and tested within the different text spheres. At the present moment, these specialized resources are scarce and even when they exist, they are rather simplistically annotated or are applicable only in the fields for which they were created. Last, but not least, most of the resources created are for the English language. The contribution we describe in this paper intends to propose solutions to the above-mentioned problems, and consists of the following points: first of all, we wanted to overcome the problem of corpora scarcity in other languages except English; we present the manner in which we compiled a multilingual corpus of blog posts on different topics of interest in three languages - Spanish, Italian and English. The second issue we tried to solve was the coarse-grained annotation schemas employed in other models. Thus, we describe the new annotation schema that we built in order to capture the different subjectivity/ objectivity, emotion/opinion/ attitude aspects we are interested in at a finer-grained level. We explain the need for a more detailed annotation model, the sources and the reasons taken into consideration when constructing the corpus and its annotation. Thirdly, we address an aspect that is strongly related to the blogs annotation: due the presence of “copy and pastes” from news articles or other blogs, the frequent quotes, we include the annotation of both the directly

power control, obstructive/conductive and active/passive. Further on, we distributed the sentiments within our list into the Scherer slots creating other smaller categories included in the abovementioned general ones. The result of this division is shown in Table 1:

Group	Emotions
Criticism	Sarcasm, irony, incorrect, criticism, objection, opposition, scepticism.
Happiness	Joy, joke.
Support	Accept, correct, good, hope, support, trust, rapture, respect, patience, appreciation, excuse.
Importance	Important, interesting, will, justice, longing, anticipation, revenge.
Gratitude	Thank.
Guilt	Guilt, vexation.
Fear	Fear, fright, troubledness, anxiety.
Surprise	Surprise, bewilderment, disappointment, consternation.
Anger	Rage, hatred, enmity, wrath, force, anger, revendication.
Envy	Envy, rivalry, jealousy.
Indifference	Unimportant, yield, sluggishness.
Pity	Compassion, shame, grief.
Pain	Sadness, lament, remorse, mourning, depression, despondency.
Shyness	Timidity.
Bad	Bad, malice, disgust, greed.

Table 1: Alternative dimensional structures of the semantic space for emotions

As we can notice from Table 1, the division among emotions has been done at a superficial level. However, our intention is to improve this initial approach and make a clear distinction between opinions and emotions. This separation would permit a double analysis and evaluation process – on emotion detection and opinion mining. These two tasks could converge or diverge in the approach, depending on the considered necessity. After having inserted the corresponding emotion, we have to select the kind of term we are labelling. It could be an adjective, a preposition or an adverb. Another term is added (modifier) indicating whether the annotated element influences the polarity of another one. Another possibility is to mark out terms as a noun or verb but we also detected cases of special use of punctuation signs, spelling mistakes or capital letter. In fact, we are convinced that capital letter and a special election of punctuation (“”, “/”, “/”, “/”) could provide precious information about the sentiment intensity. Spelling mistakes are also useful for two main reasons: on the one hand an error corresponds to a word and thus, thanks to its detection we are able to associate such a mistake with its corresponding correct word and, on the other hand, we can also obtain information about the writer/speaker (social class, education, etc.). Table 2 shows the complete model for EmotiBlog version 1.0 and Figure 3 the adjective with all its elements.

Elements	Description
Objective speech	Source
Subjective speech	Level, emotion, phenomenon, polarity, and source.
Adjectives Adverbs Prepositions	Level, emotion, phenomenon, modifier/not, polarity, and source
Verbs Names Punctuation	Level, emotion, phenomenon, polarity, and source.
Capital letter	Level, emotion, phenomenon, polarity, and source.
Other languages	Latin or English, level, emotion, phenomenon, modifier/not, polarity, and source.
Spelling mistakes	Correction, level, emotion, phenomenon, modifier/not, polarity, and source
Phrase type	Type: collocation, saying, slang, title, and rhetoric.
Emotions	Accept, anger, anticipation, anxiety, appreciation, bad, bewilderment, comfort, compassion, confidence, consternation, correct, criticism, disappointment, discomfort, disgust, despondency, depression, envy, enmity, excuse, force, fear, fright, good, grief, guilt, greed, hatred, hope, irony, interesting, important, incorrect, joy, jealousy, joke, justice, longing, lament, mourning, malice, opposition, objection, ostentation, patience, purity, preference, revenge, rivalry, remorse, rapture, respect, revendication, rage, rejection, support, scepticism, smug, sadness, shame, surprise, sluggishness, sarcasm, troubledness, thank, trust, timidity, unimportant, vexation, will, wrath, warning, yielding.
Anaphora	Type and source

Table 2: The complete model for EmotiBlog 1.0

Figure 3 shows the breakdown of an element. In this case the adjective.

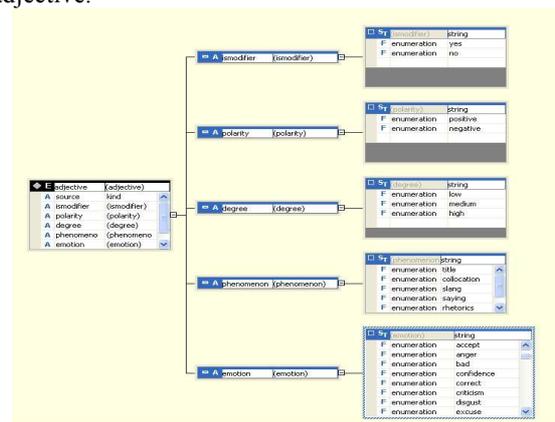


Figure 3: Breakdown of the adjective element

Finally, as we can appreciate in Table 2, when labelling the corpus, the annotator underlines cases of sayings and collocations. We can define a saying as a well-known and wise statement, which often has a meaning that is different from the simple meanings of the words it contains, while a

collocation is a word or phrase which is frequently used with another word or phrase, in a way that sounds correct to people who have spoken the language all their lives, but might not be expected from the meaning. We annotated and collected them in order to have a list, which will be extended in next annotations, because we are convinced that it could be a precious source of information the system could learn.

Further on, we present the lists of sayings and the collocations detected in the corpus object of our study:

- *Sayings*: abrirse camino, entrar en juego, aprender sobre la marcha, brotar como hongos, coger el toro por los cuernos, como de todos es conocido, condenar a muerte, correr el riesgo, dar por sentado, dar sus frutos, desaparecer bajo las aguas, desviar la atención, echar por tierra, echar marcha atrás, en un abrir y cerrar de ojos, enfrentar el peligro, ser de risa, ser hora de, estar en las manos, estar en contra, adquirir vida propia, hacer caso omiso, hacer su fortuna, hacer gala, jugar un papel muy importante, la herida sigue abierta, le sigue de cerca, les salió el tiro por la culata, lo diga quien lo diga, mantener en solitario, morir de hambre o de malnutrición, no dar ni un céntimo, no hay duda alguna, no hay que perder de vista, no le ha temblado la mano, dejar plantados, pese a que, poner en peligro, por algo fue, presionar en la dirección contraria, poner en marcha, poner en duda, sabotear al unísono, vestirse ahora de ángel, tener muy en cuenta, terminar en fracaso, tener hipo, traer suerte, vivir al límite, vivir y dejar vivir.
- *Collocations*: caballo ganador, tratamientos químicos, prueba contundente, crisis ambiental, menos mal, por lo menos, una vez por todas, sentencia de muerte, papel mojado, círculo vicioso, en plena forma, punto de inflexión, aviso a navegantes, como mínimo, camino de vuelta a casa, sueño americano, eje del mal, pasos adelante, postura impropia, primera línea, sin más sustancia, tratamientos químicos, crisis ambiental, palabras vacías, craso error, por arte de birlibirloque, sentencia de muerte, papel mojado, menos mal, por lo menos, gracias a, primer puesto, caballo ganador, en vez de, única manera, el sector de moda, tan eficiente, tigre de papel.

7. Annotation problems

The creation of an exhaustive annotation scheme for emotion is a very complex process, and it also involves different stages. The first one consists in analysing the corpus in order to understand what elements have to be annotated. The second step consists in organizing how to label them; next, we apply this “rough” model to our corpus and we annotate some texts, and then we return to the annotation scheme, we check all the possible errors with the aim of improving the annotation model. Finally, with this second version of the annotation scheme, we label the entire corpus. Finally, then we evaluate the inter-annotator agreement, and employing the results we improve the annotation scheme. Furthermore emotion annotation is a difficult task with a poor level of inter-annotator agreement, for different reasons. First of all, there are no fixed rules about the way in which words should be annotated. The instructions describe the annotations of specific examples, but they do not state that specific words should always be annotated in a certain way. On the contrary, sentences should

be interpreted with respect to the context in which they appear. In fact, the annotation guidelines show examples, but there are no fixed rules to apply the model. Finally, annotators should be extremely consistent during the annotation process. As we can deduce, it is a complex and high-responsibility task in which they have the total call on of the final decision and as a consequence of the annotation quality. Finally, there are some elements that are undoubtedly objective, such as the detection of words, verbs, adjectives, phrases, etc, but, establishing the kind of emotion, its level or its boundary is quite complex. Objective speech events are less difficult to detect and label because the annotator does not have to insert any element inside these sentences.

8. Evaluation and discussion

In order to measure the inter-annotator agreement regarding the different elements and attributes of the annotation scheme, two annotators (A and E) labelled independently 100 texts, a total of 30.000 words. We took into consideration the Spanish corpus about the ratification of the Kyoto protocol. The first step in measuring the agreement is to verify that annotators agree on which expressions should be marked. The evaluation process is extremely complex because when annotators identify the same expression in the text, they could differ in their marking of the expression boundary. Moreover there is no guarantee that the annotators will identify the same set of expressions.

Measure used

The measure we would like to use to calculate the inter-annotator agreement are the kappa value (when statistic classes are present), and the observed agreement (when non statistic classes are present). Kappa is computed according to Cohen method [29]and [30]:

$$Kappa = \frac{\text{observed proportion of agreement} - \text{chance expected proportion of agreement}}{1 - \text{chance expected proportion of agreement}}$$

But then we realized that this measure was not the appropriate for our corpus because it does not allow measuring the inter-annotator agreement for each element and its corresponding attributes, thus we decided to use the following measure:

$$agr(a||b) = \frac{|A \text{ matching } B|}{|A|}$$

The elements and attributes we evaluated are the ones that make up the model, and they are listed below:

Objective speech event

The first element we evaluated is the agreement regarding the objective speech event. Those sentences express objective facts without any subjective shade of meaning. As a consequence the annotator marks them as objective and s/he has not to add any attribute.

Adjective

Adjectives are words that describe a noun or a pronoun. In

EmotiBlog this element has different attributes that are: sentiment, level, polarity, phenomenon, modifier, and source.

Adverb

An adverb is a word which describes or gives more information about a verb, adjective, adverb or a phrase.

The elements we evaluated for this element are: level sentiment, polarity, phenomenon, modifier, and source.

Noun

A noun can be defined as a word that refers to a person, place, thing, event, substance or quality. The elements we took in consideration are the following: level, sentiment, polarity, phenomenon, and source.

Preposition

A preposition can be defined as a word which is used before a noun. We include in these elements the following attributes: level sentiment, polarity, phenomenon, modifier, and source.

Verb

A verb has the function of describes an action, condition or experience. We added to this element the following attributes: level, sentiment, polarity, phenomenon, and source.

Spelling mistakes

Spelling mistakes are errors produced in the spelling of a word. It is very important to label it in order to understand to what word they are referring to and they also could provide us with a little information about the background of the speaker/writer. The attributed we associate to this element are correction, level, emotion, phenomenon, modifier/not, polarity, and source.

Capital letter

Capital letters are significant because they are like a shout and as a consequence they assign to the phrase an higher level of intensity.

English and Latin

They are phrases that are used to give to the discourse a shade of meaning. They could be sarcastic or serious.

Punctuation

We decided to label some signs of punctuation that have the function to modify the level of the sentiment of the phrase. These signs could be ! /? /'/, etc.

Phrase

A phrase can be seen as a group of words which is part rather than the whole of a sentence. It could be a colloquialism, idioms, slang, title, or rhetoric.

Anaphora

We label the cases of anaphora that refer to other blog posts or conversations. We thought they could be useful to understand the sense of the sentence or post we are labelling. In this case, we indicate the type of anaphora and the post it is referring to. Table 3 shows the inter-annotator agreement results:

Annotation	a	b	a b	b a	Av.
Noun	A	E	0.78	0.75	0.765
Adjective	A	E	0.78	0.61	0.68
Verb	A	E	0.86	0.74	0.80
Adverb	A	E	0.83	0.76	0.79
Preposition	A	E	0.86	0.67	0.76
Punctuation	A	E	0.78	0.89	0.83
Capital letter	A	E	0.66	1	0.83
Spelling mistake	A	E	0.8	0.77	0.78
English	A	E	0.27	1	0.63
Latin	A	E	0.66	0.66	0.66
Phrase	A	E	0.52	0.66	0.59
Objective	A	E	0.76	0.73	0.74
					5
					0.68

Table 3: Inter-annotator agreement results

At the current state of the project these results are preliminary. In fact our aim is to improve the annotation scheme, to label the rest of the corpus we collected, and to improve the annotation scheme.

9. Conclusions and future work

In this article we presented the first version of *EmotiBlog*, a collection of blog posts we compiled for three different languages (English, Spanish and Italian), for three distinct topics. We present a fine-grained annotation scheme for the detection of subjective elements. We show the manner in which the gathered corpus was annotated using the described scheme. We evaluated the quality and constancy of annotation, focusing on the detection and elimination of existing problems. The results obtained are comparable to other similar works in the field; moreover, taking into consideration the fine granularity and complexity of the annotation scheme, we believe that the results are very encouraging. Further work includes the annotation of the remaining corpus, of similar corpora for other languages (due to the high flexibility of the employed model, this is easily achievable) and of text belonging to different genres. As far as the use we will give to the annotated corpus, our intentions are to employ it as a learning corpus, or the training of machine-learning algorithms, as well for the automatic recognition of fixed linguistic phenomena. Although the task, as we have seen, is highly domain-dependent, the high scale of possible annotations we employed make the corpus a valuable resource for opinion mining systems in the field.

10. Acknowledgements

This research has been partially funded by the Spanish Government under the project Intelligent, Interactive and Multilingual Text Mining based on Human Language Technologies (TIN2006-15265-C06-01), by the European project Question Answering Learning technologies in a multiLingual and Multimodal Environment (FP6 IST 033860) and by the University of Alicante scholarship of Alexandra Balahur.

11. References

- [1] Strapparava, C. Valitutti, A.. "WordNet-Affect: an affective extension of WordNet". In Proceedings of the 4th International Conference on Language Resources and Evaluation, LREC. 2004.
- [2] Esuli, A., Sebastiani, F. "SentiWordNet: A Publicly Available Resource for Opinion Mining". In Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC 2006, Genoa, Italy. 2006.
- [3] Cerini, S., Compagnoni, V., Demontis, A., Formentelli, M., and Gandini, G. "Language resources and linguistic theory: Typology, second language acquisition", English linguistics (Forthcoming), chapter Micro-WNOP: A gold standard for the evaluation of automatically compiled lexical resources for opinion mining. Franco Angeli Editore, Milano, IT. 2007.
- [4] Balahur, A.; Montoyo, A. "Applying a Culture Dependent Emotion Triggers Database for Text Valence and Emotion Classification". In Proceedings of the AISB 2008 Symposium on Affective Language in Human and Machine, Aberdeen, Scotland. 2008.
- [5] Wiebe, J. M. "Tracking point of view in narrative". Computational Linguistics, vol. 20, pp. 233–287, 1994.
- [6] Wiebe, J., Wilson, T. and Cardie, C. "Annotating expressions of opinions and emotions in language". Language Resources and Evaluation 2005.
- [7] Riloff, E., Wiebe, J. "Learning Extraction Patterns for Subjective Expressions". In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing. 2003
- [8] Stoyanov V., Cardie C., Litman D., and Wiebe J. "Evaluating an Opinion Annotation Scheme Using a New Multi-Perspective Question and Answer Corpus". AAAI Spring Symposium on Exploring Attitude and Affect in Text. 2004.
- [9] Turney, P. "*Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews*". ACL 2002: 417-424. 2002.
- [10] Pang, B., Lee, L., Vaithyanathan, S. "Thumbs up? Sentiment classification using machine learning techniques". In Proceedings of EMNLP-02, the Conference on Empirical Methods in Natural Language Processing. 2002.
- [11] Dave, K., Lawrence, S., Pennock, D. "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews". In Proceedings of WWW-03. 2003.
- [12] Mullen, T., Collier, N. "Sentiment Analysis Using Support Vector Machines with Diverse Information Sources". In Proceedings of EMNLP. 2004.
- [13] Chaovalit, P., Zhou, L.. "Movie Review Mining: a Comparison between Supervised and Unsupervised Classification Approaches". In Proceedings of HICSS-05, 2005.
- [14] Goldberg, A.B., Zhu, J. "Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization". In HLT-NAACL 2006 Workshop on Textgraphs: Graph-based Algorithms for Natural Language Processing. 2006.
- [15] Ng, V., Dasgupta, S. and Arifin, S. M. "Examining the Role of Linguistics Knowledge Sources in the Automatic Identification and Classification of Reviews". In the proceedings of the ACL, Sydney, 2006.
- [16] Gamon, M., Aue, S., Corston-Oliver, S., Ringger, E. "Mining Customer Opinions from Free Text". Lecture Notes in Computer Science. 2005.
- [17] Cui, H., Mittal, V., Datar, M.. "Comparative Experiments on Sentiment Classification for Online Product Reviews". In Proceedings of the 21st National Conference on Artificial Intelligence AAAI. 2006.
- [18] Hatzivassiloglou, V., Wiebe, J. "Effects of adjective orientation and gradability on sentence subjectivity". In Proceedings of COLING. 2000.
- [19] Wiebe, J., Riloff, E.. "Creating Subjective and Objective Sentence Classifiers from Unannotated Texts". In Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing). 2005.
- [20] Wilson, T., Wiebe, J., Hwa, R.. "Just how mad are you? Finding strong and weak opinion clauses". In: Proceedings of AAAI. 2004.
- [21] Kim, S.M., Hovy, E. "Determining the Sentiment of Opinions". In Proceedings of COLING. 2004.
- [22] Lin, W.H., Wilson, T., Wiebe, J., Hauptman, A. "Which Side are You On? Identifying Perspectives at the Document and Sentence Levels". In Proceedings of the Tenth Conference on Natural Language Learning CoNLL.2006.
- [23] Turney, P., Littman, M. "Measuring praise and criticism: Inference of semantic orientation from association". ACM Transactions on Information Systems 21. 2003.
- [24] Stoyanov, V and Cardie, C. "Toward Opinion Summarization: Linking the Sources". COLING-ACL. Workshop on Sentiment and Subjectivity in Text. 2006.
- [25] Hu, M., Liu, B. "Mining Opinion Features in Customer Reviews". In Proceedings of Nineteenth National Conference on Artificial Intelligence AAAI. 2004.
- [26] Wiebe, J., Wilson, T and Cardie, C. "Annotation Expressions of Opinions and Emotions in Language". Language Resources and Evaluation. 2005.
- [27] Scherer, K. R. "What are emotions? And how can they be measured?" Social Science Information, 44(4), 693–727. 2005.
- [28] Russell, J.A. "Pancultural aspects of the human conceptual organization of emotions". Journal of Personality and Social Psychology 45: 1281–8. 1983.
- [29] Carletta, J. "Assessing agreement on classification task: the kappa statistic". Computational Linguistics, 22(2):249–254. 1996.
- [30] Cohen, J. "A coefficient of agreement for nominal scales". Educational and Psychological Measurement, 20:37–46. 1960.

Learning a Hyperplane using Genetic Algorithm for Sentiment Classification in Text

Anjum Gupta. Spawar Systems Center, San Diego. anjum.gupta@navy.mil

Abstract—In the world of ubiquitous text data in the form of blogs, message boards and various other forums, the task of automated sentiment classification has become an important and challenging problem. In this paper, we propose a method of learning weights of different adjectives used to classify positive and negative movie reviews. We see that the hyperplane learned using genetic algorithms out performs a hyperplane found using a perceptron algorithm and out performs using various word lists where each word has the same weight. In this paper, we show the results of a learning using genetic algorithm by working on a data set involving 2000 movie reviews, written online by anonymous users. We also propose some methods to reduce the number of words or features using variations in the objective function.

I. INTRODUCTION

Text classification has been a hot research topic for years. Sentiment classification in the text documents is a fairly new but actively pursued research topic. Sentiment classification is a challenging task, given the inherent ambiguity in the natural languages for expressing sentiments. The presence of sarcasm, euphemism and presence of negations or multi-sentiment sentences makes the sentiment classifications a non-trivial task. The approach for text classification ranges from simple “bag of words” models, by counting the number or presence of some carefully chosen “keywords” such as collection of predefined adjectives, to parsing a sentence into its component by forming elaborate context free grammar rules. Another common classification method is naive bayes classification. Even though the words are taken out of context and treated as independent entities in both many cases of text classification, it works well in most cases. Our approach also looks at words out of context and treats documents as a bag of word. However, our method does not treat each word equally and uses genetic algorithm to learn the effectiveness or weight of each word in the sentiment classification. Genetic algorithms have been given little attention for text mining and sentiment classification. In our experiments and on the movie reviews data set, classification hyperplane learned using genetic algorithm performs better than a hyperplane that was learned using perceptron. Another way we used genetic algorithm was to reduce the dimensions of the feature vector. In text mining, the dimensions of the feature vectors are often in thousands if not tens of thousands. Our approach can also be used to pick the most important features thus greatly reducing the dimensions of the feature. In our classification task, we exceed or meet the accuracy levels with only a few hundred features that others have achieved with over ten thousand features.

II. RELATED WORK

The recent work done by Bo Pang and Lillian Lee at Cornell is among the most closely related work to the sentiment classification in text discussed in this paper [7]. Although, they have not tried an approach based on genetic algorithms, their work uses the same dataset with similar objectives. In addition to this, there has been work done on extracting the polarity of adjectives by Hatzivassiloglou and McKeown [8]. They introduce a method in assigning positive and negative polarity to adjectives outside of context. A sentence level polarity detection work is done by Cecilia Alm et. al. and J. M. Wiebe et. al. [11], [9] by trying to classify emotions in sentences of a fairy tale. Some other papers utilizing various machine learning algorithms with good empirical results include [10] and [12].

III. BACKGROUND

In our paper, we use a data set of 2000 movie reviews that is maintained by Bo Pang at Cornell University [1]. Many papers have utilized this data set [3] [2]. This data set has now become one of the standard data sets both for natural language processing as well as sentiment classification. It is also a part of python language based natural language toolkit, that contains about 20 standard text data sets that are used often in the text mining and natural language processing community. The data set contains 2000 hand labeled movie reviews, originally taken from internet movie database (imdb) website message boards. The movie reviews are split among two classes, namely, positive and negative. A separate data set exists where the rating for each review is given on a scale from 1 to 5, where 1 represents the most negative and 5 represents the most positive reviews. In this paper, we are using binary classification data set of negative and positive polarity of sentiments. We use genetic algorithms to train a classifier that works as a separating hyperplane. We compare our results with the results of the paper by Bo Pang and Lilian Lee [2]. Their original paper used different machine learning approaches to analyze and correctly classify the negative or positive sentiments in the data. We also compare our results of the hyperplane classification with a more common method of training a hyperplane, that is, perceptron.

IV. FEATURE SELECTION

Any classification task will involve some degree of careful feature selection. In the case of text, it also means identifying a method to represent a text document as a numerical vector using some key features. We use a list of positive and negative

adjectives as our feature. More specifically, we make a list of positive and negative adjectives and then count the number of these words in each document. The numerical vector is obtained from the text document by simply putting the counts of each adjective in its column. We get a numerical vector that has the number of dimensions equal the total adjectives in the two lists. We used two primary methods of choosing adjectives, one using a sentence polarity data set and second is using the General Inquirer list of words [4].

A. Choosing Adjectives: Sentence Polarity

The movie review data set also includes 5000 list of positive and negative individual sentences. These negative and positive sentences are in fact, one line reviews, by either critics or users that are normally used as a headline of the review or includes the one liner comments by the professional critic, such as “A must see! Best movie of the year.” Our first method, similar to the one used in [2], is by extracting all the adjectives from the positive sentence list and assuming them to be positive and extracting all the negative adjectives and assuming them to negative. This is not a perfect method of categorizing adjectives, since even a one liner negative review could have a positive sounding word such as “You will enjoy this movie because it will make you laugh at the director.” We can safely assume, however, that these will be rare and safely assume that the common adjectives present in the one liner positive reviews will be positive. Since a list of 5000 sentences generate a very high number of adjectives, we narrowed the list by removing all the adjectives that occur frequently in both negative and positive sentences. The criteria for determining if an adjective occur frequently in both positive and negative review was a based on a simple counting of the occurrence of an adjective in positive and negative review. This criteria is flexible and can be adjusted based on the desired number of adjectives in the feature vector. We also ignored the adjectives that occurred in less than 0.1% of the sentences. These two approaches reduced our adjective counts to approx. 400.

B. Choosing Adjectives: General Inquirer

General Inquirer (GI) is a list of 11 thousand words labeled with many different categories, including positivity and negativity of the word. We chose all the adjectives or adverbs from general inquirer that fell into categories that could represent positive or negative sentiments. These categories included “positive, negative”, “pain, pleasure”, “vice, virtue” etc. We could form different combination of features by combining words from different categories such as all the words that are labeled “positive” and “pleasure” etc. The list of words obtained using different categories of General Inquirer ranged in numbers from 1174 to 150. Table I shows the number of adjectives obtained using different methods.

C. Stemming

Finally, in another refinement of our feature set, we used porter stemming algorithm to used stemming to count the words. We used porter stemmer algorithm to obtain the root

Using Sentence Polarity	184 + 184 = 368
GI - Positive & Negative	541 + 633 = 1174
GI - Positive and Strong & Negative and Strong	187 + 174 = 361
GI - Positive and Positive Affinity& Negative and Negative Affinity	53 + 94 = 147

TABLE I

TABLE SHOWING THE NUMBER OF ADJECTIVES OBTAINED USING DIFFERENT METHODS

words. Stemming the words made sure that if we had “enjoy” as one of our adjectives, we also matched words such as “enjoyment”, “enjoying”, “enjoyable” etc. Stemming seemed to increase the effectiveness of our feature sets significantly.

V. LEARNING THE CLASSIFIER

After selecting the appropriate features, the next task is selecting a classifier. We classified the sentiments using three different methods. The three different methods are explained below in the following subsections. We split our data set of 2000 movie reviews into a training set of 1400 reviews and testing set of 600 reviews. The positive and negative reviews are split evenly in both training and the testing set.

A. Positive and Negative Adjective Count

This approach required simply counting the number of positive adjectives and the number of negative adjectives. To classify a document either as positive or negative, we simply subtracted the number of negative adjectives from the number of positive adjectives. If the resulting number was a positive, then the text document was labeled positive, if the result was negative, then it was labeled as negative. In case the result was zero, no label was assigned to the document. This approach treated all the words equally, that is all the positive adjectives counted as +1 and all the negative adjectives counted -1. Our next approach was to vary these weights of the words.

B. Learning Weights

Our goal was to learn the weights of each word. A weight is a numerical value that can range from -1 or +1 quantifying the polarity (negative or positive) of the given word. For example, if the weight of the word “laugh” is 0.39, it will represent that the word “laugh” carries a 0.39 positive polarity. The final classification of document was given by multiplying the weights of the adjectives with the counts of the adjectives in the document and adding these multiplied number together to obtain either a negative or a positive number. It is like taking a dot product of the weights vector with the adjective counts. The final classification can be given by the following equation, where W is the weight vector, C is the counts vector and T is the sentiment label to be either +1 or -1. $|W|$ and $|C|$ is the number of adjectives selected as features.

$$T = \text{sign}(W \cdot C) \quad (1)$$

Since it is simply an equation of the plane, our task becomes learning the classifying hyperplane. We used the following two methods to learn the classifying hyperplane.

1) *By Genetic Algorithm:* In case of genetic algorithms, we initialize a population 500 randomly generated weight vectors, where each value ranged from -1 to 1. We evaluate the value of an cost that was to be minimized. The cost function was defined as

$$Cost = |TrueLabel - PredictedLabel| \quad (2)$$

It is the absolute value of the difference between the true label and the predicted label. So the positive documents that were classified as negative and vice versa contributed +2 to the cost, where as the positive or negative documents that were classified as zero or neutral contributed +1 to the cost function. Using our genetic algorithm, we select the top 10% of the population members that had the lowest cost. We then “bred” these members and generated a new batch of 500 members. The next generation members were produced by one of the three methods. Generation G_{t+1} is produced using the top 10% vectors from generation G_t .

- 1) Crossing over 20% of the randomly selected values from a pair from $G + t$ to produce two new vectors of G_{t+1} .
- 2) Replacing 20% of the randomly selected values in G_{t+1} by an average values of the selected “genes” from a pair from G_t .
- 3) Adding a random number $m \sim N(0, 0.2)$ to a random 20% of the values in a vector in G_t to get a vector for G_{t+1} .

In addition to these variations, the top 2% of the population from G_t was carried over to G_{t+1} without any changes. This way we were guaranteed never to increase the value of the cost function during the optimization.

2) *By Perceptron:* Our last method for the classification was using the perceptron algorithm [5] to learn the separating hyperplane. Perceptron algorithm, although considerably faster than genetic algorithm, does not perform very well in this case. Since the data is not linearly separable, perceptron algorithm fails to converge of a reasonable solution. We see this in the results listed in table II.

VI. CLASSIFICATION RESULTS

The results showed a lot of variation in their accuracy. The tabulated results show an average of some of the best runs of all three algorithms. Most of the variation was in the Genetic Algorithm result and the perceptron result. The variation in the “Adjective count” method only arrived due to the changes in the random selection of the training and testing set. Genetic algorithm showed variance, most likely, due to many local minimums present in the high dimensional space. Perceptron also performed poorly since it could not converge in the absence of linearly separable classes. One interesting aspect that we noticed was that in general our method of selecting adjectives from the sentence polarity data set performed much better over the list from general inquirer. It may be due to the fact that sentence polarity data set contains positive and negative adjective that were more relevant for the movie reviews, whereas the lists from general inquirer

contained many more general adjectives that had no bearing on the movie reviews.

Feature Set	Adjective Counts	Genetic	Perceptron
Using Sentence Polarity	66.2%	80.1%	73.0%
GI - Positive & Negative	58.2%	76.1%	69.3.1%
GI - Positive and Strong & Negative and Strong	60.1%	77.1%	69.1%
GI - Positive and Positive Affinity & Negative and Negative Affinity	58.2%	74.2%	68.0%

TABLE II

TABLE SHOWING THE ACCURACY PERCENTAGE OF THE SENTIMENT CLASSIFICATION FOR THE MOVIE REVIEWS DATA

VII. REDUCING DIMENSIONS USING GENETIC ALGORITHM

We also propose using genetic algorithm for reducing the dimensions or selecting the most relevant features. To reduce the number of features, we modify our cost function. We use the modified cost function as follows,

$$Cost = |TrueLabel - PredictedLabel| + |W_{nz}| * 0.5 \quad (3)$$

where $|W_{nz}|$ is the number of features with non-zero weight. So cost increases by half for every new non-zero feature used to classify the documents. We also used a more aggressive method of reducing the features by starting all the population members to have all the initial weight equal to zero. This adds the effect of incrementally adding the features that improve the classification. While incrementally adding the features using genetic algorithm in this way, we saw the classification accuracy numbers, comparable to that listed in table II with less than 200 or in some cases even less than 100 adjectives.

VIII. CONCLUSION & FUTURE WORK

Genetic algorithms have not been tried very much in text classification. Our work shows that genetic algorithms hold promise in reducing the number of features or selecting only the relevant features for the text classification. In our experiments, genetic algorithms also produced the most optimal solution among the three methods tried. Genetic algorithms were very effective in reducing the dimensions of the data, especially when used to incrementally add the features that improved on the classification accuracy.

In the future, we plan to try different variation of the cost function and a more constrained vector space, such as weights only being integers etc. We also want to compare the results to linear support vectors classification.

IX. ACKNOWLEDGEMENTS

I would like to thank my interns, Daniel Garcia and Jason Cole, for their help with some programming and general organization of this project. I would like to thank Sparwar Systems Center at San Diego, for providing the funding for this project.

REFERENCES

- [1] <http://www.cs.cornell.edu/People/pabo/movie-review-data/>
- [2] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan, *Thumbs up? Sentiment Classification using Machine Learning Techniques*, Proceedings of EMNLP 2002.
- [3] Bo Pang and Lillian Lee, *Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales*, Proceedings of ACL 2005.
- [4] Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, Daniel M. Ogilvie, and associates. *The General Inquirer: A Computer Approach to Content Analysis*. The MIT Press, 1966.
- [5] Rosenblatt, Frank (1958), *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain*, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, pp. 386-408.
- [6] Whitley, D. (1994). *A genetic algorithm tutorial. Statistics and Computing* 4, 6585.
- [7] Bo Pang and Lillian Lee. *Opinion mining and sentiment analysis*, Now Publishers, 2008.
- [8] Vasileios Hatzivassiloglou and Kathleen R. McKeown. *Predicting the semantic orientation of adjectives* In Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics, pages 174-181, Morristown, NJ, USA, 1997.
- [9] Alm, Cecilia O. and Roth, Dan and Sproat, Richard. *Emotions from Text: Machine learning for text-based emotion prediction* HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing Vancouver, Canada, 2005, pp. 347-354.
- [10] Theresa Wilson, Janyce Wiebe and Rebecca Hwa, *Just how mad are you? Finding strong and weak opinion clauses*, In Proceedings of AAAI, 2004 761-769.
- [11] J. M. Wiebe and E. Riloff, *Creating subjective and objective sentence classifiers from unannotated texts*, in Proceedings of the Conference on Computational Linguistics and Intelligent Text Processing (CICLing), number 3406 in Lecture Notes in Computer Science, pp. 486-497, 2005.
- [12] H. Yang, L. Si, and J. Callan, *Knowledge transfer and opinion detection in the TREC2006 blog track*, in Proceedings of TREC, 2006.

SESSION

UNSUPERVISED DATA MINING

Chair(s)

Dr. Stefan Lessmann

Outlier Detection for Large High-Dimensional Categorical Data using Non-Derivable and Non-Almost-Derivable Sets

A. Koufakou, J. Secretan, M. Fox, G. Gramajo, G.C. Anagnostopoulos, M. Georgiopoulos

Abstract—Detecting outliers in a dataset has many important applications, such as credit card fraud detection or network intrusion detection. Many previous methods assume numerical data and detect outliers based on pair-wise distances among points. Recently, outlier detection methods were proposed for categorical and mixed-attribute data using Frequent Itemsets (FIs). These methods face challenges for large high-dimensional data, where many FIs are generated. To address this issue, we propose to use a well-known condensed representation of FIs, Non-Derivable Itemsets (NDIs), and an approximation of NDIs, Non-Almost Derivable Itemsets (NADIs). Both methods use a much smaller collection of sets than FIs to compute an anomaly score for each point. Experiments on real-life data show that the NDI/NADI methods offer substantial advantages in terms of speed and scalability over the FI-based method. More significantly, the NDI-based method exhibits similar detection accuracy compared to the FI-based method, while the NADI-based method exhibits accuracy close to the one achieved by the NDI-based method and further speed improvements.

1. Introduction

The task of detecting outliers has received significant attention in many applications, such as network intrusion detection [9]. Many traditional data mining fields, e.g. Association Rule Mining [2], focus on extracting common or frequent patterns in the data; in contrast, outlier detection methods aim to detect patterns that occur infrequently in the data.

Most previous research focuses on numerical attributes, and usually involves calculating similarities or distances between data points. However, in categorical data, methods that assume numerical attributes map each categorical value to a numerical value in order to compute distances, which is not a simple process (e.g., distance between values such as ‘TCP Protocol’ and ‘UDP Protocol’ [15]).

A second issue is that many applications for mining outliers involve large data, thus the proposed methods must scale well with the number of data points and dimensionality [14]. Data may also be distributed over various geographical sites; this makes data transfers difficult, due to data size,

ownership and control issues. Therefore, algorithms for such data must minimize data scans, as well as communication and synchronization.

Recently, methods for detecting outliers in categorical and mixed-attribute data were proposed, e.g. [15], [13]. Some of these methods make use of the *Frequent Itemset Mining* (FIM) concept [2]. The goal of FIM is to extract sets of items (categorical values) that co-occur frequently in the data, given a σ threshold value. FIM-based outlier detection methods first extract all Frequent Itemsets (FIs) from the data, and then assign an outlier score to each data point based on the FIs it contains. Outliers are likely to be the points that contain relatively few of the frequent patterns. However, FIM algorithms face problems for *dense data*, e.g. census data [21]. Dense datasets contain many strong correlations, and are typically characterized by items with high frequency and many frequent patterns [21]. As a result, there might be a large number of FIs to generate and store. We note that the specific FIM algorithm used is not as important; the issue addressed in this paper is due to the large number of FIs generated, not due to design or implementation of the FIM algorithm. As we show in Section 4, this issue persists even if we constrain the length of generated sets as in [15], [13].

In this paper, we use a smaller collection of sets instead of all FIs in the data, in order to detect outliers efficiently in large high-dimensional data. Specifically, we use Non-Derivable Itemsets (NDIs) which have been shown to lead to large performance gains over FIs [7]. As we discuss in Section 3, the way the NDI algorithm prunes candidate sets makes the NDI concept well-suited to the task of outlier detection. Furthermore, we propose a second outlier detection method that uses the Non-Almost Derivable Itemsets (NADIs) [19] instead of NDIs. NADIs approximate NDIs based on a user-entered parameter, δ .

Specifically, we propose two outlier detection methods: one based on a condensed representation of FIs, NDIs, *FNDI-OD*, and one based on a δ -approximation of NDIs, *FNADI-OD*. Moreover, we discuss how NDI/NADIs suit the task of detecting outliers. We compare these methods with the FI-based OD, *FI-OD*, given several data sets and parameter values. Our proposed methods are shown to be much faster and have similar detection accuracy compared to FI-OD. Specifically, *FNDI-OD* is significantly faster than FI-OD, even with large high-dimensional data and low σ values. Finally, *FNADI-OD* presents higher runtime gains compared to *FNDI-OD* usually at a small cost of detection accuracy. We also conduct experiments to explore the tradeoff of accuracy versus speed for various δ values.

Anna Koufakou, Jimmy Secretan, Michael Georgiopoulos are with the School of EECS, University of Central Florida, Orlando, FL, USA (email: {akoufako, michaelg}@mail.ucf.edu). Michelle Fox is with the School of EECS, Milwaukee School of Engineering, Milwaukee, WI, USA. Gary Gramajo is with the School of Mathematics, Florida State University, Tallahassee, FL, USA. Georgios C. Anagnostopoulos is with the ECE Dept, Florida Institute of Technology, Melbourne, FL, USA.

The organization of this paper is as follows: In Section 2 we provide a review of previous work. In Section 3, we describe the FI-based and NDI-based outlier detection algorithms. Section 4 contains our experiments and results. Finally, in Section 5, we summarize our work and provide concluding remarks.

2. Previous Work

Existing outlier detection work can be categorized as follows. *Statistical-model* based methods assume that a specific model describes data distribution [3], which leads to the issue of estimating a suitable model for a particular dataset and application. *Distance-based* methods compute distances among data points [12]; these methods become impractical for large data. A method based on randomization and pruning in [4] has complexity close to linear in practice. *Clustering* techniques assume that outliers are points that do not belong to formed clusters; however, these methods focus on clustering, not outlier detection [4]. *Density-based* methods identify outliers as those lying in relatively low-density regions. In [6], a *degree* of outlierness, local outlier factor (*LOF*), is assigned to each point based on the local density of the area around the point and the local densities of its neighbors. Although these methods can detect outliers missed by distance-based methods, they face a challenge with sparse high-dimensional data. Other examples of outlier detection work include Support Vector methods [17] among others. Finally, most of these methods require data to be in the same location, which is impractical for distributed data.

Most of these techniques are suitable for numerical or ordinal data that can be easily mapped to numerical values; in the case of categorical data, there is little sense in computing distances [15]. Another limitation of previous methods is the lack of scalability with respect to number of points and/or data dimensionality. A fast and scalable method for categorical data, AVF, is presented in [14]. The methods in [15], [11], [13] use FIs to assign an outlier score to each data point based on the subsets this point contains.

The authors in [15] propose a distributed and dynamic outlier detection method for data with both categorical and continuous attributes. For each set of categorical values (itemset), X , the method isolates the data points that contain set X , then calculates the covariance matrix of the continuous values of these points. A point is likely to be an outlier if it contains infrequent categorical sets, or if its continuous values differ from the covariance. *ODMAD* [13], an outlier detection method for mixed attribute data, can handle sparse high-dimensional continuous data. *ODMAD* first inspects the categorical space in order to detect data points with irregular categorical values or sets. Secondly, it sets aside these points and focuses on specific categorical values and the points that contain these values one at a time. *ODMAD* exhibits significant performance gains and lower memory requirements compared to [15].

The FI-based methods do well with respect to runtime performance and scalability with the tested data. Nevertheless,

these techniques rely on mining all FIs, and face problems with performance and memory requirements when applied to large high-dimensional data. This is a well-known issue for FIM, not restricted to a specific algorithm (e.g. Apriori [2]). Specifically, FIM algorithms perform well with sparse data sets, such as market-basket data, but face problems for dense data. The resulting FIs might still be numerous, an issue exacerbated when these sets contain millions of items or the σ threshold is too low. To solve this issue much work has been conducted towards *Condensed representations of FIs* (CFIs) e.g. [1], [21]. The goal of these methods is to generate a smaller collection of representative sets, from which all FIs can be deduced. Many CFIs have been proposed, e.g. *maximal*, *closed*, *δ -free*, *non-derivable* (see [8] for a CFI survey). In this paper, we use Non-Derivable Itemsets (NDIs) [7] and Non-Almost-Derivable Itemsets (NADIs) [19]. Besides CFIs, other techniques have been proposed for similar reasons, such as highly-correlated association patterns [20] or summary sets [18].

3. Algorithms

As shown in [14], the ‘ideal’ outlier in a categorical dataset is one for which each and every value of its values is extremely irregular (or infrequent). The infrequent-ness of an attribute value can be measured by computing the number of times this value is assumed by the corresponding attribute in the dataset. The algorithm in [14] assigns a score to each data point, which reflects the frequency with which each attribute value of the point occurs. In [13] this notion of ‘outlierness’ is extended to cover the likely scenario where none of the single values in an outlier point are infrequent, but the co-occurrence of two or more of its values is infrequent.

We consider a dataset \mathcal{D} which contains n data points, \mathbf{x}_i , $i = 1 \dots n$. If each point \mathbf{x}_i has m attributes, we write $\mathbf{x}_i = [x_{i1}, \dots, x_{il}, \dots, x_{im}]$, where x_{il} is the value of the l -th attribute of \mathbf{x}_i . Note that each point or record in \mathcal{D} has exact dimensionality m , and, in this sense, \mathcal{D} is not the traditional transactional database where the length of each transaction (row) may vary.

Given dataset \mathcal{D} , a support threshold σ , and a number of target outliers, k , the goal is to detect the k outlier points in \mathcal{D} . The algorithms presented here have two main phases: (1) Extract a collection of patterns or sets in the data, and (2) Compute an outlierness score for each data point and detect outliers based on the outlierness score of each point. In the following sections, we present three methods to detect outliers: a method that uses all FIs in the dataset, *FI-OD*; a method based on Non-Derivable Itemsets (NDIs), *FNDI-OD*; and a method that uses Non-Almost Derivable Itemsets (NADIs), *FNADI-OD*.

3.1 Outlier Detection based on FIs: FI-OD

Since frequent sets are “common patterns” found in many of the data points, outliers are likely to be the points that contain very few of these frequent patterns [11], [13]. On the other hand, normal points contain frequent categorical values,

Algorithm 1: FI-OD Pseudocode

Input : Database \mathcal{D} , target k , σ , $MAXLEN$
Output: k detected outliers

- 1 $G = \text{Get FIs}(\mathcal{D}, \sigma, MAXLEN)$;
- 2 $FIODScore[[\mathcal{D}], outliers[k]] := \emptyset$;
- 3 **foreach** $\mathbf{x}_i \in \mathcal{D}, i \leftarrow 0..n$ **do**
- 4 **foreach** set $f \in G \wedge f \subseteq \mathbf{x}_i$ **do**
- 5 Update $FIODScore[i]$;
- 6 **end**
- 7 **end**
- 8 $outliers[k] \leftarrow \mathbf{x}_i$ with min $FIODScore$;

or frequent sets of these values. Let $\mathcal{I} = \{i_1, i_2, \dots, i_r\}$ be a set of r items in a database \mathcal{D} . Each transaction (row) T in \mathcal{D} is a set of items such that $T \subseteq \mathcal{I}$. Given X , a set of some items in \mathcal{I} , we say that T contains X if $X \subseteq T$. The support of X , $supp(X)$, is the percentage of transactions in \mathcal{D} that contain X . We say that X is frequent if it appears at least σ times in \mathcal{D} , where σ is a user-defined threshold.

Using the frequent set information, one can define an outlier factor or score for each data point, $\mathbf{x}_i, i = 1 \dots n$. The equation for the outlier score based on FIs is designated as $FIODScore$, given in Eq. 1 below:

$$FIODScore(\mathbf{x}_i) = \sum_{f \subseteq \mathbf{x}_i \wedge supp(f) \geq \sigma} \frac{supp(f)}{|f|} \quad (1)$$

where $|f|$ denotes the length of set f , or the number of attribute values in f .

The function in Eq. 1 is based based on *FindFPOF* in [11]. It assigns a high score to points that contain many frequent values or frequent sets. The lower the score in Eq. 1 the more likely the point is an outlier. The lowest possible score is 0; this score is assigned to a point that does not contain any frequent values or sets. Note that in [11] the summation term is divided by the total number of frequent sets in \mathcal{D} , but this does not affect the detection of outlier points. Also, in Eq. 1 we divide the support of a set by the length of the set, following the concept in [15]: essentially the longer a set is, the less it contributes to the score of a point. This is because longer frequent sets contain subsets that are themselves frequent, and they have already added to the score of a point. Moreover, as shown in [15], [13], we obtain a good outlier detection accuracy by only considering sets of length up to a user-entered $MAXLEN$.

For example: let point $\mathbf{x} = [a \ b \ c]$, and $MAXLEN = 3$, the possible subsets of \mathbf{x} are: a, b, c, ab, ac, bc , and abc . If subset I of \mathbf{x} is frequent, i.e. $supp(I) \geq \sigma$, we increase the score of \mathbf{x} by $supp(I)$ divided by the length of I . In our example, if $supp(ab) = 0.3$ and ab is frequent, $FIODScore(\mathbf{x})$ will increase by $0.3/2 = 0.15$.

The pseudocode for the *FI-OD* method is shown in Algorithm 1. The first step is to mine the frequent sets from the data using a FIM algorithm such as Apriori. Again, the specific algorithm is not as important. The algorithm goes

over each data point \mathbf{x}_i and checks the subsets of \mathbf{x}_i . For each frequent set found in point \mathbf{x}_i , the anomaly or outlier score corresponding to data point \mathbf{x}_i is updated. Finally, the k data points with the lowest score are returned as outliers.

3.2 Outlier Detection based on NDIs: FNDI-OD

As noted earlier, FI-based methods may face challenges for large high-dimensional data. In the case of such data, many FIs will be generated that are spurious combinations of the same categorical values with the same or similar support. The search for a solution for this issue led to the Non-Derivable Itemsets (NDIs) [7]. The NDI representation is very well-suited for the task of outlier detection because it essentially prunes sets that have the same support as their subsets (derivable sets). Basically, NDIs retain all support information for smaller sets (i.e. single values and combinations of two values) and only prune sets of longer length that do not provide new information.

In the following, we present background on the NDI representation from [7]. Calders et al. [7] showed that itemsets whose support can be deduced or derived from their subsets, i.e. *derivable sets*, can be pruned; this can dramatically reduce the amount of sets generated.

Let a general itemset, G , be a set of items and negations of items, e.g. $G = \{abc\bar{c}\}$. The support of G in this case is the number of transactions where items a and b are present while item c is not present. We say that a general itemset $G = X \cup \bar{Y}$ is based on itemset I if $I = X \cup Y$. The deduction rules in [7] are based on the inclusion-exclusion (IE) principle [10]. For example, using the IE principle we write the following for the support of another general itemset $\{abc\bar{c}\}$:

$$supp(abc\bar{c}) = supp(a) - supp(ab) - supp(ac) + supp(abc).$$

Given a database \mathcal{D} and threshold σ , the NDI algorithm produces the condensed representation $NDIRep(\mathcal{D}, \sigma)$. For itemset I , NDI calculates lower ($LB(I)$) and upper ($UB(I)$) bounds on the support of itemset I based on information from the subsets of I . The NDI representation contains only the non-derivable frequent sets as below:

$$NDIRep(\mathcal{D}, \sigma) = \{I \in \mathcal{I} : supp(I) \geq \sigma \wedge LB(I) \neq UB(I)\}.$$

In order to find the frequent itemsets, the NDI algorithm uses Apriori-gen to generate candidate sets, and then prune infrequent candidates. If the lower and upper bounds are equal, then the itemset is derivable. If a set I is NDI, i.e. $LB(I) \neq UB(I)$, the algorithm needs to count the support of I ; if it is found that $supp(I) = LB(I)$ or $supp(I) = UB(I)$, all strict supersets of I are derivable and they can be pruned. This process is repeated until candidate sets cannot be generated further. Several properties of the NDIs were presented in [7] which we briefly summarize below.

Monotocity: If I and J are itemsets, $J \subseteq I$, and J is derivable, then I is derivable.

The NDI collection cannot be very large, because the width of the support intervals, $W(I) = UB(I) - LB(I)$, decreases exponentially with the cardinality of itemset I . Also, every set I with cardinality larger than the logarithm of the size of database, i.e. $|I| > \log_2(n) + 1$, must be derivable.

Algorithm 2: FNDI-OD Pseudocode

Input : Database \mathcal{D} , target k , σ , $MAXLEN$
Output: k detected outliers

- 1 $NDIRep = \text{Get NDI sets } (\mathcal{D}, \sigma, MAXLEN)$;
- 2 $FNDIODScore[|\mathcal{D}|], outliers[k] := \emptyset$;
- 3 **foreach** $\mathbf{x}_i \in \mathcal{D}$, $i \leftarrow 0..n$ **do**
- 4 **foreach** $I \in NDIRep(\mathcal{D}, \sigma) \wedge I \subseteq \mathbf{x}_i$ **do**
- 5 Update $FNDIODScore[i]$;
- 6 **end**
- 7 **end**
- 8 $outliers[k] \leftarrow \mathbf{x}_i$ with min $FNDIODScore$;

Given a database \mathcal{D} and the $NDIRep(\mathcal{D}, \sigma)$ set, we propose an algorithm named FNDI-OD to find outliers as follows. For each data point in the database, $\mathbf{x}_i = [x_{i1}, \dots, x_{im}]$, we assign an outlier score based on the itemsets in $NDIRep(\mathcal{D}, \sigma)$ that are subsets of \mathbf{x}_i . Every time an itemset in $NDIRep(\mathcal{D}, \sigma)$ is found to be a subset of \mathbf{x}_i , we update the score of \mathbf{x}_i as follows:

$$FNDIODScore(\mathbf{x}_i) = \sum_{I \subseteq \mathbf{x}_i \wedge I \in NDIRep} \frac{supp(I)}{|I|} \quad (2)$$

where I is a non-derivable frequent set, i.e. it belongs in $NDIRep$ (instead of f , a frequent set, in Eq. 1). Therefore, only those frequent sets in $NDIRep$ are used to find outliers (thus eliminating the need to use all frequent sets as previously with FI-OD). Finally, the algorithm finds the k lowest scores and labels the respective data points as outliers, where k is a positive integer provided as input. The pseudocode for FNDI-OD is in Algorithm 2.

As we see in section 4, when detecting outliers using frequent itemsets generated by a FIM algorithm, we are scanning through a much larger number of sets than the ones generated by the NDI algorithm. Also, the sets generated by traditional FIM methods are much longer than the ones generated by NDI. It is shown in [7] that with certain datasets, the number and length of frequent itemsets generated by the Apriori algorithm was so large that the execution of the algorithm had to be terminated. Our experiments also support these result (see Section 4).

On the other hand, using only the sets generated by the NDI algorithm, i.e. $NDIRep$, greatly speeds up the outlier detection process while not greatly affecting the accuracy of outlier detection. This is because both the NDI and Apriori (or other FIM algorithms) generate almost identical sets of length 1 and length 2. The sets of length greater than 2 not maintained by NDI, i.e. the frequent derivable sets, do not provide significant additional information when compared to their subsets. In fact, as the σ threshold decreases, most of the derivable sets, especially those of longer length, are increasingly longer combinations of highly frequent items. This implies that it is not necessary to continue adding the support information of every possible combination of these items for the outlier score, because it will be repeatedly

Algorithm 3: FNADI-OD Pseudocode

Input : Database \mathcal{D} , target k , σ , $MAXLEN$, δ
Output: k detected outliers

- 1 $NADIRep = \text{Get NADI sets } (\mathcal{D}, \sigma, \delta, MAXLEN)$;
- 2 $FNADIODScore[|\mathcal{D}|], outliers[k] := \emptyset$;
- 3 **foreach** $\mathbf{x}_i \in \mathcal{D}$, $i \leftarrow 0..n$ **do**
- 4 **foreach** $I \in NADIRep(\mathcal{D}, \sigma, \delta) \wedge I \subseteq \mathbf{x}_i$ **do**
- 5 Update $FNADIODScore[i]$;
- 6 **end**
- 7 **end**
- 8 $outliers[k] \leftarrow \mathbf{x}_i$ with min $FNADIODScore$;

adding the same support number for most of the normal points.

We also note that not generating sets longer $MAXLEN$ also speeds up the algorithm. E.g. the collection of sets with length less than 4 is also much smaller than all FIs that can be generated. Indeed, we used this parameter setting in all of our experiments (see Section 4). However, the NDI collection is still smaller than the FI collection. Therefore, using NDI increases the performance of the algorithm, while, at the same time, freeing the user from choosing an arbitrary value for the maximum length of a set.

Any other CFI scheme e.g. *closed* sets [16], can be used to generate all FIs and then proceed with outlier detection based on FIs. However, the issue we are addressing is that a very large number of FIs is generated either way. In this case, if we used another CFI and then generated the FIs, the second phase of computing the score for each point would still be a very slow process. Therefore, it is imperative in this case to use a representative set of FIs instead of FIs for the computation of our score.

Finally, other condensed representations, e.g. *maximal* or *closed* sets, are not as straightforward to use in the task of detecting outliers because they retain high-level information about a collection of sets. We need a representation that preserves smaller sets and their support, which is achieved by NDI. We also need to preserve as many of the longer sets as needed, if they do provide new information, which is also achieved by NDI. Thus, NDI-OD provides a very good trade-off between accuracy and efficiency, which can also be seen in section 4.

3.3 Outlier Detection based on NADIs: FNADI-OD

FNDI-OD uses all frequent itemsets I for which $UB(I) = LB(I)$. However, on closer inspection, some of the NDIs have a small support interval, $W(I)$. We may also be able to achieve good outlier detection accuracy by eliminating such sets I with small $W(I)$, because their support is close to the supports of their subsets.

In [19], given a user-defined error-tolerance threshold δ , an itemset I is Almost Derivable (ADI) if: $UB(I) - LB(I) \leq \delta$. Otherwise, set I is called a non almost-derivable itemset

Table 1: Datasets: Number of rows, columns, distinct single items, and outlier percentage.

Dataset	Rows	Columns	Items	outlier %
BC	483	9	87	8%
Mushroom	4644	22	113	9.4%
KDD1999	98587	39	1179	1.3%

(NADI). As shown in [19] if set X is an ADI and $X \subset Y$, Y is also ADI. The NADIRep is the collection of all NADIs that are also frequent:

$$\text{NADIRep}(\mathcal{D}, \sigma, \delta) = \{I \in \mathcal{I} : \text{supp}(I) \geq \sigma \wedge W(I) \leq \delta\}.$$

Finally, assuming set X is non-almost derivable, if $UB(X) - \text{supp}(X) \leq \delta$, or $\text{supp}(X) - LB(X) \leq \delta$, we do not generate any supersets of X because they will be ADIs.

We propose to use all frequent NADI sets instead of frequent NDIs to detect outliers. The following equation displays the score for the algorithm based on frequent NADIs, FNADI-OD:

$$\text{FNADIODScore}(\mathbf{x}_i) = \sum_{I \subseteq \mathbf{x}_i \wedge I \in \text{NADIRep}} \frac{\text{supp}(I)}{|I|} \quad (3)$$

As shown in [19], the NADI representation contains a smaller number of sets than the NDI representation. However, as the δ value increases, the accuracy of the algorithm is expected to drop. Since we are interested in detecting outliers and not extracting frequent sets, we can allow larger δ values than the ones presented in [19]. As we see in the next section, we can expect similar accuracy to FNADI-OD with larger performance gains.

4. Experiments

4.1 Experimental Setup

We conducted all experiments on a Pentium 2.61 GHz processor with 2 GB RAM. We used the NDI code available online¹, and implemented the rest in C++. All datasets were obtained from the UCI Repository [5] (see Table 1).

Wisconsin Breast Cancer (BC): The attributes in BC are computed from an image of a fine needle aspirate (FNA) of a breast mass, and describe characteristics of the cell nuclei (e.g. radius or texture). The original set contains 444 benign points and 212 malignant. As in [14], to make the dataset more imbalanced, we kept every sixth malignant record, resulting in 39 outliers (malignant), 444 non-outliers (benign).

Mushroom: This set represents samples for 23 species of gilled mushrooms. The set contains 8,124 points and 22 categorical attributes. Mushrooms are poisonous (48.2%) or edible (51.8%). To make the dataset more imbalanced, we kept every tenth poisonous record. The final set contains 113 unique categorical values, 4,644 total points, and 436 outliers (poisonous) or 9.4% of new set.

KDD1999: This set represents connections to a military computer network and multiple intrusions and attacks by

Table 2: Correct Detection (False Alarm) rates and Total Sets generated by FNADI-OD and FI-OD for the *Mushroom* set ($k = 700$; actual outliers: 436).

σ	FNADI-OD		FI-OD	
	CD (FA)	NDIs	CD (FA)	FIs
50%	72.0 (9.2)	122	72.0 (9.2)	535
20%	76.1 (8.8)	1156	75.0 (8.9)	5323
15%	76.4 (8.7)	2068	75.0 (8.9)	9402
10%	76.8 (8.7)	3846	76.1 (8.8)	18255
5%	77.3 (8.6)	9069	76.8 (8.7)	44741
2%	77.3 (8.6)	21217	77.1 (8.7)	115725

Table 3: Correct Detection (False Alarm) rates, Generated Sets and Runtime Performance in seconds for *Mushroom* ($k = 700, \sigma = 2\%$; actual outliers: 436).

Algorithm	δ	CD (FA)	Sets	Runtime
FI-OD	-	77.1 (8.7)	115725	51.1
FNADI-OD	-	77.3 (8.6)	21217	14.2
FNADI-OD	2%	77.5 (8.6)	8626	6.5
FNADI-OD	5%	77.5 (8.6)	3659	3.2
FNADI-OD	10%	77.3 (8.6)	1241	1.4
FNADI-OD	20%	76.1 (8.8)	441	0.8

unauthorized users. The raw binary TCP data were processed into features such as connection duration, protocol type, etc. There are three available sets: training set, test set, and 10% of the training set. For our experiments we used the 10% training set. All *KDD1999* sets contain 33 continuous attributes and 8 categorical attributes. Due to the large number of attacks, we preprocessed the data such that attack points are a small percentage of the data, and these points were chosen randomly from the collection of attack points. Network traffic packets tend to occur in *bursts* for certain intrusions. While we preserved the proportions of the various attacks in the data, we selected outlier points at random without necessarily preserving the burst length. We followed [15], [13] and detected bursts of packets in the data. Our resulting set contains 98587 points, 1309 attacks. We discretized continuous attributes using equal-width discretization (20 intervals), and removed 2 attributes that contained the same value for all records. The resulting set has 39 columns and 1179 distinct categorical values (single items).

Evaluation. We compare runtime performance of the algorithms in seconds using the same data. We then evaluate the accuracy based on the following measures:

- *Correct Detection rate (CD)*: ratio of the number of outliers correctly identified as outliers over the total number of outliers;
- *False Alarm rate (FA)*: ratio of the number of normal points erroneously flagged as outliers over total number of normal points.

4.2 Results

We ran several experiments with various values for support threshold, σ , and the desired number of outliers, k . The δ parameter for FNADI-OD is presented as a percentage of the total number of points in the set. We used *MAXLEN*

¹<http://www.adrem.ua.ac.be/goethals/software>

Table 4: Correct Detection (False Alarm) rates and Total Sets generated by FNDI-OD and FI-OD for *KDD1999* ($k = 5000$; actual outliers: 1309).

σ	FNDI-OD		FI-OD	
	CD (FA)	NDIs	CD (FA)	FIs
99.4%	70.1 (4.5)	177	70.1 (4.5)	1486
98%	71.1 (4.4)	406	71.1 (4.4)	5092
97%	71.1 (4.4)	459	71.1 (4.4)	5898
95%	59.0 (4.5)	661	70.6 (4.3)	8903
90%	32.8 (4.5)	1499	35.7 (4.5)	19628

Table 5: Correct Detection (False Alarm) rates, Generated Sets and Runtime Performance in seconds for *KDD1999* ($k = 5000$, $\sigma = 97\%$; actual outliers: 1309).

Algorithm	δ	CD (FA)	Sets	Runtime
FI-OD	-	71.1 (4.4)	5898	721
FNDI-OD	-	71.1 (4.4)	459	74
FNADI-OD	0.02%	71.1 (4.4)	146	24
FNADI-OD	0.05%	71.1 (4.4)	116	21
FNADI-OD	0.10%	71.1 (4.4)	59	18
FNADI-OD	0.15%	61.6 (4.4)	49	12

equal to 4 for all experiments as in [15], [13].

We did not observe a difference in CD or FA rates for the BC Dataset between FI-OD and FNDI-OD, so these results are omitted. For this dataset, σ less than or equal to 20% gives the best accuracy and false alarm rates.

Table 2 contains accuracy results and sets generated for FNDI-OD versus FI-OD on the Mushroom set for various σ values. From Table 2, FNDI-OD has better CD and FA rates than FI-OD for the Mushroom set for all σ values, while the best rates are achieved for lower σ values ($\sigma \leq 5\%$). As we also see in this Table, FNDI-OD achieves better accuracy by using far less sets; e.g. NDIs are 3846 versus 18255 FIs for $\sigma = 10\%$. This is after we set *MAXLEN* to 4. This shows that an FI-based algorithm still faces problems with large high-dimensional data even when using *MAXLEN* in order to stop generating longer sets. We could set the *MAXLEN* to a lower value, e.g. 3. However, even if this value results in good accuracy rates, the collection of NDIs and NADIs will still be less than the generated FIs.

Table 3 contains accuracy and runtime results for FNADI-OD with several δ values versus FI-OD and FNDI-OD for the Mushroom set and σ equal to 2%. FNADI-OD has similar accuracy to FNDI-OD for δ less than 20%. For example, for δ equal to 10%, FNADI-OD achieves slightly better accuracy than FNDI-OD (and better accuracy than FI-OD). It does so while generating only 1241 sets and finishing execution in slightly over 1 second versus 8626 NDIs (14 seconds) and 115725 FIs (51 seconds). Accuracy (correct detection) results for FNADI-OD and FNDI-OD for the Mushroom set are shown in Fig. 1 for various δ and σ values ($\delta = 0$ reflects FNDI-OD). As seen in this figure, the accuracy of FNADI-OD is very close to FNDI-OD for most combinations of the parameter values.

Table 4 contains CD and FA rates as well as total generated sets by FNDI-OD and FI-OD for the *KDD1999* set and various σ values. Table 4 contains the average accuracy based

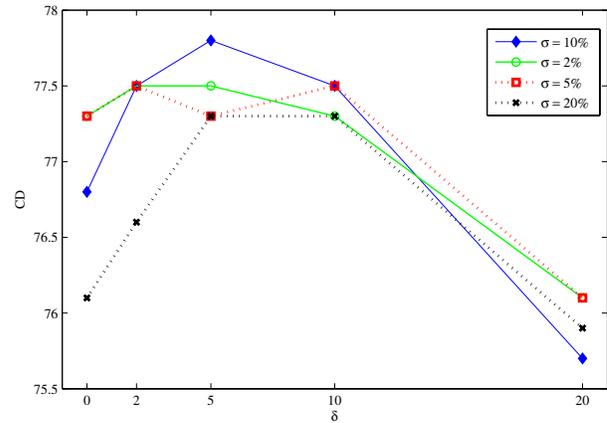


Fig. 1: Correct Detection for FNADI-OD versus FNDI-OD for the Mushroom set and various δ values.

on bursts of attacks that were correctly detected. Essentially, if we detect one point in a burst of packets as an outlier we mark all points in the burst as outliers and we count the burst as detected, similar to [15] and [13]. As seen in this Table, FNDI-OD achieves the same accuracy as FI-OD using only a fraction of the number of sets compared to FI-OD. For example, for $\sigma = 97\%$, both algorithms achieve the same accuracy but FNDI-OD needs only 459 NDIs versus 5898 FIs needed by FI-OD. We also notice that for lower σ values the accuracy of FNDI-OD and FI-OD deteriorates, while FNDI-OD does so faster than FI-OD. This is because more FIs or NDIs are added to the score of each point which in the case of this data set make it difficult to distinguish between outliers and normal points.

Fig. 2 gives a pictorial representation of the sets generated by FNDI-OD versus FI-OD as well as total runtime for the *KDD1999* set. As this figure shows, FNDI-OD scales much better as σ decreases compared to FI-OD. In fact, for σ equal to 10% the FI-OD algorithm generates more than 110 thousand FIs, and its execution was terminated. This also shows the importance of using a small collection of sets when dealing with large data sets. For example, for σ equal to 90%, FNDI-OD took 3 minutes to detect the outliers in the *KDD1999* dataset, while FI-OD needed 29 minutes to accomplish the same task.

Table 5 compares FNADI-OD with FNDI-OD and FI-OD for $\sigma = 97\%$. FNADI achieves the same accuracy as FNDI-OD and FI-OD for δ less than 0.15% (or 150). However, for δ equal to 0.1%, FNADI-OD takes 18 seconds to finish the task, while FNDI-OD takes 74 seconds, and FI-OD takes more than 12 minutes. These results indicate that FNADI-OD closely approximates the performance of FNDI-OD, for a range of δ values, while using a smaller number of sets, and thus offers higher runtime performance gains.

5. Conclusions

Recently, outlier detection techniques were proposed for categorical and mixed-attribute datasets [11], [13], [15] based

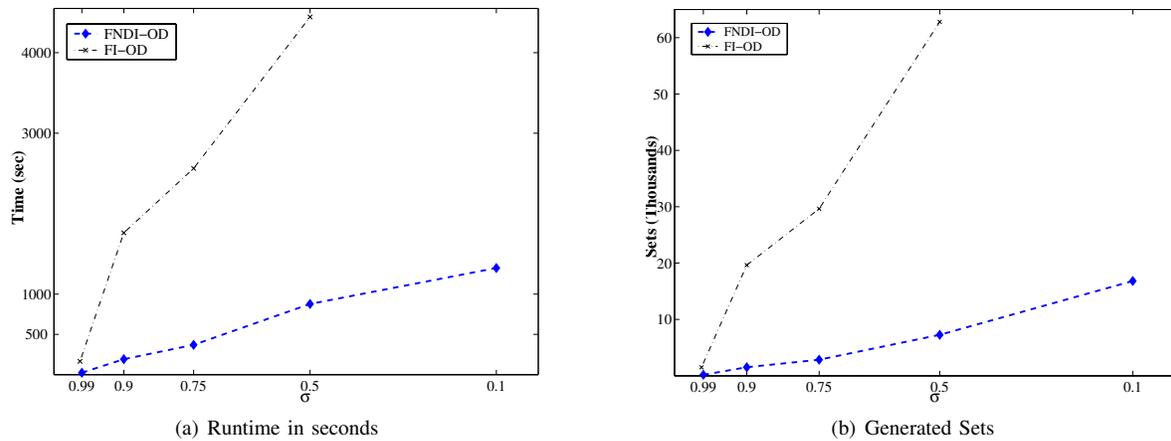


Fig. 2: Runtime and Sets for FNDI-OD versus FI-OD ($MAXLEN = 4$), for $KDD1999$ set, as σ decreases.

on Frequent Itemset Mining (FIM) [2]. These methods aim to extract all frequent sets, or common patterns, in the data and then identify as outliers the points that contain few of these patterns. Even though these methods have been shown to perform well, they face significant challenges for large high-dimensional data where a large number of frequent sets is generated. In this paper, we propose two outlier detection schemes that use a collection of sets that is much smaller than the FI collection. The first method uses frequent Non-Derivable Itemsets [7], FNDI-OD, while the second method uses an approximate NDI representation, Non-Almost Derivable Itemsets [19], FNADI-OD. Our experiments show that the FNDI-OD method presents significant runtime advantages compared to its FI-based counterpart. Overall, FNDI-OD has similar correct detection and false alarm rates compared to FI-OD. On the other hand, the approximate NDI method, FNADI-OD, is even faster than FNDI-OD, and exhibits accuracy very close to the one achieved by FNDI-OD depending on a δ parameter. Future research includes extending our ideas for data with both categorical and continuous attributes, and for distributed datasets.

Acknowledgments

This work was supported in part by NSF grants: 0341601, 0647018, 0717674, 0717680, 0647120, 0525429, 0806931, 0837332.

References

- [1] R. Agarwal, C. Aggarwal, and V. Prasad. Depth first generation of long patterns. *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pages 108–118, 2000.
- [2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. *Proc. Int'l Conference on Very Large Data Bases*, pages 487–499, 1994.
- [3] V. Barnett. *Outliers in Statistical Data*. John Wiley and Sons, New York, 1978.
- [4] S. Bay and M. Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. *Proc. of the ACM SIGKDD Int'l conference on Knowledge Discovery and Data Mining*, pages 29–38, 2003.
- [5] C. Blake and C. Merz. *UCI Repository of Machine Learning Databases*. <http://archive.ics.uci.edu> (Accessed September 2008), 1998.
- [6] M. Breunig, H. Kriegel, R. Ng, and J. Sander. LOF: identifying density-based local outliers. *ACM SIGMOD Record*, 29(2):93–104, 2000.
- [7] T. Calders and B. Goethals. Non-derivable itemset mining. *Data Mining and Knowledge Discovery*, 14(1):171–206, February 2007.
- [8] T. Calders, C. Rigotti, and J. Boulicaut. A survey on condensed representations for frequent sets. *LNCS Constraint-based mining and Inductive Databases*, 3848:64–80, 2004.
- [9] P. Dokas, L. Ertöz, V. Kumar, A. Lazarevic, J. Srivastava, and P. Tan. Data mining for network intrusion detection. *Proc. NSF Workshop on Next Generation Data Mining*, pages 21–30, 2002.
- [10] B. Ganter and R. Wille. *Formal concept analysis*. Springer-Verlag, 1999.
- [11] Z. He, X. Xu, J. Huang, and S. Deng. FP-Outlier: Frequent Pattern Based Outlier Detection. *Computer Science and Information System*, 2(1):103–118, 2005.
- [12] E. Knorr, R. Ng, and V. Tucakov. Distance-based outliers: algorithms and applications. *Int'l Journal on Very Large Data Bases VLDB*, 8(3):237–253, 2000.
- [13] A. Koufakou, M. Georgiopoulos, and G. Anagnostopoulos. Detecting Outliers in High-Dimensional Datasets with Mixed Attributes. *WORLD COMP Int'l Conference on Data Mining DMIN*, pages 427–433, 2008.
- [14] A. Koufakou, E. Ortiz, M. Georgiopoulos, G. Anagnostopoulos, and K. Reynolds. A Scalable and Efficient Outlier Detection Strategy for Categorical Data. *IEEE Int'l Conference on Tools with Artificial Intelligence ICTAI*, pages 210–217, 2007.
- [15] M. Otey, A. Ghoting, and S. Parthasarathy. Fast Distributed Outlier Detection in Mixed-Attribute Data Sets. *Data Mining and Knowledge Discovery*, 12(2):203–228, 2006.
- [16] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. *7th International Conference on Database Theory ICDT*, pages 398–416, 1999.
- [17] D. Tax and R. Duin. Support Vector Data Description. *Machine Learning*, 54(1):45–66, 2004.
- [18] J. Wang and G. Karypis. On efficiently summarizing categorical databases. *Knowledge and Information Systems*, 9(1):19–37, 2006.
- [19] Y. Xiaoming, W. Zhibin, L. Bing, Z. Shouzhi, W. Wei, and S. Bole. Non-Almost-Derivable Frequent Itemsets Mining. In *Proceedings of the The Fifth Int'l Conference on Computer and Information Technology*, pages 157–161, 2005.
- [20] H. Xiong, G. Pandey, M. Steinbach, and V. Kumar. Enhancing Data Analysis with Noise Removal. *IEEE Transactions Knowledge and Data Engineering*, pages 304–319, 2006.
- [21] M. Zaki and C. Hsiao. Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. *IEEE Transactions Knowledge and Data Engineering*, pages 462–478, 2005.

A Sparse Coding Based Similarity Measure

Sebastian Klenk, and Gunther Heidemann

Intelligent Systems Department, Stuttgart University, Stuttgart, Germany

Abstract—*In high dimensional data sets not all dimensions contain an equal amount of information. Further global aspects of a signal are usually more important than local ones. Therefor a similarity measure should be indifferent to small scale differences an sensitive to changes in the global structure of a signal. This makes it difficult to select a similarity measure that inherently considers these differences in weighting. We present a sparse coding based similarity measure that is capable of extracting and emphasizing relevant elements of a signal given a reference data set. The measure we propose is set to work on raw and unprocessed data which allows for a representation more closely to the actual information than it would be possible with extracted features.*

1. Introduction

The analysis and interpretation of continuous data requires the definition of suitable similarity measures. Popular examples are clustering or classification procedures based on object similarities such as the Nearest Neighbor Methods, some types of Neural Networks or Hierarchical Clustering Methods (for a broad overview on the subject see e.g. Hastie et. al. or Ripley [1], [2]). Applications of similarity measures range from areas such as Content Based Image Retrieval (CBIR) to sound recognition and time series analysis to name but a few [3].

Due to the broad range of applications, similarity measures are usually highly domain dependent. Thus most applications employ their own type of similarity measure that was thoroughly tuned to the task. For example, CBIR systems use often more than 200 different measures each of which has its own special purpose, such as color histogram based distance measures that describe the similarity of images in terms of their color distribution. Other measures judge the similarity of textures. The list can be extended almost endlessly.

Here we present ideas on a similarity measure that can exhibit different kinds of characteristics, depending on the choice of parameters. Our main tool to achieve this is sparse coding. It is based on the idea that high dimensional signals can be represented by a very small number of elements taken from a large dictionary with high redundancy. The choice of these elements is highly signal specific, a fact our approach makes use of. An important implication is that it is very unlikely that two different signals share a common set of dictionary elements [4]. As opposed to basis systems redundant dictionary systems don't describe different signal

by different weightings of a common set of basis elements but by different sets of dictionary elements. Further we have to keep in mind that the dictionary is highly redundant and that a sparse representation is a close to optimal but not the only possible representation. Depending on the distribution of the signals, most signals can be represented fairly good with the other signals elements. How good this other representation describes the signal depends on their similarity.

In this paper we describe how the properties of sparse coding can be used to calculate signal similarities. We start with a brief introduction to sparse coding, then the proposed similarity measure will be described. We will present numerical results and conclude with an outlook on future tasks.

2. Sparse coding

High dimensional signals can be represented in a number of ways. A common representation is the (unique) superposition of sin- and cosine functions or fractions thereof. The underlying sets are usually maximally linear independent and free of any redundancy. But besides such basis systems there is also a large number of dictionaries which are highly linearly dependent and not at all free of redundancy. Usually such dictionaries are made up from a combination of different basis systems such as wavelets or windowed fourier basis functions. Such systems allow a sparse representations. One drawback of this method is that the decomposition of a signal into its basis components is no longer unique. Consequently, significant effort is required to choose the representation of a particular signal such that it is suitable for the designated use. Therefore new methods have been developed to calculate an optimal, i.e. sparse, representation [5], [6].

A sparse representation is commonly thought of in the l_0 sense. This means that the representation contains a minimal number of non-zero entries. Finding these is a combinatorial problem the solution of which is infeasible for real world applications. Therefore, sparse coding must resort to a less strict norm. A solution was proposed by Chen et. al. [6] for the basis pursuit method. They used a minimal l_1 norm as a means of sparsity. A representation x of a given signal y is chosen such that for a given basis matrix A the l_1 norm $\|x\|_1$ is minimal:

$$\min \|x\|_1 \quad \text{and} \quad Ax = y.$$

Donoho showed [7] that, for sufficiently sparse x , the above representation is actually also the sparsest solution in the l_0 sense, i.e. the l_0 and l_1 optimum are equivalent. So the solution found with the basis pursuit method is actually the one with the fewest non-zero entries.

3. Sparse coding based similarity measure

There are a number of approaches to use representations in different bases or dictionary systems to create similarity measures. These are mostly based on special basis systems such as wavelets or Fourier bases. The choice of these systems is usually motivated by the characteristics of a particular signal domain [8]. The better the basis is adapted to the signal, the “clearer” the signal can be represented, i.e., the more sparse becomes its representation.

As mentioned above, highly redundant systems allow sparser representations than orthogonal basis systems. This property, known as super resolution, is a curse and a blessing at the same time. The difficulty lies in the fact that if the representation of a domain is very sparse, then for two different signals the number of common dictionary elements (with non-zero coefficients) is very small. Therefore the knowledge about the (euclidian) difference – which is in such a case simply the sum of the squared values – between two signals is of limited use. Donoho shows that for very high dimensional data with a normal distribution almost all points of a data set form the vertices of its convex hull [4].

To circumvent this problem we employ the redundancy of the dictionary elements. This means that there are many ways to represent a signal and also that one set of dictionary elements fits to a large group of signals. However, the thereby obtained representations are not as good as they could be with an optimally chosen set. The similarity between two signals can now be approximated by estimating how good the set of dictionary elements of one signal represents the other signal. For two signals f and f' we first calculate the best representation x_f of f given all dictionary elements A

$$\min \|x_f\|_1 \quad \text{subject to} \quad \|Ax_f - f\| \leq \epsilon.$$

This representation can now be used to calculate the set of dictionary elements A_f necessary to describe f

$$A_f^T = A \cdot D_f \quad \text{with} \quad D_f = \text{diag}(x_f).$$

Here $\text{diag}(x)$ is a diagonal matrix with 1's, where the elements of x are non-zero and 0's otherwise, as its diagonal.

Once this reduced matrix is obtained one can calculate for f an adapted version of the signal f' :

$$\min \|x_{f'}\|_1 \quad \text{subject to} \quad \|A_f x_{f'} - f'\| \leq \epsilon. \quad (1)$$

This is being achieved by using only the most significant – significant for the representation of f – dictionary elements

to represent the other signal f' . Note that only those rows of the matrix A_f are non-zero that were most important to the representation of f . Therefore an exact solution is highly unlikely and equation (1) minimizes within a given error tolerance ϵ .

We now have the approximations of the two signals f and f' . Both of them were obtained using the same set of dictionary elements $\{d | I_{x_f}(d) = 1\}$. Now it is possible to calculate a difference between f and f' in terms of the difference between the two sets of coefficients belonging to the same set of dictionary elements

$$d_1 = \|x_f - x_{f'}\|_2.$$

This information is not sufficient to fully represent the difference. The quality of the approximation

$$q_f = \langle A_f x_f, f \rangle \quad \text{v.s.} \quad q_{f'} = \langle A_f x_{f'}, f' \rangle$$

might vary dramatically between two very different signals. Therefore this information has to be considered, too. It can be cast into a simple fraction

$$d_2 = \frac{q_{f'}}{q_f}.$$

Together these two form a similarity measure

$$d_{sp} = w_1 \cdot d_1 + w_2 \cdot d_2$$

which is based on the information given by the sparse representation. The weighting terms w_1 and w_2 can be chosen depending on the quality of the signal. In our setting we had $w_1 = w_2 = 1/2$.

The following algorithm (1) describes the whole procedure in terms of processing steps.

Algorithm 1 SP-Dist(f_1, f_2, D)

```

 $x_1 \leftarrow 0$ 
min  $\|x_1\|_1$  subject to  $\|Dx_1 - f_1\| \leq \epsilon$ 
 $x_1 \leftarrow \text{GETLARGESTCOEFFICIENTS}(x_1, \sqrt{\text{size}(D)})$ 
 $D \leftarrow \text{GETCOLUMNSTOCOEFFICIENTS}(x_1, D, \sqrt{\text{size}(D)})$ 
 $x_1 \leftarrow 0$ 
 $x_2 \leftarrow 0$ 
min  $\|x_1\|_1$  subject to  $\|Dx_1 - f_1\| \leq \epsilon$ 
min  $\|x_2\|_1$  subject to  $\|Dx_2 - f_2\| \leq \epsilon$ 
return  $\{d_{sp}(x_1, x_2, D)\}$ 

```

The functions `getLargestCoefficients` can be implemented as a simple sorting procedure where only the $k = \sqrt{\text{size}(D)}$ largest coefficients are returned. The same goes for the function `getColumnsToCoefficients`, which selects only those columns that correspond to the largest coefficients in x . The computationally most expensive areas of the algorithm are the optimization procedures. Here a Linear Programming Problem has to be solved which for most practical problems can be done in $2m$ to $3m$ iterations, where m is the number of dictionary elements

[9]. Fortunately only the first optimization procedure goes over the entire data set, whereas the last two only work on a very small subset.

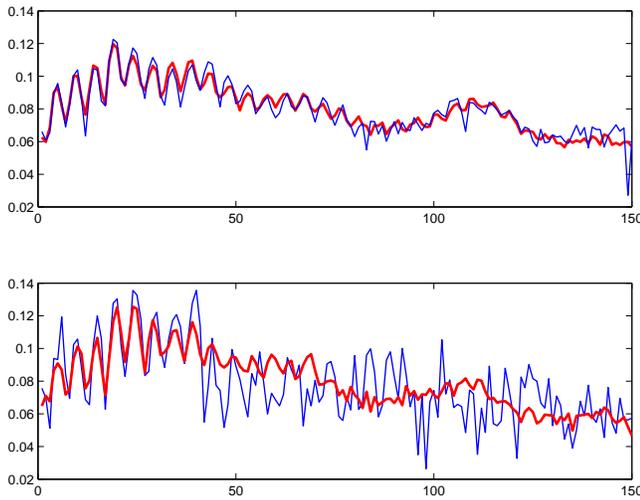


Fig. 1: Similarity measurement between two functions, the blue curve is the actual signal and the red curve shows the approximation. As can be seen the figure above shows a match whereas the one below is a mismatch.

4. Numerical experiments

We have conducted a number of experiments on the use of the sparse coding based similarity measure. To obtain comparable and reproducible results, we have used publicly available data sets like the phoneme data set [1], [10] or stock market data which can be downloaded for example from Yahoo (finance.yahoo.com).

In the following we describe the experiments. First we have estimated the influence of noise on the measurement results. We therefore calculated the similarity for two randomly chosen signals, taken from the phoneme data set. We repeated this step for each pair of signals and with increasing noise level. The idea here is that the more robust a similarity measure is towards noise the closer it stays to its initially calculated similarity. Figures 2 and 3 displays the resulting curves. The upper curve displays the results for the euclidean distance (as a $1 - d(x, y)$ similarity measure) whereas the lower curve shows the one for the sparse coding based measure. The dotted curves show the standard deviation of the measure. Obviously the sparse coding based measure stays, on average, close to the initial similarity (measurement between the two curves with no influence of noise). The curve for the euclidean distance drops sharply which means that with the increasing noise level also the similarity decreases. Here the influence of the compression step can clearly be seen as the similarity remains relatively stable. Figure 3 demonstrates this for 50 randomly chosen

sets of signals, each of which contained 50 signals. Here the sharp difference is not that obvious any more, but still clearly visible. The slightly larger deviation for the sparse coding based similarity which can be seen in both plots is partly due to the fact that the set of possible dictionary elements varies with each calculation as is described later.

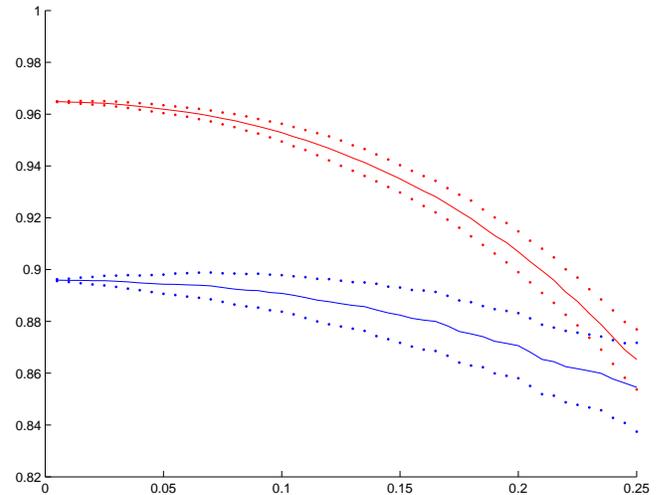


Fig. 2: Similarity measure with increasing noise level for a single curve.

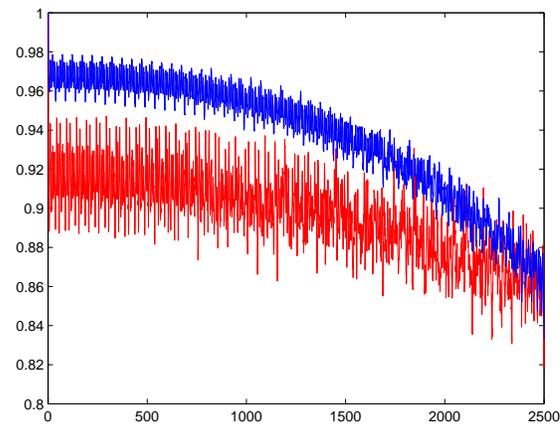


Fig. 3: Similarity measure with increasing noise level for all curves.

The calculations follow algorithm 1 which we implemented in Matlab. The optimization procedures needed for the basis pursuit method were calculated with the CVX toolbox, a package for specifying and solving convex programs [11], [12]. For a improved calculation speed we only took a randomly chosen subset of all possible dictionary elements. This subset was chosen sufficiently large, about 300 elements

out of 800 for a signal of dimension 150, such that a possible influence of this selection can be ignored.

Our second experiment is set to test the sparse coding based similarity measure for its ability to classifying continuous data. We therefore used the popular phoneme data. This is a labeled set of log-spectral transformed phonemes recorded from different speakers. In the experiments we only considered the phonemes *sh* as in "she", and *iy* as the vowel in "she". Figure 4 shows the average curve of all of the two phoneme log spectrums. There are rather strong differences between the two curves which can be clearly detected. For the ease of computation we used a data set reduced in size (as was also done by Ferraty and Vieu [10]) with 800 signals and $m = 150$ measurements per signal. We also employed a reduced set of dictionary elements to increase the computation speed. The performance of the optimization procedure depends not only on the dimension of the signal, in our case 150, but also on twice the size of the dictionary as is described by Chen et. al. [6].

The test procedure was as follows. We divided the data into two random sets, one test with 25 signals and the dictionary set with 775 signals. We now took the first signal of the test set and calculated the sparse coding based similarity to all other 24 signals. Now we estimated the class of the first signal by taking (i) the Nearest Neighbor and (ii) the k -Nearest Neighbors. We then compared our estimation with the actual class. This procedure was repeated 50 times to get an average error. The entire error estimation procedure was also repeated 50 times to get information on the mean and the deviation of the error. The results can be seen in Figure 5 and Table 5.

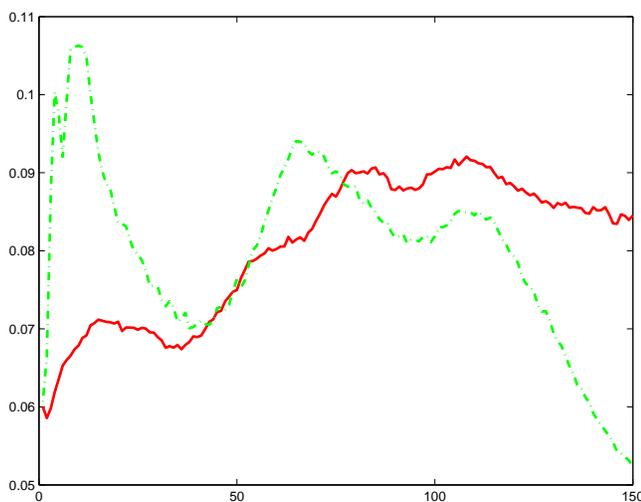


Fig. 4: The mean of all curves shown for the two different phonemes.

5. Related Work

The use of sparse coding for pattern recognition is fairly new. There are some papers dealing with classification, particularly interesting is the work of Wright and Yang [13], [14] and Haupt et. al. [15], [16]. Even though the former have proposed a method where the basic idea is fairly similar, both are considering classification tasks, and are heavily relying on knowledge about the class and its structure. In contrast, our work aims at developing a general similarity measure.

Haupt et. al. describe a method for signal detection or classification by using the fact that the true representation x of a compressed signal f can be reconstructed if x is known to be sufficiently sparse. Here the idea of sparse coding is used when reconstructing x from f . In [15], [16] the use of a so called measurement ensemble — a matrix A with usually i.i.d. columns — is described which serves in Compressive Sensing [17] as a means of generating a compressed version of a signal. From this compressed version the original signal can be recovered if it is sufficiently sparse. For a given signal f a compressed representation can be obtained by multiplication with the measurement ensemble

$$y = A^T f.$$

With the help of sparse coding the true signal can now be reconstructed and be compared to class representatives c_i . Those with the largest similarity will be chosen as matching class.

$$C = \arg \min_i \|c_i - Ay\|$$

The difficulty of this approach is that the reconstruction — by definition — will be sparse and most likely has no common non-zero entries with the class representatives.

Wright and Yang [13], [14] use a different approach, where they employ the knowledge of the class membership of the dictionary elements. They first calculate a sparse representation of a given signal

$$\min \|x_f\|_1 \quad \text{subject to} \quad \|Ax_f - y\| \leq \epsilon$$

and then compare the reconstruction quality of the signal to the original signal. The reconstruction is calculated by only using dictionary elements of a certain class

$$C = \arg \min_i \|A\delta_i(x) - y\|.$$

Here $\delta_i(x)$ allows only those coefficients to be non-zero that stand for a dictionary element belonging to class i .

This last approach is fairly similar to the measure proposed in this paper. We are also only using a small set of dictionary elements and measure the reconstruction error, but in addition we incorporate the differences between their influence on the signal. Further, the method proposed in [13], [14] is strongly related to feature extraction, whereas our approach is intended to work on the unprocessed, raw data.

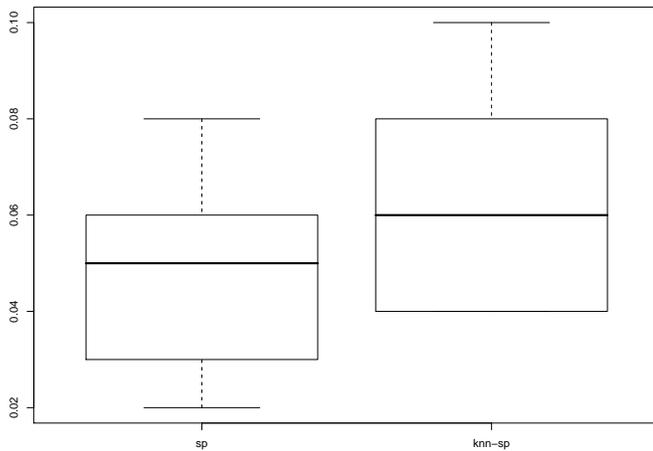


Fig. 5: The error measure for the classification of the two phonemes.

	d_{sp}	$k - m_{sp}$
\bar{x}	0.046	0.061
$\text{var}(x)$	0.0004	0.0006

Table 1: The error measure for the classification of the two phonemes in numbers.

6. Conclusions and future work

We have presented a sparse coding based distance measure. The goal was to develop a measure which incorporates the advantages of sparse coding. In section 2 we have introduced the main ideas of sparse coding and demonstrated how these can be used for a novel distance measure. We have shown how the new distance measure relates to other commonly used methods such as wavelets or Fourier transform based measures. In section 4 we described numerical experiments with very promising results. However, also much room for optimization in future work is left.

A very interesting aspect that remains to be worked on is the choice of the dictionary. It would be nice to see whether there are procedures for estimating a suitable dictionary or if this remains to be the prior knowledge that has to be

incorporated in the classification process.

References

- [1] T. J. Hastie, R. J. Tibshirani, and J. H. Friedman, *The elements of statistical learning*, corrected print. ed. Springer, 2002.
- [2] B. D. Ripley, *Pattern recognition and neural networks*, 7th ed. Cambridge Univ. Press, 2007.
- [3] J. Han and M. Kamber, *Data mining*. Morgan Kaufmann Publ., 2001.
- [4] D. L. Donoho and J. Tanner, "Counting faces of randomly-projected polytopes when the projection radically lowers dimension," *Journal of the AMS*, vol. 22, no. 1, pp. 1–53, Jan 2009.
- [5] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *Signal Processing, IEEE Transactions on*, vol. 41, no. 12, pp. 3397–3415, Dec 1993.
- [6] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.
- [7] D. L. Donoho, "For most large underdetermined systems of equations, the minimal ℓ_1 -norm near-solution approximates the sparsest near-solution," *Communications on Pure and Applied Mathematics*, vol. 59, no. 7, pp. 907–934, Dec 2006.
- [8] J. O. Ramsay and B. W. Silverman, *Functional data analysis*. Springer, 2002.
- [9] S. J. Nocedal, Jorge ; Wright, *Numerical optimization*, corr. print. ed., ser. Springer series in operations research. New York: Springer, 2000.
- [10] F. Ferraty and P. Vieu, *Nonparametric Functional Data Analysis: Theory and Practice (Springer Series in Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
- [11] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*, ser. Lecture Notes in Control and Information Sciences, S. B. V. Blondel and H. Kimura, Eds. Springer, 2008, pp. 95–110.
- [12] —, "Cvx: Matlab software for disciplined convex programming (web page and software)," Feb 2009. [Online]. Available: <http://stanford.edu/~boyd/cvx>
- [13] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, Feb. 2009.
- [14] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry, "Feature selection in face recognition: A sparse representation perspective," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2007-99, Aug 2007.
- [15] J. Haupt, R. Castro, R. Nowak, G. Fudge, and A. Yeh, "Compressive sampling for signal classification," *Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on*, pp. 1430–1434, 29 2006-Nov. 1 2006.
- [16] J. Haupt and R. Nowak, "Compressive sampling for signal detection," *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 3, pp. III–1509–III–1512, April 2007.
- [17] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, 2008.

An Enhanced Density Based Spatial clustering of Applications with Noise

A. Fahim¹, G. Saake¹, A. Salem², F. Torkey³ and M. Ramadan⁴

¹Faculty of Information, Magdeburg University, Magdeburg, Germany

²Faculty of Information, Ain Shams University, Cairo, Egypt

³Kafrelshiekh University President, Kafrelshiekh, Egypt

⁴Faculty of Science, Menufiya University, Shebin El-Kom, Egypt

Abstract – Cluster analysis is a primary method for data mining. Finding clusters with varying sizes, shapes and densities is a challenging job. DBSCAN can find clusters with varying shapes and sizes. But it has a trouble in finding clusters with varying densities, because it depends on a global value for its parameter *Eps*. This paper presents enhanced DBSCAN which clusters databases containing clusters with varying densities effectively. The idea is to use varied values for *Eps* according to the local density of the starting point in each cluster. The clustering process starts from the highest local density point towards the lowest local density one. For each value of *Eps*, DBSCAN is adopted to make sure that all density reachable points with respect to current *Eps* are clustered. At the next process, the clustered points are ignored, to avoid merging among denser clusters with sparser ones.

Keywords: DBSCAN, Data Clustering, Density Clustering.

1 Introduction

Spatial data clustering is one of the promising techniques of data mining. The main goal of clustering is to group data objects into clusters such that objects belonging to the same cluster are similar, while those belonging to different ones are dissimilar. Data clustering algorithms can be classified into four categories; (1) partitioning, (2) hierarchical, (3) density-based and (4) grid-based. However, some algorithms may fall into more than one category. By clustering one can identify dense and sparse regions and, therefore, discover overall distribution patterns. Finding clusters in data is challenging when the clusters are of widely differing sizes, shapes and densities and when the data contains noise and outliers. Although many algorithms exist for finding clusters with different sizes and shapes, there are a few algorithms that can detect clusters with different densities.

Basic density based clustering techniques such as DBSCAN [5] and DENCLUE [6] treat clusters as regions of high densities separated by regions of no or low densities. So they are able to suitably handle clusters of different sizes and

shapes besides effectively separating noise and outliers. But they fail to identify clusters with varying densities unless the clusters are separated by sparse regions [3]. There are some algorithms which can handle clusters of different densities, like OPTICS [1] but it does not produce explicit clusters. Traditional DBSCAN may have trouble with clusters of varying densities. For example, in the dataset shown in Fig. 1.a, DBSCAN fails to find the four clusters, because this data set has four levels of density, the clusters are not totally separated by sparse regions and the value of *Eps* is global for the dataset. So we need an approach able to stop the expanding of cluster at different levels of density. In Fig. 1.b there are two problems; they are in the smallest sparse cluster and the largest cluster which contains three dense clusters within it. In Fig. 1.c, DBSCAN discovers only the three small clusters and considers the other two large clusters as outliers or merges the three small clusters in one cluster to be able to find the other two large clusters.

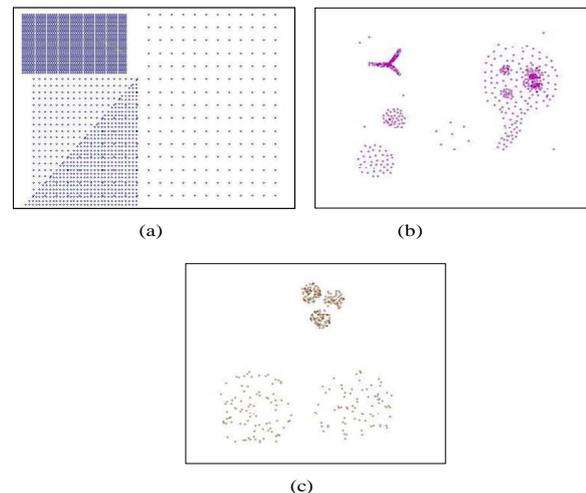


Fig. 1. clusters with varying densities

This paper introduces an enhanced version of the DBSCAN algorithm which is able to discover clusters with varying densities. It selects suitable values for its parameter *Eps* for each cluster. It is based up on the local density of the

starting point in each cluster, and adopts the traditional DBSCAN for each value of Eps . The idea of the proposed algorithm depends on discovering the highest density clusters first, and then the Eps is adapted to discover the next low density clusters with ignoring the previously clustered points. The proposed algorithm requires two input parameters; they are $Minpts$ and $Maxpts$. The $Maxpts$ allows the value of Eps to be different from one cluster to another according to the local density of the initial point in each one. The $Minpts$ determines the lowest level of density allowed inside the cluster, while the $Maxpts$ limits the highest level of density allowed inside the cluster in addition to controlling the values of Eps neighboring radius of the traditional DBSCAN algorithm. We can note that the DBSCAN starts to create a cluster from any core point, while the enhanced DBSCAN starts the clustering process from the highest density core point. We refer to this core point as the starting point or the initial point.

Rest of the paper is organized as follows. Section 2 surveys the main definitions of DBSCAN. Section 3 briefly surveys some of recent clustering methods. The proposed algorithm is presented in section 4. Section 5 presents some experimental results to evaluate the algorithm. Finally, section 6 presents a conclusion.

2 Density based clustering (DBSCAN)

Since the proposed algorithm is an enhanced version of DBSCAN algorithm, it is important to review the main definitions of DBSCAN algorithm, for details see [5]. Let D is a database of N points in d -dimensional space R^d . The distance between two data points p and q is given by the Euclidean distance and denoted by $d(p,q)$. The DBSCAN depends on the following definitions:

- The Eps -neighborhood of a point p , denoted by $NEps(p)$, is defined by $NEps(p) = \{q \in D \mid d(p,q) \leq Eps\}$.
- A point p is *directly density-reachable* from a point q wrt. Eps and $Minpts$ if $p \in NEps(q)$ and $|NEps(q)| \geq Minpts$ (core point condition).
- A point p is *density-reachable* from a point q wrt. Eps and $Minpts$ if there is a chain of points $p_1, \dots, p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i .
- A point p is *density-connected* to a point q wrt. Eps and $Minpts$ if there is a point o such that both, p and q are density-reachable from o wrt. Eps and $Minpts$.
- A *cluster* C wrt. Eps and $Minpts$ is a non-empty subset of D satisfying the following conditions:
 - $\forall p, q$: if $p \in C$ and q is density-reachable from p wrt. Eps and $Minpts$, then $q \in C$. (Maximality).
 - $\forall p, q \in C$: p is density-connected to q wrt. Eps and $Minpts$. (Connectivity).

- If C_1, \dots, C_k be the clusters of the database D wrt. parameters Eps_i and $Minpts_i, i = 1, \dots, k$. Then the noise is defined as the set of points in the database D not belonging to any cluster C_i , i.e. noise = $\{p \in D \mid \forall i: p \notin C_i\}$.
- If $d(p,q) \leq Eps$, q is core point and $|NEps(p)| < Minpts$ then p is a border point.

DBSCAN searches for clusters by checking the Eps neighborhood of each point in the database. If the Eps neighborhood of a point P contains $Minpts$ points or more, a new cluster with P as *core point* is created. Then DBSCAN iteratively collects directly *density-reachable* points from these core points, which may involve the merge of a few *density-reachable* clusters. The process terminates when no new points can be added to any cluster. In DBSCAN the $Minpts$ is fixed to 4 points, and the number of discovered clusters depends on the Eps value which is fixed during the execution time, and this value is not suitable for discovering clusters with different densities except when the clusters are totally separated. But when the clusters are not totally separated the DBSCAN algorithm faces the problem of varying densities and produces inaccurate clusters.

3 Related works

The DBSCAN (Density Based Spatial Clustering of Applications with Noise) [5] is a basic density based clustering algorithm. The density associated with a point p is obtained by counting the number of points in a region of specified radius Eps around p .

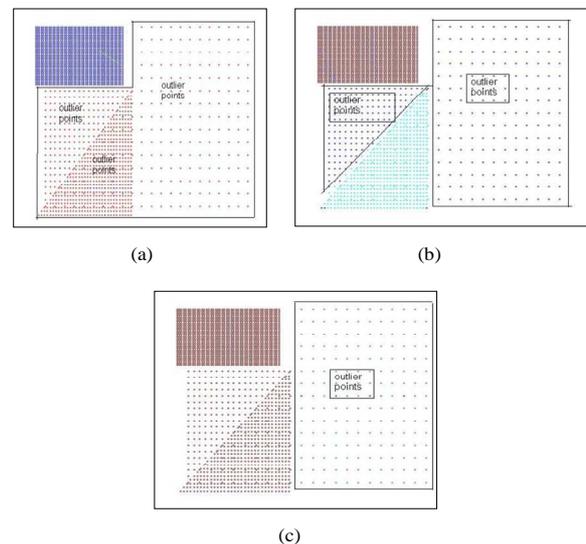


Fig.2. (a) DBSCAN discovers only the densest cluster ($Eps=0.25$). In (b) it discovers two clusters satisfying the lowest possible density ($Eps=0.4$). In (c) it merges the two clusters in (b) with the intermediate points to produce only one cluster ($Eps=0.5$) and discards the other points as noise.

A point with density greater than or equal to a specified threshold $Minpts$ is treated as core point (dense), otherwise non-core (sparse). Non-core points that do not have a core point within the specified radius are discarded as noise. Clusters are formed around core points by finding sets of density connected points that are maximal with respect to density-reachability. DBSCAN can find clusters having varying sizes and shapes, but there may be wide variation in local densities within a cluster since it uses global density parameters $Minpts$ and Eps , which specifies only the lowest possible density of any cluster without restricting the highest possible density. Fig. 2 shows the results of applying the DBSCAN on the dataset in Fig. 1.a. From Fig. 2, we note that for some datasets we need many different values for the Eps parameter of DBSCAN algorithm to discover clusters with different densities, and we can not depend on a single value for this sensitive parameter. This problem motivates us to propose an enhanced version of DBSCAN which is able to discover clusters from datasets having varying densities.

To find clusters that are naturally presented in dataset very different local densities needed to be identified and separated into clusters. The proposed algorithm performs this task efficiently. DBSCAN is very interesting algorithm; it has received a lot of attention from many researchers. But most of them concentrated their efforts in improving the scalability of it. We briefly review some of these works; for example in IDBSCAN [4], the researchers improved the scalability of the algorithm by reducing the number of query region. This has been done by sampling the points in the Eps neighborhood of core point p and marking these points as Marked Boundary Objects (MBO), for each of these MBO s, the closest point in the Eps -neighborhood -as in Fig. 3- is identified and selected as a seed. If the same point is identified as the nearest point for more than one MBO then this point must be regarded only once as a seed. Therefore total number of seeds selected may be less than or equal to eight in 2-dimensional case. In general, if the points are of dimension d , then there will be (3^d-1) MBO s, 2^d quadrants and number of seeds selected is at most 3^d-1 . This number is lower than the corresponding number in DBSCAN. IDBSCAN [4] is more efficient than DBSCAN, but it occasionally may treat some boundary objects as noise producing a few more noise points than DBSCAN.

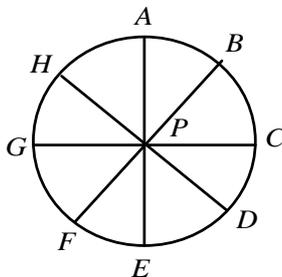


Fig. 3. Circle with eight MBO

Another algorithm was introduced to improve the efficiency of DBSCAN by merging between the k-means and

IDBSCAN, this algorithm is known as KIDBSCAN [9]. K-means yields the core points of clusters. Then, clusters are expanded from these core points by executing IDBSCAN. The purpose of the k-means is to determine the k highest density center points and the points that are closest to these center points. These points are moved to the front of data. And the IDBSCAN starts the expansion of clusters from high density center points. This process reduces the number of redundant steps and increases efficiency of IDBSCAN.

In [8] the authors introduced VDBSCAN (Varied Density Based Spatial Clustering of Applications with Noise). First, VDBSCAN calculates and stores k-dist for each point. K-dist plot is drawn for selection of parameters Eps_i , and analysis of density levels of the dataset. Second, the number of densities is given intuitively by k-dist plot. Third, the user chooses parameters Eps_i for each density. Fourth, VDBSCAN scans the dataset and clusters different densities using corresponding Eps_i . And finally, it displays the valid clusters corresponding with varied densities. VDBSCAN has two steps: choosing parameters Eps_i and clustering in varied densities. This method depends on seeing several smooth curves connected by greatly variational ones, and in many cases we see only one smooth curve, referring to our experimental results we see only one curve in dataset 8 and two curves in dataset 7 and more than 12 curves in 4-dist plot in dataset 3. And this is only true for dataset 4 where the points of each region are uniformly distributed.

Recently DDSC (Density Differentiated Spatial Clustering) [3] was proposed. It is an extension of the DBSCAN algorithm to detect clusters with differing densities. Adjacent regions are separated into different clusters if there is significant change in densities. DDSC starts a cluster with a homogeneous core object and goes on expanding it by including other directly density-reachable homogeneous core objects until non homogeneous core objects are detected. A homogeneous core object p is a core object whose density (the count of points in its Eps -neighborhood) is neither more nor less than α time the density of any of its neighbors, where $\alpha > 1$ is a constant. DDSC requires three input parameters, they are Eps , $Minpts$ and α . As we know Eps and $Minpts$ depend on each other. However, finding the correct parameters for standard density based clustering [5] is more of an art than science. In addition to the third parameter α , which represents the allowed variance in densities within each cluster, and proper tuning of the parameter values is very important for getting good quality results.

4 The Proposed Algorithm (Enhanced DBSCAN)

In this section, we describe the details of the proposed algorithm. As DBSCAN is sensitive to Eps , the proposed algorithm will adjust different values for this parameter in each cluster. The algorithm finds the k-nearest neighbors for each point in given dataset as DBSCAN does, but here the

enhanced algorithm does not build the *sorted k-dist graph*. Based on the k -nearest neighbors, a local density function is used to find the local density at each point which is an approximation of the overall density function [6]. The overall density of the data space can be calculated as the sum of the influence functions of all data points. The influence function can be seen as a function which describes the impact of a data point within its neighborhood, and it is applied to each data point. Local density at a data point is computed according to the following functions:

Influence function represents the impact of point x on point y as the Euclidean distance between them.

$$INF(x, y) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2} \quad (1)$$

Where d is the dimensionality of points. As the distance between the two points decrease the impact of x on y increase, and vice versa.

The **local density function** at point x is defined as the sum of the (influence functions within the k -nearest neighbors) distances among the point x and its k -nearest neighbors, this is shown in (2).

$$DEN(x, y_1, \dots, y_k) = \sum_{i=1}^k INF(x, y_i) \quad (2)$$

The definition of local density based on the sum of distances of the k -nearest neighbors is better than counting the points in the neighborhood radius (Eps). Consider the following example as in Fig. 4.

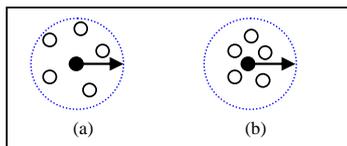


Fig. 4. Density based on k distances.

As we see in Fig. 4.a and 4.b both black points have five points in Eps -neighborhood (the Eps is the same represented by black arrow) but Fig. 4.b is denser than Fig. 4.a. The sum of distances reflects this fact accurately. In this example $Maxpts = 5$, and $Minpts = 4$. So the proposed algorithm detects appropriate value for Eps in each cluster that will be $Maxpts$ -distance from the starting point in the cluster. But based on fixed Eps -neighborhood radius as in DBSCAN algorithm there is no difference, because each point has five points in its Eps . And we can note that the value of Eps varies according to local density by accumulating the distances to the $Maxpts$ -neighbors. Also the point inside the cluster has high density, on the other hand the point at the edge (border) of cluster has low density, since the neighbors for this point lie on one side

of it, but the point at the core of cluster has its neighbors surrounding it from all sides. So density based on summing of distances is better than counting points in neighborhood radius.

We use the advantage of k -nearest neighbors to determine a suitable value for Eps in each cluster separately. The k -nearest neighbors capture the concept of neighborhood dynamically. The neighborhood radius of a data point is determined by the density of the region in which this data point resides. In a dense region, the neighborhood is defined narrowly and in a sparse region, the neighborhood is defined more widely [7]. By examining Figure 4, if we determine the value of Eps to be the distance to the 5th neighbor for the densest point in each cluster, we will find two different values for the Eps parameter, the first value will be for the black point in cluster b and the other will be for the black point in cluster a. The exact two input parameters for the proposed algorithm are $Minpts$ and $Maxpts$; $Minpts < Maxpts < 20$. These two parameters determine the minimum and maximum density for core points respectively. $Maxpts$ also determines the Eps for each cluster according to the highest local density of its starting point. The $Maxpts$ -distance is the same as k -distance or the distance to the k -nearest neighbor.

The algorithm finds the k -nearest neighbors for each point p , and keeps them in ascending order according to their distances to p . (i.e. $N_k(p) = \{q \in D, d(p, q_{i-1}) \leq d(p, q_i), i=1, \dots, k\}$), for each neighbor q_i of p its distance to p and its input order in the input dataset are kept in one vector. The algorithm computes the density on each point p according to (2), and arranges the points in dataset in descending order according to the local density of each point using the quick sort. So the first point will be the densest point and the last will be the sparsest one. The algorithm initializes all points as unclassified ($ClusId = -1$), and sets value of $Minpts$ to 4. The user inputs $Maxpts$ that will be used to determine Eps parameter. Starting from the unclassified point p with the highest local density, the algorithm determines the Eps as the distance to $Maxpts$ -neighbor for this point p .

The algorithm expands the cluster around the point p starting from the nearest directly density-reachable point q , if q is a core point, then its unclassified neighbors at Eps will be appended to the seed list, otherwise q is border point and no points are directly density-reachable from it. This process is continue until the seed list is empty, and the algorithm starts new cluster with new value for Eps , and ignores the clustered points. Because the points are arranged in descending order according to their local density, the denser clusters will be created first and the sparser clusters will be created later. The distance to the k -nearest neighbor in dense region tends to be small while in sparse region it tends to be large. This distance is a very good indicator for a suitable value of Eps in each cluster (region). Based on this idea, the proposed algorithm is able to change Eps to be suitable for the current cluster. The

experimental results confirm this idea. And this algorithm is able to discover clusters having different densities even if there is no separation among them. The following are the main steps of the proposed algorithm:-

1. Find the k -nearest neighbors for each point p . (i.e. $N_k(p)$) and keep them in ascending order from p , in this step $k=20$.
2. Set local density value for each point p as $DEN(p, y_1, \dots, y_k)$, in this step $k=10$.
3. Rearrange the data points in descending order according to their local densities.
4. $ClusId=1$.
5. Starting from the first unclassified point p in the sorted data do the following:-
 - a. Eps = distance to $maxpts$ -neighbor for the point p .
 - b. Assign the point p to the current cluster ($ClusId$).
 - c. Append its unclassified neighbor q_i , wrt. Eps and $Minpts$ to the seed list SL_p in ascending order of their distance to p , Continue expanding current cluster until no point can be assigned to it.
6. $ClusId = ClusId + 1$.
7. Assign the next unclassified point to the current cluster and go to step 5 until all points are classified.

Step 1 of the proposed algorithm imposes an ordering on the k -nearest neighbors, and these neighbors will be appended to the seed list SL_p in the same order as in step 1. Step 3 makes the creation of clusters start from the densest cluster and ended with the sparsest one, at the same time allows variance in density within a cluster according the deference between $Maxpts$ and $Minpts$. And $Maxpts$ is the only input parameter the user set to the algorithm. Step 5.c appends the points in ascending order to the seed list to impose growing of cluster in contiguous regions. Since the density of every point in the cluster compared with the density of the starting point of it, the algorithm may discover small high density clusters. The algorithm discards the small clusters as outlier.

4.1 Time complexity

The most time consuming part of the algorithm is to find the k -nearest neighbors. The neighborhood size ($k=20$) is very small compared to the size of the dataset. So, the different tests performed on the neighborhood to arrange them will not consume much time. While expanding a cluster the list of newly contributed seeds by each object of the cluster are already sorted before. For all objects only a small fraction of the neighbors become new seeds, whereas some points contribute no new seeds at all. The time required for a neighborhood query is $O(\log n)$ by using a spatial access method such as R*-tree [2]. Neighborhood query is performed for each of the n points in the dataset. Also we arrange the points according to their local densities using quick sort, which requires $O(n \log n)$. If we don't arrange the points, we must search for the densest point in each new cluster creation, this process requires $O(nm)$; where m is the number of

obtained clusters and n is the size of the input dataset. So the run time complexity is $O(n \log n)$ which is the same as that of DBSCAN.

5 Experimental Results

In this section we evaluate the performance of the proposed algorithm. We implemented this algorithm in C++. We have used many synthetic datasets to test the proposed algorithm. We experimented with eight different data sets containing points in two dimensions whose geometric shapes are shown in Fig. 5. The first dataset has six clusters of different sizes, shapes, and orientation, as well as random noise and special artifacts such as streaks running across clusters.

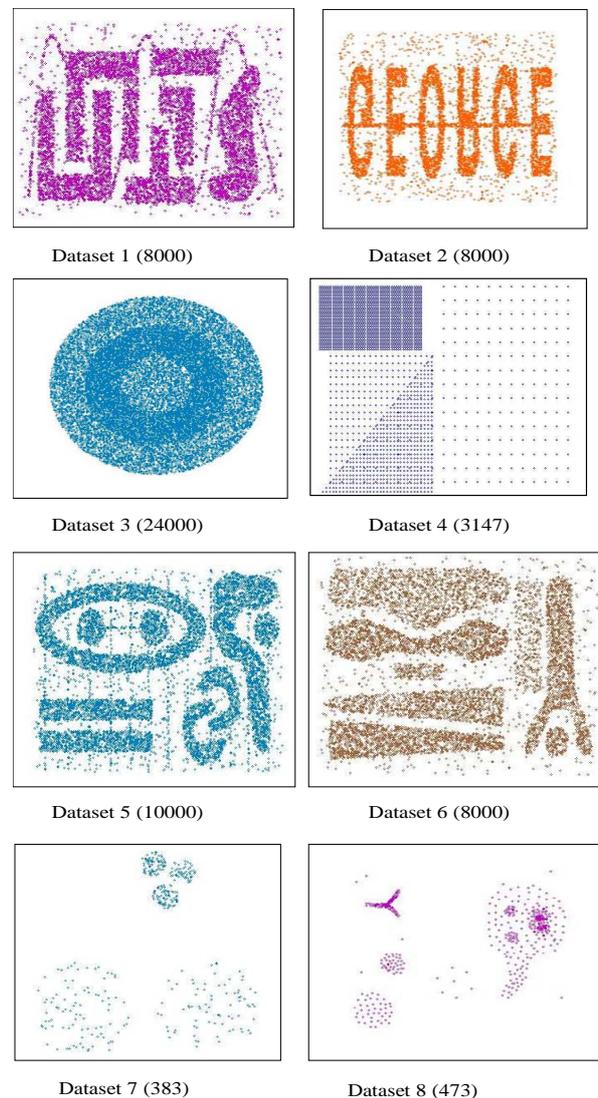


Fig. 5. Datasets used to evaluate the algorithm and their sizes

The second dataset has six clusters of different shapes. Moreover, it also contains random noise and special artifacts, such as a collection of points forming horizontal streak. The

third dataset has three nested circular clusters of different sizes, and densities. A particularly challenging feature of this dataset is that clusters are very close to each other and they have different densities and there is no separation between them. The fourth dataset has four clusters of different shapes, sizes, and densities. There are no sparse regions among clusters. The fifth and sixth data set contain clusters of different shapes and densities which are very close to each other. The seventh data set contains five clusters, three of them have high density and are very close, which motivate DBSCAN to merge them in single cluster to detect the other two, or to detect these three clusters and discard the two large as noise. Finally, in eighth data set that contains clusters with varying densities in addition to three dense clusters involved in large intermediate dense cluster. The size of these data sets ranges from 383 to 24000 points, and their exact sizes are indicated in Fig.5.

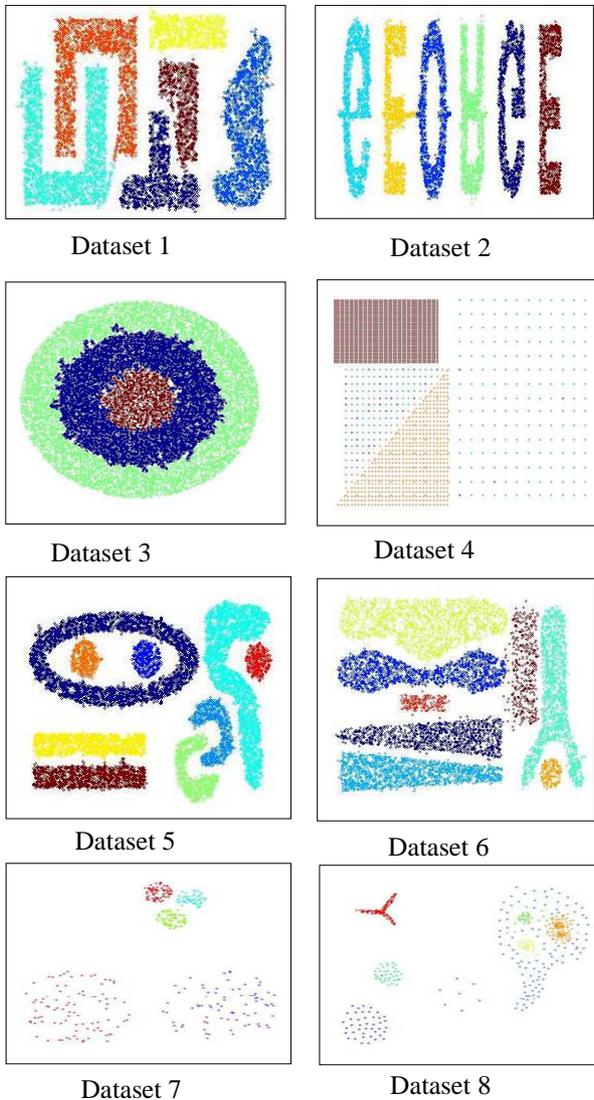


Fig. 6. The results from the proposed algorithm

The clusters resulting from applying the proposed algorithm on these eight datasets in Fig.5 are shown in Fig.6. Different colors are used to indicate the clusters. The very small clusters are discarded as noise. It can be seen from Fig.6 that the triangular and rectangular clusters are discovered and extracted based on differences in densities although they are not separated by sparse regions. This is will be clear when we compare the result of dataset 4 in Fig.6 with that in Fig.2 that result from the DBSCAN algorithm.

Table 1 the different values for Eps in each dataset according to the Maxpts-distance of the starting point in each cluster

	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	Dataset6	Dataset7	Dataset8
Eps1	5.233	3.045	4.243	0.224	7.006	9.191	0.672	1.374
Eps2	6.666	4.111	7.280	0.424	6.248	7.189	0.783	1.772
Eps3	7.169	3.488	7.071	0.539	6.805	7.580	0.763	3.100
Eps4	5.233	4.250		1.170	6.765	8.977	2.785	3.236
Eps5	5.735	4.226			7.689	8.474	2.911	4.955
Eps6	6.052	3.851			5.918	8.696		6.576
Eps7					6.334	9.610		6.519
Eps8					8.501	16.812		18.355
Eps9					7.976			

From Fig. 6, the proposed algorithm discovers the correct clusters in datasets 3 and 4, where there is no separation between the clusters which have varying densities. Also it discovers the right clusters in datasets 7 and 8. We concentrate on these four datasets because they contain

clusters with varying densities. Table 1 shows the exact value for the *Eps* in each cluster according to the *Maxpts*-distance of the starting point in each one. We can't determine these accurate values from *k*-dist plot as stated in VDBSCAN [8], because this method depends on seeing several smooth curves connected by greatly variational ones. The following Fig. 7 presents the 4-dist plot for some datasets that have clusters with varying densities. Examining this Figure, we should select values that cluster dataset 3 into more than three clusters. For dataset 7 it is very difficult to determine from the *k*-dist plot those values that cluster it correctly, and this is the same situation for dataset 8.

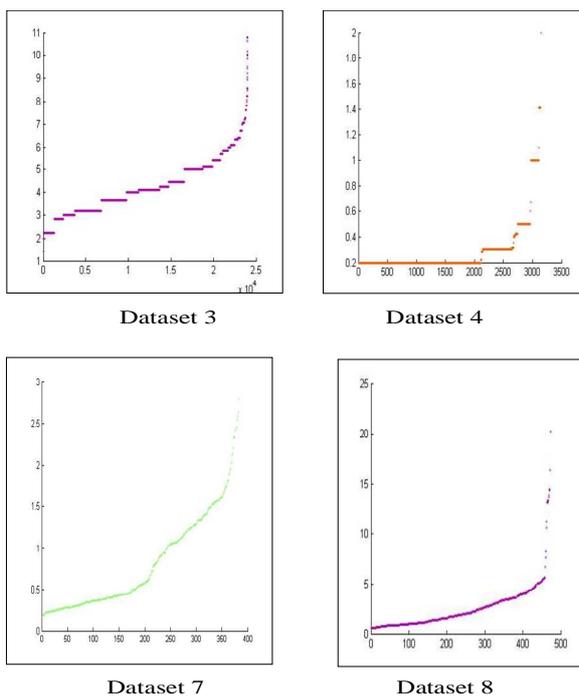


Fig. 7. The 4-dist plot

6 Conclusion

In this paper, we have introduced a competent idea to improve the results of DBSCAN algorithm by detecting clusters with varying densities without requiring any separation between clusters. The *Minpts* is fixed to 4 as in traditional DBSCAN. The proposed algorithm requires only one input parameter (*Maxpts*) which specifies the maximum number of points around a core point; in other word, it specifies maximum level of density for a core point within any cluster. As the value of *Maxpts* decrease the number of discovered cluster increase and the points within a cluster are more similar to each other, and the vice versa. The difference between *Maxpts* and *Minpts* represents the allowed difference in density inside the cluster. The time complexity of the algorithm remains $O(n \log n)$. The experimental results are the evidence on the efficiency of the proposed algorithm.

7 References

- [1] M. Ankerst, M. Breunig, H.P. Kriegel, and J. Sandler. "OPTICS: Ordering Points to to Identify the Clustering Structure"; proceedings of the Int. Conf. on Management of Data (SIGMOD'99), pp. 49-60, 1999.
- [2] N. Beckmann, H.P. Kriegel, and R. Schneider. "The R*-tree an Efficient and Robust Access Method for Points and Rectangles"; proc. ACM SIGMOD, Utlantic City, USA, pp. 322-331, May 1990.
- [3] B. Borah, and D.K. Bhattacharyya. "DDSC: A Density Differentiated Spatial Clustering Technique"; Journal of Computers, Vol. 3, No. 2, pp. 72-79, February 2008.
- [4] B. Borah, and D.K. Bhattacharyya. "An Improved Sampling-Based DBSCAN for Large Spatial Databases"; proceedings of International Conference on Intelligent Sensing and Information, pp. 92-96, 2004.
- [5] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. "A density based algorithm for discovering clusters in large spatial data sets with noise"; in 2nd International Conference on Knowledge Discovery and Data Mining, pp. 226-231, 1996.
- [6] A. Hinneburg and D. Keim. "An efficient approach to clustering in large multimedia data sets with noise"; in 4th International Conference on Knowledge Discovery and Data Mining, pp. 58-65, 1998.
- [7] G. Karypis, E. H. Han, and V. Kumar. "CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling"; Computer, 32, pp. 68-75, 1999.
- [8] P. Liu, D. Zhou, and N. Wu. "VDBSCAN: Varied Density Based Spatial Clustering of Applications with Noise"; International Conference on Service Systems and Service Management (ICSSSM), pp.1-4, 2007.
- [9] C-F. Tsai, and C-W. Liu. "KIDBSCAN: A New Efficient Data Clustering Algorithm"; ICAISC, PP. 702-711, 2006.

GCLOD : A clustering algorithm for improved intra-cluster similarity and efficient local outliers detection

Abdul Aleem, Reena Srivastava, A K Singh and M M Gore
 Department of Computer Science and Engineering
 Motilal Nehru National Institute of Technology
 Allahabad, U.P., India

Abstract—Clustering is a well-known functionality of data mining. Clustering aims for high intra-cluster similarity and low inter-cluster similarity. Most of the existing techniques for clustering use distance measures, which consider the overall similarity/dissimilarity among the records. These measures ignore the effect caused by individual attributes, thus leading to intra-cluster dissimilarity. In a relational database, it happens sometimes that some records have the attribute values which are drastically different to the usual values taken up for those attributes in other records. Such records, known as local outliers, are absorbed in the clusters during the clustering mechanism. In clustering, the local outliers' absorption sometimes produces low quality clusters and provides inaccurate results. This paper proposes a clustering algorithm - GCLOD, that achieves high intra-cluster similarity for clusters and detect local outliers. The GCLOD algorithm avoids local outliers' absorption through the usage of a new distance measure - sd , which considers the dissimilarity caused by individual attributes along with the overall dissimilarity. The computation of intra-cluster dissimilarity is done through measure of clustering error. Experimental results over three well-known datasets show that the GCLOD produces clusters with minimal clustering error as compared to the existing methods and detect local outliers in nearly linear time.

1. Introduction

Clustering is one of the various functionalities of Data Mining. Clustering finds vast application in market research, pattern recognition, data analysis, and image processing. Clustering is defined as the process of grouping a given set of records, into various groups, such that the members of the same group are similar and the members from different groups are dissimilar. The computation of similarity/dissimilarity is done through distance measures, based on the values of the attributes of the records. To store the dissimilarity computed among all the pairs, a dissimilarity matrix is used. The clustering algorithms compute dissimilarity on almost all types of data such as interval-scaled, categorical, ordinal, ratio-scaled, binary and mixed-type variables.

The quality of clusters obtained can be measured through clustering error [8], [3]. Clustering error is defined as the

sum of the squares of deviations of all points within a cluster from its mean. Mathematically clustering error can be computed as:

$$E(m_1, m_2, \dots, m_M) = \sum_{i=1}^N \sum_{k=1}^M I(x_i \in C_k) |x_i - m_k|^2$$

where C_1, \dots, C_M are clusters with their centers as m_1, \dots, m_M [8]. The intra-cluster similarity is inversely proportional to the clustering error.

The term Data Segmentation has been used interchangeably for clustering due to its ability to segment similar types of data. This behavior of clustering leads to outlier detection, which is sometimes of utmost interest for data analyst. Outliers are those values, which lie far apart from any of the clusters, in terms of their attributes. An Outlier can be a local outlier or a global outlier. Global outliers are those which lie far apart from any of the clusters due to some unusual values of attributes exhibited by the particular record. On the contrary, local outliers are those which do not have such unusual entries but they are not fit to be absorbed into clusters due to the distribution of other records in the space.

The clustering algorithms applied to cluster the data are based on similarity/dissimilarity computed among the records. The similarity/dissimilarity measure yields the overall effect depicted by all the attributes. A relational database containing binary, categorical and ordinal variables may frequently lead to absorption of local outliers into clusters, leading to low intra-cluster similarity. The local outliers having drastically different values for some of their attributes may produce an overall dissimilarity equivalent to that of non-outlier records due to the compensational behavior of the attributes' values. This absorption of local outliers into clusters may yield disastrous result when clustering real world data like Comparative Genomic Hybridization (CGH) as done in [4]. Researchers have diverted their focus towards density-based methods for better clustering result. But density measure is computed through distance measure and so the chances of creeping local outliers persists.

The requirement for any clustering algorithm is the ability to deal with different type of attributes. Further, the algorithm should discover clusters of arbitrary shape and must be able to deal with noise and outliers. The final

clusters formed are expected to be independent of initial points i.e. the clusters should not converge locally on the initial points chosen. But the existing clustering algorithms do not satisfy all these requirements at the same time. The local convergence of K-means algorithm has been dealt in [8], [12], [15], but absorption of local outliers in the final clusters obtained has been overlooked. Moreover, if an obvious distance measure does not exist; it is needed to be explicitly defined, which is not always an easy task.

Apart from the deficiency of absorption of these local outliers through clustering techniques, recent outlier detection techniques do not take corrective measures to detect such local outliers. A comparison of various distance-based techniques have been done for better local outliers' detection in [2], but the distance measure under consideration calculates dissimilarity collectively for all the attributes rather than considering individual attributes. A combination of distance-based and density-based methods have been applied to detect local outliers for data partitions in [6] but it depends on the sampling scheme followed for partition. An enhancement of local outliers' detection has been done in [9], which is an extension to density-based methods. The improved formulation of LOF is based on nearest distance and hence, local outliers tend to be absorbed. To detect local and global outliers in mapping studies, use of mathematical models have been done in [1], but it can be used only for outliers' detection and does not yield clusters.

This paper proposes a distance-based clustering algorithm - GCLOD (Global Convergence and Local Outlier Detection), which produces clusters having high intra-cluster similarity and yield local outliers that may get absorbed into clusters. The algorithm starts with the selection of refined initial cluster points, thus avoiding local convergence of distance-based mechanism, and then stores the dissimilarity of individual attributes among the record pairs along with the overall dissimilarity. The clustering process counts both the overall dissimilarity as well as the dissimilarity contributed by individual attributes. Finally, clusters having high intra-cluster similarity are obtained and the records having vast deviation for some of the attributes' values are detected as local outliers.

The rest of the paper is organized as follows. Section 2 discusses about the related work. Section 3 explains the problem of local outlier absorption and has been elaborated with an example. Section 4 introduces the GCLOD algorithm. Section 5 discusses the complexity analysis of the algorithm. Section 6 provides experimental results and finally section 7 provides conclusions and future work.

2. Related Work

Clustering can be done through various methods namely partitioning method, hierarchical method, density-based methods, etc. The simplest form of clustering is done through partitioning methods, which partition the data into

required number of clusters and then use an iterative relocation technique. Hierarchical methods rely on the hierarchical decomposition of data. The distance-measure used by partitioning methods and hierarchical methods is altered to give a new parameter as density for clustering through density-based methods. Distance-based clustering is more efficient than density-based clustering [6].

The basic algorithm, specifically known as Naive K-Means [3], does not meet all the requirements of clustering algorithms and is slow. The algorithm converges to different local minima based on the initializations. The distance measure used considers overall dissimilarity, which leads to low intra-cluster similarity. In addition to these drawbacks, the algorithm can not be used to detect local outliers.

N. Hussein [12] presented a fast Greedy K-means that overcomes some of the drawbacks of K-means. The fast Greedy K-means is a hybrid approach of the two algorithms - The Blacklisting Algorithm and The Filtering Algorithm. These two algorithms work on hyper-rectangles as constructed through kd-trees. kd-trees lead to formation of regions, whose centers/centroids could be used for further computation instead of working on whole dataset. This leads to acceleration of K-means. The Greedy K-means algorithm tends to be fast and scalable for high-dimensional dataset yet it relies on overall dissimilarity computed and effect of individual attributes' dissimilarity is ignored. The method yields clusters having low intra-cluster similarity and cannot be used for local outliers' detection.

Aristidis Likas et. al. [8] presented Global K-means clustering algorithm, an incremental approach to clustering. The Global K-means starts with one cluster and then dynamically adds one cluster at a time. The underlying idea being that the solution for $K=1$ is always known and optimal. The algorithm places next cluster through a deterministic global search procedure. The K^{th} cluster is then placed in $K-1$ clusters. The algorithm makes use of k-d trees and explores the concept that if there are lesser number of elements than b (pre-specified number, also called bucket size) in a bucket (or cluster), then the algorithm terminates, indicating that there is no need of adding new clusters. The Global K-means is independent of initial points, converges globally and does not depend on empirically adjusted parameters. On the contrary, it relies on overall distance computed and effect of individual attributes' dissimilarity is ignored. The algorithm does not consider the intra-cluster dissimilarity among clusters. The Global K-means cannot be used for local outliers' detection.

Apart from clustering mechanism, various distance-based and density-based techniques exist to detect outliers. The Local Outlier Factor(LOF) [2] use measures based on the overall dissimilarity rather than dissimilarity contributed by individual attributes. Hence, the chances of detecting such local outliers become negligible. Density-based methods may comparatively detect more local outliers than distance-based but compensational behavior exhibited by some of

the attributes of the relational database record may not be exposed. Some other methods like distribution-based and deviation-based techniques do perform a better task than the above mentioned techniques yet they require the data to follow a mathematical model and hence are not suitable to be used as a general approach for local outliers' detection.

R. Kashef et. al. [2] have proposed a new local outlier factor CBLOF to determine the prioritization for a record to be an outlier. Pie Sun et. al. [6] have applied a combination of distance-based and density-based methods to detect local outliers in high-dimensional relational database. A. L. Chiu et. al. [9] have done the enhancement on local outliers' detection technique by grading the degree of outlieriness. L. M. Ainsworth et. al. [1] have made use of mathematical models to detect local and global outliers in mapping studies. The deviation mechanism limits the approach in [1] to data distribution conforming to a mathematical model. Apart from being the non-clustering techniques, these methods [2], [6], [9], [1] make use of density measures based on nearest overall distance and hence, local outliers tend to be absorbed.

Paul S. Bradley et. al. [15] have given a method to determine initial points through the distribution of the data. Rather than analyzing complete dataset, they have worked upon the random samples of dataset to be clustered. After taking sufficient number of sub-samples of the dataset, they clustered sub-samples through K-means. The modes of the joint probability of the data and placing a cluster at each mode are found to find the initial cluster points. The initial points with minimum distortion are accepted as initial points for clustering. The algorithm is superior to the one for obtaining initial points through Spherical Gaussian distribution method. It is effective to work with more than one subsample for better result. The algorithm solves well the purpose for which it was intended and provides initial points for a clustering algorithm to cluster the data. The clustering algorithm (may be K-means, EM etc.) applied make use of commonly used distance measure, which may lead to intra-cluster dissimilarity. The algorithm cannot be used for local outliers' detection.

3. Local Outlier Absorption - An Elaboration

Currently existing distance-based clustering algorithms make use of distance measures, which are computed considering all the attributes collectively. The dissimilarity measure does not consider the values of individual attributes. Hence, there is a high probability that the records having vastly distinct values for few of the attributes get absorbed into clusters. Let us take an example to illustrate the mechanism. Consider a relational database having 5 attributes and 5 records to be clustered into 2 groups. The attributes are conventionally notated as *atr1*, *atr2*, ..., *atr5*. To demonstrate the heterogeneity in relational database, we consider versa-

tile attribute-types. *atr1*, *atr2* are interval-scaled variable. *atr3* is an ordinal variable having 3 categories. *atr4* and *atr5* are binary variables. The table 1 shows the normalized entry on a scale of 0-100.

Table 1: A normalized example relational database

	Atrb1	Atrb2	Atrb3	Atrb4	Atrb5
Record1	75	41	100	100	0
Record2	97	9	100	100	0
Record3	46	95	0	0	100
Record4	13	66	0	0	100
Record5	12	67	50	100	100

Let the Record2 and Record4 be initial points for clustering. Conventional approaches would form a matrix for dissimilarity (considering Manhattan distance) among clusters' centre and records as shown in Matrix 1. The rows are used for clusters' centre and columns are used for Records. We will use the notations CC and EI for clusters' centre and records respectively.

$$\begin{bmatrix} 54* & 0* & 437 & 441 & 293 \\ 387 & 441 & 62* & 0* & 152* \end{bmatrix}$$

Matrix 1: First iteration of Dissimilarity Computation

The first iteration groups records EI1 and EI2 into first cluster and records EI3, EI4 and EI5 into second cluster. The smaller distance for each record from a cluster is marked with an asterisk(*). The new cluster centers are now CC1(86, 25, 100, 100, 0) and CC2(23.67, 76, 17.33, 33.3, 100). The second and final iteration of clustering would give the matrix for dissimilarity computation as shown in Matrix 2.

$$\begin{bmatrix} 27* & 27* & 410 & 414 & 269 \\ 269 & 323 & 92* & 71.33* & 170* \end{bmatrix}$$

Matrix 2: Second iteration of Dissimilarity Computation

The second iteration yields dissimilarity, which does not bring any changes in the cluster assignment. So, the clustering algorithm stops with the result that EI1, EI2 forms cluster1 and EI3, EI4, EI5 forms cluster2. Looking carefully into the table for Record5, it can be observed that out of the three ordinal variables, two of its values are complimentary to its cluster members Record3 and Record4. Binary variables just represent two states and may depict some important property of the data in a yes/no form. Apart from this, the dissimilarity computed for Record5 never emphasize such vast deviation in few of its attributes and the Record5 get easily absorbed into cluster2. In a high-dimensional large relational database, there could be many instances of such cases and the clustering technique may yield clusters having high intra-cluster dissimilarity.

In the example mentioned above, Record1 and Record2, both have deviation from cluster1's mean of 27. The total clustering error for cluster1 is 0.0754 on the standard scale of 0-1. Similarly, Record3, Record4 and Record5 have devia-

tions of 92, 71.33 and 170 respectively from cluster2's mean. The total clustering error for cluster2 is 1.5392. Hence, the clustering error for the clusters formed is 1.6146. The high value of clustering error demonstrates that the intra-cluster similarity of clusters formed is too low and the quality of clusters formed is poor.

4. The GCLOD Clustering Algorithm

The GCLOD algorithm provides solution for local convergence and local outliers absorption. The local convergence problem has been solved by refining initial points for clustering as in [15] and explained in section 2. Along with the inclusion of global convergence, the GCLOD algorithm detects those local outliers which may be absorbed. To deal with the absorption of local outlier, a new distance measure has been defined, which does not overlook the contribution made by individual attribute. To avoid dissimilar records to be in the same cluster, each attribute is treated distinctly such that the dissimilarity in one attribute does not compensate the dissimilarity of other attributes. This leads to the detection of local outliers with respect to a single cluster. Any such candidate which is outlier to all the clusters is a local outlier for the dataset. The separate treatment for dissimilarity of all the attributes of a particular record is done through a distance measure - **sd** - a vector to store the dissimilarity of a record from any other record with respect to all the attributes. The **sd** vector also stores the sum of the dissimilarities depicted by all the individual attributes. Thus, the **sd** vector has length 1 greater than number of attributes.

A **diff** matrix is used to store various **sd** vectors calculated with respect to all the refined initial cluster points. The number of rows in **diff** matrix is equal to the number of clusters to be formed and the number of columns is 1 greater than the number of attributes. So, **diff[cc][a+1]** matrix stores dissimilarity of a record from **cc** number of initial cluster points with respect to all the attributes of the dataset. The additional column is appended to store the Manhattan sum of all the individual dissimilarities existing between attributes. The **diff** matrix is used to make comparisons and decides to which cluster a particular record belongs to, in case if it is not an outlier.

The algorithm assumes that the dataset is transformed to numeric form and **record[n][a+1]** is a corresponding matrix for normalized dataset with **n** records and **a** attributes. The extra column is appended in the matrix to attach cluster id (of allocated cluster) with each record. The normalization is assumed to be done on a scale of 0-100. However, scaling can be done on any scale depending upon the precision to be preserved. Here, it is for two decimal places. Hence, the preprocessing step requires the transformation of all the data to numeric form, followed by normalization of the transformed values to a scale of 0-100. The transformed normalized value is stored in matrix **record**.

The algorithm starts with the refinement of initial points avoiding local convergence. The first step of the algorithm obtains the initial cluster points and gives total number of clusters to be formed. The second step of the algorithm computes the dissimilarity of a record with all the initial cluster points. The computed dissimilarity is with respect to all the attributes individually along with the sum of all the dissimilarities. The first expression computes dissimilarities existing among attributes, whereas the second expression computes the sum of all dissimilarities using Manhattan distance measure.

The heart of the algorithm lies in the third step, where for each record the smallest sum of dissimilarities is computed and then all the categorical, ordinal and binary variables are checked for the match. Besides the match for above mentioned attributes, all the interval-scaled variables are checked to have dissimilarity not greater than the threshold value, which is calculated based on the ratio of distribution range to the number of clusters. If the smallest sum satisfies these criteria, then the record is allocated to the cluster for which the dissimilarity sum is minimal. In case, if the criterion is not met, the algorithm repeats the above procedure with the next smallest value. The process goes on until the record is allocated a cluster or it is not allocated to any of the clusters. If it is not allocated to any of the clusters then the record is a local outlier.

The fourth step of the algorithm is just a repetitive structure, which executes the second and third step for all the records in the dataset. Algorithm 1 shows the GCLOD clustering algorithm.

Depending on the dataset and the clustering quality required, the number of mismatches may be increased in the step 3(a)(i) of the algorithm. Moreover, the threshold value in step 3(a)(ii) can be changed depending on the type of the distribution followed by the interval-scaled variables. The salient feature of GCLOD is that the algorithm is not needed to be run again for the new clusters' means as the clustering is done for refined initial cluster points. This saves a maximum of **n** times scan of the dataset and thus reduces the overall complexity of the algorithm.

Let us execute the GCLOD algorithm on the problem example, which we considered in section 3. The first step of the algorithm gives the Record2 and Record4 as the refined initial cluster points. The threshold value here is 50 (100/2). The diff matrix for first record is shown in Matrix 3.

$$\begin{bmatrix} 22 & 32 & 0 & 0 & 0 & 54 \\ 62 & 25 & 100 & 100 & 100 & 387 \end{bmatrix}$$

Matrix 3: Diff matrix for record1

The matrix lists the Manhattan distance between various attributes of Record1 and cluster1(Record2) in the first row. Similarly, the second row lists the Manhattan distance between various attributes of Record1 and cluster2(Record4). The last column shows the sum of these dissimilarities.

Algorithm 1 The GCLOD Clustering Algorithm

- 1) Obtain the initial cluster points as given by Bradley and Fayyad in [15]. Let **cc** be total number of clusters to be formed and matrix **ccentre** stores the attributes' value of every cluster.
- 2) Starting with the first record, compute the dissimilarity with all the cluster means. The dissimilarity is computed as
 $diff[i][j] = |ccentre[i][j] - record[i][j]|$, where $i = 1, 2, \dots, cc$ and $j = 1, 2, \dots, a$
 $diff[i][a+1] = diff[i][1] + diff[i][2] + \dots + diff[i][a]$
- 3) Compute the smallest dissimilarity value ($diff[i][a+1]$) of the record from all cluster centers. For the smallest **diff[i][a+1]**
 - a) Check whether all **diff** values
 - i) are zero for all categorical/ordinal/binary variables
 - ii) are less than or equal to $int(100/cc)$ for all interval-scaled variables. If yes, assign the record to cluster **i** ($record[x][a+1] = i$). Otherwise
 - b) Ignore the smallest value
 - c) Compute second smallest dissimilarity value
 - d) Perform similar checks on second smallest value and move (if required) to next smallest
 - e) The process is repeated until a cluster has been assigned or no **diff** values satisfy the above criteria
- 4) If no cluster is assigned to a record then it is an outlier. Repeat steps 2 and 3 with other records until all records are assigned clusters or detected as outlier.

The smaller value of overall dissimilarity is 54. For all the categorical and binary variables there is a match. All the interval-scaled attributes are less than the threshold (50). So, the algorithm does not look for next smaller value and allocate Record1 to cluster1. Record2 and Record4 does not need to be analyzed as they are the initial cluster point. Similar treatment for Record3 (diff matrix shown in Matrix 4) allocates it to cluster2.

$$\begin{bmatrix} 51 & 86 & 100 & 100 & 100 & 437 \\ 33 & 29 & 0 & 0 & 0 & 62 \end{bmatrix}$$

Matrix 4: Diff matrix for record4

The diff matrix for record5 is shown in Matrix 5. The smaller overall dissimilarity is 152 for cluster2. All the interval-scaled attributes are less than threshold (50). But there are mismatches for categorical attribute (atr3) as well as in one of the binary attributes (atr4). So, the algorithm does not allocate Record5 to cluster2. Now the algorithm looks for next overall smaller dissimilarity, which is 293 for cluster1. The individual dissimilarity for interval-scaled

attributes (corresponding to overall dissimilarity 293) has a distance greater than threshold (50) and so the algorithm will not allocate it to cluster1. Since, Record5 has not been assigned to any cluster and all the cluster centres have been examined, therefore Record5 is detected as a local outlier.

$$\begin{bmatrix} 85 & 58 & 50 & 0 & 100 & 293 \\ 1 & 1 & 50 & 100 & 0 & 152 \end{bmatrix}$$

Matrix 5: Diff matrix for record5

The algorithm gives two clusters as cluster1 and cluster2, where cluster1 contains Record1 & Record2 and cluster2 contains Record3 & Record4. The other record, namely Record5 is detected as a local outlier. If the clustering quality is degraded and one mismatch is allowed for categorical/ordinal/binary variables, then also the clustering result remains same. Further degradation of clustering quality, allows Record5 to be grouped into cluster2, which has two mismatches for categorical/ordinal/binary variables. In relational data clustering, the clustering quality can be slightly degraded to allow some of the nearest outliers to be absorbed into clusters. Ignoring more mismatches would lead to absorption of all the local outliers and yield clusters having low intra-cluster similarity.

The effectiveness of the GCLOD algorithm can be seen through the measure of clustering error. When the number of allowed mismatch is 0 or 1 for ordinal attributes, the clustering error for cluster1 is 0.0754 and for cluster2 is 0.0966. Hence, the total clustering error is 0.172, which is almost one-tenth of the clustering error obtained in section 3. Thus, it can be argued that the GCLOD algorithm yields clusters of better quality even with the inclusion of few mismatches. Further permitting two mismatches induces an error of same magnitude as obtained in section 3, thus exhibiting the intra-cluster dissimilarity due to absorption of local outlier.

5. Complexity Analysis of GCLOD Algorithm

The GCLOD algorithm works in three stages - data preprocessing, refining initial points and clustering records. The GCLOD algorithm processes one record at a time. The disk-scan for algorithm is just one time. In case of large disk resident dataset partitioning may be employed to limit disk-scan to one. The GCLOD algorithm during data preprocessing scans **n** records from the relational database, which has **a** attributes. Hence, the complexity of scanning the database for data retrieval is $O(na) \sim O(n)$, since **a** is a small constant. The complexity for refinement of initial points is $O(m^2) \sim O(n)$, where **m** is a small fraction of **n**.

The clustering process is done in three phases - dissimilarity computation, smallest dissimilarity value selection and the selection of next smallest dissimilarity value, if required. Dissimilarity computation for a record costs $O(ac)$, where

c is the number of clusters to be formed. The selection of smallest dissimilarity value can be done in $O(ac)$ time complexity. The selection of next smallest dissimilarity is required at most $c - 2$ times, which can be done in $O(ac^2)$ time complexity. Hence, the total time complexity for assigning cluster to a single record is $O(ac^2)$, assuming all other operations require unit time. Thus, the total time complexity for assigning clusters to n records is $O(nac^2) \sim O(n)$, since a and c are small constants. So, overall time complexity for preprocessing, refining initial points and cluster assignment is $O(n + n + n) \sim O(3n) \sim O(n)$.

6. Experimental Results

The GCLOD algorithm was compared with Min K-means and Global K-means. All the algorithms were executed with three relational databases namely Iris [13], Synthetic[16] and Students' Database. The implementation was done on a 2.79 GHz Pentium P4 PC, with 512 MB of RAM, 80 GB of Hard Disk and Windows XP as operating system. The algorithms Min K-means, Global K-means and GCLOD were implemented with Microsoft Visual Basic 6.0 and Microsoft Access. The results were compared on the basis of the quality of the obtained solution, where the quality of clusters is described in terms of the final clustering error as described in section 1.

The Iris database had 150 four-dimensional records and all the attributes were interval-scaled. The synthetic database had 1000 two-dimensional records and both attributes were interval-scaled. The Students' database is of students, who seek to take admission in a particular course at a particular institution. The original students' database had nine attributes, namely Serial Number, Name, Age, Sex, Marks obtained in written test, Marks obtained in personal interview, Rate of response from student, Caste-Category and Physically Handicapped. The preprocessed database contained schema Student(S.No., Age, Sex, Category, PhyHan, MarksWT, MarksPI, Responses), where all the attributes excluding S. No. were normalized to scale of 0-100. The attributes Age, MarksWT and MarksPI are interval-scaled variables. The attribute Category is an ordinal attribute with three categories - General, OBC and SC/ST. The attribute Responses is also an ordinal variable with 3 categories - Quick, Medium and Slow. The attributes Sex and PhyHan are binary variables having two states as Male/Female and Yes/No respectively. The database had 100 records.

For all the datasets, one run of GCLOD algorithm, one run of Global K-means and N runs of K-means were executed. For K-means, out of N runs, the result that yielded minimal clustering error was recorded. The number of clusters formed with these data sets varied from 3 to 15. The results of comparisons for Iris, Synthetic and Students' database are shown in figure 1 to 3. The outliers detected through GCLOD are shown in figure 4. The results show GCLOD as a better option for clustering all the datasets.

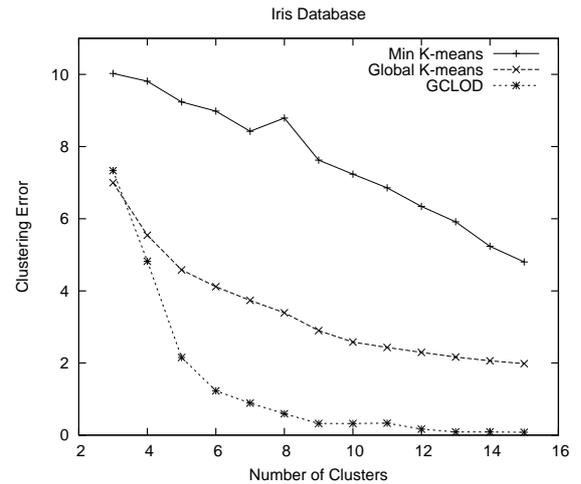


Figure 1: Comparison of various algorithms on Iris Database

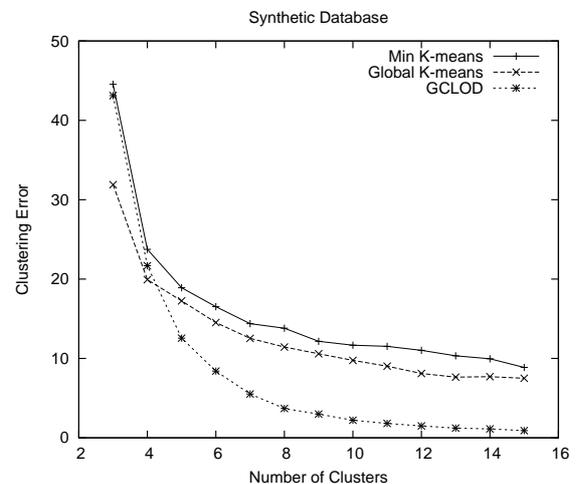


Figure 2: Comparison of various algorithms on Synthetic Database

In Iris and Synthetic dataset, where the attributes are only interval-scaled variables, the results show better performance for GCLOD with respect to other algorithms. In relational database having ordinal variables, GCLOD provides more accurate results as depicted by figure 3. The clustering error in Students dataset is near to zero, when the other algorithms have high clustering error. The number of outliers in figure 4 increases as we increase the number of clusters, since increasing number of clusters leads to more refined clustering.

7. Conclusion and Future Scope

Clustering real-world data may involve the absorption of local outliers into the cluster, resulting inaccurate result. The absorption is primarily due to the distance measure considered, which is based on the overall dissimilarity existing between cluster points and records. This paper proposes

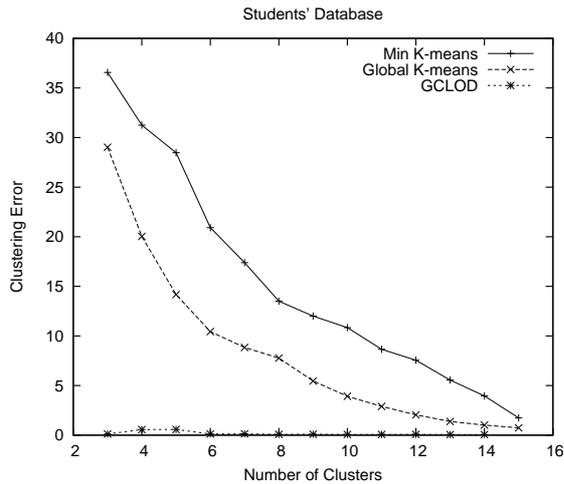


Figure 3: Comparison of various algorithms on Students' Database

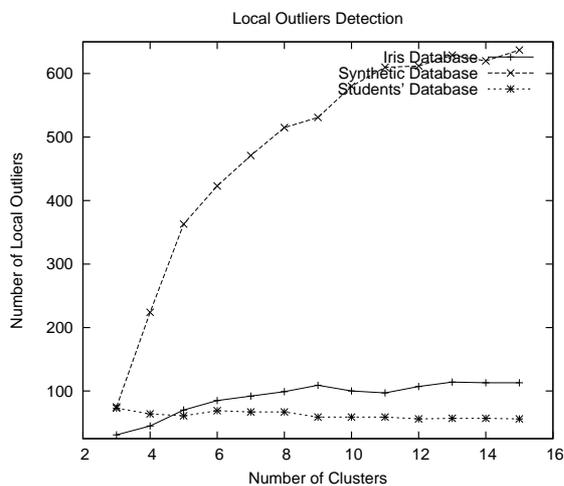


Figure 4: Local Outliers detected through GCLOD

GCLOD clustering algorithm, which groups data considering the dissimilarity exhibited by individual attributes as well as the overall dissimilarity. The method provides excellent result in the context of intra-cluster similarity, which is depicted through the measure of clustering error. The proposed algorithm is independent of initial points, thus avoiding the problem of local convergence for distance-based methods. Besides clustering, the algorithm also detects outliers and prevents the local outliers to be absorbed into the clusters. The algorithm has an edge over other existing algorithms, where more accurate clustering is required. Further, the quality of the clusters formed can be slightly degraded to absorb the nearest outlier into the cluster, depending on the clustering quality required.

As a future work, implementation of the GCLOD algorithm can be done on a parallel architecture, where cluster

assignment for records can be done at parallel processors. Another direction for future work is related with the implementation of the GCLOD algorithm in other databases like spatial database etc. Another research direction concerns the application of proposed method to other types of clustering like fuzzy clustering etc. Finally, the proposed algorithm can be combined with other clustering techniques like density-based or deviation-based techniques.

Acknowledgment

The authors would like to thank Mr. Girish Patnayak, Mr. Manas Mishra and Mr. Vijayendra Singh for their help and kind support.

References

- [1] L. M. Ainsworth, C. B. Dean. Detection of local and global outliers in mapping studies. *Environmetrics*, 19:21-37, 2008
- [2] R. Kashef and M. S. Kamel. Towards Better Outliers Detection for Gene Expression Datasets. In the proceedings of International Conference on Biocomputation, Bioinformatics, and Biomedical Technologies. May, 2008
- [3] Jiawei Han and Micheline Kamber. *Data Mining : Concepts and Techniques*. Second Edition. Morgan Kaufmann Publishers ÁT2006 by Elsevier.
- [4] Jun Liu, Jaaved Mohammed, James Carter, Sanjay Ranka, Tamer Kahveci and Michael Baudis. Distance-based clustering of CGH data. Advance Access Publication. Oxford University press. May 2006
- [5] Fabrizio Angiulli, Stefano Basta, and Clara Pizzuti. Distance-Based Detection and Prediction of Outliers. In Proceedings of the 18th IEEE Transactions on Knowledge and Data Engineering, NO. 2, February, 2006
- [6] Pei Sun, Sanjay Chawla, Bavani Arunasalam. Mining for Outliers in Sequential Databases (Won the Best Application Paper Award). In proceedings of the Sixth SIAM International Conference on Data Mining, Bethesda, Maryland, USA, 2006, pp. 94-105. Along with complete Master's Thesis of Pei Sun.
- [7] Girish Keshav Palshikar. Distance-Based Outliers in Sequences. In Proceedings of the International Conference on Distributed Computing and Internet Technology (ICDCITÁÁ05). Springer-Verlag Berlin Heidelberg. pp. 547-552, Decemeber, 2005
- [8] Aristidas Likas, Nikos Vlassis, Jacob J. Verbeek. The global K-means clustering algorithm. *Pattern Recognition*, 36:451-461, 2003.
- [9] Anny Lai-mei Chiu, Ada Wai-chee Fu. Enhancements on Local Outlier Detection. In Proceedings of the Seventh International Database Engineering and Applications Symposium (IDEASÁÁ03)
- [10] Stephen D., Bay Mark Schwabacher. Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule. In Proceedings of the SIGKDDÁÁ03, August 24-27, 2003, Washington DC, USA
- [11] Yeou-Min Lin. MBA: A Matrix-Based Partitioning Clustering Algorithm for Non-Numerical Data. Master's Thesis. National Cheng Kung University, Tainan, Taiwan, R.O.C., June 2003.
- [12] N. Hussein. A Fast Greedy K-Means Algorithm. Master's Thesis. University of Amsterdam, Netherlands, November 2002.
- [13] C. L. Blake and C. J. Merz, "UCI repository of machine learning databases", University of California, Irvine, Dept. of Information and Computer Sciences, 1998
- [14] Edwin M. Knox and Raymond T. Ng. Algorithms for Mining Distance-Based Outliers in Large Datasets. In Proceedings of the Twenty-fourth VLDB Conference, New York, USA, 1998
- [15] Paul S. Bradley, Usama M. Fayyad. Refining initial points for K-Means Clustering. In Proceedings of the 15th International Conference on Machine Learning(ICMLÁÁ98), J. Shavlik(ed.), pp. 91-99, Morgan Kaufmann, San Francisco, May 1998.
- [16] B. D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, U.K., 1996.

A Graph-based Similarity Metric and Validity Indices for Clustering Non-numeric and Unstructured Data

Chuan Zhao and Krishnamoorthy Sivakumar

School of EECS, Washington State University, Pullman, Washington, USA

Abstract—We consider the problem of clustering for an unstructured dataset consisting of non-numeric attributes. Examples of such datasets include emails, news articles, publication/citation indexes, movies etc. In particular, we consider the clustering problem where the interrelationships are represented in a graphical format. We propose a node similarity metric to assess the similarity between nodes. We consider the K -medoid clustering method and introduce some postprocessing steps to improve the quality of clustering. A set of validity indices are proposed to assess the quality of the clustering results. To reduce computational complexity, a sampling strategy is introduced. Effect of sampling on the values of validity indices and clustering result is discussed. Finally, influence of different similarity metrics, postprocessing steps, and cluster number on the quality of clustering is discussed, both analytically and experimentally.

Keywords: clustering, unstructured data, similarity metric, validity index

1. Introduction

Data mining and knowledge discovery in databases often requires dividing data into groups where similar objects are assigned to the same group. This task is often executed by clustering. Clustering groups objects with high similarity into a single cluster and separates objects with low similarity into different clusters. Data that requires clustering is often unstructured, consisting of non-numeric attributes (e.g., movie data, Email data, news articles). A movie contains a variety of different attributes including director, actor, role, etc. An Email has attributes like sender, receiver, subject, date, etc. All these attributes are non-numeric (with the exception of date). Clustering is usually performed based on distance between objects in the data set. When the attributes are non-numeric, there is no obvious notion of distance between two objects. Instead, we measure the degree of similarity between objects and consider an efficient way to calculate this similarity value.

In this paper, we consider a graphical representation — bipartite graph — of objects and relationships between them in a non-numeric dataset. The Relationship Generating Graph Analysis Engine (REGGAE) [1] developed by Applied Technical Systems (ATS) is used as our database and programming engine. We propose a similarity metric between objects based on REGGAE graph structure. We implement our clustering algorithm using different variations of the proposed similarity metric. Our graph representation scheme, similarity metric,

and clustering method are generally applicable to a variety of datasets. For concreteness, we use a movie dataset (IMDB) [2] to describe our work. We also propose metrics to evaluate the clustering results. We present two validity indices — Cohesion-Separation or CS for short and Representativeness of Medoid or RoM for short — which are based on the cohesion and separation of clusters. Furthermore, we present another validity index called Concentration of Similarity (CoS for short) which measures how well the nodes similar to a given node are concentrated in the same cluster.

To reduce computations involved in calculating the validity indices, we sample the nodes in each cluster and calculate the validity indices based on the sampled nodes. We compare different sampling methods based on their sampling errors and time complexities. Inspired by our experimental results, we present two theorems relating the validity index and cluster number. Based on our experimental results, the influence of different similarity metrics, clustering postprocessing steps, and cluster number on the quality of clustering is discussed. We conclude this section with a brief review of work related to node similarity and clustering validity indices.

Blondel *et al.* [3] calculated the similarity matrix for each node pair in a graph with adjacency matrix for synonym extraction and web searching. Huang *et al.* [4] proposed a node structural metric to measure the similarity between nodes making use of the number of shared edges. They used this metric to cluster graph to simplify the graph representation. Lu *et al.* [5] explored the definition of similarity based on connectivity only, and proposed several algorithms for this purpose. Their metrics took advantage of the local neighborhoods of the nodes in the networked information space. Jeh [6] measured the structural-context similarity using bipartite similarity rank. Their basic idea was that two objects are similar if they are related to similar objects. Lifshits [7] discussed similarity search in bipartite graphs. For a bipartite graph with people and movies in each partite, the person-person similarity is the number of 2-step (person to movie then to person) chains in the graph. Person-movie similarity is the number of 3-step chains. Cook [8] described the similarity measure between two entity clusters as a weighted combination of the attribute similarity and graph-based similarity between them. Finally, see [9], [10] for a review of clustering validity and methods.

2. Graph Representation

Similarity between non-numeric data points can be quantified by analyzing attributes they have in common and relationships

they share. Naturally, we use a graph to represent these data objects and their relationships. Each value of each attribute is represented by a node. An edge connects two nodes which have a direct relationship. For example, for movie data, we have node of title, name (for people in the movie such as actor), role, etc. Names and roles in one movie are connected to each other and to that movie title node.

Unfortunately, this kind of graph representation can lead to some problems. Consider the following scenario: Two movies m_1 and m_2 have a same role called r_1 , an actor a_1 played role r_1 in m_1 and role r_2 in m_2 , and another actor a_2 played role r_1 in m_2 ; then the graph depicting this scenario would be as shown in Figure 1 (a):

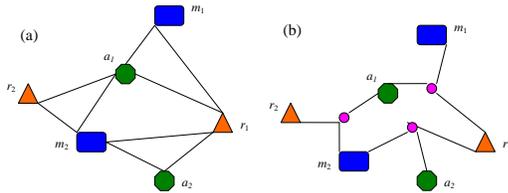


Fig. 1: Graph structures depicting relationships between movies

In Figure 1 (a), ambiguity is introduced as it seems to imply that a_1 also played (role) r_1 in (movie) m_2 . How can we resolve this problem without introducing duplicate nodes for the same data element? REGGAE solves this problem by storing context information.

Traditional graph approaches to representing relationships between objects have only one layer, the data layer. REGGAE is a bipartite graph structure consisting of a data layer and a context layer. Data layer contains only data (e.g., “Name: a ”, “Role: r_1 ”, “Title: m_1 ”) and the context layer stores connections between data nodes. Nodes in the context layer may only connect to nodes in the data layer, and nodes in the data layer may only connect to nodes in the context layer. The formal definitions of data layer and context layer are as follows [1]:

Data Layer: Populated with entities (cells), with each entity represented by a basic Type-Value construct.

Context Layer: Populated with context nodes (links), which provide contextual relationships between entities on the graph.

Figure 1 (b) is the REGGAE bipartite graph depicting the previous scenario. The small circles (colored purple) are nodes on the context layer and the rest (colored other than purple) are nodes on the data layer. A node on the data layer is called an “entity” and a node on the context layer is called a “link node.” We can see that the problem in the traditional graph is solved through the introduction of link nodes.

For entity nodes of REGGAE graph, besides the nodes for properties’ values, we should also include one other kind of node called “Data” node to represent each data object, then property value’s node can be called “Property” node which takes the form of “PropertyName: PropertyValue”, e.g., “Role: Super man”. For Data node, we can use the unique ID of

this data object in the dataset as its value, such as “Movie: MovieID” or “Email: MsgID”. Note that if the property’s value is a set, we should use one node for each value in the set. To summarize, REGGAE provides us the following advantages:

- Facilitates the analysis and programming for programmer and users;
- Eliminates ambiguity and provides a unified platform for algorithm design and programming.

3. Similarity Metric

Similarity metrics play a key role in our clustering method; it is a measure of “closeness” between two objects instead of distance. Different similarity metrics can lead to very different content and quality of the clusters.

We now discuss how to measure similarity between objects represented by our bipartite graph. Let us use the movie example for illustration. First, let us see how two movies are similar in a natural way; then we will see how it is reflected in a graph. We know two movies can be similar in many ways. They may have the same director(s), same actor(s), or same role(s). These are direct similarities. Two movies can also be similar in some indirect ways. For example, they do not have any common actors, but their actors played roles in a same third movie (we call this kind of movie “common neighbor movie”). In this situation, we consider these two movies as similar in an indirect way.

Generally speaking, similarity led by common Property nodes is called *Direct similarity*; while similarity led by the common Data nodes is called *Indirect similarity*. This is because in our REGGAE data representation scheme, Data node is connected to Property node directly (through link node), while Data node is never directly connected to another Data node.

We then have the following similarity metric for unstructured data (for Data node, unless specified otherwise):

$$S = w_d S_d + w_i S_i, \quad (1)$$

where S_d is the similarity value for direct similarities, S_i is the similarity value for indirect similarity, and w_d, w_i are their corresponding weights with $w_d + w_i = 1$. Typically $0 \leq S \leq 1$.

For S_d , we have: $S_d = \sum_{j=1}^m w_j S_{d_j}$, where S_{d_j} is similarity value associated with j th property of the data objects and w_j is the corresponding weight for j th property. Typically, $0 \leq S_{d_j} \leq 1$, $w_j \geq 0$ and $\sum_{j=1}^m w_j = 1$, m being the number of properties considered. Calculation of S_{d_j} depends on the data type of the property. It compares the similarity between property j ’s values for two data objects. Generally, extended Jaccard coefficient [11] is used. For example, for Director property of movie data, the similarity value should be the size of the intersection of two movies’ Director sets (the common neighbor Director nodes on the graph) divided by the size of the union of two movies’ Director sets. Note that some properties like actor and role form a natural pair in original dataset; treating them as one property

set in the similarity metric is reasonable. We will discuss this in detail in section 6. The link node in REGGAE is used to correctly identify such pairs.

There are two situations for S_i . One where two Data nodes have common neighbor Data node through the same common neighbor Property node; the other where two Data nodes have common neighbor Data node through different neighbor Property nodes. Because common neighbor Property nodes are already used for calculate S_d , we should only take the second situation into consideration while calculating S_i . Generally, extended Jaccard coefficient is also used to calculate S_i . For example, for movie data, S_i should be the number of common neighbor Movie nodes (only considering the second situation described above) divided by the size of the union of two movies' neighbor Movie nodes sets.

Existing similarity/distance metrics consider only the direct similarity, or similarity due to shared nearest neighbor (SNN) [11], whereas our similarity metric also considers indirect similarity. Moreover, it does not need a data object to be similar to its neighboring objects; this helps capture the relations between data objects more comprehensively. As we will see in section 6, inclusion of indirect similarity improves some aspects of clustering quality. The similarity metric described in [8] appears similar to our similarity metric, but they have following differences: 1) our similarity metric considers node similarity, not cluster similarity; 2) our (REGGAE) graph includes property nodes, not just data nodes; 3) our indirect similarity part (common neighbor data nodes) will not consider the data nodes which are derived by the common property node(s) since they are already used for the direct similarity part; the metric in [8] cannot or did not do this with the reference graph.

4. Clustering Algorithm

K -medoid is a simple prototype-based clustering algorithm that uses the medoid (the most representative one) of the objects in a cluster as the prototype of the cluster [12]. We use K -medoid because it requires only a proximity measure for a pair of objects which we already have through the similarity metric. Also, we use K -medoid because it tends to produce globular clusters in which each object is sufficiently similar to the cluster's medoid or to other objects in the cluster. We enhance the basic K -medoid algorithm in several respects which will be discussed below. Based on the algorithm, we introduced a set of validity indices to direct the clustering process, evaluate and improve the clustering quality.

4.1 Algorithm description

Our K -medoid clustering method has four steps: initialization, association, re-initialization, and postprocessing.

Initialization: The initialization step is to find K initial medoids. Our initialization method consists of three basic steps: sampling, determining first medoid, and successively determining rest of the $K - 1$ medoids. First, we select a small

sample of the nodes (usually 5-10% nodes); this can be done either randomly or by picking say one node every 20 or 10 nodes. Then, we take the median of the sample as the first medoid. We define the median as a node with the largest sum of similarities to other nodes in the sample. Finally, for each successive medoid, we select the node that is most dissimilar to any of the previously selected medoids.

Association: During Association, we associate each non-medoid node to its most similar medoid. We break ties randomly or associate the nonmedoid node to the smallest cluster, unless the node is most similar to the medoid of its current cluster, in which case we do not change its cluster membership.

Re-initialization: In this step, we compute the median node (node with largest sum of similarities to other nodes in the same cluster) for each cluster (after Association) and assign it as the new medoid of the cluster. If the new medoid for a cluster is a node that was used as a medoid in a previous iteration, we keep the current medoid. This is done to avoid oscillatory behavior where the medoid of a cluster repeatedly switches between two nodes.

Postprocessing: When no new medoids are generated, the Association – Re-initialization loop should be terminated. To improve the quality of clustering, we perform additional postprocessing steps. In our K -medoid clustering method, we use three different postprocessing strategies: *split*, *merge*, and *move*.

The general idea of a split is to split the largest or sparsest cluster. We split the cluster which (1) is too big, say, has more than 50% of all nodes; (2) has the smallest total/normalized nonmedoid-medoid/node-node (intra-cluster) similarity. In our experiments, we tried different split metrics in (2) and we observed that normalized node-node similarity was a good choice. After split, the medoid of the split cluster is replaced by two randomly selected nodes from the respective clusters (locally), or replaced by two nodes from the respective clusters or whole dataset (globally) which are farthest from current medoids.

The general idea of a merge is to merge the smallest clusters or closest clusters. We merge (1) the clusters which are too small, say, each has less than 1% of all nodes, or just has one node; (2) the two clusters which have the largest total/normalized medoid-medoid/node-node(inter-cluster) similarity. Because a merge often results in a decrease in cohesion, we impose some conditions on merge. We merge the two clusters which are: 1) the most similar (has biggest inter-cluster similarity); 2) densest (has biggest intra-cluster similarity); 3) particularly close (the similarity between them is bigger than some factor (e.g., twice) of the second biggest inter-cluster similarity. After merge, the medoids of the merged clusters are replaced by a randomly selected new node from the merged cluster (locally), or replaced by the node from the merged cluster or the whole dataset (globally) which is farthest from current medoids.

We alternate between the split and merge steps to control the number of the clusters. A sequence of split/merge steps

can be used to fine tune the total number of clusters, since this is not always known or given. Finally, note that split/merge postprocessing is performed after and followed by several iterations of Association – Re-initialization steps.

The third kind of postprocessing we perform is *move*. We call a node *lonely* if it shares no common Property node with any other nodes in its cluster. During a *move* we move lonely nodes to a different cluster where it shares the most (at least one) common Property node with other nodes in the cluster. This helps increase the quality of the clustering by increasing the cohesion and decrease the separation further. Note that this postprocessing step is executed only at the end of the whole clustering process.

Termination: Different termination conditions can be used to stop the execution of the clustering process. For example, when an validity index (like *CS*, *RoM*, or *CoS*) is bigger or smaller than some threshold. Alternatively, after split and merge are executed a given number of times, the clustering process can be stopped. In our experiments, we stop the clustering process after five split/merge steps (splitting of big cluster and the merging of small clusters are not counted).

4.2 Validity Indices

Cluster evaluation (validation) is an important and necessary step for any clustering algorithm. Cohesion and separation are often used for unsupervised cluster evaluation [11]. Cohesion indicates how closely nodes in one cluster are grouped together; separation indicates how well separated clusters are between each other. We propose three different validity indices; two of them are based on the idea of cohesion and separation. Each index measures a different aspect of clustering quality. Some notations used in our validity indices are formalized in Table 1.

Table 1: Notations used in validity indices

C_i	i th cluster
$ C_i $	Size of the i th cluster
c_i	medoid of i th cluster
K	Number of clusters
$s(x_i, x_j)$	Similarity between nodes x_i and x_j

Before we introduce our validity indices let us look at four definitions first. *Cohesion_{medoid}* or C_m for short is the normalized nonmedoid-to-medoid similarity within all the clusters:

$$C_m = \sum_{i=1}^K \left(\sum_{j=1}^{|C_i|} s(c_i, x_j) - 1 \right) / \sum_{i=1}^K (|C_i| - 1). \quad (2)$$

Here $s(c_i, x_j)$ is the similarity between medoid c_i and node x_j . It tells us how similar are nodes to their cluster medoid; it could be regarded as an indication of the average “size” of each cluster in similarity space (analogous to “radius” in Euclidean space). In general, for a fixed dataset, we want to maximize the value of C_m . Note that we subtract 1 from the numerator and denominator of equation (2) for removing the similarity

between medoid and itself for each cluster (according to our similarity metric, a node’s similarity to itself is 1).

Cohesion_{average} or C_a for short is the normalized node-to-node similarity within all the clusters:

$$C_a = \sum_{i=1}^K \left(\sum_{j=1}^{|C_i|} \sum_{l=j+1}^{|C_i|} s(x_j, x_l) \right) / \left(0.5 \sum_{i=1}^K (|C_i|(|C_i| - 1)) \right). \quad (3)$$

It indicates how “dense” the clusters are. Again, we want to maximize the value of C_a for a given dataset.

Separation_{medoid} or S_m for short is the average medoid-to-medoid similarity between all clusters:

$$S_m = \sum_{i=1}^K \sum_{j=i+1}^K s(c_i, c_j) / (0.5K(K - 1)). \quad (4)$$

A small value for S_m is desirable for a good clustering.

Separation_{average} or S_a for short is the normalized node-to-node similarity between all clusters:

$$S_a = \sum_{i=1}^K \sum_{j=i+1}^K \left(\sum_{m=1}^{|C_i|} \sum_{n=1}^{|C_j|} s(x_m, x_n) \right) / \sum_{i=1}^K \sum_{j=i+1}^K (|C_i||C_j|). \quad (5)$$

This value should also be small for a good clustering result.

The first validity index is called Cohesion-Separation or *CS* for short because it is based on C_a and S_a :

$$CS = C_a / S_a. \quad (6)$$

For a good clustering result, we expect a large value for C_a and a small value for S_a and hence a large value for *CS*.

The second validity index is called Representativeness of Medoids (*RoM* for short) because it measures to what degree the medoids are representative of the cluster nodes:

$$RoM = |\log(C_a / C_m)| + |\log(S_m / S_a)|. \quad (7)$$

If the medoid is representative of the cluster, we expect C_a and C_m to be close and hence the ratio would be close to one. Similarly, we also expect the ratio of S_m to S_a to be close to one. Therefore, a small value for *RoM* is preferred for good representativeness of medoids.

The time complexity for calculating *CS* or *RoM* is $O(n^2)$, where n is the size of the dataset. Clearly, C_a and S_a are the main time-consuming parts. To reduce complexity, we can sample the nodes in the clusters and then only use the sampled nodes to calculate C_a and S_a . We calculate the relative error (RMSE and STDEV) due to sampling, and the similarity of clustering results with and without sampling, which means we compared the contents of the clusters of non-sampling version and sampling version.

In addition to the above two validity indices, we also propose another measure called *Concentration of Similarity (CoS)* which measures how well the nodes similar to a given node are concentrated in the same cluster. For each node, we calculate its top m most similar nodes (using our similarity metric) within the same cluster — call this set A . We compare this with set

B consisting of the top m similar nodes among all the nodes in the dataset. The average size of $A \cap B$ (over all the nodes) expressed as a fraction of m is defined as CoS . Ideally, we want CoS to be as close to 1 as possible.

Note that CoS and C_m can be used to evaluate not only the whole clustering result but also a single cluster, while CS and RoM generally are only used for the whole clustering result. The value of CS , RoM , or CoS is independent on the dataset or similarity value, so they can be used as abstract validity indices to compare any two clustering results. Furthermore, unlike RoM , CS and CoS are independent on the clustering method.

Some most commonly used validity indices can identify only the well separated hyper sphere shaped clusters. As these indices measure the variance of the clusters around some representative points but some clusters, especially the arbitrary shaped clusters, do not have representative center point [13]. But for our validity indices, introduction of CoS can be used to measure arbitrarily shaped clusters.

The traditional K -medoid target function/validity index is to minimize the total sum of the distance between each data object and its medoid [14]. It is easy to see that as the value of K increases, each data object generally will be closer to its medoid, which would decrease the value of this target function. So this validity index is not a good measure when K is allowed to vary. In general, our validity index CS does not have this problem, because the denominator S_a may also increase with K (except in some special cases discussed in Theorem 5.2).

5. Influence of Postprocessing on Validity Indices

We now present three theoretical results about the influence of postprocessing on validity indices.

LEMMA 5.1. *Given a clustering of a dataset of size n , let $k + 1 \geq 2$ be the number of clusters, and x be the size of the biggest cluster. After a postprocessing (followed by re-association) of the given clustering result, let x' be the size of the biggest cluster (again, assuming we have at least two clusters). If $x' < (1 + \sqrt{1 + 2((n - x)^2/k - n + x^2)})/2$, then the total number of pairs of nodes, both belonging to the same cluster, after postprocessing is smaller than that before postprocessing.*

For example, with $n = 500$, $k = 10$, and $x = 100$, the condition in lemma 5.1 is satisfied if the size of the biggest cluster after postprocessing is $x' \leq 113$. Following is a simple sufficient condition for the condition in lemma 5.1: $x < (\sqrt{kn^2 - k^2n + kn} - n)/(k - 1)$, $k > 1$ and $x' < x$. For example, with $n = 500$ and $k = 10$, this condition is satisfied if the size of the biggest cluster before postprocessing is $x \leq 118$. Note that after a split, we often have $x' < x$.

The following Theorem (which uses Lemma 5.1) explains the effect of postprocessing on C_a .

THEOREM 5.1. *For a given clustering of a dataset, let n_1 be the total number (over all clusters) of pairs of nodes, both*

belonging to the same clusters, n_2 be the total number of pairs of nodes, each belonging to a different cluster. After a postprocessing followed by re-association of the the given clustering result, let n_{11} (out of the original n_1) be the number of pairs of nodes still in same cluster, $n_{12} = n_1 - n_{11}$ be the pairs of nodes now separated into two different clusters. Furthermore, let n_{22} (out of the original n_2) be the number of pairs of nodes now assigned to same cluster. Let s_{11} , s_{12} , s_{22} respectively be the average similarity value of the n_{11} , n_{12} , n_{22} pairs of nodes. Let C_a be the C_a value of the clustering result before split and C'_a be the C_a value after postprocessing (followed by re-association). We then have:

If the assumptions of Lemma 5.1 are satisfied, $\frac{s_{11}}{s_{22}} \triangleq \theta > 1$, and $\frac{s_{12}}{s_{22}} < \delta$, then $C'_a > C_a$, where $\delta \triangleq (n_{11}n_{22} + n_{12}n_{22} + \theta(n_{11}n_{12} - n_{11}n_{22})) / (n_{11}n_{12} + n_{12}n_{22}) > 1$.

This theorem shows that if $s_{11} > s_{22}$ (which is usually true for a postprocessing step) and the condition of Lemma 5.1 is satisfied, then postprocessing (split/merge/move) usually results in an increase in C_a , as long as $\frac{s_{12}}{s_{22}} < \delta$. Note that C_a would increase, even if $s_{12} > s_{22}$ (which indicates a bad reassociation), since $\delta > 1$. Moreover, the sufficient condition for Lemma 5.1 is often satisfied after a split; this explains why after a split, C_a often increases.

We now present a result about the relation between the number of clusters K , C_a , and S_a .

THEOREM 5.2. *Given C_a and S_a for some nontrivial ($1 < K < n$) clustering result with $K = k_1$ on a dataset of size n , and for any $1 \leq i \leq k_1$, $|C_i| = \frac{n}{k_1}$; i.e., all the clusters are of the same size. Furthermore, given C'_a and S'_a for a different nontrivial clustering result with $K = k_2$ (caused by postprocessing on the same dataset), and for any $1 \leq i \leq k_2$, $|C_i| = \frac{n}{k_2}$; i.e., all the clusters are again of the same size. Then $\frac{S'_a}{S_a} < \frac{k_2(k_1-1)}{k_1(k_2-1)}$ if and only if $\frac{C'_a}{C_a} > \frac{k_2(n-k_1)}{k_1(n-k_2)}$.*

In particular, if $k_2 > k_1$; i.e., the number of clusters increases, then $\frac{k_2(k_1-1)}{k_1(k_2-1)} < 1$, and $\frac{k_2(n-k_1)}{k_1(n-k_2)} > 1$. Therefore, if $\frac{S'_a}{S_a} < \frac{k_2(k_1-1)}{k_1(k_2-1)}$ or equivalently $\frac{C'_a}{C_a} > \frac{k_2(n-k_1)}{k_1(n-k_2)}$, then $S_a > S'_a$ and $C_a < C'_a$. In other words, an increase in C_a to some degree is accompanied by a decrease in S_a (which ultimately increases CS), when the number of clusters increases. This is often the case for a postprocessing split step. Some clustering methods like OPPOSUM partition the data into roughly equal-sized clusters [11], and for some applications like clustering market baskets, each cluster should contain roughly the same number of samples [15]. Theorem 5.2 can be directly applied to these methods and applications.

6. Experiments

We tested the K -medoid clustering with enhancements described in Section 4, using different initial cluster numbers and different sets of weights for the similarity metric. In each case, we calculated the validity indices.

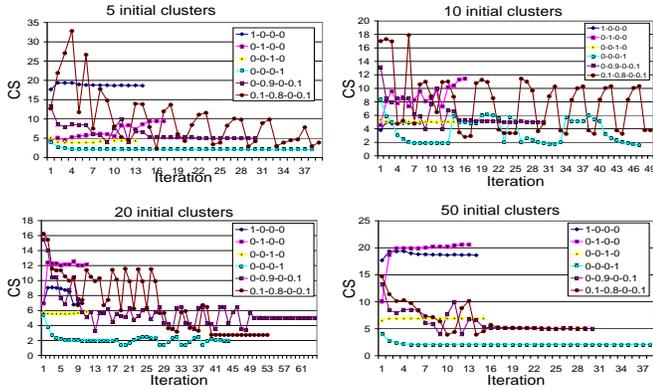


Fig. 2: Initial cluster number vs. similarity metric vs. CS

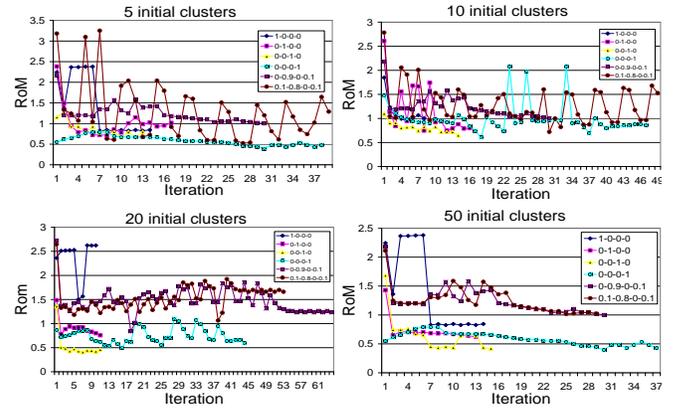


Fig. 3: Initial cluster number vs. similarity metric vs. RoM

6.1 Datasets

The IMDB movie dataset was imported into REGGAE for the experiments. Due to lack of space, we do not report our results on Netflix ratings and Enron email databases. This dataset has more than 480K movies; we generally used a subset of it. The experiments reported here were done using the movies released from year 1991 to year 2005; we used movies with a certain minimum number of actors, actresses etc. to obtain our test dataset of 2571 movies.

6.2 Similarity Metrics

For direct similarity S_d (see equation (1)), we used the following properties: *Director*, *Actor*, *Actress*, *Role*, *Producer*, *Editor*, *Writer*, *Composer*, and *Cinematographer*. For simplicity, we group *Actor*, *Actress*, *Producer*, *Editor*, *Writer*, *Composer* and *Cinematographer* together into one property called *Name*. Thus we have three properties determining S_d , which together with the indirect similarity measure S_i (see equation (1)) based on common neighbor movie (Data node), gives us four components in our similarity metric: *Director*, *Name*, *Role*, and *NeighborMovie*. We tried six different weight combinations for them: 1-0-0-0, 0-1-0-0, 0-0-1-0, 0-0-0-1, 0-0-9-0-0.1 and 0.1-0.8-0-0.1. Note that we consider *director* as a separate component since director plays a very important role in the style of a movie.

Treating *Actor/Actress* and *Role* as a pair (as mentioned in section 3), we used another similarity metric with four components: *Director*, *Actor-Role*, *otherName*, and *NeighborMovie*. We used weight combination 0.25-0.5-0.25-0 for this metric and compared that with the previous similarity metric where we put *Actor* and *Role* together but not in pairs (with the same weight combination).

6.3 Results

Figures 2–4 depict our validity indices (CS , RoM , and CoS) for different initial cluster numbers and similarity metrics. Figure 5 compares the clustering quality (with respect to CS and RoM) by treating *Actor* and *Role* in pairs with those

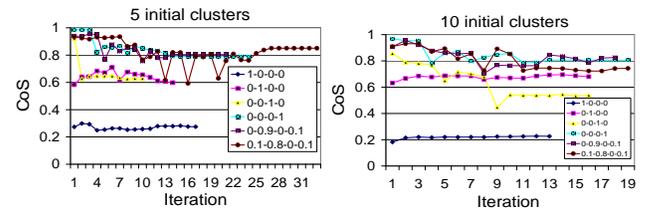


Fig. 4: Initial cluster number vs. similarity metric vs. CoS

obtained by treating them individually. Different subsets of the original dataset were used; their sizes are as indicated in the legend.

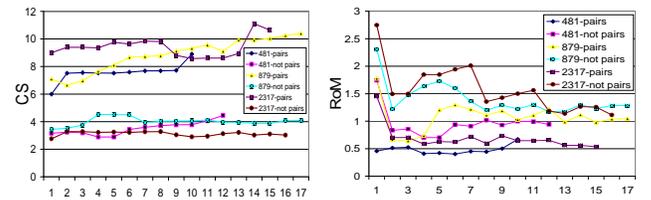


Fig. 5: Effect of Actor-Role pairs on cluster validity indices

To reduce computational complexity, we use a small sample of the nodes in each cluster to compute the cohesion C_a and separation S_a . In each case, we select a fraction (expressed as a sampling percentage) of the nodes from each cluster, using different sampling schemes. In one scheme we simply perform a uniform random sampling over each cluster (designated “random”). Our second sampling scheme was based on the similarity space for each cluster. For each cluster, we first choose either the medoid or a border node (node in cluster most dissimilar to medoid) as starting node (we designate this scheme as “medoid” or “border”, respectively). We then select nodes based on its similarity with the starting node (uniform sampling in similarity space). Finally, we used a “mixture” scheme where we pick up the medoid, a border node and one other node as starting node and take the average over the three results. Figure 6 shows the relative root-mean-squared-error (RMSE) and standard deviation of C_a and S_a due to sampling.

We present results for ten initial clusters and similarity metric weights 0-1-0-0; results are similar for other metrics/initial cluster numbers. For each sampling percentage we report the average and standard deviation over ten runs (top and bottom row, respectively, of Figure 6).

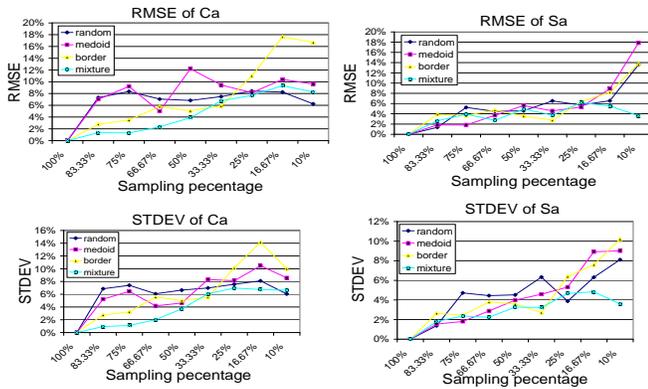


Fig. 6: Sampling percentage vs. sampling scheme vs. sampling error

6.4 Discussion

For index CS , we can see from Figure 2 that it increases with iteration for weights 0-1-0-0 and 0-0-1-0; it also increases with initial cluster number for the same weights, especially for 0-1-0-0. We generally use the clustering result from the iteration which has the best CS value; therefore from the perspective of peak CS value, weights 0.1-0.8-0-0.1 is the best for 5 and 10 initial clusters, whereas weight 0-1-0-0 is the best for higher number of initial clusters. From Figure 3, the index RoM is small for weights 0-1-0-0, 0-0-1-0, and 0-0-0-1, which means that the medoids are good cluster representatives. We also note that overall RoM decreases with iteration, except for 20 initial clusters. From Figure 4, we note that CoS is better when we use common neighbor movies in the similarity metric (last component non-zero). Similar results were obtained with 20 or 50 initial clusters. This indicates that inclusion of indirect similarity improves the similarity concentration quality (CoS measure) of the clustering. We can see that both CS and CoS reach a high value (e.g., 10 for CS and 0.8 for CoS), which indicates a good clustering quality. Figure 5 shows that the clustering quality improves significantly by considering *Actor* and *Role* in pairs.

Overall, sampling error decreases with increase in sampling percentage, which is to be expected. According to Figure 6, “random” and “border” sampling strategy is better for C_a , whereas for S_a medoid sampling scheme is better. Although the mixture strategy looks the best for all the cases, we cannot use it because it is an average over three schemes and hence has added complexity (it is presented only for comparison).

7. Conclusions

In this paper, we proposed a graph-based similarity metric used for clustering non-numeric and unstructured data. We

also presented a set of validity indices to evaluate the clustering quality. Theoretical and experimental results about the influence of different similarity metrics, cluster numbers, and postprocessing steps on the validity indices were presented and discussed. We presented a sampling scheme to reduce the computational complexity for the cohesion and separation measures. Overall, we can reduce the complexity by a factor of 9 (with 33.3% sampling) and still obtain quite good results.

The CS validity index is somewhat limited by the shape of the clusters, which needs to be improved. Further improvements to our algorithm can be achieved by jointly optimizing our validity indices. Since our similarity metric is derived from a graph representation of the dataset, in future work, we would like to design a graph-based clustering method which could better exploit the capabilities of the REGGAE database engine.

Acknowledgment

This work was supported by Washington Technology Center and Applied Technical Systems (awards RTD08WSFMC2 and RTD09WSFMC2). We thank J. Larson for his invaluable help and numerous discussions about the graph representation scheme, and providing his insight about the dataset.

References

- [1] J. Larson, *REGGAE Whitepaper*, Applied Technical Systems, Inc., 2008.
- [2] “The internet movie database,” <http://www.imdb.com/interfaces>.
- [3] V. D. Blondel, A. Gajardo, M. Heymans, P. Senellart, and P. V. Dooren, *A Measure of Similarity between Graph Vertices: Applications to Synonym Extraction and Web Searching*. SIAM Review, 2004, vol. 46:4.
- [4] X. D. Huang and W. Lai, “Clustering graphs for visualization via node similarities,” *Journal of Visual Languages & Computing*, vol. 17, no. 3, pp. 225–253, June 2006.
- [5] W. Z. Lu, J. Janssen, E. Milios, and N. Japkowicz, “Node similarity in networked information spaces,” in *Proceedings of the 2001 conference of the Centre for Advanced Studies on Collaborative research*, ser. IBM Centre for Advanced Studies Conference. Toronto, Ontario, Canada: IBM Press, 2001, p. 11.
- [6] G. Jeh, “Simrank: A measure of structural-context similarity,” in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.
- [7] Y. Lifshits, “Similarity search: a web perspective,” October 2007, available at <http://yury.name/algoweb/lifshits-retreat.pdf>.
- [8] D. J. Cook and L. B. Holder, *Mining Graph Data*. Hoboken, NJ John Wiley & Sons, Inc. (US), 2007.
- [9] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, “Clustering validity methods: part I,” in *ACM SIGMOD Record (ACM Special Interest Group on Management of Data)*, vol. 31, June 2002, pp. 40–45.
- [10] —, “Clustering validity checking methods: part II,” in *ACM SIGMOD Record (ACM Special Interest Group on Management of Data)*, vol. 31, September 2002, pp. 19–27.
- [11] P. N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Pearson Education, 2006.
- [12] L. Kaufmann and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, 1990.
- [13] C. Legány, S. Juhász, and A. Babos, “Cluster validity measurement techniques,” in *AIKED’06: Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2006, pp. 388–393.
- [14] I. Saha and A. Mukhopadhyay, “An improved crisp and fuzzy based clustering technique for categorical data,” *International Journal of Computer Science and Engineering*, vol. 2, no. 4, pp. 184–193, Fall 2008.
- [15] N. R. Pal and L. Jain, *Advanced Techniques in Knowledge Discovery and Data Mining*. Springer, 2005.

Comparison of Standard and Optimized K-means in SQL

Prof. R. Nedunchelian¹, Dr.R.Muthucumarasamy², Dr.E. Saranathan³

¹Department of Computer Science and Engineering, Sri Venkateswara College of Engineering
Anna University, Chennai, Tamil Nadu, India.

²Department of Applied Mathematics, Sri Venkateswara College of Engineering
Anna University, Chennai, Tamil Nadu, India.

³Department of Civil Engineering, Sastra University, Thanjavur, Tamil Nadu, India.

Abstract

Integrating data mining algorithms with a relational DBMS is an important problem for database programmers. We introduce two SQL implementations of the popular K-means clustering algorithm to integrate it with a relational DBMS: 1) A straight-forward translation of K-means computations into SQL. 2) An optimized version based on improved data organization, efficient indexing, sufficient statistics, and rewritten queries. We experimentally show the proposed K-means implementations work correctly and can cluster large data sets. We identify which K-means computations are more critical for performance. The optimized K-means implementation exhibits linear scalability. We compare K-means implementations in SQL and C++ with respect to speed and scalability and we also study the time to export data sets outside of the DBMS. Experiments show that SQL overhead is significant for small data sets, but relatively low for large data sets, whereas export times become a bottleneck for C++.

1. Introduction

Clustering algorithm with a relational DBMS using SQL, that is nowadays the standard language in relational databases. Clustering algorithm, partition a data set into several groups such that points in the

same group are close (similar) to each other and points across groups are far (different) from each other. There is work on improving speed and quality of solutions of K-means but the problem of integrating it into a relational database has received little attention. Having a clustering algorithm implemented in SQL provides many advantages. SQL is available in any relational DBMS. SQL isolates the application programmer from internal mechanisms of the DBMS. Many data sets are stored in a relational database. Trying different subsets of data points and dimensions is more flexible, faster, and, generally, easier to do inside a DBMS with SQL queries than outside with alternative tools. Managing large data sets without DBMS support can be a daunting task. Space management, fault tolerance, secure access, concurrency control, etc., are automatically taken care of by the DBMS for the most part. Although it is possible to efficiently cluster a very large data set outside a relational database, the time to export it to a workstation can be significant. Data mining has evolved to a point where a data mining algorithm is just one step inside a bigger process. Therefore, being able to cluster a data set stored inside a relational database can solve these issues. Clustering technique that selects cluster centers directly from the data set, allowing it to speed up the fitness evaluation by constructing a look-up table in advance, saving the distances between all pairs of data points, and by using binary representation rather than string representation to encode a variable number of cluster centers.

2. Existing Approaches

Partitioning algorithms: Construct various partitions and then evaluate them by some criterion.

Hierarchy algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion. This is an agglomerative approach and a divisive approach. Major *weakness* of agglomerative clustering methods: do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects can never undo what was done previously.

Integration of hierarchical clustering with distance-based method:

BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters.

CURE (1998): selects well-scattered points from the cluster and then shrinks them towards the center of the cluster by a specified fraction.

CHAMELEON (1999): hierarchical clustering using dynamic modeling.

Current clustering techniques do not address all the requirements adequately (and concurrently). Large number of dimensions and large number of data items. Strict clusters vs. overlapping clusters.

3. Proposed Work:

In this paper, we propose an optimized version of the standard K-means to improve the algorithm in terms of scalability with high dimensional data, speed and performance. The number of clusters cannot always be known a priori. Different distance measures lead to different types of clusters (e.g. compact hyperspheres, compact hyper-ellipsoids, lines, shells, etc.). We improve the way clusters select their members by using the single pass k-means by which the data point most unlikely to move at all till the convergence condition is discarded completely. This can be implemented in Visual Basic by storing the datasets temporarily in what is known as lists in Visual Basic compared to a buffer in SQL. The database shall be reorganized so that records are arranged in an efficient way so that the hard disk header doesn't spend much time in searching for the related records of the same type. The proposed system shall have a front end interface for user interaction to show results in a more interactive way while the database will be an MS Access database file. Since the database isn't large compared to the database used in commercial clustering operations an MS Access file will be sufficient enough to show that the above optimizations will improve many factors of the Standard K-means several other optimizations are

related strictly to the RDBMS in SQL servers and may not be feasible at this level.

Density-based: based on connectivity and density functions.

Grid-based: based on a multiple-level granularity structure.

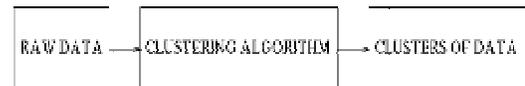
Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other. Most of the times clustering results need to be related back to other tables in a data warehouse to get reports or they can be used as input for other data mining tasks.

Considerable progress has been made in scalable clustering methods:

- *Partitioning:* k-means, k-medoids, CLARANS
- *Hierarchical:* BIRCH, CURE
- *Density-based:* DBSCAN, CLIQUE, OPTICS
- *Grid-based:* STING, WaveCluster.
- *Model-based:* Autoclass, Denclue, Cobweb.

3.1 Standard K-means algorithm:

A clustering algorithm attempts to find natural groups of components (or data) based on some similarity. The clustering algorithm also finds the *centroid* of a group of data sets.



The **centroid** of a cluster is a point whose parameter values are the mean of the parameter values of all the points in the clusters.

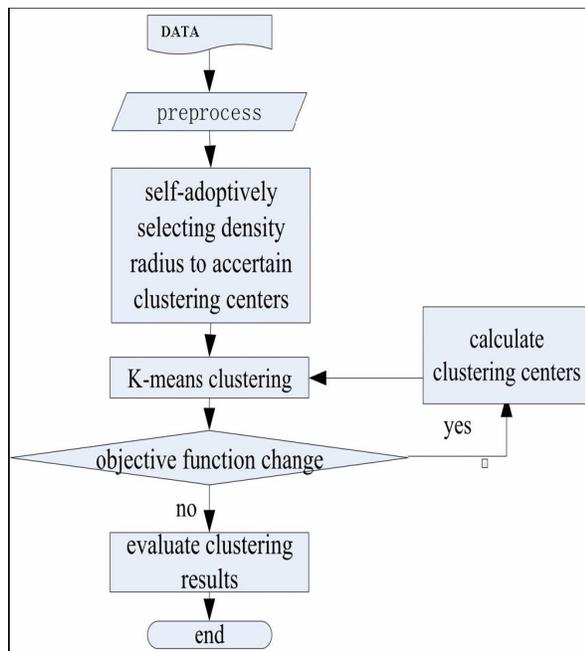
3.2 Advantage and Disadvantage of k-means:

Advantages:

- Relatively efficient: $O(nkt)$, where n is # of objects, k is # of clusters, and t is # of iterations. Normally, $k, t \ll n$.
- Often terminates at a local optimum.

Disadvantages:

- Applicable only when mean is defined.
- Need to specify k , the number of clusters, in advance.
- Unable to handle noisy data and outliers.
- Not suitable to discover clusters with non-convex shapes.

3.3 Optimized K-Means Algorithm

Advantage of Optimized K-Means algorithm:

- Engineering sciences: pattern recognition, artificial intelligence, cybernetics etc. Typical examples to which clustering has been applied include handwritten characters, samples of speech, fingerprints, and pictures.
- Life sciences (biology, botany, zoology, entomology, cytology, microbiology): the objects of analysis are life forms such as plants, animals, and insects.
- Information, policy and decision sciences: the various applications of clustering analysis to documents include votes on political issues, survey of markets, survey of products, survey of sales programs, and R & D.

4.1 Calculating Distance Measurements

K-means, distance computation accounted for 90 percent of time during one iteration. In contrast, for Optimized K-means distance computation went down to about 65 percent of total time. In the first alternative, YD is indexed on i and the k distances are stored on the same disk block. It is remarkable that execution time goes down to roughly one third. For the second alternative, all k distances are stored together in one row (as opposed to same block) and the index column remains i only. This further change reduces the time to almost one tenth, compared to Standard K-means. Determining the nearest cluster requires a scan on YD, reading kn rows, to get the minimum distance per point, and then a join to determine the subscript of the closest cluster.

4.2 Document Clustering

The purpose of clustering is to divide the given objects into a number of groups (clusters), in order that the objects in a particular cluster would be similar among the objects of the other ones. This technique tries to solve how to distribute N object in M clusters according to the minimization of some optimization criterion additive over the clusters. Once the optimization criterion is selected, the clustering problem is to provide an efficient algorithm in order to search the space of the all possible classifications and to find one on which the optimization function is minimized.

4.3 Retrieval of Documents Based on Time

It is to be noted that term weighting strategies have a great impact on the similarity calculation and, thus, a great impact on the final search and routing performance. Any change in the term weighting strategy will essentially change the Ranking function. Ranking selection, which sorts all the individuals by fitness and the probability of individual, will be selected, as the proportional for its rank in this sorted list.

5. Details of Design and Related Algorithms

The Common Metric for Clustering Techniques are the **distance between two points** is taken as a common metric to assess the similarity among the components of a population. The most commonly used distance measure is the **Euclidean metric** which

defines the distance between two points $p = (p_1, p_2, \dots)$ and $q = (q_1, q_2, \dots)$ as :

$$d = \sqrt{\sum_{i=1}^k (p_i - q_i)^2}$$

Input condition: The number of clusters k , as well as the sample collection which contains n data objects;

Output condition: k clusters which satisfy the variance smallest criterion;

5.1 Process flow Algorithm:

- (i) Select k objects randomly from n data objects to take as initial clustering centers;
- (ii) Circulate the following step 1 to 2 until no cluster change any longer;
 - 1) Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
 - 2) Assign each object to the group that has the closest centroid.
 - 3) When all objects have been assigned, recalculate the positions of the K centroids.
 - 4) Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

6. Implementation Results

We conducted experiments on many data sets, and the experimental results show the superiority of the proposed algorithm over the GCUK method in terms of clustering validity and time efficiency. Using the Euclidean distance as the dissimilarity metric yields circular clusters. For some of the test data such

clusters may not be as natural as those provided by people.

For a very large data set, the number of iterations required by K-means may turn clustering into a difficult task, even for the Optimized K-means version. A related issue is that, since clustering is a hard optimization problem (nonconvex and NP-complete, it is customary to make several runs to find a high quality or acceptable solution.

This may be prohibitive for massive data sets, where it may be preferable to get an acceptable solution faster. Example with $d \approx 3$, $k \approx 2$, and $n \approx 5$. the incremental learning nature of the underlying algorithm. For a very large data set, optimized K-means becomes faster by reducing the number of iterations. However, it has one incremental iteration with high overhead. Standard K-means may take slightly more time than Optimized K-means, but it will find better clusters.

7. Conclusion

Experiments evaluate correctness and performance with real and synthetic data sets. We showed that Incremental K-means converges in fewer iterations than Standard and Optimized K-means. A set of experiments benchmarked queries individually. The most critical operation is distance (*: Estimated) Times in seconds, updating clustering results in each iteration.

These two aspects are used as guidelines for optimization. Optimized K-means computes all Euclidean distances for one point in one I/O, exploits sufficient statistics, and stores the clustering model in a single table. Experiments evaluate performance with large data sets focusing on elapsed time per iteration. Standard K-means presents scalability problems with increasing number of clusters or number of points.

Even though we proposed an efficient way to compute Euclidean Distance, there may exist more optimizations. Clustering very high-dimensional data where clusters exist only on projections of the data set is another interesting problem, especially for transaction data.

Large data sets could be clustered in a single scan using SQL combining the ideas proposed here with User-Defined Functions and more efficient indexing. Certain computations may warrant defining SQL primitives inside the DBMS to allow general applicability. Such constructs would include Euclidean distance computation, pivoting a table to have one dimension value per row, and another one to find the nearest cluster subscript given several distances.

8. Future Work

In the future, we will test other distance metrics such as the Mahalanobis distance and the point symmetry distance against a variety of data sets with various shapes. In addition, we are investigating the correlation between the convergence speed and the number of clusters in the data set. We are also studying the similarity/ dissimilarity metric and expect to further improve the unsupervised clustering algorithm.

Davies-Bouldin index is a feature that could be employed to measure the validity of clusters. The development of our algorithm has demonstrated an ability to properly cluster a variety of data sets. The experimental results show that the proposed algorithm provides a more stable clustering performance in terms of number of clusters and clustering results. This result in considerable less computational time required, when compared to other GA-based clustering algorithms. While Genetic algorithms draw a very good picture in terms of better clustering, providing training sets, mutating and performing crossovers are very crucial to the formation of the Hyper- Quad trees which are slowly gaining popularity in a more acclaimed name of Alternative to K-means.

9. References

- [1] C. Aggarwal and P. Yu, "Finding Generalized Projected Clusters in High Dimensional Spaces," Proc. ACM SIGMOD Conf., pp. 70- 81, 2000.
- [2] P. Bradley, U. Fayyad, and C. Reina, "Scaling Clustering Algorithms to Large Databases," Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining, pp. 9-15, 1998.
- [3] Fernández V.1,2; García Martínez R. 1,2; González R. 1; Rodríguez L., "Algorithms Applied to Clustering"
- [4] Hwei-Jen Lin*, Fu-Wen Yang and Yang-Ta Kao, "An Efficient GA-based Clustering Technique," Department of Computer Science and Information Engineering, Tamkang University, Tamsui, Taiwan 251, R.O.C.
- [5] J.Clear, D. Dunn, B. Harvey, M.L. Heytens, and P. Lohman, "Nonstop SQL/MX Primitives for Knowledge Discovery," Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining, pp. 425-429, 1999.
- [6] PDF : College of Computer and Information Science, Northern University, USA <http://www.ccs.neu.edu/home/rjw/csg220/lectures/k-means.pdf>
- [7] F.Fanstrom, J.Lewis, and C.Elkan, "Scalability for Clustering Algorithms Revisited," SIGKDD Explorations, vol.2, no.1, pp.51-57, June 2000.
- [8] G.Hammerly and C.Elkan, "Alternatives to K-Means Clustering that Find Better Solutions," Proc.ACM conf. Information and knowledge Management, pp.600-607.
- [9] K-Means application to find a set with high nutrient content for soldiers http://www.cs.gmu.edu/cne/modules/dau/stat/clustgals/clust5_bdy.html
- [10] C.Ordonez and P..Cereghini, "SQLEM : Fast Clustering in SQL Using the EM Algorithm," Proc.ACM SIGMOD Conf., pp.559-570,2000.
- [11] C.Ordonez and E.Omiecinski, "FREM : Fast and Robust EM Clustering for Large Data sets," Proc.ACM Conf. Information and Knowledge Management, pp.590-599, 2002.
- [13] C.Ordonez and E.Omiecinski, "Efficient Disk-Based K-Means Clustering for Relational Databases," IEEE Trans. Knowledge and Data Eng., vol.16, no.8,pp. 909-921, Aug.2004
- [15] T.Zhang, R.Ramakrishnan, and m.Livny, "BIRCH: An efficient Data Clustering Method for Very Large Databases," Proc. ACM SIGMOD Conf., pp.103-114, 1996.

Opportunistic Consensus Clustering

Arko Banerjee, Arun K Pujari
The LNM IIT, Jaipur, Rajasthan, India

Abstract - Consensus clustering (also referred to as clustering ensembles) has emerged as a method of improving quality and robustness in clustering by optimally combining the results of different clusterings generated in different ways. In this paper, we propose a new way of arriving at a consensus clustering using a cumulative voting strategy. In our algorithm we use a greedy strategy to select the clusterings in an iterative consensus generation technique that ensures the quality of clustering to be monotonically non-decreasing. In strict sense, the process is not a consensus of 'all' available clusterings and hence we term it as opportunistic consensus. We show empirically that the consensus clustering obtained by our algorithm gives better clustering accuracy than those of all major existing methods for many datasets.

Keywords: Data clustering, cluster analysis, consensus clustering, clustering aggregation, clustering ensemble, voting.

1 Introduction

The need for consensus clustering arises due to the fact that none of existing clustering techniques can yield satisfactory partition for all instances of input data. In such situations, consensus clustering (also referred to as clustering ensembles) attempts to combine the results of different clusterings obtained in different ways in order to get a better clustering. In last few years, several approaches have been proposed and these can be broadly categorized into hypergraph based, information theory based, mixture model (EM) based, voting based and co-association based methods.

In this paper, we propose a new way of arriving at a consensus clustering from a set of clusterings with fixed number of clusters. Our approach is a sort of voting aggregation and is novel in many ways. We view the outcome of each clustering run as the opinion of one expert expressing his opinion in terms of a ranking function of an element's chance of being a member of a cluster. The experts are also graded based on some criterion function. The consensus clustering, in this sense, means arriving at a consensus of experts. When there is unanimity (all experts have total agreement) then the problem becomes trivial. Normally experts differ among themselves and we gain from such

diverse opinion. Sometime the experts' opinions are too widely diverse to arrive at a meaningful consensus. Since the objective is to get a combined opinion that is qualitatively better than the individual opinion, it may be worthwhile ignoring some experts' opinions that are not positively contributing to a meaningful consensus. Our algorithm adopts this principle and unlike the existing voting-based consensus clustering techniques, we selectively and iteratively choose one clustering at a time while ensuring that the quality of consensus clustering is monotonically nondecreasing. As mentioned earlier we use quality of clustering as the weights assigned to the experts. But our method is general enough to handle any criterion function. The usual objection in the voting method of relabeling the clusters is handled by using Hungarian method in our formulation. We experimented with some benchmark data for which the clustering labels are known. We compare our results with other existing consensus algorithms on these datasets and it reveals that our algorithm gives better clustering results for most of the datasets.

The rest of the paper is organized as follows. In section 2, we discuss the problem formulation of opportunistic consensus. Section 3 deals with the proposed technique of opportunistic consensus. We report the experimental results of our method in section 4. Section 5 discusses the complexity analysis of the proposed algorithms and Section 6 draws the conclusion.

2 Consensus Clustering

Formally the problem of combining multiple clusterings can be described as follows: Let S be the set of data points. $S = \{s_1, s_2, \dots, s_n\}$. We are given a set of T partitions, $P = \{P_1, P_2, \dots, P_T\}$ of the data points in S . A partition P on S is defined as $P = \{C_1, C_2, \dots, C_k\}$ such that $C_i \subseteq S$, $C_i \cap C_j = \emptyset$ and $\cup C_i = S$. Our goal is to find a final clustering $P^* = \{C_1^*, C_2^*, \dots, C_k^*\}$ that optimizes a consensus function. A consensus function maps a given set of partitions $P = \{P_1, P_2, \dots, P_T\}$ to a final partition P^* . The P^* is a sort of median of P_1, P_2, \dots and P_T .

The problem of opportunistic consensus clustering is formulated in a generic sense as follows. Suppose that we are given a set of T partitions, $\mathcal{P} = \{P_1, P_2, \dots, P_T\}$ of the data points in S . The goal is to find a clustering $P^* = \{C_1^*, C_2^*, \dots, C_k^*\}$ that optimizes a criterion function over the power set of \mathcal{P} , $2^{\mathcal{P}}$. This consensus clustering problem is more general

than the earlier formulation and leads to a new area of investigation. There has been some attempts in this direction where the problem is referred to as *ensemble selection*. Recently, Fern et al [1] proposed three ensemble selection approaches based on external quality and diversity of partitions. It is shown empirically that among the three methods, *Cluster and Select (CAS)* method achieves the best overall performance. CAS organizes different solutions into groups such that similar solutions are grouped together. The solution with the highest quality is selected from each group to form the ensemble. Finally consensus on the ensemble is done using some known consensus algorithm. But such method suffers from serious disadvantages. The performance of CAS is very sensitive to the similarity measure and it is hard to find an optimal similarity measure for a specific dataset. The average performance of CAS is not impressive. Moreover, the quality of consensus clustering obtained from CAS is dependent on the consensus algorithm. We observed, from our experiments, that some algorithms (which are different from graph based technique) do not yield satisfying results for CAS.

3 Opportunistic Consensus Clustering

In this section we introduce a rank matrix for a partition and use this rank matrix to define a cumulative voting based consensus of a set of partitions. Since it is necessary to relabel the clusters of a partition with respect to a given reference partition, we formulate the relabeling problem when the partitions are represented as rank matrices and show that the relabeling problem can be efficiently solved by Hungarian method. We elaborate our earlier formulation of generic opportunistic consensus clustering problem and give a more specific formulation.

3.1 Ranking

Given a partition $P = \{C_1, C_2, \dots, C_k\}$ on $S = \{s_1, s_2, \dots, s_n\}$. From this 'hard' partition, we generate a soft partition such that μ_{ij} denotes the weight of the data point s_i belonging to the cluster C_j . We assume the weights μ_{ij} are normalized such that $\sum_j \mu_{ij} = 1$.

3.2 rank μ

For a partition $P = \{C_1, C_2, \dots, C_k\}$, we define μ_{ij} for each s_i , $1 \leq i \leq n$ and for each C_j $1 \leq j \leq k$ in the following way.

$$\mu_{ij} = \frac{\bar{\mu}_{ij}}{\sum_{j=1}^k \bar{\mu}_{ij}}, \text{ where } \bar{\mu}_{ij} = \min_{s_h \in C_j} d(s_i, s_h)$$

$d(.,.)$ denoted the Euclidean distance.

3.3 rank(s_i, C_j)

The rank of C_j for the s_i , $rank(s_i, C_j)$, is the position of C_j when μ_{ij} $1 \leq j \leq k$, are arranged in decreasing order. For a partition P , we construct a $n \times k$ rank matrix U with a row for each s_i $1 \leq i \leq n$ and a column for each C_j $1 \leq j \leq k$ and each entry u_{ij} is $rank(s_i, C_j)$. For a given partition P , we can compute the corresponding rank matrix U and similarly, for a given rank matrix U , we can generate the corresponding partition (Algorithm 1).

Algorithm 1. To find a partition from a rank matrix

```

INPUT: U
OUTPUT: P = {C1, C2, ..., Ck}
INITIALIZE Cj = ∅, ∀ i
  do for every si ∈ S
    j = argmaxh Uih
    Cj ← Cj ∪ {si}
  enddo

```

Let $P_1 = \{C^1_1, C^1_2, \dots, C^1_k\}$ and $P_2 = \{C^2_1, C^2_2, \dots, C^2_k\}$ be two partitions on S . We assume that the clusters in both partitions are labeled so that C^1_i and C^2_i are corresponding partitions. This correspondence problem can be solved easily by relabeling the clusters of P_2 with respect to P_1 , by applying Hungarian matrix method. Assuming that the partitions are properly labeled, we define below the process of cumulative voting based consensus. The consensus is obtained through the rank matrix and we first aggregate the rank matrix and use Algorithm 1 to get the consensus partition from the aggregated rank matrix.

3.4 Cumulative Voting

Let the rank matrices corresponding to P_1 and P_2 be U_1 and U_2 , respectively. The rank matrix as a result of cumulative voting of two partitions P_1 and P_2 is $U_1 + U_2$.

3.5 Consensus of P_1 and P_2

The cumulative voting based consensus of P_1 and P_2 , denoted as $P_1 \oplus P_2$, is defined as the partition obtained from the rank matrix $U_1 + U_2$ using Algorithm 1.

3.6 Ranked based Voting Consensus (RVC)

For a set of partitions $Q = \{P_1, P_2, \dots, P_w\}$, the consensus of all partitions in Q is $\oplus Q = P_1 \oplus P_2 \oplus \dots \oplus P_w$. In the rest of the paper we call this consensus method as ranked based voting consensus (RVC). To establish the merits of RVC over other existing consensus algorithms, we have performed extensive experimental comparisons on different datasets, which is presented in section 4 under the title experimental results.

As an illustration of the working of RVC, we consider Cassini dataset [2] and compare it with an algorithm IPVC which is designed by Nguyen and Caruana [6]. They find that an iterative EM-like method is remarkably effective for consensus clustering problem. They introduce Iterative Voting Consensus (IVC) algorithm and its two variations, Iterative Probabilistic Voting Consensus (IPVC) and Iterative Pairwise Consensus (IPC), for finding clustering consensus. Out of the three algorithms IVC, IPVC and IPC, IPVC has been reported to be the best performing algorithm. The mechanism of IPVC is as follows: For each data point s_i in S a corresponding T -dimensional feature vector is constructed, where the i^{th} feature is simply the cluster label from the clustering P_i . In each iteration of the algorithm, each data point is reassigned to different clusters based on a defined distance function between the considered data point and the previous established clusters in the target consensus clustering. The main operation in each iteration is to find a closest cluster for each data point via a defined distance measure between them. The Cassini dataset contains 1000 two dimensional points which are grouped into three clusters. We choose cassini dataset, because the three clusters are well separated but the distribution of data offers challenging task to any clustering algorithm. An ensemble of 100 partitions is generated with random initialization of k-means over the dataset. The Adjusted Rand Index (ARI) [7] for consensus due to IPVC and RVC are found to be 0.512 and 0.7836, respectively. Figures 1(a), 1(b) and 1(c) depict the original clustering, consensus clustering due to IPVC and consensus clustering due to RVC, respectively.

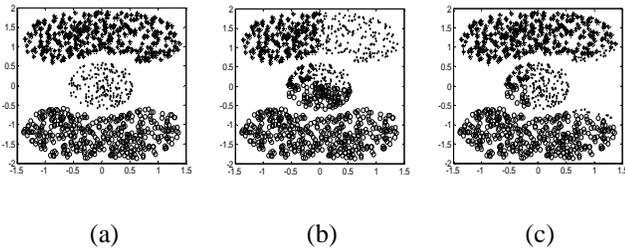


Figure 1. (a) Cassini dataset in two dimensions. Three different shapes represent three ground truth clusters. (b) Consensus clustering due to IPVC. (c) Consensus clustering due to RVC.

Figure 1 reveals that RVC performs better than IPVC on Cassini dataset.

3.7 Relabelling

We assume that each partition of the ensemble has fixed number of clusters i.e., k and the consensus partition also has k clusters. The problem of matching the cluster labels is necessary to get the most consistent labeling of clusters with reference to a given clustering. The cluster label matching problem is formulated as follows.

Let U_1 and U_2 be the rank matrices corresponding to partitions P_1 and P_2 , respectively. The problem is to determine the best cluster label permutation (permutation of the columns of U_2) π that minimizes

$$\sum_i \left(\sum_j |u_{ij}^1 - u_{i\pi(j)}^2| \right)$$

We show that the above entity can be reduced to a $k \times k$ assignment matrix V where each element v_{pq} is defined as follows.

$$v_{pq} = |u_{.p}^1 - u_{.q}^2|$$

We denote u_{pq}^1 (u_{pq}^2) for p^{th} row and q^{th} column of U_1 (U_2 , respectively).

$$\begin{aligned} & \min_{\pi} \sum_i \sum_j |u_{ij}^1 - u_{i\pi(j)}^2| \\ &= \min_{\pi} \sum_j \sum_i |u_{ij}^1 - u_{i\pi(j)}^2| \\ &= \min_{\pi} \sum_j |u_{.j}^1 - u_{.\pi(j)}^2| \\ &= \min_{\pi} \sum_j v_{j\pi(j)} \end{aligned}$$

Hence the optimal permutation can be obtained by applying the Hungarian method on the assignment matrix V .

3.8 Criterion function for Consensus

In order to measure the quality of clustering, we use internal measures and in the present study the cluster quality is defined as the ratio of intercluster distances to intracluster distances. Given a partition $P = \{C_1, C_2, \dots, C_k\}$ on S , let c_i be the cluster mean of C_i , $\forall i$. The quality of P , denoted by $q(P)$ is defined as follows.

$$q(P) = \frac{\sum_{j_1, j_2} d(c_{j_1}, c_{j_2})}{\frac{k-1}{2} \sum_{C_j} \sum_{s_i \in C_j} d(s_i, c_j)}$$

There can be many ways of defining the criterion function for opportunistic clustering. We formulate the problem with quality as the criterion function.

To show that clustering quality does not satisfy monotonicity property we consider an ensemble of 100 partitions of Iris dataset [4] which is generated with random initializations of k-means algorithm. Out of the 100 partitions only 3 partitions are found to be distinct. We take the entire ensemble and form the consensus using RVC. The quality of the consensus clustering is found to be 0.5021; whereas the quality of the consensus clustering by considering the last two partitions is found to be 0.3826. This leads to the conclusion

that clustering quality does not improve if any subset of the ensemble is taken; hence subset of the ensemble should be chosen judiciously.

3.9 Opportunistic Consensus (quality as criterion)

Given set of T partitions, $\mathcal{P} = \{P_1, P_2, \dots, P_T\}$ of S , the opportunistic consensus P^* of \mathcal{P} is defined as follows.

$$P^* = \arg \max_Q q(\oplus Q)$$

The maximum is taken over all subsets Q of \mathcal{P} . Thus the opportunistic consensus function is a criterion function that maximizes the quality of cumulative ranking consensus over the power set $2^{\mathcal{P}}$.

3.10 Opportunistic Consensus Clustering algorithm

It is evident that finding the opportunistic consensus clustering for a set of partitions by maximizing the consensus over all subsets of partitions is not going to be easy. We propose here a greedy heuristic to determine the consensus partition.

Given a set of T partitions, $\mathcal{P} = \{P_1, P_2, \dots, P_T\}$ of S , without loss of generality, let $q(P_1) \geq q(P_2) \geq q(P_3) \geq \dots \geq q(P_T)$. Algorithm 2 formally states the method of finding opportunistic consensus clustering.

Algorithm 2. Opportunistic Consensus

```

Initialization:  $P_{i-1}^* \leftarrow P_1$ 
do for all  $i, 2 \leq i \leq T$ 
  compute the consensus  $P_{i-1}^* \oplus P_i$ 
  if  $q(P_{i-1}^* \oplus P_i) \geq q(P_{i-1}^*)$ 
  then  $P_i^* \leftarrow P_{i-1}^* \oplus P_i$ 
  else  $P_i^* \leftarrow P_{i-1}^*$ 
end do

```

By our method, the starting point is the clustering with the best quality and we gradually add additional clusterings ensuring the quality to be non-decreasing. This does not rule out another set of partitions that yield yet another opportunistic consensus.

4 Experimental Results

4.1 Ensemble generation

The proposed concept of opportunistic consensus is not dependent on the method of ensemble generation, to build our ensemble, we used the k-means algorithm as our base algorithm. k-means is chosen because it is one of the most widely used clustering algorithms and has been used in many previous cluster ensemble studies. Different clustering

solutions are obtained by applying k-means to the same data with different random initializations.

4.2 Evaluation criteria

Evaluating the quality of clustering is nontrivial and ill-posed task [3]. Hence it is difficult to determine an ideal way of evaluating the quality of clusterings. In this paper we adopt three approaches to evaluate consensus clusterings: external consensus criteria, relative consensus criteria and internal consensus criteria.

In external consensus criteria it is measured how well the target clustering performs in comparison to the external true label of the data points. In this paper we use Adjusted Rand Index (ARI) as the similarity measure between two partitions. We choose ARI because it is considered as a good measure of similarity, it is used in many previous clustering studies and it is easy to compute.

In relative consensus criteria it is measured how the target consensus clustering is in agreement with all clusterings in the ensemble. In our experiment, we denote the measure as Median(P,E) and define it as the average distance between the consensus partition P and the ensemble E . Mathematically,

$$\text{Median}(P,E) = \frac{1}{|E|} \sum_{P_i \in E} \text{ARI}(P, P_i)$$

More the value of Median(P,E) better is the consensus as mean.

Internal consensus criteria is based on calculating properties of the resulting clusters, such as compactness, separation and roundness. Here we take the measure as defined in section 3.8.

4.3 Data sets

Our experiments use the datasets CHART, SEGMENTATION, WINE, GLASS, IRIS, YEAST, ECOLI which are benchmark data sets from the UCI machine learning data repository [4]. It should be noted that all the datasets are labeled and contain supervised class information.

Table 1. Basic Information of the Data sets

Datasets	# Instances	# features	# classes
Chart	600	60	6
Segmentation	2100	19	7
Ecoli	336	7	8
Yeast	1484	9	10
Iris	150	4	3
Glass	214	10	6
Wine	178	13	3

The reason that we have chosen those datasets because some are very well known datasets in cluster analysis studies and

some are not so well known but due to their high dimensionality they present significant challenge to standard clustering algorithms.

4.4 Experimental settings

In our experiment we generate three different ensembles of size 100 for all datasets and we compare the average performances of 6 existing consensus clustering methods CSPA [5], HGPA [5], MCLA [5], IVC [6], IPVC [6] and IPC [6] with our methods. Strehl and Ghosh [5] define the cluster ensemble problem as an optimization problem and maximize the normalized mutual information of the consensus clustering. They introduce three different algorithms to obtain good consensus clustering, namely Cluster-based Similarity Partitioning (CSPA), HyperGraph Partitioning (HGPA), and Meta-Clustering (MCLA) algorithms.

4.5 Results

In the following tables EC, RC and IC represents external consensus criteria, relative consensus criteria and internal consensus criteria, respectively. Here we mention that, we do not evaluate RC for OPC, as they do not operate on the whole ensemble.

Table 2a-2g shows results of consensus algorithms applied on the whole ensemble, whereas Table3 represents the EC performance of different consensus algorithms on the ensemble of distinct partitions.

Table 2a. Results for Iris dataset.

	EC	IC	RC
CSPA	0.714	0.3508	0.7172
HGPA	0.4379	0.0444	0.3236
MCLA	0.7302	0.5021	0.8719
IVC	0.7302	0.5021	0.8719
IPVC	0.7302	0.5021	0.8719
IPC	0.7302	0.5021	0.8719
RVC	0.7302	0.5021	0.8719
OPC	0.7302	0.5021	-

Table 2b. Results for Glass dataset.

	EC	IC	RC
CSPA	0.552	2.2611	0.8959
HGPA	0.2113	0.1335	0.39
MCLA	0.5523	2.2681	0.8963
IVC	0.5459	2.2626	0.8891
IPVC	0.5466	2.2214	0.8963
IPC	0.5355	2.2524	0.8934
RVC	0.5523	2.2681	0.8963
OPC	0.5619	2.2716	-

Table 2c. Results for Wine dataset.

	EC	IC	RC
CSPA	0.3711	0.3554	0.8587
HGPA	0.1599	0.1793	0.276
MCLA	0.3711	0.3554	0.8587
IVC	0.3711	0.3554	0.8587
IPVC	0.3711	0.3554	0.8587

IPC	0.3711	0.3554	0.8587
RVC	0.3711	0.3554	0.8587
OPC	0.3711	0.3554	-

Table 2d. Results for Chart dataset.

	EC	IC	RC
CSPA	0.5393	0.0695	0.6069
HGPA	0.3978	0.0364	0.4578
MCLA	0.582	0.0787	0.6624
IVC	0.5016	0.067	0.5649
IPVC	0.5263	0.0639	0.6519
IPC	0.5196	0.0617	0.6484
RVC	0.5222	0.0903	0.6697
OPC	0.5896	0.0961	-

Table 2e. Results for Ecoli dataset.

	EC	IC	RC
CSPA	0.2957	0.0908	0.5316
HGPA	0.3328	0.1081	0.5161
MCLA	0.3476	0.1717	0.6731
IVC	0.4475	0.1122	0.4868
IPVC	0.3981	0.1757	0.6808
IPC	0.345	0.171	0.6834
RVC	0.4651	0.2113	0.6314
OPC	0.5058	0.0757	-

Table 2f. Results for Segmentation dataset.

	EC	IC	RC
CSPA	0.3622	0.0757	0.548
HGPA	0.3371	0.0365	0.625
MCLA	0.3828	0.1907	0.5943
IVC	0.3495	3.2309	0.4663
IPVC	0.3877	4.612	0.6206
IPC	0.3763	4.8609	0.6709
RVC	0.4318	4.9166	0.6909
OPC	0.4318	4.9166	-

Table 2g. Results for Yeast dataset.

	EC	IC	RC
CSPA	0.097	0.0136	0.5429
HGPA	0.0815	0.0063	0.4325
MCLA	0.1193	0.0263	0.6407
IVC	0.1305	0.0182	0.5555
IPVC	0.1349	0.0222	0.5595
IPC	0.1387	0.0183	0.7375
RVC	0.1729	0.0586	0.7388
OPC	0.1649	0.0498	-

Table 3 . Results for distinct partitions of datasets

	MCLA	CSPA	IVC	IPVC	IPC	RVC	OPC
Iris	.7302	.688	.54	.7302	.7302	.56	.7302
Glass	.545	.545	.554	.545	.545	.545	.554
Wine	.3768	.4380	.140	.3389	.3711	.375	.3781
Chart	.5654	.564	.561	.5622	.54	.575	.5881
Ecoli	.3456	.3008	.442	.359	.363	.4402	.4631
Segm	.3849	.3657	.313	.2536	.355	.3626	.4316
Yeast	.1193	.0997	.149	.1405	.141	.1718	.1634

From our experiment we observe that in most of the datasets at least one of our proposed algorithms achieves the best result.

4.6 Experimental comparison of CAS and OPC

We apply CAS on our generated ensembles and for consensus we use CSPA and RVC separately. In the following table CAS+CSPA/RVC refers to consensus due to CSPA/RVC on a subset selected by CAS.

Table 4. Comparison of CAS and OPC

	CAS+CSPA	CAS+RVC	OPC
iris	0.7148	0.7148	0.7302
glass	0.5494	0.5229	0.5619
wine	0.4437	0.3254	0.3711
chart	0.5455	0.4863	0.5896
ecoli	0.3929	0.3081	0.5058
Segmentation	0.3525	0.3367	0.4318
yeast	0.1009	0.1308	0.1649

The experiment shows that OPC outperforms CAS+CSPA for most of the datasets, whereas performance of CAS+RVC is not at all significant. It shows that quality of consensus due to CAS is not impressive with all consensus algorithms.

4.7 Visualization using manifold learning

To provide an intuitive feel of OPC algorithm, we use a manifold learning based visualization method. For a pair P_i, P_j of partitions, $1-ARI(P_i, P_j)$ are taken as pairwise distance matrix. Locally Linear Embedding (LLE) on this matrix is computed for the target dimension 5. The LLE is a nonlinear dimensionality reduction method that can be employed for any type of data if the pairwise distances of the objects are known. It preserves the local neighbourhood properties and projects the high dimensional data into lower dimensional space by preserving the coefficients of affine combination of the K-nearest neighbors of each object. In our experiment, we take $K=10$ and show the embedding of the partitions in ensemble in 2-dimensional space (selected 3rd and 4th dimension). The partitions selected by different consensus algorithms are highlighted with different symbols. In the Figures, black dots (\bullet) represent distinct partitions. Square (\square) represent partitions selected by OPC. Diamond (\diamond), upper triangle (Δ), triangle left (\triangleleft), lower triangle (∇), triangle right (\triangleright) and cross (\times) represent consensus partitions due to MCLA, CSPA, HGPA, IVC, IPVC and IPC, respectively. Circle (\circ) and star (\star) signs represent consensus partitions due to RVC and OPC, respectively. The plus ($+$) represents the true partition. The following figures show some interesting results

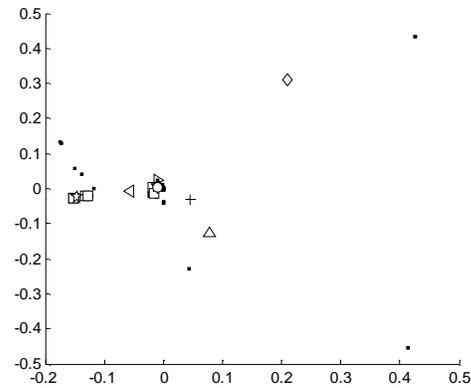


Fig. 2. Embedding of the partitions in ensemble of Chart data in 2-dimension.

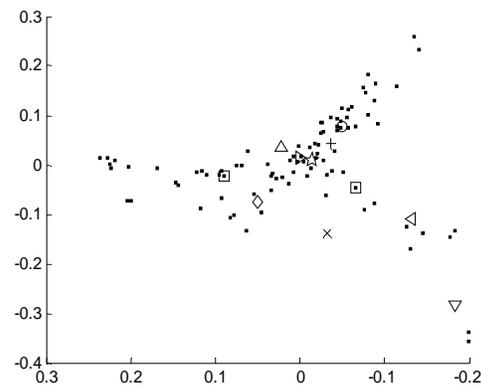


Fig. 3. Embedding of the partitions in ensemble of Ecoli data in 2-dimension.

The maximum, minimum and mean ARI values of distinct partitions of Chart dataset in the previous generated ensemble with respect to the ground truth partition are 0.5607, 0.4937 and 0.5269, respectively. Figure 2(a) shows that OPC chooses 5 partitions out of 35 distinct partitions to form the final consensus partition with ARI 0.5881. The maximum, minimum and mean ARI values of distinct partitions of Ecoli dataset in the previous generated ensemble with respect to the ground truth partition are 0.4210, 0.3103 and 0.3534, respectively. Figure 2(b) shows that OPC chooses 2 partitions out of 60 distinct partitions to form the final consensus partition with ARI 0.4631. OPC shows such interesting results for most of the other datasets too.

5 Complexity Analysis

Space complexity of RVC as well as OPC is $O(nk + k^2)$, where n is the number of data points and k is the number of clusters required. $O(nk)$ is due to storage of rank matrix and $O(k^2)$ is due to labeling. Time complexity of RVC is $O(n^3 + Tnk)$, where T is the size of the ensemble. Here, evaluating rank takes $O(n^2)$ time, cumulative voting takes $O(Tnk)$ time, consensus takes $O(Tnk)$ time and labeling takes $O(n^3)$ time. In OPC quality computation takes $O(n^2 + k^2)$ time. Hence Time complexity of OPC is $O(n^3 + Tnk + k^2)$.

6 Conclusion

In this paper, we demonstrated that it is necessary to select a subset of clusterings that can yield better resulting clusters than combining all the available clusterings. Furthermore, we proposed a new cumulative voting ensemble approach based upon the ranking of object-cluster associations. In our approach we also suggested that clustering quality could be a good measure for selecting significant partitions that may contribute to a meaningful consensus. Our algorithm overcomes the computationally infeasible simultaneous combination of all partitions by converting the relabeling problem into cost assignment problem, and hence solving it by Hungarian method in polynomial time. Evaluations on different datasets have shown this approach to be effective in improving clustering accuracy, particularly when true number of clusters of ground truth partition is considered. We have shown that the proposed method performs well compared to other well known consensus methods, although more experiments are needed to assess the real value of the method. It is expected that our technique can lead to even better consensus partition, if more powerful clustering methods than k -means is used to generate initial partitions. In our future work, we will be implementing our algorithms on larger and challenging datasets, like, 20-newsgroups. We also need to measure the performance of our approach as the number of partitions increases. Further we are investigating on the generalized version of opportunistic consensus.

7 References

- [1] Fern, X. Z. and Lin, W., Cluster Ensemble Selection. *Proceedings of SIAM Data Mining*, Atlanta, Georgia, USA, pp. 787-797, 2008.
- [2] Hornik, K., A CLUE for CLUster Ensembles, *In Journal of Statistical Software*, Vol. 14, No. 12, pp. 1-25, September 2005.
- [3] Fred, A. L. N. and Jain, A. K., Robust data clustering. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Madison, Wisconsin, USA, II: 128-133, 2003.

[4] UCI Machine Learning Repository (website: <http://archive.ics.uci.edu/ml/>).

[5] Strehl, A. and Gosh, J., Cluster ensembles – a knowledge reuse framework for combining partitionings. *Proceedings of AAAI*. Edmonton, Canada, pp. 93–98, 2002.

[6] Nguyen, N. and Caruana, R., Consensus Clusterings. *Proceedings of the 7th IEEE International Conference on Data Mining*. Omaha, NE(USA), pp. 607-612, 2007.

[7] Steinley, D., Properties of the Hubert-Arabic adjusted Rand index. *In Psychol Methods*, 9:386-396, 2004.

Learning a Similarity Measure to Objectively Evaluate Image Segmentation Quality

Anjum Gupta. Spawar Systems Center, San Diego. anjum.gupta@navy.mil

Abstract—Image segmentation is an important and active area of research in image processing and computer vision. It is often difficult to objectively evaluate the quality of the segmentation, especially in the case of natural images. In fact, even two different human subjects will generally segment a given natural image differently. We propose an objective and robust measure to evaluate the quality of a computer segmented image by comparing the segmented image to other segmentation of the same image that may be done by human subjects. We also show an objective evaluation of our proposed method by demonstrating that our method always gives a very high similarity index when comparing two different segmentations of the same image, and a low similarity when the segmentation do not belong to the same image. Our method can be used to automatically evaluate the accuracy of image segmentation algorithms by using human segmented images as the ground truth. We compare our method with other methods that have been proposed in the past and show that our method gives a more accurate evaluation of the segmentations.

I. INTRODUCTION

Many algorithms have been proposed for automatically evaluating image segmentation algorithms [1]. It is relatively easy to get an idea of the accuracy of the segmentation by just observing the segmentation and the image side by side. However, it is hard to objectively quantify the overall accuracy of the algorithm. This is due to the fact that, in case of most natural images, there are multiple ways of “correctly” segmenting the image. In fact, when the same image is given to different human subject, each human subject tends to segment the image differently. Figure 1 shows a few different cases of the same image being segmented differently by different human subjects. This makes it difficult to compare two different algorithms or compare the performance of an algorithm on a large set of images. So we need a standard way to measure the accuracy or the quality of the image segmentation algorithm, so that a fair assessment can be made of a particular algorithm or a particular set of parameters used for segmenting images. There have been many different measures proposed for such comparisons in the past [1]. We will briefly cover these measures in the next section. Our measure uses a set of general measures that are used to compare two different clusterings of the data. One of the key aspects of our approach that sets it apart from the methods that have been proposed in the past is the fact that we use machine learning algorithms to adapt and learn the parameters of the final expression that is used to compare two or more different segmentations of an image.

Our approach works in the following way: Given a set

of natural images and a set of corresponding segmentations done by humans, our algorithm learns to distinguish between two different segmentations of the same image versus two segmentations belonging to different images. Two different segmentations of the same image can be a result of the image being segmented by two different human subjects or a result of one human subject producing two reasonable or acceptable (but different) segmentation of the same image. We can imagine having a data set of images and two sets of segmentations done by human subjects. Once we have developed a function that can recognize two different segmentations as belonging to the same image, we can replace one set of segmentations by an output of the image segmentation algorithm that we wish to evaluate. If the output of the algorithm is recognized as an acceptable segmentation, or another human segmentation, of a given image, then we can measure to what degree the algorithm is performing at a human level.

II. RELATED WORK

In our paper, we use a data set of natural images that is maintained by the University of California, Berkeley [8].

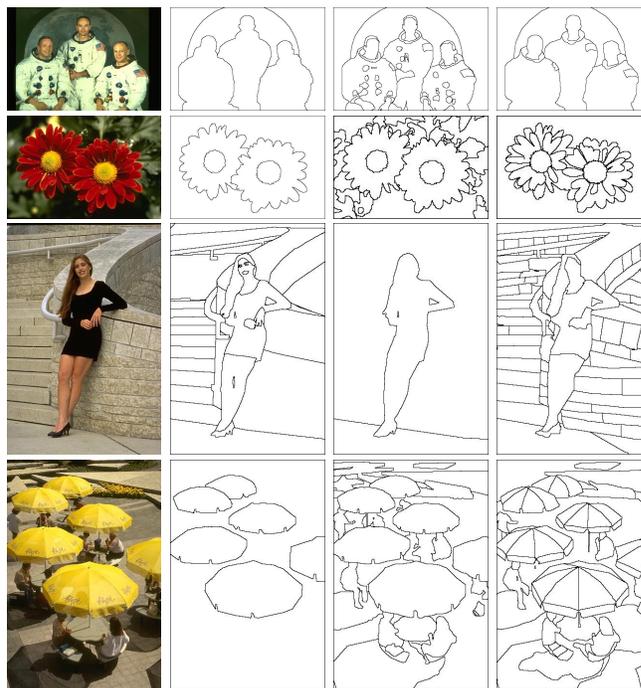


Fig. 1. Figure showing three different segmentations done of natural images by different human subjects.

The data set contains 200 natural images where each image has been segmented by different human subject. The main purpose of the data set is to serve as a standard and a ground truth that can be used by different image segmentation algorithms. The data set contains 200 training and 100 testing images. Each of the training image has been segmented by 3 to 10 different human subjects. Since the data set is to be used to compare different image segmentation algorithms, there is a simple yet effective measure proposed by [8]. The details of their proposed measure are outlined in the next section. In our paper, we use measures that have been proposed to compare two sets of data clusterings. These measures include, *Rand* index, *Fowlkes* index, *Jaccard* index etc. We will visit these measures again in the next section. Strictly speaking, an index is a distance measure whose values are restricted between 0 and 1. Since most of the measures discussed in the paper are indices, we will use the words “index” and “distance measure” interchangeably. The approach of using these measures individually to compare image segmentation has been applied in a paper by Xiaoyi Jiang, Cyril Marti, C. Irniger and H. Bunke [9]. Our paper, in a way works as an extension of the work done by Xiaoyi et. al. We extend their approach by learning an optimal combination of these measures using a classifier such as a hyperplane or nearest-neighbor etc.

III. BACKGROUND

In this section, we will briefly describe the various similarity or distance measures that we use in our classifier function that are used to evaluate image segmentations. A thorough survey of these methods has been done in [2]. Since most of these measures were proposed to measure the similarity between two different clusterings of the data, to apply these measures to the image segmentations, we translate the clusters to be the segments within an image and two different clusterings of the data to be two different segmentations of an image. Following is the common notation used in the formulas of these distance measures: Let the segmentations of the image be S_1 and S_2 . The total number of pixels in the image is n .

N_{00} is the number of pairs of pixels that were in different segments in S_1 and in S_2 .

N_{01} is the number of pairs of pixels that were in different segments in S_1 but are in the same segment in S_2 .

N_{10} is the number of pairs of pixels that were in the same segment in S_1 but are in different segments in S_2 .

N_{11} is the number of pairs of pixels that were in the same segments in both S_1 and in S_2 . N is the total number of pairs, which is equal to, $n(n-1)/2$.

First in our series of measures is *Rand* index [3]. *Rand* index is defined as:

$$R_{similarity} = \frac{N_{00} + N_{11}}{N} \quad (1)$$

Since the above expression gives a similarity measure, we can define the distance index by subtracting the similarity measure from 1. Next, we define *Jaccard* and *Fowlkes*

indices [5], [10] and [4]:

$$Jaccard = 1 - \frac{N_{11}}{N_{11} + N_{01} + N_{10}} \quad (2)$$

$$Fowlkes = 1 - \sqrt{W_1(C_1, C_2)W_2(C_1, C_2)} \quad (3)$$

Where W_1 and W_2 are defined as,

$$W_1 = \frac{N_{11}}{\sum_{i=1}^{k_1} n_i(n_i - 1)/2} \quad (4)$$

$$W_2 = \frac{N_{11}}{\sum_{i=1}^{k_2} n_i(n_i - 1)/2} \quad (5)$$

where k_1 and k_2 are the number of segments in S_1 and S_2 , respectively. Another distance measure, known as the *Dongen* Index works by measuring the overlaps between clusters or segments [6]. It is defined as,

$$Dongen = 2m - \sum_{s_i \in S_1} \max_{s_j \in S_2} (s_i \cap s_j) \quad (6)$$

Where m is the number of pixels in each image. This measure starts with all the pixels in the two images, $2m$, then subtracts all the overlapping regions in the two segmentations.

Next, “Local Consistency Error,” (LCE), is proposed in the original paper describing the Berkeley image data set and its use for image segmentation evaluation. Let $R(S, p)$ be the segment (set of pixels) in Segmentation S that contains pixel p . We define a function E as follows,

$$E(S_1, S_2, p_i) = \frac{R(S_1, p_i) \setminus R(S_2, p_i)}{R(S_1, p_i)} \quad (7)$$

where \setminus denotes the set difference. LCE is defined as,

$$LCE = \frac{1}{m} \sum_{p_i \in Pixels} \min\{E(S_1, S_2, p_i), E(S_2, S_1, p_i)\} \quad (8)$$

Our last distance measure is based on concepts of entropy and mutual information between the two segmentations. The distance measure is proposed by Marina Meila and is called Variational Information (VI) [7]. Entropy, H of a segmentation S can be defined as

$$H(S) = - \sum_{s_i \in S} p(s_i) \log(p(s_i)) \quad (9)$$

Mutual information, between segmentation S_1 and segmentation S_2 is

$$M(S_1, S_2) = \sum_{s_i \in S_1} \sum_{s_j \in S_2} p(s_i, s_j) \log \frac{p(s_i, s_j)}{p(s_i)p(s_j)} \quad (10)$$

Variational information, VI , is then defined as,

$$VI(S_1, S_2) = H(S_1) + H(S_2) - 2MI(S_1, S_2) \quad (11)$$

IV. IMAGE SEGMENTATION EVALUATION

Our task is to distinguish between two segmentations belonging to the same underlying image versus the two segmentation that belong to a different underlying image. If we have a measure or an algorithm that can identify when two segmentations are similar enough to belong to the same image, we can then use this algorithm to automatically evaluate the quality of segmented images that are segmented by some image segmentation algorithm.

We have 200 natural images in our data set. Even though, in the original dataset each image has been hand segmented by 3 to 10 different humans, to be more consistent with the number of human subjects per image, we only use up to 5 different segmentation of each image. So in our experiments, each image has been hand segmented by 3 to 5 subjects. Very few images had less than 5 human segmentations. So for 200 images, we have at least 600 hand segmentations. Out of 600 individual segmentations, we extract 1800 pairs of segmentations that belong the same image and 1800 pairs that belong to different images.

A. Each Measure Individually

We first explore the effectiveness of using each of these distance measures individually. In figure, 2 we see the distribution of the values of each one of the distance measures when the two segmentations of the same image and different images are compared. For example, in Figure 2 part (A) was generated by computing *Rand* index for 1800 pairs of segmentations that belong to the same image and plotting the distribution of the values of the *Rand* index (solid line), then we computed the index values for 1800 pairs of segmentations that belonged to different underlying images, and plotted distributions of those 1800 measures (dotted line). We do this for each one of the 6 measures described in section 3. As expected, each of these measures show a good split between the segmentation of same image versus different images. Each of the measure also show a significant overlap in the values under dotted line and the solid line, showing that each measure also produces significant ambiguity in determining when the two segmentations belonged to the same image and when they belonged to different images. To quantify this ambiguity produced by each one of these measures individually, we try to determine the accuracy in classifying the pairs as belonging to the same image or different image. Out of the total 3600 pairs available, we first

Rand	Fowl	Jac	Donge	VI	LCE
89.2	86.2	87.1	89.8	85.4	84.5

TABLE I

PERCENTAGE ACCURACY IN THE CLASSIFICATION OF SAME IMAGE AND DIFFERENT IMAGES PAIRS WHEN USING EACH MEASURE INDIVIDUALLY. THIS ACCURACY IS IN CLASSIFYING 1200 PAIRS OF SEGMENTATIONS.

make a training set of 2400 pair, evenly split between the two classes, "same image" and "different images." We use rest of the 1200 pairs as our testing set, where we try to determine

whether a given pair of segmentations are coming from the same image or different images. We can see how each one of these measures is capable of measuring the similarity between two segmentations of an image. Following is a table of the result of how each one performs in identifying weather the two segmentations, that have been done by two different human subjects, belong to the same image or different images. These results are the accuracy based on a nearest-neighbor classifier. For example, to compute the accuracy of *Rand* index in classifying the segmentations pair, we compute the *Rand* index for each pair in the testing set, and label it according the labels of the closest k neighbor in the training set. Table I shows the results of the accuracies of the nearest-neighbor classifier.

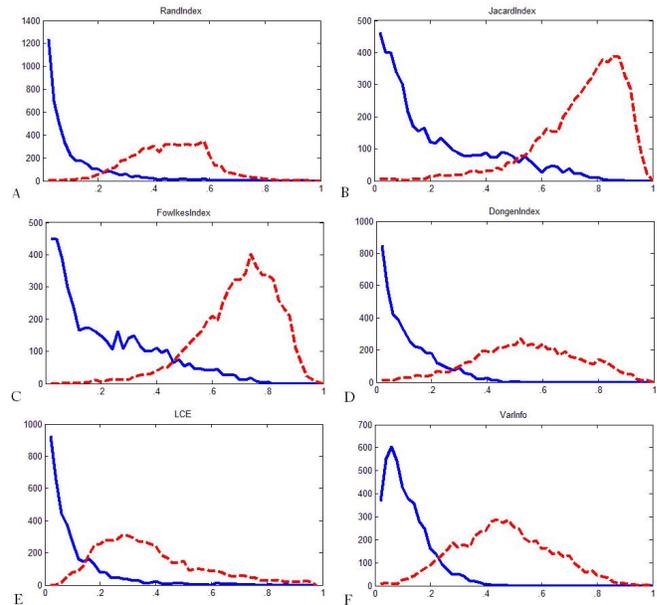


Fig. 2. Distribution of the values generated by each individual distance measure on 3600 pairs. Pairs belonging to the same image (solid line) and pairs belonging to different images (dotted line.)

B. Learning a Classifier

We now focus on combining these distance measures to get much better classification accuracy. Instead of using each distance measure by itself, we want to treat the 6 measures as a 6 dimensional vector. So each pair produces this 6 dimensional vector and we want to classify the points into two classes in the 6 dimensional space. We can use various learning algorithms, where the input feature vector is the value of each of our indices and the output is a classification, of same or different image. For the sake of simplicity, we use two of the basic classifiers, that is, nearest neighbor and a linear hyperplane learned by perception algorithm. Our ideas can be extended by using other ensemble of classifiers [14].

1) *Nearest Neighbor Classifier*: Nearest neighbor is used to classify a data point by looking at the label of its k nearest neighbors in the training set [11]. We used Euclidean distance for nearest neighbor. The value of the k can be varied but

the results listed here were obtained by using $k = 3$ and a majority vote was taken to determine the final label. Once again we split the data set of 3600 pairs into a training and a testing set as we did while evaluating the performance of each individual measure by itself. Following are the results of the nearest neighbor classifier for classifying the pairs in two classes of "same image" and "different images." We get much higher accuracy, when we use the nearest neighbors in 6 dimensions versus a single dimension.

Perceptron	97.2%
Nearest-Neighbor	98.6%

TABLE II

THE AVERAGE ACCURACY OF OUR TRAINED CLASSIFIERS. THE RESULTS ARE FOR THE NEAREST-NEIGHBOR IN 6-DIMENSIONAL EUCLIDEAN SPACE AND HYPERPLANE OBTAINED BY THE PERCEPTRON ALGORITHM

2) *Linear Classifier: Perceptron:* In our second classifier, we learn a hyperplane in 6 dimensional space using perceptron algorithm [12], [13]. We chose perceptron for its simplicity and an easy representation of the classifier as linear weights for each of the measures. Even though, the data is not linearly separable we get a very high accuracy of the classification. III shows the corresponding linear weights learned by the perceptron algorithm. These weights do not represent the importance of each measure but just show the collective normal vector for the separating hyperplane. Figure 3 also shows a distribution of the value of the linear combination of the distance measures as proposed by the separating hyperplane computed by the perceptron algorithm. We can see that there the overlap between the two values is significantly less. The accuracy results for the classification of pairs of segmentations have been shown in table II.

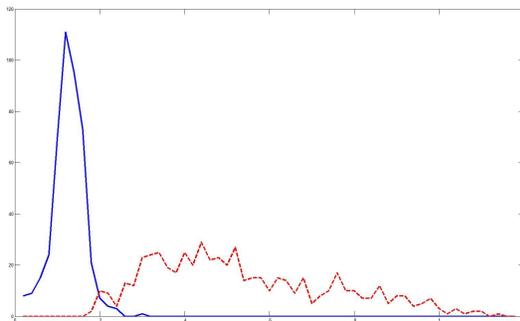


Fig. 3. Distribution of the values generated by the linear combination of the individual distance measures. The linear weights are from the hyperplane obtained by the perceptron algorithm. The overlap between the two classes i.e. pairs belonging to different images (dotted line) and same image (solid line) is much smaller than in the case when each measure was used individually.

V. CONCLUSION

The fact that there is no objective measure for the quality of a segmentation, makes the task of designing and, especially, evaluating an image segmentation algorithm a difficult

Constant	Rand	Fowl	Jac	Donge	VI	LCE
-10	40.0	12.2	-69.8	-13.16	27.7	16.4

TABLE III

WEIGHTS GIVEN TO INDIVIDUAL DISTANCE MEASURES BY THE SEPARATING HYPERPLANE OBTAINED FROM THE PERCEPTRON

one. One of the best ways to think about an objective criteria to evaluate the segmentation quality is to try and answer the question whether the given segmentation is good enough to be taken as a segmentation that could be done by a human subject. In this paper, we propose a classifier that classifies the segmentation of the same image and the different images with approximately 98% over a data set of 1200 pairs. Such a classifier can be used to find out how much of the time a given image segmentation algorithm is operating at a human subject level, based on how often our classifier classifies the image segmented by the algorithm and a human subject as two acceptable segmentations of the same image. Our approach of combining the different similarity measure and learning a classifier over these values increases the overall robustness of the classification as opposed to using just one distance measure. There are various ways this approach can be further refined and extended. We can try and use different and more complex classifier such as boosting, neural networks or support vector machines to get even a better classification. We have also done some preliminary work to use this approach to evaluate the performance of different image segmentation algorithms on the Berkeley image data set [8].

VI. ACKNOWLEDGEMENTS

I would like to thank my intern, Daniel Garcia, for his help with some programming and general organization of this project. I would like to thank Sparwar Systems Center at San Diego, for providing the funding for this project.

REFERENCES

- [1] C. Pantofaru, M. Hebert, *A Comparison of Image Segmentation Algorithms* Tech. CMU-RI-TR-05-40, Sep. 2005. Carnegie Mellon University
- [2] M. Meila, *Comparing Clusterings*, Technical report 419, Univ. of Washington, 2002. www.stat.washington.edu/reports.
- [3] W.M. Rand, *Objective Criteria for the evaluation of the clustering methods.*, Journal of the Americal Stastical Association, vol 66, no. 336, pp. 846-850, 1971.
- [4] E.B. Fowlkes and C.L. Mallows, *A Method for comparing two hierarchical clusterings*, Journal of the Americal Stastical Association, vol 78, no. 383, pp. 553-569, 1983.
- [5] A. Ben-Hur, A. Elisseeff, and I. Guyon, *A stability based method for discovering structure in clustered data*, Proc. 7th pacific Symposium on Biocomputing, vol 7, pp. 6-17, Lihue, Hawaii, USA, January 2002
- [6] S. van Dongen, *Performance criteria for graph clustering and Markov cluster experiments*, Tech. Rep. INS-R0012, Centrum voor Wishekunde en Informatica (CWI), Amsterdam, The Netherlands, 2000
- [7] M.Meila, *Comparing clusterings by the variation of information.*, 16th Annual Conference of Computational Learning Theory and 7th Workshop on Kernel Machines. pp 173-187, Washington, DC, USA, August 2003.
- [8] D. Martin and C. Fowlkes and D. Tal and J. Malik, *A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics*. Proc. 8th Int'l Conf. Computer Vision. July 2001, Vol. 2, pages 416-423

- [9] Xiaoyi Jiang and Cyril Marti and Christophe Irniger and Horst Bunke, *Distance measures for image segmentation evaluation*, EURASIP J. Appl. Signal Process. Vol. 2006 No. 1 Pages 209–209 Hindawi Publishing Corp.
- [10] Jaccard, P. 1908. *Nouvelles recherches sur la distribution florale*. Bul. Soc. Vaudoise Sci. Nat. 44:223270.
- [11] Cover, T.M., Hart, P.E. *Nearest neighbor pattern classification*. IEEE Trans. Inform. Theory, IT-13(1):2127, 1967.
- [12] Rosenblatt, Frank , *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain*, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, pp. 386-408 1958.
- [13] Gallant, S. I. *Perceptron-based learning algorithms*. IEEE Transactions on Neural Networks, vol. 1, no. 2, pp. 179-191 1990.
- [14] G. Valentini and F. Masulli, *Ensembles of learning machines*, in Neural Nets WIRN Vietri-02, Series Lecture Notes in Computer Sciences, M. Marinaro and R. Tagliaferri, Eds.: Springer-Verlag, Heidelberg (Germany), 2002

Improving the Results of Partial Match

Aikaterini Krotopoulou

Department of Computer Engineering and Informatics, University of Patras, 26504, Patras, Greece

Abstract: Similarity search detection in high dimensional databases constitutes an emerging wide task in database research. Most studies focus on full similarity, while partial similarity – although it is equivalently significant – is less investigated. This matters because partial similarity is characterized by relativity, forming a demanding problem. Even *k-n-match* approach[5], which creates a solid foundation for the investigation of partial similarities, does not result in a safe method for the retrieval of this kind of similarities.

Our paper's task is the proposition of a method which overcomes the drawbacks of *k-n-match*, detecting partial matching effectively and accurately enough. The core ideas behind the proposal is (a) human-computer interaction and (b) minimizing of (a), by the exploitation of certain key-experimental and theoretical observations of *k-n-match*.

Keywords: Multidimensional Data Mining, Partial Similarity, K-n-match, Human – Computer Interaction.

1. Introduction

In most existing work, similarity search among high dimensional objects [7, 8, 9, 10, 11, 12, 13] has been modeled as the nearest neighbor (NN) problem, which considers the distance between the query object and the data objects over a fixed set of features. The main disadvantage of this modeling is that the computed distance is often affected by a few dimensions with high dissimilarity. Thus, usually the answer-set of a similarity detection query does not include the similar objects which have a few high dissimilarities (e.g. 'very' different color) with the given one. This kind of similarity where objects have certain basic similarities and a few (high or not) dissimilarities, constitutes the *partial similarity* and reflects a *global – not full – aspect of similarity*; there are a numerous applications – e.g. in medical informatics – where this kind of similarity can give a forecast or other information, elevating its great significance.

In order to overcome the drawbacks of NN model responsible for 'losing' partial similarities from the query answer set, *k-n-match* [5] problem is introduced. The latter models similarity search as matching between the query object and the data objects in n dimensions, where n is a given integer smaller than dimensionality d and these n

dimensions are determined dynamically to make the query object and the data objects returned in the answer set match best. Although the *k-n-match query* is superior to the kNN query in discovering partial similarities, this superiority depends on a good choice of the n value. The problem here is that no method for the definition of such a 'best' n for each case exists. This is addressed by introducing the *frequent k-n-match* problem [5], which finds a set of objects that appear in the *k-n-match* answers most frequently for a range $[n_0, n_1]$ of N values. Finally, *frequent k-n-match query* – which is not sensitive to the choice of n – captures the full similarity of objects, leaving partially similar objects out of the answer set. Therefore, the problem about the effective detection of partial similarities among high dimensional objects remains open.

In this paper, we propose the *LUI (Limited User-Interactive) k-n-match* method, which provides a more accurate *k-n-match*, without the need of the selection of a certain 'good' n . *Lui k-n-match* is performed for a relatively small range $[n_{lui0}, n_{lui1}]$, using human-computer interaction for the final selection of the partially similar objects for each n . The range $[n_{lui0}, n_{lui1}]$ is limited because the final value of n_{lui1} is determined according to the intermediate results of *Lui k-n-match* and it always lies between $d/2$ and d . The method resembles *frequent k-n-match* but we differentiate it in order to *assure partial similarity detection in an effective way*. Although it 'costs' a human-computer interaction time overhead, this time is reduced considerably by that range ($[n_{lui0}, n_{lui1}]$) limitation.

This paper is organized as follows: A short overview of the principles of *k-n-match* and *frequent k-n-match problem*[5] is presented in the second section. Our proposal is discussed and evaluated in the third section while we conclude in the forth section.

2. K-N-Match Problem Overview

To have a better idea about *k-n-match*[5] and *frequent k-n-match query*[5], we present and discuss the corresponding definitions, in the following:

N-match difference:

*Given two d-dimensional points (objects) $P(p_1, p_2, \dots, p_d)$ and $Q(q_1, q_2, \dots, q_d)$, let $\delta_i = |p_i - q_i|$, $i=1, \dots, d$. Sort the array $\{\delta_1, \delta_2, \dots, \delta_d\}$ in increasing order and let the sorted array be $\{\delta'_1, \dots, \delta'_d\}$. Then δ'_n is the *n-match difference* of point (object) P with regard to Q .*

K-N-Match Problem:

Given a d -dimensional database DB of cardinality c , a query point(object) Q , an integer n ($1 \leq n \leq d$), and an integer $k \leq c$, find a set S which consists of k points (objects) from DB so that for any point(object) $P1 \in S$ and any point(object) $P2 \in DB-S$, the n -match difference between $P1$ and Q is less than equal to the n -match difference between $P2$ and Q . The S is the k - n -match set of Q .

As mentioned above, k - n -match detects partial similarities among objects but its efficiency depends on a good choice of n . Although experimental and theoretical study has led to the conclusion that *neither small nor large values of n can provide a good result*, no method can ensure the selection of the accurate value of the best n in each case. To address this, *frequent k - n -matching* has been introduced as follows:

Frequent K-N-Match Problem:

Given a d -dimensional database DB of cardinality c , a query point (object) Q , an integer $k \leq c$, and an integer range $[n_0, n_1]$ within $[1, d]$, let S_0, \dots, S_i be the answer sets of k - n_0 -match, ..., k - n_i -match, respectively. Find a set T of k points (objects), so that for any point(object) $P_1 \in T$ and any point(object) $P_2 \in DB-T$, P_1 's number of appearances in S_0, \dots, S_i is larger than or equal to P_2 's number of appearances in S_0, \dots, S_i .

In other words, in *frequent k - n -match* method, firstly the k - n -match answer sets for a range $[n_0, n_1]$ of N values are found and afterwards the k points (objects) that appear most frequently in the k - n -match answer sets, are chosen.

In the framework of *partial similarity search*, the drawbacks of this method are:

(i) As it has been referred in [5], it finally captures only full similarity, while it is affected by all n 's and not by a suitable subset of them, as in partial match case.

(ii) As it has been proved, *frequent k - n -match* query is more effective than some of the 'best' similarity search techniques (IGrid[2] and Human-Computer Interactive NN search[1]) but none the less, the accuracy of its results is still 10%-30% below 100%.

Our proposal attempts to overcome these drawbacks, while it retrieves partially similar objects with high accuracy, trying at the same time to minimize the expected time overhead.

3. Our Proposal – Lui K-N-Matches

3.1. The Definition of Lui K-N-Matches

To eliminate the drawbacks discussed above, we first of all introduce the human-computer interaction in the k - n -match method in order to provide a reliable filter for every k - n -match answer-set (for each of the predefined n 's). More precisely, in each k - n -match, the results of the method are presented to the (reliable) user and he/she selects the (0- k)

objects that are really similar to the given one. Based on this selection, a new object set, named k' set is created, consisting of those k or less (0- k) selected by the user objects, which have not already been selected in the previous k - n -matches (distinct objects). When all the k' sets have been created, they will be presented to the user who will select the k objects being more similar to the given one. This method ensures more reliability of the results but one can even intuitively understand that working with all k - n -matches that correspond to a certain integer range $[n_0, n_1]$ is not the smarter solution. It can be obvious if one take into account the conclusion [5] – discussed above – that *neither small nor large values of n can provide a good partial match result*. More precisely, this conclusion stands because the results of k - n -match for:

- *small n 's*, reflect similarities in a small subset of object dimensions which does not represent the objects globally and
- *large n 's*, can lead to rough full similarity detection which results in less similar objects than partial match does [5].

None the less, there is no formula for the definition of the *best n 's* because they depend on each certain application. Thus, instead of trying to generate such a formula I studied the behavior of each k - n -match (for each n) theoretically and experimentally (both in our experiments and in the experiments of [5]). In the latter, while no certain limit between small and large n 's exists, I roughly defined this limit equal to $d/2$ in order to have a specific and safe framework. The main observations were the following:

- for *small n 's*, the answer-set usually hosts some of the real partially similar to the query object objects. It happens when some of the similar objects have high similarity to the query object in some of the dimensions related to this small n . These dimensions create the corresponding low n -match differences which determine the results in each k - n -match answer-set.

- for *large n 's*, there is a n that this and all the next n 's of the $[n_0, n_1]$ set, return answer-sets from their k - n -matches where no object has a real partial similarity with the given object, leading to empty k' sets. More precisely, the higher the dimensionality of the data set the greater the set of n 's which return empty k' sets in our method. The explanation of this is that as the dimensionality increases the calculated differences can not represent similarity or dissimilarity, because the distance to the nearest data object approaches the distance to the farthest data object [3,4]. Although this observation has been proved [3,4] for other distance metrics (e.g. Euclidian distance) it is not limited in them, while the basis of the proof is that distances for high dimensional objects return an average distance which – as dimensionality increases – can not reflect the particular characteristics and the differences of the objects. In other words, one can not know if the difference is mostly affected by many small or mediocre differences or by few high differences among dimensions. In the first case (for many

small or mediocre differences) the related objects are more possibly dissimilar, while in the second case the opposite stands.

Taking into account these observations, it is clear that the efforts for the limitation of the useless *k-n-matches* have to focus on large *n*'s. Thus, using the human-computer interaction described above, we succeed to reduce the executed *k-n-matches* by testing when no new similar objects are selected by the user, for large *n*'s. Finally, we concluded to the following method:

Lui K-N-Matches:

The *k-n-matches* for a set of *n*'s [n_{lui0} , n_{lui1}] can be found and for each *n* the results can be presented to the user, who will decide the *k* or less real partially similar objects. For each *n*, a set named *k*' set will be created consisting only from the selected objects which have not already been selected in the previous *k-n-matches* (*distinct objects*). The procedure stops when for a certain *n* larger than $d/2$, the *k*' set remains empty; we call this *n*, *stop n*. Finally, all the *k*' sets are presented to the user and the *k* (or less) most similar objects will be selected to constitute the final answer set.

3.2. Evaluation of the Method

Lui k-n-match can achieve high partial match accuracy because it is based on two filters: (a) the *k-n-matches* and (b) *human-computer interaction which can sort out the results of (a)*. The advantage behind the method is its ability to share the best skills of a human and a computer in finding the partial similarities. It is obvious that the resultant accuracy of our method is higher than the accuracy of *frequent k-n-match* which in its turn, outperforms the accuracy of other effective similarity search methods [5].

We have tested¹ *Lui k-n-match* on two synthetic data sets of 100 objects. The first set (A) has 40 dimensions while the second one (B) has 15. We used 5 query objects for each data set and as it was expected, the mean accuracy achieved was higher in B set, where the dimensionality is lower than that of A set (Table 1). As we have discussed above, *Lui k-n-match* is the only method that captures partial similarities, so there is not the capability of comparing its results with the results of an other method. None the less, in order to have a global comparative aspect of the method, we indicatively use the most relevant algorithm - *frequent k-n-match* - for the evaluation (Table 1).

Human-computer interaction may be time-consuming, but *Lui k-n-match* reduces time delay, by using *stop n* detection. In this way - as we have experimentally confirmed - a percentage of almost 30% of the *k-n-matches*

which are performed in *frequent k-n-match* case, are safely avoided.

Table 1. Accuracy on Synthetic Data

Data Set	Accuracy of Freq. k-n-match	Accuracy of Lui k-n-match
A(40dimensions)	83%	92%
B(15 dimensions)	87%	96%

4. Conclusions

Although partial similarity search among high dimensional objects forms an interesting research area, only a few studies have investigated it, while it is a hard task to isolate partial similarities in a high-dimensional vector space. Even [5] which proposes an effective partial match detection approach - *k-n-match* - can not guarantee the retrieval of real partially similar objects in each case, because a good *n* value has to be selected without having a method to achieve it. To overcome this, *frequent k-n-match* is introduced [5], but while it finally detects full similarity, the initial problem of partial match remains open.

The contribution of this paper, is the proposal of an other version of *k-n-match* approach, *Lui k-n-match*, in order to eliminate the defects of *k-n-match*. *Lui k-n-match* detects the area of *n*'s which can return real partial similarity. This is achieved via a smart diagnosis of upper 'bad' *n*'s in order to avoid the useless calculation and selection of their corresponding *k-n-matches*. This smart area detection and the human-computer interaction which is involved, *maximizes the accuracy of the results*, keeping the corresponding time overhead low enough.

More precisely, *Lui k-n-match*:

- a) Performs successive *k-n-matches*
- b) Involves the user for the final selection of the partially similar objects
- c) Stops *k-n-matches* in that large *n* (*stop n*), where none of the objects proposed by the method are selected as similar by the user.

Estimating the final performance of the method, one can acknowledge that it succeeds to capture partial similarities among high dimensional objects, with great accuracy and relatively low time overhead.

5. Acknowledgment

I thank Christos Makris for many helpful discussions on the subject of this paper. I also thank English teacher Sophia Krotopoulou for her linguistic support.

¹ The experiments were run on a desktop computer with 1.1 GHz CPU and 512 RAM.

6. References

- [1] Aggarwal C.C.: Towards meaningful high-dimensional nearest neighbor search by human-computer interaction, ICDE 2002.
- [2] Aggarwal C.C. and Yu P.S.: The igrid index: Reversing the dimensionality curse for similarity indexing in high dimensional space. In KDD 2000.
- [3] Beyer, K., Goldstein, J., Shaft, U.: When is Nearest Neighbors Meaningful? In ICDDT 1999.
- [4] Hinneburg, A., Aggarwal, C., Keim, D.: What is the nearest neighbor in high dimensional spaces? In VLDB 2000.
- [5] Tung A.K.H, Zhang R., Koudas N., Beng C.O., Similarity Search: A Matching Based Approach, VLDB 2006.
- [6] Weber R., Schek H.J, Blott S., A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces, VLDB 1998.
- [7] Zezula, P., Amato, G., Dohnal, V., and Batko, M. Similarity Search - The Metric Space Approach. Springer, 2006.
- [8] Bouteldja, N., Gouet, V., Scholl, M.: HiPer: Hierarchical Progressive Exact Retrieval in Multi-dimensional Spaces. In SISAP 2008.
- [9] Fagin, R., Kumar, R., Sivakumar, D.: Efficient similarity search and classification via rank aggregation. In SIGMOD 2003.
- [10] Faloutsos, C., Equitz, W., Flickner, M., Niblack, W., Petkovic, D., Barber, R.: Efficient and effective querying by image content. In Journal of Intelligent Information Systems, 3(3):231-262,1994.
- [11] Stricker, M., Orengo, M.: Similarity of color images. In Storage and Retrieval for Image and Video Databases, SPIE, 1995.
- [12] Weber, R., Blott, S.: An approximation based data structure for similarity search. Technical Report 24, ESPRIT project HERMES (no. 9141) October 1997.
- [13] Arya, S., D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. Journal of the ACM, vol. 45, no. 6, pp. 891-923
- [14] Korn, P., Sidiropoulos, N., Faloutsos, C., Siegel, E., Protopapas Z.: Fast and Effective Retrieval of Medical Tumor Shapes. In IEEE Transactions on Knowledge and Data Engineering, Vol 10, No 6, 1998.

SESSION

DATA MINING IN THE SOCIAL AND BEHAVIORAL SCIENCES

Chair(s)

Dr. Anthony Scime

The Use of Logistic Regression Analyses and Data Classification Mining to Examine Variables Predictive of Long-term Healthcare Staff Giving Cessation Advice

Celia A. Watt, Ph.D., Jill W. Lassiter, Ed.D., and Douglas M. Scheidt, Ph.D.
 Department of Health Science, State University of New York – College at Brockport¹
 Brockport, New York, USA

Abstract - *This paper examines variables that might be predictive for healthcare staffs' advising tobacco cessation to residents via logistical regression analysis and classification data mining. Independent variables that might influence the dependent variable outcome (healthcare workers' willingness to give advice) are the staffs' smoking status; professional license or position in the facility, their health beliefs regarding the harmful effects of smoking or environmental smoke, the efficacy of quitting for this older population, their beliefs about safety related to smoking, their views of policy and/or administrative supports or barriers, their opinion about older adults' quality of life and how cessation might impact it, their stance on resident autonomy issues, and other various barriers to advising. Logistic Regression analysis is the more common method for examining variables predictive of an outcome in social sciences, however data classification can also be used. Findings of this research found parallels across the two types of analyses.*

Keywords: Data Classification Mining, Logistic Regression Analysis, Tobacco Control.

1 Introduction

This paper examines one set of data using two analytical techniques: logistical regression analysis and classification data mining. An abbreviated background outlining the domain from which the data was collected and they types of variables analyzed are discussed followed by an overview of the two techniques and results.

Tobacco control is an expansive domain and the data presented in this paper are specific to resident cigarette smoking in nursing homes or long-term care facilities. Cigarette smoking has been banned in the majority of healthcare facilities however concern for individuals' rights to self-determination in long-term care settings [1] has traditionally resulted in fewer restrictions for the residents residing in them. While this remains a controversial topic, healthcare providers' attitudes toward resident smoking

and their likelihood of delivering cessation advice to residents provides an initial glimpse into this debate from a staff perspective.

Factors that might influence healthcare workers' willingness to give advice are as follows: their own smoking status; their professional license or position in the facility, their health beliefs regarding the harmful effects of smoking or environmental smoke and the efficacy of quitting for this older population, their beliefs about safety related to smoking, their views of policy and/or administrative supports or barriers, their opinion about older adults' quality of life and how cessation might impact it, their stance on resident autonomy issues, and other various barriers to advising.

Logistic Regression analysis is the more common method for examining variables predictive of an outcome in social sciences, however data classification mining can also be used. As an analytic tool, data mining provides knowledge about the structure and interrelationships among data (expressed in a decision tree and translated into classification rules) which may provide new knowledge or confirmation of theory in the domain from which the data comes. For example, using classification mining [2] confirmed that campaigns do matter in presidential elections and [3] mined crime data to confirm the relationship between police presence and incidence of crime. This paper will examine the use of data mining in the domain of tobacco control and the more traditional logistic regression techniques.

2 Methods

Questionnaires completed by the staff at long-term care facilities (that allowed smoking) were examined to explore their attitudes and beliefs regarding resident smoking and giving cessation advice. Two methodologies were used to examine healthcare providers' likelihood to give cessation advice to residents; logistic regression analyses and classification data mining.

¹This research was funded in part by NIH (1 R03 CA097742-01 Watt - Primary Investigator)

2.1 Logistic Regression Analyses

Logistic regression analyses are used to reveal predictive relationships when the response variable of interest is dichotomous. This study examines healthcare providers self-report of giving cessation advice to patients or not. Logistic regression analyses are able to determine whether multiple independent variables (for example, personal smoking status, license, views on resident's autonomy) have a predictive relationship to the dichotomous dependent variable (advising or not). The independent variables do not have to be dichotomous. Logistic regression analyses determine the relationship between the defined independent variables and a dichotomously defined dependent variable in terms of the likelihood (odds ratio) of data fitting a logistic curve [4].

In logistic regression, the data are represented in a plot graphing the independent (predictor) variable on the X axis. The Y axis represents the dependent variable. Since the dichotomous dependent variable is coded 0 (i.e., not advise) or 1 (i.e., advise), the Y axis represents the proportion of cases coded as 1. With these two categories coded 0 and 1, the proportion is the same as the mean of that variable. The logistic curve is an estimated plot of the independent variable on the X axis (the predictor variable) to the predicted mean of the dependent variable [5]. The predicted value is based on the formula below:

$$P = \frac{e^{(a+bX)}}{1 + e^{(a+bX)}} \quad (1)$$

In this formula, the constant e is the base of the natural log (approximately 2.718). The variables a and b are the variables which will be optimized by the logistic regression to create a curve that best fits the data. The constant a produces the value of the above formula when X is 0. The constant b is the value that indicates how strongly the curve increases along the Y axes as the value of X increases.

Logistic regression analysis involves an iterative process of estimating the parameters a and b and evaluating the goodness of fit between the logistic curve and the actual plot of x and $P(y)$. Goodness of fit is measured using a likelihood (probability) estimate of the observed data from the predicted curve. The iterative process continues to reach a threshold of maximum likelihood, often set at .01 or .001. Likelihood would be measured based on the variance of the difference between the observed value of Y and the predicted value of Y , aggregated across the values of the independent variable.

In a logistic regression, the odds ratio is calculated by raising e to the exponent b or e^b . The odds ratio is the factor by which the odds that the dependent variable is increased by an increase in the independent variable. If the

odds ratio is less than 1.0, the relationship is inverse, yielding a decrease in the dependent variable. If the odds ratio is 1.0 (or its confidence interval includes 1.0), the independent variable is determined to have no relationship to the dependent variable. If the odds ratio is greater than 1.0, an increase in the independent variable corresponds to an increase in the dependent variable.

2.2 Classification Data Mining

Data mining is a process that involves the analysis of data to find patterns and previously unknown relationships in data. Classification algorithms construct decision trees by looking at the past performance of independent variables with respect to a dependent variable.

The decision tree is constructed from data instances or cases with known values for the dependent variable, referred to as the training data. Using a divide-and-conquer algorithm, the node with the most information gain is selected from the data attributes. The training data are divided based on the decision at this node, thereby creating two subsets of data. Each subset is evaluated independently to select the next node along its edge. The process of dividing the data and selecting the next node, which is the one with the greatest information gain at that point, continues until a leaf node is constructed or an outcome is identified. The variable with the greatest information gain is selected as the node for dividing the data in the decision tree at each node. The information gain with the overall highest gain with respect to the dependent attribute is the root of the tree.

During the tree-construction process, branches may be pruned to increase the classification performance. Pruning simplifies the tree by replacing or removing branches that exceed an established error rate. Beginning at the bottom of the tree, for each non-leaf node in the tree the expected error rate is calculated as if that node were to become a leaf with the class of its most-frequent children (the nodes immediately below a given node). The expected error rate is also calculated for the branch with the children not pruned. If the error is greater for the pruned sub-tree, then the sub-tree is maintained. This process proceeds up the tree until the root node is reached. The accepted error is based on an acceptable confidence that the error will not exceed an established level and the instances will be correctly classified. A confidence level of 25% strives to produce a 75% success rate at each node individually [6, 7].

3 Results

3.1 Overall

A total of 646 staff questionnaires (64 physicians, 277 licensed nurses, and 305 CNAs) from 47 smoking facilities across the United States were examined in the

following analyses. Regionally, 10 were from the Northeast, 14 from the South, 11 from the West, and 12 from the Midwest. The majority of facilities (77%; n = 36) allowed all residents to smoke. When asked to estimate the percentage of residents in the facility that smoked, contact persons indicated a range from 0 to 30% (M = 5%).

3.2 Logistic Regression Analyses

To examine factors that might be associated with healthcare staff reports of advising, logistic regression models were used to examine the relationship of items in several domains (health beliefs, safety, policy/administrative, quality of life, autonomy, and barriers to advising) to the dichotomous outcome variable of advising. Because descriptive analyses found numerous differences in these domains across license and smoking status, preliminary logistic regression runs examined the predictive value of these two variables. Smoking status was not predictive of advising therefore not controlled for in subsequent logistic analyses. However, healthcare staff license was predictive of advising and was therefore included in analyses examining the domains to mediate the impact of license. Five separate logistic regression analyses examined the domains as illustrated in Table 1, all controlling for healthcare staff license. These analyses indicate that nurses' personal health beliefs and license are most strongly predictive of giving cessation advice.

3.3 Data Mining

The data set of 646 cases was split into training (215 cases) and testing (431 cases) data sets with 49 of 164 attributes. The 49 attributes were selected by a domain expert based upon smoking cessation theory. The C4.5 classification algorithm as implemented in WEKA was applied to the training data using 10-fold cross validation and 25% confidence. This resulted in a 21 branch tree with a 64% success rate (64% of the cases were classified correctly). The test set was evaluated using this model with 63.8% success. The 21 branches contained 41 nodes. When the branches were converted into rules, the rule that accounted for the majority of the cases resulted:

IF the healthcare worker doesn't strongly disagree with the statement, "I don't advise because it is not effective"
 AND their professional license is a Certified Nursing Assistant
 AND they don't strongly disagree with the statement, "Physicians are the only staff that should give cessation advice"
 THEN they do not give cessation advice.

In other words, data mining revealed that the most common trend regarding cessation advising was that CNAs who tend to believe that physicians are the only staff that should give advice and that cessation advising is not effective are not likely to advise residents to quit smoking.

Table 1. Variables Associated with Healthcare Staff Advising Residents to Quit

	OR	95% CI
Analysis 1: Healthcare Staff License		
License	1.58***	1.23, 2.05
Analysis 2: License and Health Beliefs		
License	1.48**	1.13, 1.91
Composite Score	1.62***	1.25, 2.09
Analysis 3: License and Safety Issues		
License	1.43**	1.10, 1.86
Composite Score	1.46***	1.19, 1.80
Analysis 4: License and Policy/Administrative Issues		
License	1.37*	1.05, 1.78
Composite Score	1.54***	1.26, 1.87
Analysis 5: License and Autonomy Issues		
License	1.53**	1.17, 2.00
This facility is home to residents; they should have the right to smoke.	0.86*	0.74, 1.00
Residents aren't capable of making decisions regarding smoking/health.	1.14	0.97, 1.35
Analysis 6: License and Barriers to Giving Advice		
License	1.31*	1.01, 1.72
Composite Score	0.57***	0.45, 0.72

***p<.001
 **p<.010
 *p<.050

4 Conclusion

The data mining rule that accounted for the majority of cases parallels the findings from the logistic regression analyses that found the healthcare workers license along with barriers (beliefs regarding effectiveness of giving advice) and administrative/policy issues (authority to give advice) were predictive of not giving advice. However, examining the variables from a data classification model point of view may lend itself to a more “reader-friendly” interpretation of the data rather than odds ratios since it describes adherence to a rule. For example, data mining results can be interpreted to infer that if nurses classify within particular rules (specifically they agree that advising is effective, that staff other than physicians should advise, and are LPNs or RNS) they are more likely to advise smoking cessation. A predictive interpretation of logistic regression would read that nurses with a higher license (i.e. RN’s) are 1.58 times more likely to advise residents to quit smoking, as well certain health beliefs and policy issues have similar predictive strength on advising (see Table 1). While both data mining and logistic regression analyses revealed the same domains to be most predictive of healthcare staff advising behaviors, data mining provided more specific insights into the characteristics of the domains that were most relevant. In this case, data mining classification allows for more succinct output describing specific areas within each domain that are associated with higher likelihood for advising, while logistic regression output would require much more advanced analysis and interpretation to provide such detailed findings.

Several iterations of data mining varying the included cases are often necessary to theoretically interpret the resultant rules. As such, future analyses comparing these methods should exclude professional license to ascertain how much the resultant rules are impacted by the healthcare workers’ license.

Traditional statistical methods for examining social and/or behavioral research questions are valuable resources but are not the only tool for researchers. Data classification mining offers additional valuable insight that may be equally valid for descriptive and predictive purposes.

5 References

- [1] Omnibus Budget Reconciliation Act of 1987, *Public Law 100-203, Sections 4201(a), 4211(a)*.
- [2] G. Murray, and A. Scime, “Micro-targeting and Electorate Segmentation: Data Mining the American National Election Studies,” *Journal of Political Marketing*, in press.
- [3] S. Bagui, “An approach to mining crime patterns,” *International Journal of Data Warehousing and Mining*, vol. 2, no. 1, pp. 50-80, 2006.
- [4] D. W. Hosmer, and S. Lemeshow, *Applied Logistic Regression*, Chichester, UK: Wiley, 1989.

- [5] W. L. Hays, *Statistics*, 5th ed., Fort Worth: Harcourt College Publishers, 1994.
- [6] J. Han, and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed., Boston: Morgan Kaufmann, 2006.
- [7] I. H. Witten, and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed., San Francisco, CA: Morgan Kaufmann, 2005.

Testing Terrorism Using Iterative Expert Data Mining

G. R. Murray¹, L. Y. Hunter¹, and A Scime²

¹Department of Political Science, Texas Tech University, Lubbock, Texas, USA

²Department of Computer Science, The College at Brockport, Brockport, New York, USA

We analyze a unique dataset comprised of terrorist events and measures of social, political, and economic contexts in 185 countries worldwide between the years of 1970 and 2004 using the Iterative Expert Data Mining (IEDM) Methodology. The IEDM process allows researchers to identify a small number of input variables that exhibit theoretical and predictive significance in datasets composed of large numbers of variables and cases. We test current debates within the terrorism literature regarding the causes of terrorism, including level of democracy, economic development, modernization, and social fractionalization. The worldwide data and comprehensive analysis suggest that the utility of the final model is likely to persist over time.

Keywords: terrorism, intrastate violence, data mining, IEDM, decision trees

1 Introduction

Scholars have identified a number of social, political, and economic contexts within countries that are associated with the likelihood of a state falling victim to terrorism. Some researchers focus on geographical location, type of political system, and degree of modernization [1]. Others associate terrorist activity at the country level with regime type, income inequality, and economic globalization [2], [3]. Still other researchers relate terrorism to a state's level of democracy, religious composition, and level of wealth [1], [4]-[7]. Debate, however, continues over the effects these and other factors have on the likelihood that a country will experience terrorism.

In an attempt to sort out the variety of national conditions that have been identified as contributors to the occurrence of terrorism in a country, we present and analyze a unique dataset using a methodology that is well suited for unraveling divergent theoretical arguments. The dataset is comprised of terrorism events and measures of social, political, and economic contexts in 185 countries worldwide between the years of 1970 and 2004. We analyze this dataset using the Iterative Expert Data Mining Methodology (IEDM) [8] to evaluate current debates within the literature regarding the predictors of terrorism. The IEDM process allows researchers to identify a small number of input variables that exhibit theoretical and predictive significance in datasets

composed of large numbers of variables and cases. In this endeavor, we first present a review of the literature related to the causes of terrorism. Then we follow with a brief description of the IEDM methodology and its application to our dataset. We conclude with a discussion of the contributions of this research.

2 National Conditions and Terrorism

Researchers have associated a number of social, political, and economic conditions at the national level with the probability that a state will fall victim to a terrorist event. These conditions generally fall into at least one of four broad categories: level of democracy, economic development, modernization, and social fractionalization. The first factor is a state's level of democracy. Some scholars argue that higher levels of democracy lead to fewer terrorist attacks due to less political discontent [9]. Others contend that higher levels of democracy lead to more terrorist attacks. Some of these researchers suggest that democratic states are easier targets due to the political freedoms they must provide to their citizens that make stopping terrorist attacks more difficult [1], [3], [10]-[12]. Similarly, some contend that democratic governments may be more likely to be targets of terrorism because of the greater opportunity terrorists have to publicize an attack and link it with their cause in the free media [3], [13], [14]. Other scholars find that terrorists may perceive that their chances of influencing government decisions are greater in democratic states because these governments "shape their policies in response to public opinion while autocracies may be insulated from the influences of the masses" [15].

While levels of democracy have been related to terrorism, state wealth and factors regarding economic development have also been associated with terrorist attacks. Broadly speaking, as a country's wealth increases, the probability that it will be targeted for a terrorist attack increases [1], [4], [6], [7]. While there is evidence that terrorism becomes a less viable option for individuals as social factors such as poverty recede [16], other evidence indicates that countries with higher levels of economic growth experience a greater number of terrorist attacks as do states with higher overall levels of Gross Domestic Product (GDP) [7].

One of the reasons wealthier societies suffer from more terrorist attacks may be associated with the effects of modernity [5]. Modernity refers to economic changes that take place within a country that lead to societal changes such as increased urbanization and education, easier communication and social networks, and the growth of the middle and upper classes [17], [18]. An expanding urban environment increases physical proximity, which provides individuals with greater opportunities to centralize their agenda for change through personal networking and the exchange of goods and ideas [5]. In urban environments, terrorists have access to greater media resources, a larger audience to observe their actions, and more potential sympathizers and recruits. Generally speaking, a larger population provides more opportunities for terrorists to carry out their attacks [3] through an increase in resources, group visibility, and access to human capital [1], [5]. Modernization also often makes communication easier through such means as increased media access and literacy rates and new social networks and forms of transportation. During the early phases of modernization, there is often a clash between modern and traditional cultures [17], [18]. As a society becomes more educated and has access to a greater pool of resources, some individuals may be more likely to pressure the government for reform. And if the channels of political representation cannot effectively aggregate societal interests, dissatisfied individuals in a modernizing society may be able to organize more effectively and/or pursue violent strategies to promote policy change [19].

Finally, although some scholars contend that social fractionalization plays only a small role in terrorism [7], others argue that terrorism is related to racial, ethnic, and religious group identification within a state [6], [20]. These researchers suggest that aggrieved identity groups within a country may initiate social movements in an attempt to gain government attention for their concerns. In the event identity groups perceive that their position within society is weakening, an individual or third party may intervene to channel the disenfranchised group's discontent into organized violence directed against the state [1], [21], [22].

3 The IEDM Methodology

The Iterative Expert Data Mining (IEDM) methodology [8] combines classification data mining techniques with domain expertise, the knowledge and skills held by a person who is considered especially qualified in a domain [24], to construct models. Data mining is a process of inductively analyzing data to assess known relationships as well as to find interesting patterns and unknown relationships. The term "data mining" encompasses a number of techniques and involves both human and computational resources [8], [23]. Classification data mining analysis is a technique that may be both predictive and explanatory [25]. While there are a variety of classification algorithms, the algorithm used here constructs decision tree models. These models use a divide-

and-conquer algorithm to evaluate the past performance of input variables with respect to an outcome variable. Specifically, the IEDM methodology proceeds as follows:

1. The domain expert selects variables and cases that are likely to produce useful results. The cases can also be selected randomly.
2. Cases are randomly assigned to the "training" set, which is used to construct the model, and the "test" set, which is composed of independent data that are reserved for testing the model's robustness.
3. The input variables are rank ordered by association with the outcome variable using the training set.
4. A series of models is created by repetitively executing the data mining algorithm on the training set and progressively removing input variables beginning with the least associated input variables.
5. The domain expert reviews and compares the resulting models and identifies the most promising model.
6. The expert uses domain knowledge to adjust the model, such as adding variables that increase the model's accuracy and theoretical grounding or deleting variables that have little impact on the model's performance.
7. These adjustments are used to create and compare additional models.
8. The domain expert identifies the preferred, final model.
9. This final model is tested for robustness using the test set.

4 The Data

We created a unique dataset comprised of terrorism events and measures of social, political, and economic contexts in 185 countries worldwide between the years of 1970 and 2004.¹ The domain expert (a political scientist with expertise in international relations and the causes of terrorism) selected pertinent variables from a variety of datasets including the Global Terrorism Database, Correlates of War, Database of Political Institutions (2006), and World Development Indicators (2008). The dataset includes a broad range of variables that appear in the scholarship on contextual effects and terrorism at the national level. The analysis, which covers incidents over a number of decades, is designed to capture long-term predictors that have persisted, and are more likely to persist, over time.

Beginning the IEDM process, in terms of case selection, the domain expert selected as the outcome variable a dichotomous variable indicating whether a country was victim of a terrorist event in a given year (coded 1) or not (coded 0). That is, the unit of analysis is a country-year. Terrorist events were identified in and defined by the Global Terrorism Database (GTD). GTD defines terrorism as "the threatened or actual use of illegal force and violence by a

¹ Data on our dependent variable, terrorist events, are not available in the Global Terrorism Database for 1993; therefore, the dataset does not include that year.

non-state actor to attain a political, economic, religious, or social goal through fear, coercion, or intimidation” [29]. The GTD includes data from 1970 to 2004², which define our years of analysis, and 202 countries, which are reduced to 185 in our analysis due to missing data from other datasets. In terms of variable selection, the domain expert initially selected from the disparate datasets more than 200 input variables informed by the scholarly literature [1], [3], [5], [16]. The expert reduced the initial set of input variables by excluding measures that appeared to be variations of other input variables, such as excluding indicators of GDP in local currency while retaining indicators of GDP in constant dollars, and alternate measures of urban population. Of the remaining variables, the expert retained the 126 input variables that held the most theoretical and empirical significance.

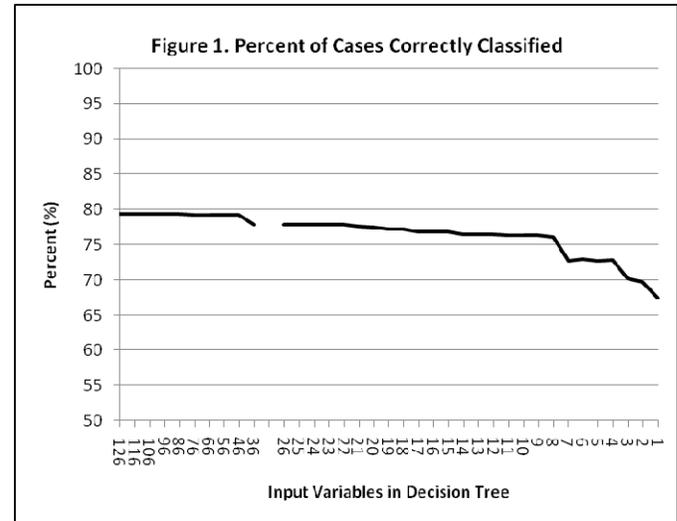
The resulting dataset includes 5431 cases or country-years. Given that larger training sets construct more accurate models, we randomly assigned two-thirds of the country-years ($N = 3652$) to the training set and one-third ($N = 1779$) to the test set.

5 Results: The Model

We next used the training set to estimate the performance of each input variable with respect to the outcome variable using the Chi-squared Automatic Interaction Detection (CHAID) algorithm [26], [27] as implemented by Answer Tree 3.1, a software package of decision tree algorithms and related tools. Table 1, Columns 1 and 2 present the top 22 input variables by rank. The χ^2 estimates indicate that the best performing input variables are urban population ($\chi^2 = 760.23$, $p < 0.001$), gross national expenditure ($\chi^2 = 751.50$, $p < 0.001$), income inequality as measured by the GINI coefficient ($\chi^2 = 748.70$, $p < 0.001$), and military expenditures ($\chi^2 = 617.88$, $p < 0.001$). These are followed in the top 10 by military personnel, net income from abroad, the National Capabilities Index, crisis state status, level of political competition, and level of competitiveness of political participation. The top 10 input variables reflect a diverse theoretical mix of level of democracy (e.g., levels of political competition and competitiveness of political participation), economics (e.g., net income from abroad, income inequality, and gross national expenditure), and modernization (e.g., urban population, energy production, and national capabilities). The remaining variables in the top 22 also contribute to the theoretical mix with, for instance, measures of state capacity (e.g., energy imports) and regime persistence.

Following the IEDM methodology, we constructed a series of decision tree models by using the Answer Tree implementation of CHAID and progressively eliminating input variables (10 at a time until reaching the top 25

variables then one at a time) by weakest performance relative to the outcome variable. That is, the first tree was constructed from all 126 input variables (DT126), the second tree with the 116 best performing input variables relative to terrorist incidents (DT116), and the last tree was constructed from the one best performing input variable relative to terrorist incidents (DT1). Figure 1 reports the accuracy of a series of trees as represented by the proportion of cases correctly classified as experiencing a terrorist event or not. It indicates that accuracy declines from around 80% for the full model of all 126 variables to about 70% for the restricted model composed of the last three variables.



The next step in the IEDM methodology is to identify the most promising model from the perspective of knowledge discovery and then to adjust that model using domain expertise. Our starting point in this research is the most efficient model; that is, the decision tree that correctly classifies the greatest number of country-years using the smallest number of input variables. The full decision tree model that includes all 126 input variables (DT126) correctly classifies 79.3% of the country-years. As Table 1, Column 5 indicates, the smallest model that is statistically indistinguishable from DT126 in terms of correctly classifying country-years is the tree that includes the top 22 input variables (DT22) ($t = 1.52$, $p = 0.07$, one-tailed test), which correctly classifies 77.8% of cases (Column 4). That is, DT22 is the most efficient model before the domain expert makes further theoretical adjustments. The statistical test employed is the t-test of independent samples with a one-tailed test of statistical significance. The one-tailed test is appropriate given the general expectation that the percent correctly classified decreases as the number of input variables decreases.

The IEDM process next calls for the domain expert to evaluate and to make theoretical adjustments to the input variables in DT22. In particular, social fractionalization [6], [20] stands out as a theoretical domain that is unrepresented in

² See footnote 1.

the top 22 input variables. In order to assess this domain, the expert forced back into the dataset measures of religious, ethnic, and language fractionalization as well as indicators of a country's dominant religion, the chief executive's affiliation with a religious party, the chief executive's party ideology, and nationalistic tendencies of the chief executive's party. Conflict [28] also stands out as an unrepresented theoretical domain in DT22, so the expert also forced back into the dataset eight measures of internal and external conflict ranging from indicators of riots and civil war to minor conflict (less than 1000 deaths per year) and major conflict (1000 or more deaths per year). Finally, the domain expert forced back into the dataset a measure of media access to capture the effect the presence of free media has on terrorism [3], [12]. In total, then, the domain expert forced back into the dataset 16 input variables for further analysis in conjunction with the top 22 input variables.

These 38 input variables (i.e., DT22 plus 16 domain expert additions) construct a decision tree model that correctly classifies 78.0% of country-years, which is statistically indistinguishable from DT22 ($t = 0.20$, $p = 0.83$, two-tailed test). That is, the expert adjustments did not statistically improve the accuracy of the model. Although five of the expert additions emerged in the tree, which suggests that they are theoretically related to the occurrence of terrorist activity, their relationship is substantively inconsequential in the context of the other input variables.

Table 1. Input Variables and Identification of Naïve and Final Trees

Rank	Input Variables	χ^2*	Cumulative Correctly Classified	DT Statistically Different v. DT126†	In Final Tree
1	Urban population	760.23	0.674	Yes	Yes
2	Gross national expenditure	751.50	0.696	Yes	No
3	Income inequality (GINI)	748.70	0.701	Yes	Yes
4	Military expenditure	702.43	0.728	Yes	Yes
5	Military personnel	617.88	0.726	Yes	No
6	Net income from abroad	601.28	0.729	Yes	Yes
7	National Capabilities Index	600.86	0.727	Yes	Yes
8	Crisis state	574.78	0.760	Yes	Yes
9	Political competition	558.17	0.763	Yes	Yes
10	Competitiveness of participation	554.17	0.763	Yes	No
11	Energy production (COW)	540.42	0.763	Yes	Yes
12	Regulation of participation	519.77	0.764	Yes	Yes
13	Executive constraints	493.03	0.764	Yes	Yes
14	Executive recruitment	481.40	0.764	Yes	No
15	Country's political regime	481.05	0.768	Yes	Yes
16	Competitiveness of executive recruitment	476.78	0.768	Yes	No
17	Iron/steel production	465.26	0.768	Yes	Yes
18	Energy imports, net	462.36	0.772	Yes	Yes
19	Energy production (kt of oil)	440.85	0.772	Yes	No
20	Population density	394.79	0.774	Yes	No‡
21	Legislative Index of Electoral Competitiveness	383.60	0.776	Yes	Yes
22	Regime's continuous rule (Yrs)	382.42	0.778	No	Yes

Outcome variable: country was victim of terrorism in a given year (coded 1) or not (coded 0).

* $p < 0.001$ for all input variables.

† Statistically different at the 0.05 level, t-test of independent samples, one-sided test.

‡ Appears in reduced tree of 16 input variables, but removed due to theoretical correlation with urban population.

To complete the IEDM process and identify the final model, it is important to note that not all variables put into a decision tree algorithm appear in the tree. Some input variables are not significantly related to the outcome variable in the context of the model and, therefore, are not used to classify cases. Examination of Table 1, Column 6 shows that

six of the top 22 input variables are not used to classify cases and do not appear in the tree (e.g., gross national expenditure), which reduces the set of input variables to 16. It is also important to note in Table 1 that two highly theoretically correlated variables also appear in the tree: urban population and population density. We removed population density, the lower ranked of the two variables, from the set of input variables because it is theoretically unlikely to add to the predictive power of the model. This adjustment did not statistically affect the model. As such, the final model produced by the IEDM process includes the 15 input variables identified in Table 1, Column 6. This model, which can be converted into 44 rules that apply to at least 1% of the cases, correctly classifies whether a country experienced a terrorist event or not in a given year in 78.0% of the cases. The proportional reduction in error is 50.9%, which indicates that the model classifies country-years substantially more effectively than simply guessing the modal category (i.e., the most frequently occurring category: No Attacks, 55.3%).

To complete the IEDM process, we tested the final model for robustness by comparing the accuracies of the training and test sets. The final model correctly classified 78.0% of the country-years in the training set, while it correctly classified 76.2% in the test set. Analysis of these success rates indicates that the two trees are statistically indistinguishable ($t = 0.15$, $p = 0.19$, two-tailed test); therefore, we conclude that the final model is generalizable and not over-fitted to the data. Further, the model is not statistically distinguishable from the full model DT126 ($t = 1.31$, $p = 0.10$, one-tailed test), which correctly classified 79.3% of cases. This suggests that the model is also efficient.

6 Discussion and Conclusion

In an attempt to clarify the national conditions that contribute to the occurrence of terrorism in a country, we presented and then analyzed a unique dataset using Iterative Expert Data Mining [8], a methodology that is well suited for empirically unraveling divergent theoretical arguments. The final, restricted model of 15 input variables correctly classified whether a country experienced a terrorist event or not in a given year in almost eight of 10 cases. Further, the final model correctly classified these cases at a rate that is statistically indistinguishable from the full model composed of all 126 input variables.

We initially restricted the full set of variables to create a model of 22 input variables based on variable rank and efficiency (i.e., the smallest number of input variables with the greatest statistical rate of correct classification). The DT22 model included measures of many of the widely noted national contexts that have been related to terrorism in the extant literature [1]. For instance, the model captures level of democracy (e.g., political competition and political regime), economic development (e.g., gross national expenditure and net income from abroad), and modernization (e.g., urban

population and national capabilities). The domain-expert then reviewed the model and found that it did not capture the theoretically significant construct of social fractionalization [6], [20] or more minor constructs such as conflict [28] and media access [3], [12]. Further analysis with these variables indicated that they did not improve the predictive power of the model, which suggests that their relationship is substantively inconsequential in the context of the other input variables. Thus, social fractionalization and the other minor constructs may be theoretically related to terrorism, but the IEDM process suggests that they are not among the primary factors that affect terrorism.

Returning to DT22 to identify the final model, Table 1, Column 6 indicates that six of the 22 input variables do not appear in the final model because they are not used to classify cases. These variables include some, but not all, of the measures of democracy such as competitiveness of participation and the two measures of executive recruitment. We believe the eliminated variables may be collinear; that is, their associations with the outcome variable are confounded because they capture very similar aspects of the outcome variable. The terrorism literature is unsettled on the relationship between levels of democracy and terrorist attacks [1], [5], [11]. This research suggests that level of democracy is related to terrorism, but the effect is constrained. Similarly, a highly ranked and broad-based economic variable, gross national expenditure, also fails to appear in the final model. On the other hand, economic measures such as overall national capabilities, net income from abroad, and income inequality continue to demonstrate a significant relationship with the outcome variable. These results suggest that economic factors are important and diversified predictors of terrorism [1], [7].

In sum, our findings demonstrate the effects of social, political, and economic conditions that are often associated with terrorist activity in a country. Democracy and economic development appear to be related to terrorism, but the results offer particularly strong support for the argument that modernization is associated with terrorist activity. On the other hand, the IEDM process suggests that social fractionalization is theoretically related to terrorism, but that it is not among the primary factors that affect terrorism. The IEDM process allowed us to identify a small number of input variables that exhibit theoretical and predictive significance in this unique dataset, which captures a broad range of national conditions in a sample of 185 countries over a multi-decade timeframe. The worldwide data and comprehensive analysis suggest that the utility of the model is likely to persist over time.

7 References

- [1] J. I. Ross, "Structural causes of oppositional political terrorism: Towards a causal model," *J. Peace Research*, vol. 30, no. 3, pp. 317-329, 1993.
- [2] Q. Li and D. Schaub, "Economic globalization and transnational terrorist incidents: A pooled time series cross sectional analysis," *J. Conflict Resolution*, vol. 48, no. 2, pp. 230-258, 2004.
- [3] M. Koch and S. Cranmer, "Testing the Dick Cheney hypothesis: Do governments of the left attract more terrorism than governments of the right?" *Conflict Management and Peace Science*, vol. 24, pp. 311-326, Sept. 2007.
- [4] T. R. Gurr, "Some characteristics of political terrorism in the 1960s," in *The Political Terrorism*, M. Stohl, Ed. New York: Marcel Dekker, 1979.
- [5] M. Crenshaw, "The causes of terrorism," *Comparative Politics*, vol. 13, no. 4, pp. 379-399. 1981.
- [6] A. T. Turk, "Social dynamics of terrorism," *Annals AAPSS*, vol. 463, pp. 119-128, Sept. 1982
- [7] A. B. Kugler, *What makes a terrorist: economics and the roots of terrorism*. Princeton: Princeton University Press, 2007.
- [8] A. Scime and G. R. Murray, "Vote prediction by iterative domain knowledge and attribute elimination," *I. J. Business Intelligence and Data Mining*, vol. 2, no. 2, pp. 160-176, 2007.
- [9] P. Kurrild-Klitgaard, J. Mogens, and R. Klemmensen, "The political economy of freedom, democracy and transnational terrorism," *Public Choice*, vol. 128, no. 1-2, pp. 289-315, 2007.
- [10] P. Rosendorff and T. Sandler, "Too much of a good thing? The proactive response dilemma," *J. Conflict Resolution*, vol. 48, no. 5, pp. 657-671, 2004.
- [11] J. Eyerman, "Terrorism and democratic states: Soft targets or accessible systems," *International Interactions*, vol. 24, no. 2, pp. 151-170, 1998.
- [12] A. Braithwaite and Q. Li, "Transnational terrorism hot spots: Identification and impact evaluation," *Conflict Management and Peace Studies*, vol. 24, no. 4, pp. 281-296, 2007.
- [13] W. Eubank and L. Weinberg, "Does democracy encourage terrorism?" *Terrorism and Political Violence*, vol. 6, no. 4, pp. 417-435, 1994.
- [14] W. Enders and T. Sandler, "Is transnational terrorism becoming more threatening?" *J. Conflict Resolution*, vol. 44, no. 3, pp. 307-32, 2000.
- [15] N. Sambanis, "Poverty and the organization of political violence," *Brookings Trade Forum*, pp. 165-211, 2004.
- [16] A. Kruglanski and S. Fishman. "The psychology of terrorism: 'Syndrome' versus 'tool' perspectives," *J. Terrorism and Political Violence*, vol. 18, no. 2, pp. 193-215, 2006.

- [17] S. M. Lipset, 1959, "Some social requisites of democracy: Economic development and political legitimacy," *Am. Pol. Sci. Rev.*, vol. 53, no. 1, pp. 69-105, 1959.
- [18] S. M. Lipset and S. Rokkan, "Cleavage structures, party systems, and voter alignments: An introduction," in *Cleavage Structures, Party Systems and Voter Alignments*, S. M. Lipset and S. Rokkan, Eds. New York: Free Press, 1967.
- [19] S. Huntington, *Political Order in Changing Societies*. New Haven: Yale University Press, 1968.
- [20] L. E. Dutter, "Ethno-political activity and the psychology of terrorism," *Terrorism: An International Journal*, vol. 10, no. 3, pp. 145-163, 1987.
- [21] M. B. Brewer, "Ingroup identification and intergroup conflict: When does ingroup love become outgroup hate?" in *Social Identity, Intergroup Conflict, and Conflict Reduction* (3rd ed.), R. D. Ashomre, L. Jussim, and D. Wilder, Eds. New York, NY: Oxford University Press, 2001, pp. 17-41.
- [22] J. Sidanius and J. R. Petrocik, "Communal and national identity in a multiethnic state: A comparison of three perspectives," in *Social Identity, Intergroup Conflict, and Conflict Reduction* (3rd ed), R. D. Ashomre, L. Jussim, and D. Wilder, Eds. New York, NY: Oxford University Press, 2001, pp. 101-129.
- [23] M. Hofmann and B. Tierney, "The involvement of human resources in large scale data mining projects," *Proc. 1st Int. Sym. Information and Communication Technologies*, Dublin, Ireland, 2003, pp. 103-109.
- [24] J. C. Giarratano and G. D. Riley, *Expert Systems: Principles and Programming*, 4th Ed. New York: Course Technology, 2004.
- [25] K. Osei-Bryson, "Evaluation of decision trees: A multi-criteria approach," *Computers and Operations Research*, vol. 31, no. 11, pp. 1933-1945, 2004.
- [26] G. Kass, "An exploratory technique for investigating large quantities of categorical data," *Applied Statistics*, vol. 29, no. 2, pp. 119-127, 1980.
- [27] J. Magidson, "The CHAID approach to segmentation modeling: Chi-squared automatic interaction detection," in *Advanced Methods of Marketing Research*, R. P. Bagozzi, Ed. Cambridge, MA: Basil Blackwell, 1994.
- [28] D. Lektzian and B. C. Prins, "Taming the Leviathan: Examining the impact of external threat on state strength," *J. Peace Research*, vol. 45, no. 5, pp. 613-31, 2008.
- [29] National Consortium for the Study of Terrorism and the Study of Terrorism: GTD1 and GTD2 [Online] Available: http://www.start.umd.edu/start/data/gtd/gtd1_and_gtd2.asp accessed 2/22/09.

SESSION

LATE PAPERS

Chair(s)

TBA

Mining Spreadsheet Complexity Data to Classify End User Developers

Stephen Hole, Duncan McPhee and Alex Lohfink

Abstract—End user computing is a phenomenon that has existed since desk top computers arrived in the professional workplace. Many studies have considered the concept of end user computing classification so the phenomenon can be understood, controlled, and nurtured. Spreadsheets are the major environment in which end users operate, developing complex models through in-built functions and macro code. The environment, whilst being extremely powerful, has a well documented history of costly errors. The more complex the application being modeled the greater the scope for disaster. This paper seeks to minimize these risks by proposing a data mining solution to classify end users, using complexity data that has been gathered by a software agent. Using test data it was possible to identify the developers of the most complex spreadsheets for in-depth error checking.

I. INTRODUCTION

End users have responded to the substantial efforts to empower them to create their own programs [1, 2]. Typical end user programming environments include web authoring tools, graphical languages, and spreadsheet systems [1]. Spreadsheet systems are the most common form of environment for end user programming [2]. Spreadsheet are an adaptable business tool used extensively in organizations and perform a vital role in the decision making process. They come supplied with a wide range of in-built functionality for mathematical, statistical and financial modeling. End users create spreadsheet models for many purposes including accounting, financial modeling, data analysis, rapid prototyping in engineering, and for teaching [2].

Using the in-built functionality, end users are producing ever more complex spreadsheet applications, often consisting of numerous worksheets that encompass hundreds and sometimes thousands of formulae [2]. End users often have no formal programming training and are more likely to make mistakes than professional programmers [3]. Also spreadsheet packages can be used to produce models that appear to be thoughtfully constructed but in reality are based on significant shortcomings [4].

Spreadsheet capabilities can be extended via the use of user-defined macro based functionality or third party

supplied add-ins. The power available to spreadsheet modelers is immense, as is the possibility of errors; the more complex the model the greater the likelihood of hidden errors.

To make the developers of spreadsheets more productive and technically competent requires both training and experience. To identify the training requirement necessitates detailed knowledge of the users' skill set and the functions and macro code they currently use in order to generate complexity data [5]. Over time, end users will create and develop many spreadsheet applications. Generating the spreadsheet complexity data from these applications requires a significant amount of storage that ideally is saved within a relational database environment. The saved data is rich with hidden information, and with the right transformations and tools, can be used to classify spreadsheet-based end user developers. Thus, the people who most require training can be identified. Many of these tools fall within the realm of data mining. Data mining is a term for a loose collection of statistical techniques or algorithms often used to predict future behavior based on data about past behavior. Data mining technology has a proven track record over a number of years. It is used to extract patterns of information from large datasets and as such, enhances business intelligence [6]. The classification process carried out by Govindarajulu [7] involved the use of on-line questionnaires. This process is superior to that of paper-based questionnaires in that the data received is in a machine readable format and immediately available for analysis. The process takes time out of the end users normal schedule and questions can be misinterpreted. A superior approach is to gather the data automatically via desktop tools. This creates an accurate picture of the end user and with data gathered on a continuous basis enables many snap-shots of the data to be captured. Multiple snap-shots over time gives the opportunity to measure the progress an end user has achieved.

To address these problems and minimize errors, we propose a data mining solution to classify end users using complexity data that has been gathered by a software agent. Wooldridge [8] defines an agent as “*a computer system situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives.*”

S. Hole is a Senior Lecturer at Swansea Metropolitan University, Swansea, SA1 6ED, UK (phone: 0044 1792 481152; fax: 0044 1792 481192; e-mail: stephen.hole@smu.ac.uk).

D. McPhee is a Senior Lecturer at the University of Glamorgan, Pontypridd, SA1 6ED, UK (phone: 0044 1443 483602; e-mail: dmcphoe@glam.ac.uk).

A. Lohfink is a Lecturer at the University of Glamorgan, Pontypridd, SA1 6ED, UK (phone: 0044 1443 483616; e-mail: alohfink@glam.ac.uk).

The experiment uses test data to identify the developers of

the most complex spreadsheets for in-depth error checking. Due to its ubiquity, this study concentrates on the Microsoft Excel family of spreadsheet products.

II. SPREADSHEET COMPLEXITY

Price, Waterhouse and Coopers (PWC), assert that spreadsheets are a fundamental tool in the functional areas of financial reporting, analysis of management information, and tracking and monitoring of workflow to support operational processes [9]. Spreadsheets vary greatly in their complexity and PWC differentiate this complexity with a three band categorization of low, moderate and high [9].

The *low* category consists of spreadsheets that serve to log data and information tracking systems.

The *moderate* category is typified by spreadsheets that perform simple calculations such as using formulae to total certain fields or calculate new values by multiplying two cells.

High complexity spreadsheets typically support complex calculations, valuations and modeling tools. These spreadsheets tend to use macros and multiple supporting spreadsheets where cells, values and individual spreadsheets are linked.

PWC identified a number of potential risks, errors and issues with spreadsheets [9]:

- “Complexity of the spreadsheet and calculations
- Purpose of the spreadsheet
- Number of spreadsheet users
- Type of potential input, logic and interface errors
- Size of the spreadsheet
- Degree of understanding and documentation of the spreadsheet requirements by the developer
- Uses of the spreadsheet output
- Frequency and extent of changes and modifications to the spreadsheet
- Development, developer (and training) and testing of the spreadsheet before it is utilised
- Logic errors in which inappropriate formulae are created and generate abnormal results”

In order to ensure that the users get the correct training for their needs, the complexity of the macro code and functions used has to be scrutinized. Software complexity is the investigation of what makes program code difficult for humans to comprehend [10]. There are several well-documented software complexity measures in the literature such as McCabe’s Cyclomatic Number [11], Halstead’s Programming Effort [12], and Oviedo’s Data Flow Complexity Measures [13]. The motivation of complexity measures is to gauge the correctness, effectiveness and clarity of software and to quantify the costs of testing, maintenance, development time, and number of errors [10, 14].

III. END USER CLASSIFICATION

Bullen *et al* [15] classify Excel developers into five general categories. The allocation is dependent on their knowledge of both Excel and the Excel macro language Visual Basic for Applications (VBA). The categories are Excel User, Excel Power User, VBA Developer, Excel Developer, and Professional Excel Developer. Table 1 summarizes the categories.

To provide a wide ranging classification, the schemes of PWC and Bullen *et al* are combined to provide the classification schema as shown in Table 2.

Table 1 Excel Developer Classification

Category Name	Description
Excel User	In the first instance uses Excel for storing lists and repetitive calculations. With experience inclusion of worksheet functions, pivot tables and charts.
Excel Power User	Wide understanding of Excel’s functionality. Creates complex spreadsheets for own use and helps develop and debug colleagues’ spreadsheets. Occasional use of VBA code from macro recorder or web sites.
VBA Developer	Extensive use of VBA. Typically they are power users who started to learn VBA (often too early) or Visual Basic developers who switched to VBA development but often lack sufficient knowledge of Excel to make full use of its features
Excel Developer	Constructs efficient and maintainable applications by making the best use of Excel’s in-built functionality, augmented by VBA when appropriate. Excel developers are confident at developing Excel based applications for both their colleagues and as part of development teams.
Professional Excel Developer	Designs and develops Excel-based applications and utilities that are robust, fast, easy to use, maintainable and secure. Excel forms the core of their solutions but they can also make use of other languages and applications that are appropriate.

Table 2 Combined Spreadsheet Developer Classifications

Level	Description
Novice User	Recording and storing of data.
Basic User	Use of functions for simple calculations
Experienced User	Pivot tables and charts
Comprehensive User	Creates complex spreadsheets. Occasional use of Visual Basic for Applications (VBA) from macro recorder, books etc.
VBA Developer	Extensive use of Visual Basic for Applications (VBA).
Power User	Constructs efficient and maintainable applications by making use of Excel’s in-built functionality, augmented by Visual Basic for Applications (VBA) when appropriate.
Professional Developer	Designs and develops Excel-based applications that are robust, fast, easy to use, maintainable and secure. Can make use of languages other than Visual Basic for Applications (VBA).

IV. DATA MINING ENVIRONMENT

With the release of SQL Server 2005, Analysis Services and several other tools became an integrated part of the database management system. These tools are [16]:

- The relational engine (RDBMS) to manage and store the dimensional data warehouse database.
- Integration Services to create the extract, transformation and load system (ETL).
- Analysis Services
 - An OLAP (Online Analytical Processing) database to support user queries.
 - Data mining to develop statistical data mining models.
- Reporting Services to build predefined reports.
- Development and management tools
 - SQL Server Business Intelligence Development Studio.
 - SQL Server Management Studio

The toolset provides a complete environment in which to operate. The tools are designed to complete data warehouse/business intelligence systems without the need of third party applications, giving confidence that all the components work together correctly. The Business Intelligence Development Studio is integrated into the Visual Studio Development Suite giving the Visual Basic or C# developer a familiar environment in which to operate.

Analysis Services supports a powerful implementation of the MDX formula language that allows for a range of capabilities from simple calculated ratios to complex financial calculations. It is simple enough for small uncomplicated organizations and powerful enough to meet the needs of large complex organisations [17].

V. MINING MODELS

Four mining models were used in this study: Association Rules, Clustering, Naïve Bayes, and Neural Networks.

A. Association rules

This mining algorithm is used to determine the affinity between objects and as such display items that naturally fit together [18]. This type of data mining algorithm is useful in recommendation engines. A typical recommendation engine proposes products to customers based on their buying habits. Examples can be seen on Amazon and eBay web sites and with grocery store loyalty cards.

The algorithm is used to examine a dataset to identify items that appear together in an end user profile. The Microsoft Association Algorithm uses two parameters named support and probability. These are used to describe the item-sets and the rules that are generated. If X and Y are two products that could be in a shopping basket/cart, the support parameter is the count of occurrences in the dataset that contain the combination of items X and Y. The probability parameter is the percentage of instances in the dataset that contain both X and Y. The resultant rules are used to predict the existence of an attribute value based on the presence of other specific values within specified fields

[19].

B. Clustering

Clustering analysis works by dividing a data set into groups (clusters) that are either meaningful, useful or both [20]. A cluster gathers together data items that are similar to one another and are dissimilar to data items in other clusters. Cluster analysis can be used to discover distribution patterns and correlations among data attributes [21].

C. Naïve Bayes

Naïve Bayes is a simple form of the Bayes classifier. It can be used to predict class membership probabilities such as determining the likelihood that a sample item belongs to an identified class. Bayesian classifiers display high accuracy and speed when used in conjunction with large databases [21].

D. Neural Networks

Neural Networks are structured to simulate human biological neural systems [20]. In simple terms they consist of a set of connected and weighted input/output units. The network learns by adjusting the weights in order to be able to predict the correct class labels for the input data set. Neural Networks exhibit a high tolerance to noisy data and can be used to classify patterns for which they have not been trained [21].

VI. METHODOLOGY

The method to classify end user spreadsheet developers involves three distinct phases: data gathering, transformation, and prediction (see Figure 1). The data gathering phase has two processes: a File Watching Agent (FWA) and spreadsheet analyzer. The transformation phase has a single process to transform the gathered data into a form that can be easily data mined. The final phase of prediction mines the transformed data to forecast the likely classification of each end user developer according to the classes described in Table 2.

A. Data Gathering

The FWA runs as an autonomous Windows service constantly monitoring a specified disc location for the saving of a spreadsheet file that is under development. When saving occurs, the FWA stores the machine name, user name, and file name (including path details) to a centralized SQL Server database table named 'WorkingSpreadsheets'. A software application running on the desktop computer of the end user instructs the analyzer entity to process all unique working spreadsheets for the session. These are then analyzed in terms of functions used, charts present, program code constructs, together with file properties 'Author' and 'Last Saved By' creating an entry in the 'tblExcelFileData' table. This analysis represents the raw data and before data mining commences it is necessary for it to undergo a three stage transformation process.

B. Transformation

An extraction, transformation and load (ETL) system is used to transform the data and involves three stored procedures.

spWorkbookStatsETL1: A spreadsheet that has been developed over multiple sessions will have been analyzed for complexity several times – one for each monitored session. The analyzed data is written to the 'tblExcelFileData' table. This stored procedure is used to group together the multiple table entries for each unique spreadsheet to find the maximum value for each individual complexity measure, i.e. Statistical Functions. For every spreadsheet under development a single entry is made in the 'WorkbookStatistics' table (see Listing 1).

spMiningAuthorStatisticsETL2: An author may be responsible for multiple applications created with spreadsheet technology. This stored procedure brings together in a single profile the history of all complexity measures associated with a single author's development activities. This is achieved by summarizing the values for every spreadsheet for which each user is both author *and* the last user to save the file (see Listing 2).

spMiningTransformAuthorStatisticsETL3: The numerical values from the summarization are then transformed into one of the following categories: none, low, medium or high (see Listing 3). The data is now transformed and ready for mining to identify the classification of each end user.

Listing 1 Workbook Statistics Stored Procedure

```
USE [ExcelComplexity]
CREATE Procedure spWorkBookStatsETL1
AS
-- Empty the Table before processing
DELETE FROM WorkBookStatistics
--Insert the new records from the Agent
generated data
INSERT INTO WorkBookStatistics
SELECT Author, DocumentName, DocumentPath,
-- Functions
MAX (GDateAndTime) AS MDateAndTime,
MAX (GLookAndRef) AS MLookUpAndRef,
MAX (GDatabase) AS MDatabase,
MAX (GText) AS MText,
MAX (GLogical) AS MLogical,
MAX (GInfo) AS MInfo,
MAX (GFinancial) AS MFinancial,
MAX (GMathAndTrig) AS MMathAndTrig,
MAX (GStatistics) AS MStatistics,
MAX (GFrequent) AS MFrequent,
-- Code Structures
MAX (CPrivateSub) AS MPrivateSub,
MAX (CPublicSub) AS MPublicSub,
MAX (CPrivateFunction) AS MPrivateFunc,
MAX (CPublicFunction) AS MPublicFunc,
MAX (CAsNew) AS MAsNew,
MAX (CForEach) AS MForEach,
MAX (CDoLoop) AS MDoLoop,
--Others
MAX (CellArith) AS MCellArith,
MAX (ChartCount) AS MChartCount
FROM tblExcelFileData
GROUP BY Author, DocumentName, DocumentPath
```

C. Prediction

All data mining algorithms have their strengths and weaknesses and their prediction precision depends upon the situation in which they are used. The mining model that can deal with all data better than other methods is yet to be found [22]. To find the best performing algorithm for a given dataset requires both experimentation and analysis of the results. To demonstrate this experimentation and analysis a small dataset and test data was applied to the following Microsoft data mining algorithms: - Naïve Bayes, Association Rules, Neural Network and Clustering.

Listing 2 Author Statistics Stored Procedure

```
CREATE PROCEDURE
[dbo].[spMiningAuthorStatisticsETL2]
AS
BEGIN
DELETE FROM MiningAuthorStatistics
--Insert the new records from the Agent
generated data
INSERT INTO MiningAuthorStatistics (
Author, InBuiltFunctions, Subroutines,
Functions, AsNew, ForEach, DoLoops,
CellArithmetic, ChartCount)
SELECT Author,
-- Functions
SUM (DateAndTime) + SUM (LookUpAndRef) +
SUM ([Database]) + SUM ([Text]) +
SUM (Logical) + SUM (Info) + SUM (Financial) +
SUM (MathAndTrig) + SUM ([Statistics]) +
SUM (Frequent) AS SInBuiltFunctions,
-- Code Structures
SUM (PrivateSub) + SUM (PublicSub) AS
SSubroutines,
SUM (PrivateFunction) + SUM (PublicFunction)
AS SFunctions,
SUM (AsNew) AS SAsNew,
SUM (ForEach) AS SForEach,
SUM (DoLoop) AS SDoLoop,
SUM (CellArithmetic) AS SCellArith,
SUM (ChartCount) AS SChartCount
FROM WorkBookStatistics
GROUP BY Author
END
```

Listing 3 Transform Stored Procedure

```
USE [ExcelComplexity]
CREATE PROCEDURE
spTransformAuthorStatisticsETL3
AS
BEGIN
-- Empty the Table before processing
DELETE FROM MiningAuthorCharacteristics
INSERT INTO MiningAuthorCharacteristics (
Author, InBuiltFunctions, Functions,
Subroutines, AsNew, ForEach, DoLoops,
CellArithmetic, ChartCount)
Select Author, InBuiltFunctions =
CASE
WHEN InBuiltFunctions = 0 THEN 'None'
WHEN InBuiltFunctions < 100 THEN 'Low'
WHEN InBuiltFunctions BETWEEN 100 And
250 THEN 'Medium'
ELSE 'High'
End,
Functions =
-----
FROM MiningAuthorStatistics
END
```

VII. IMPLEMENTATION

A sample of 859 spreadsheets from 35 authors was supplied for analysis. The majority of the authors are students studying on the European Computer Driving License course. Forty of the spreadsheets were unusable because of password protection and a further 4 failed to be analyzed. The sample for mining was reduced to 815 spreadsheets from 28 authors. The data set is in three parts (see Table 3): author details, Excel functions, and programming constructs.

A training set of data containing 15 records was used to teach each model, this will need to be significantly enlarged for full scale trials. To test the validity of each model the training data is used to predict a class label (DeveloperType) for the spreadsheet author data set. The predicted class label is then compared against an actual value assigned by the project team. The training and test data could have been derived from discretised values obtained by a spreadsheet complexity agent.

Table 3 Data Set Description

Part Name	Attribute Name	Attribute Description
Author Details	Author	Name of the spreadsheet author who created the spreadsheet.
Excel Functions	InBuiltFunctions	A combination of all the Excel functions from areas such as Statistics, Database, Text, Logical, etc.
	CellArithmetic	Excel cell arithmetic that includes simple arithmetic operator but not involving any in built functions such as Sum, StDev, etc.
Programming Constructs	Functions	Author created functions.
	Procedures	Author created sub procedures.
	AsNew	Use of classes and objects.
	DoLoops	Use of any form of iteration based code.
	For Each	Looping based on a collection of objects.

A. Results

The Clustering algorithm performed poorly only predicting the correct classification for 8 out of 28 authors. Association rules provided a better fit with 19 successes. The Neural Network achieved 21 successes. The best match by far was provided by Naïve Bayes with 25 successful author classifications.

The results are plotted on a Lift Chart that can be viewed in Figure 2. "A lift chart plots the results of prediction queries from a testing dataset against known values for the predictable column that exist in the dataset. The chart displays the results of the mining model, together with a representation of the results that an ideal model would produce, and a representation of the results of random guessing.", [19].

The Naïve Bayes Dependency Association Chart (Figure 3) displays the dependencies between the input attributes and the predictable attributes in the model. It clearly shows that each attribute in the data set predicts the Developer Type class label.

VIII. CONCLUSION AND FUTURE WORK

The data gathering element of the application has been successfully launched on three test machines and is gathering data as expected. The ETL system has successfully transformed the raw complexity data to a state that is fit for data mining. The Microsoft Analysis Services has provided a suitable data mining environment that has led to the sample end users being successfully categorized. Of the four algorithms used the best classifier is the Naïve Bayes.

A substantial number of spreadsheets have been successfully analyzed from 28 authors. The next stage is to identify more authors with a greater diversity of backgrounds, preferably those with greater experience of using VBA code to extend their spreadsheet applications. The data gathering application will then be installed on their computers and the extra data gathered can be used to test the robustness of the system. The end users who agree to participate in the study will be given a questionnaire to complete that will seek to categorize users in terms of their software development activities. From the questionnaire data the users will be manually classified. A comparison can then be made between both data gathering mechanisms and how well the data mining can classify against the manual method.

REFERENCES

- [1] B. Myers, A. , M. Burnett, and M. B. Rosson, "End users creating effective software," in *CHI '05 extended abstracts on Human factors in computing systems* Portland, Oregon, USA: ACM, 2005.
- [2] T. Antoniu, P. Steckler, A., S. Krishnamurthi, E. Neuwirth, and M. Felleisen, "Validating the Unit Correctness of Spreadsheet Programs," in *Proceedings of the 26th International Conference on Software Engineering: IEEE Computer Society*, 2004.

- [3] R. Panko, R. , "What we know about spreadsheet errors," *J. End User Comput.*, vol. 10, pp. 15-21, 1998.
- [4] T. Babbitt, G., D. Galletta, F., and B. LopesAlexandre, "Influencing the success of spreadsheet development by novice users," in *Proceedings of the international conference on Information systems* Helsinki, Finland: Association for Information Systems, 1998.
- [5] S. Hole, D. McPhee, and M. Mhereeg, "Developing a Windows Service Agent to Analyze Spreadsheet Macro and Function Complexity," in *4th International Conference on Computer Science and Information Systems* Athens, Greece: Athens Institute for Education and Research, 2008.
- [6] J. Rajan and V. Saravanan, "A Framework of an Automated Data Mining System Using Autonomous Intelligent Agents," in *Proceedings of the 2008 International Conference on Computer Science and Information Technology - Volume 00*: IEEE Computer Society, 2008.
- [7] C. Govindarajulu, "End users: who are they?," *Communications of the ACM*, vol. 46, pp. 152-159, 2003.
- [8] M. Wooldridge, *An Introduction to MultiAgent Systems*. Chichester: John Wiley & Sons, 2002.
- [9] W. Price, Coopers, "The Use of Spreadsheets: Considerations for Section 404 of the Sarbanes-Oxley Act," Price, Waterhouse, Coopers 2004.
- [10] M. B. O'Neal, "An empirical study of three common software complexity measures," in *Proceedings of the 1993 ACM/SIGAPP symposium on Applied computing: states of the art and practice* Indianapolis, Indiana, United States: ACM Press, 1993.
- [11] T. J. McCabe, "A Complexity Measure," *IEEE Transactions on Software Engineering*, vol. SE-2, pp. 308-320, December 1976 1976.
- [12] M. H. Halstead, *Elements of Software Science*. New York: Elsevier North-Holland, 1977.
- [13] E. I. Oveido, "Control Flow, Data and Program Complexity," in *Proceedings of the IEEE COMPSAC*, Chicago, Illinois, 1980, pp. 146-152.
- [14] S. Misra and A. K. Misra, "Evaluation and comparison of cognitive complexity measure." vol. 32: ACM Press, 2007, pp. 1-5.
- [15] S. Bullen, R. Bovey, and J. Green, *Professional Excel Development: The Definitive Guide to Developing Applications Using Microsoft Excel and VBA*. Upper Saddle River: Addison-Wesley, 2005.
- [16] J. Mundy, W. Thornthwaite, and R. Kimball, *The Microsoft Data Warehouse Toolkit with SQL Server 2005 and the Microsoft Business Intelligence Toolset*. Indianapolis: Wiley, 2006.
- [17] R. Jacobson and S. Misner, *Microsoft SQL Server 2005 Analysis Services Step by Step*. Redmond: Microsoft Press, 2006.
- [18] M. J. A. Berry and G. S. Linoff, *Mastering Data Mining: The Art and Science of Customer Relationship Management*. New York: Wiley, 2000.
- [19] Microsoft, "SQL Server Books Online," 2008.
- [20] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Boston: Addison Wesley, 2006.
- [21] J. Han and M. Kamber, *Data Mining Concepts and Techniques*. San Diego: Morgan Kaufmann Publishers, 2001.
- [22] G. Wang, C. Zhang, and L. Huang, "A Study of Classification Algorithm for Data Mining Based on Hybrid Intelligent Systems," in *Proceedings of the 2008 Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing - Volume 00*: IEEE Computer Society, 2008.

Figure 1 End User Spreadsheet Developer Classification Process.

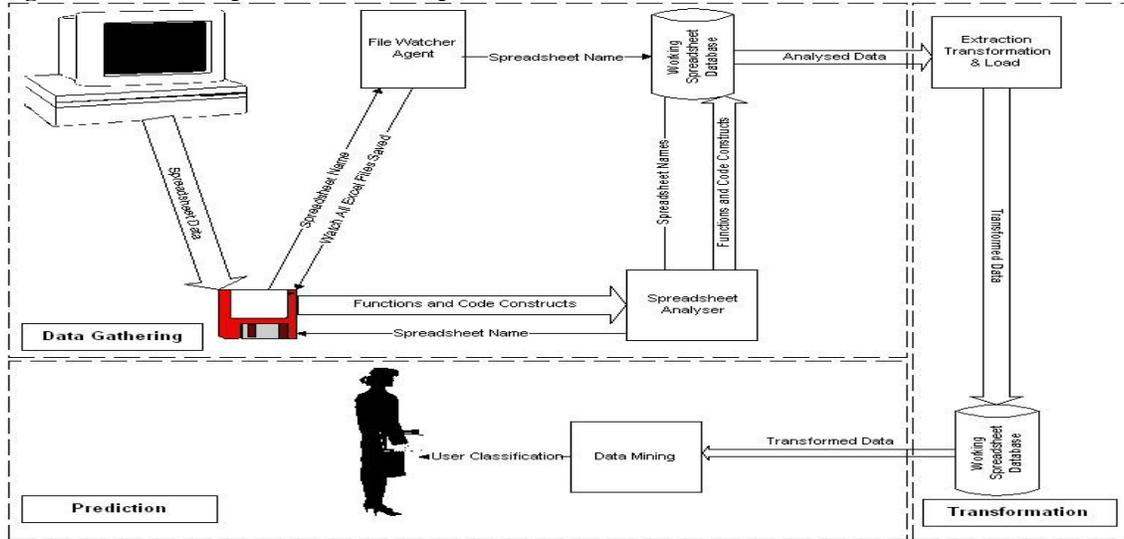


Figure 2 Lift Chart

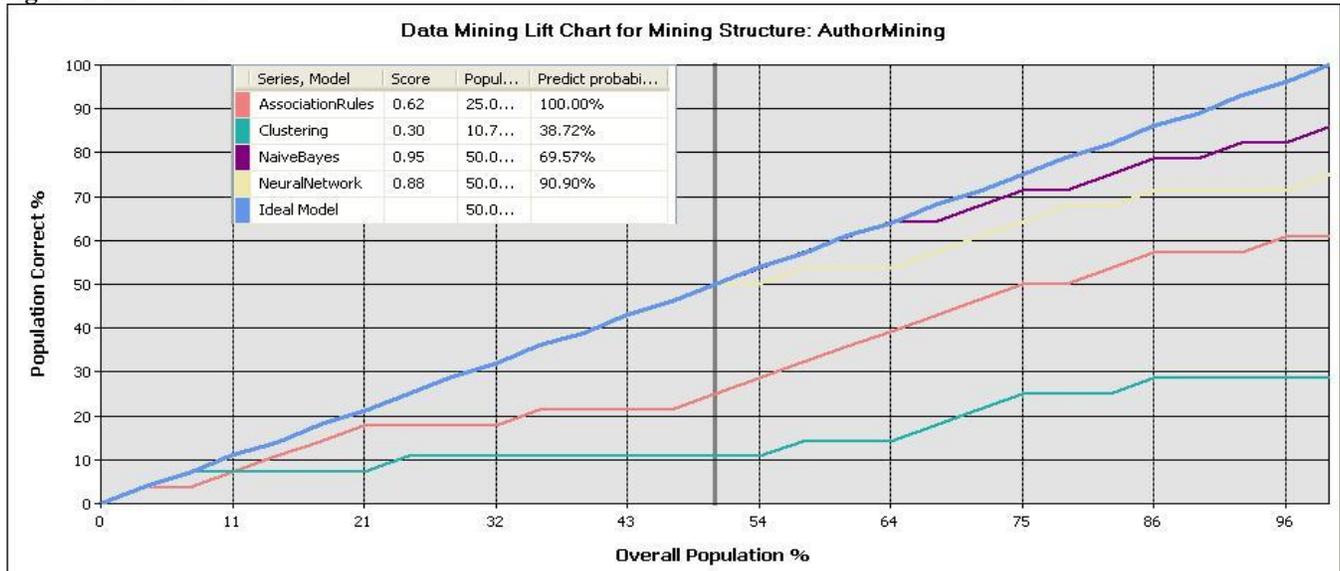
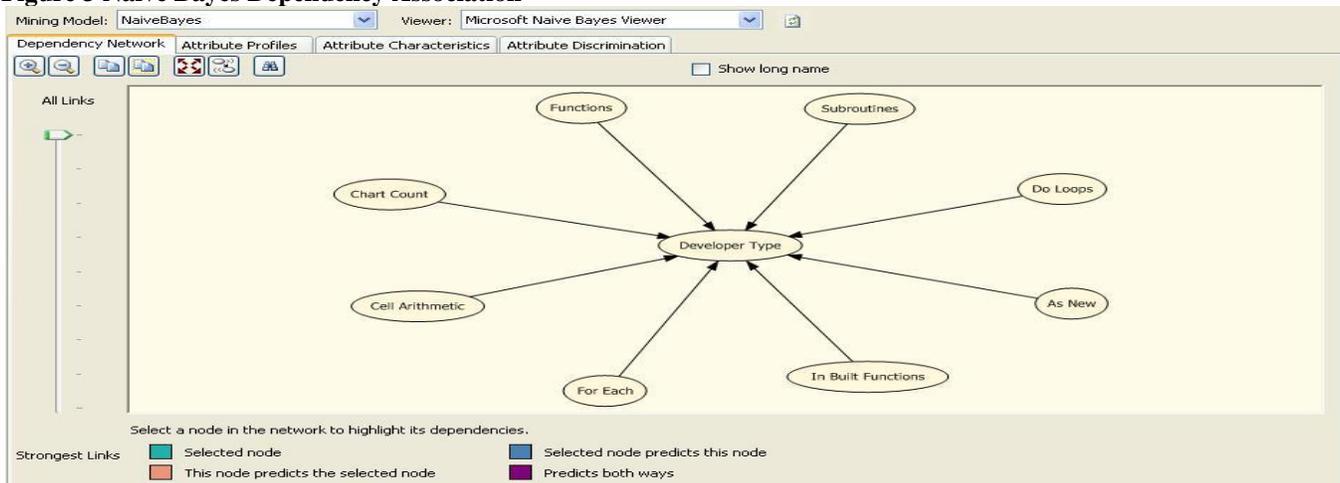


Figure 3 Naïve Bayes Dependency Association



EXTREME LEARNING MACHINE CLASSIFIER CAPABILITIES IN SOLVING MULTICATEGORY DISEASE CLASSIFICATION PROBLEMS

Emad A. El-Sebakhy[†], Tarek R. Sheltami*, Ognian Asparouhov[‡], Krassimir Latinski[‡], and Zeesham Rasheed H**

[†]MEDai, Inc. an Elsevier Company, Millenia Park One, 4901 Vineland Road, Suite 450, Orlando, Florida 32811

*Computer Engineering Department, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

**Information and Computer Science, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

Abstract— This paper presents recently introduced learning algorithm called extreme learning machine (ELM) for single-hidden layer feed-forward neural networks (SLFNs) which randomly chooses hidden nodes and analytically determines the output weights of SLFNs. ELM avoids problems like local minima, improper learning rate and over fitting commonly faced by iterative learning methods and completes the training very fast. We have evaluated the multicategory classification performance of ELM on five different datasets related to bioinformatics namely, the Breast Cancer Wisconsin data set, the Pima data set, the Heart-Statlog data set, the Hepatitis data set and the Hypothyroid data set. A detailed analysis of different activation functions with varying number of neurons is also carried out which concludes that algebraic sigmoid function outperforms all other activation functions on these datasets. The evaluation results indicate that ELM produces better classification accuracy with reduced training time and implementation complexity compared to earlier implemented models.

Index Terms— Extreme Learning Machine, Bioinformatics, Classification, Machine Learning, Bayesian Network, Support Vector Machine

I. INTRODUCTION

In the feature based classification area for the diagnosis of different diseases types, binary classification problems have been more extensively studied but multi category classification problems are still in focus. Studies also indicate that direct multi-class classification is much more difficult than binary classification and the classification accuracy may drop dramatically when the number of classes increases [1].

Instead of directly dealing with multi-category problems, many classification methods actually use some combination of binary classifiers on a One-Versus-All (OVA) or One-Versus-One (OVO) comparison basis [2,3,4,5], but this way of implementation costs greater computational burden, longer training time as well as higher system complexity. If we consider the well-known Support Vector Machine (SVM) as an example; binary classifier tries to map the data from a lower-dimensional input space to a higher-dimensional feature space to make the data linearly separable into two classes [6]. When using the one-versus-all approach to make binary classifiers applicable to multi category problems where number of classes are more than two; binary classifiers should

be built for SVM to distinguish one class from all the rest of the classes. Thus, it can be seen that, when the number of classes c increases, the complexity of the overall classifier also increases. The authors in [2] have used the SVM-OVA as a multi class classifier and results were compared with other methods. Artificial neural network (ANN) methods provide an attractive alternative to the above approach for a direct multi category classification problem [3,5,7]. Neural networks map the input data into different classes directly with one network; also non linear features can also be accommodated using where we need predictions [6]. However, conventional Neural networks usually produce lower classification accuracy than SVM [8].

To overcome the computational time along with classification accuracy problems, Huang et al. in [9] proposes a learning algorithm called extreme learning machine (ELM) for single hidden layer feed-forward neural networks (SLFNs) which randomly selected the input weights and analytically determines the output weights of SLFNs. They stated that algorithm tends to provide the best generalization performance at extremely fast learning speed. We have evaluated the multicategory classification performance of ELM on five different datasets related to bioinformatics namely, the Breast Cancer Wisconsin data, the Pima data, the Heart-Statlog data, the Hepatitis data and the Hypothyroid data, respectively. We then compared the performance of ELM with the results stated in [12].

This paper is organized as follows. Section II contains related work. Section III gives the brief overview of ELM. Section IV talks about the investigated data. Performance evaluation is presented in Section V. Discussions and conclusions are drawn in Section VI.

II. RELATED WORK

Many classifiers were developed in the last decade exploring various fields with the help of computer science. In fact, most of the research work found in the literature related to disease classification either make use of statistical models or artificial neural networks [3,5,7], though some recent work have been done using some other artificial intelligence techniques such as Bayesian Belief Networks (BBN) [12].

Some optimization techniques were also used such as Hill Climbing BN, Laplace smoothing, C4.5L to improve the ranking of classifiers [13]. SVM, When using the one-versus-all approach to make binary classifiers applicable to multi category problems, it can be seen that, when the number of classes c increases, the complexity of the overall classifier also increases. So the system becomes more complex and requires extra computations [6]. As far as Neural Networks are concerned, they usually produce less low classification accuracy and needs much training time while updating the input output weights [3,5,6,8].

Recently, Huang et al [9,10,11] have proposed a new learning algorithm called the *Extreme Learning Machine* (ELM) for *single-hidden layer feedforward neural networks* (SLFNs). In ELM, one may randomly choose (according to any continuous sampling distribution) and fix all the hidden node parameters and then analytically determine the output weights of SLFNs [9]. After the hidden nodes parameters are chosen randomly, SLFN can be considered as a linear system and the output weights can be analytically determined through a generalized inverse operation of the hidden layer output matrices. Studies have shown [9] that ELM has good generalization performance and can be implemented easily. Many nonlinear activation functions can be used in ELM, like sigmoid, algebraic sigmoid, sine, hard limit, radial basis functions [10,11], and complex activation functions [12]. The empirical results show that ELM outperforms other classifiers in terms of accuracy as well as computational time.

III. EXTREME LEARNING MACHINE

Let us first define the standard SLFN (single-hidden layer feed-forward neural networks). If we have N samples (x_i, t_i) , where $x_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in \mathbb{R}^n$ and $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbb{R}^m$, then the standard SLFN with \tilde{N} hidden neurons and activation function $g(x)$ is defined as:

$$\sum_{i=1}^{\tilde{N}} \beta_i g(w_i \cdot x_j + b_i) = o_j, j = 1, \dots, N,$$

where $w_i = [w_{i1}, w_{i2}, \dots, w_{in}]^T$ is the weight vector that connects the i th hidden neuron and the input neurons, $\beta_i = [\beta_{i1}, \beta_{i2}, \dots, \beta_{im}]^T$ is the weight vector that connects the i th neuron and the output neurons, and b_i is the threshold of the i th hidden neuron. The “ \cdot ” in $w_i \cdot x_j$ means the inner product of w_i and x_j .

SLFN aims to minimize the difference between o_j and t_j . This can be expressed mathematically as:

$$\sum_{i=1}^{\tilde{N}} \beta_i g(w_i \cdot x_j + b_i) = t_j, j = 1, \dots, N,$$

or, more compactly in matrix format, that is,

$$H \beta = T,$$

where the matrix $\mathbf{H}(w_1, \dots, w_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, x_1, \dots, x_N)$ is defined as:

$$\mathbf{H}(w_1, \dots, w_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, x_1, \dots, x_N) = \begin{bmatrix} g(w_1 \cdot x_1 + b_1) & \dots & g(w_{\tilde{N}} \cdot x_N + b_{\tilde{N}}) \\ \vdots & \dots & \vdots \\ g(w_1 \cdot x_N + b_1) & \dots & g(w_{\tilde{N}} \cdot x_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} \quad \text{and} \quad T = \begin{bmatrix} T_1^T \\ \vdots \\ T_{\tilde{N}}^T \end{bmatrix}_{N \times m}$$

As proposed by Huang and Babri (1998), H is called the neural network output matrix.

The ELM learning algorithm based on the provided information within Huang et al. (2004) can be expressed as follow:

- Given $N = \{(x_i, t_i) | x_i \in \mathbb{R}^n, t_i \in \mathbb{R}^m, i = 1, \dots, N\}$ training set; activation function $g(x)$; and \tilde{N} of hidden neurons; then assign random value to the input weight, w_i and the bias b_i , for $i = 1, \dots, \tilde{N}$.
- Find the hidden layer output matrix H .
- Find the output weight β : $\beta = H^\dagger T$; where H^\dagger is the Moore-Penrose generalized inverse [14] of the hidden layer output matrix H ; H ; and T are defined in the same way they were defined in the single layer feedforward neural networks (SLFN) specified above.

IV. DATA ACQUISITION AND IMPLEMENTATION PROCESS

To implement the extreme learning intelligence system paradigm and the most common data mining schemes, namely, statistical regression, multilayer perceptron neural networks, and support vector machines, we have designed our own programming codes in MATLAB with the required parameters. We utilized four different real-world databases from distinct scientific medical sources; such as, University of California Irvine (UCI) machine learning repository. For the sake of simplicity, we did not include the background of the data, but the reader can know the details description of such database by taking a look at the first table in the next section.

During this study, we use the cross-validation process for all the utilized modeling schemes, and then the available dataset has to be divided into two sets, the training and the testing set. This division is usually done by selecting the data

points in a random order. The training set is used to build up the ELM network model while the testing set is used to

Table 2: Experimental Results on Classification Accuracy

Data Set	ELM	SVM	FBC	C4.5L
Wisconsin Breast	98.15%	95.95%	97.25%	91.86%
Pima Diabetes	79.61%	74.43%	74.85%	73.88%
Heart -statlog	88.44%	82.48%	83.81%	79.85%
Hepatitis	90.34%	80.86%	86.90%	81.50%
Hypothyroid	95.39%	93.48%	93.16%	93.24%

ELM: Extreme Learning Machine, SVM: Support Vector Machine, FBC: Fully Bayesian Network Classifier, C4.5L: The traditional decision trees induction algorithm C4.5 using Laplace smoothing

evaluate the predictive capability of that model. The predictive performance of the network is then assessed using cross validation parameters.

In the implementation process, the multilayer perceptron feed-forward network (MLPFFN) is used here with pure linear and sigmoid activation neuron functions, with two or three hidden layers. The input datasets were normalized to interval of [0,1] using equation (11), as given below.

$$x_i^{(new)} = \frac{(x_i^{(old)} - \min(x_i))}{(\max(x_i) - \min(x_i))}; \quad i = 1, \dots, n;$$

This led to a set of input variables that are independent of their measurement units. The normalized inputs are then used for both statistical regression and functional network techniques. Through the implementation, a stratified sampling technique is used to make sure that the same pattern as in the original data is used. In addition, different quality measures can be used to judge the performance of the models. The most common statistical quality measures can be written in mathematical formulae as follows:

Root Mean Squares Error: Measures the data dispersion around zero deviation:

$$RMSE = \left[\frac{1}{n} \sum_{i=1}^n E_i^2 \right]^{1/2},$$

where E_i is a relative deviation of an estimated value from an experimental input data sets.

$$E_i = \left[\frac{(y)_{exp} - (y)_{est}}{(y)_{exp}} \right] \times 100; \quad i=1(1)n;$$

Correlation Coefficient: It represents the degree of success in reducing the standard deviation by regression analysis, defined as:

$$r = \sqrt{1 - \frac{\sum_{i=1}^n [(y)_{exp} - (y)_{est}]^2}{\sum_{i=1}^n [(y)_{exp} - \bar{y}]^2}}, \quad \text{where } \bar{y} = \frac{1}{n} \sum_{i=1}^n [$$

The Mean Absolute Percentage Error: To compare and evaluate the accuracy of the estimators, the mean absolute percentage error (MAPE) of the forecasted values with

respect to the actual amount of the development effort-hours (EFH) for each project was calculated:

$$MAPE = \frac{100}{n} \sum_{i=1}^n \left[\frac{|(V)_{est} - (V)_{act}|}{(V)_{act}} \right],$$

where $(V)_{est}$ is the forecasted value of EFH and $(V)_{act}$ is the actual EFH consumed by the project; n is the number of cases used in the test sample.

V. DATA DESCRIPTION

In our experiment, we use four types of datasets selected from UCI machine learning repository namely the Breast Cancer Wisconsin data set, the Pima data set, the Heart-Statlog data set and the Hepatitis data set. A brief description of these datasets is in Table 1.

Table 1: Description of Datasets

Dataset	Size	# of Att	Missing	Class
Wisconsin Breast	699	9	Yes	2
Pima Diabetes	768	8	No	2
Heart -statlog	270	13	No	2
Hepatitis	155	19	Yes	2

VI. EXPERIMENTAL SETUP AND RESULTS

A. Activation Functions

The activation function two sub-functions: the combination function and the transfer function. The combination function commonly uses the "standard weighted sum" that is the summation of the input attribute values multiplied by the weights that have been assigned to those attributes, to calculate a value to be passed on to the transfer function. The transfer function applies either a linear or non-linear transformation to the value passed to it by the combination function. The "hidden layer" then employs this transfer function in moving

data to the output nodes.

In our experiment, we used many activation function like sin, tan, piecewise linear, hyperbolic tangent, logistic sigmoid, algebraic sigmoid and analyzed their effects on accuracy. Among all, algebraic sigmoid activation function gives high precision and accuracy. It is defined as

$$f(x) = \frac{x}{\sqrt{1+x^2}}$$

This proves the significance of non linear sigmoid functions. If the transfer function were linear, each of the neuronal inputs would get multiplied by the same proportion during training. This could cause the entire system to drift during training runs. That is, the system may lose outputs it has already tracked while attempting to track new outputs. A non-linearity in the

in Table 2. The performance measure is based on Confusion Matrix showing number of correct and incorrect classified samples shown in Table 3. Best cases are also represented in Table 4.

A. Receiver Operating Characteristics (ROC)

Receiver operating characteristics graph are very useful in organizing classifiers and visualizing their performance. We performed another experiment reflecting how ELM performs with different number of neurons. By calculating true positive rate and false positive rate with different number of hidden neurons used in ELM, we generated an ROC graph showing discrete classifiers. The graph is given in Figure 1 and 2. Figure 1 stated the results of only four different configurations of neurons while Figure 2 is generated on the basis of 50

Table 3: Confusion Matrix

Data Set		Class 1	Class 2	Class 3	Class 4
Wisconsin Breast	Class 1	309	7	N/A	N/A
	Class 2	14	152	N/A	N/A
Pima Diabetes	Class 1	131	17	N/A	N/A
	Class 2	32	49	N/A	N/A
Heart -statlog	Class 1	94	12	N/A	N/A
	Class 2	15	68	N/A	N/A
Hepatitis	Class 1	17	5	N/A	N/A
	Class 2	1	88	N/A	N/A
Hypothyroid	Class 1	918	6	8	11
	Class 2	6	923	7	7
	Class 3	8	11	915	9
	Class 4	15	6	12	910

system helps to isolate specific input pathways and for the same reason, our experiment shows best classification accuracy when algebraic sigmoid is used as an activation function.

B. Classification Accuracy

The performance comparison of the proposed ELM algorithm and many other popular algorithms has been conducted for many real medical diagnosis problems data sets. We tried several activation functions like sigmoid, algebraic sigmoid, sine and after analyzing each of them, we come to conclusion that algebraic sigmoid is supposed to be the best activation function for such types of data sets. As far as missing values in the data sets are concerned, they are processed using means and modes mechanism.

We conducted our experiments based on MATLAB platform and we utilized five different types of datasets each of them have different number of attributes and classes. 70% and 30% stratified samples are chosen for several numbers of trials. Each data set is trained and tested using different machine learning techniques like Fully Bayesian Networks Classifier, Decision Trees C4.5L, SVM and ELM. The classification accuracy is given as an average of several trials which is given

iterations, every time increasing the number of neurons.

Table 4: Best Case on Classification Accuracy

Dataset	Accuracy
Wisconsin Breast	99.20 %
Pima Diabetes	82.10 %
Heart -statlog	91.36 %
Hepatitis	93.69 %
Hypothyroid	97.15 %

ELM Best Case with 25 Neurons

In this paper, the notion behind this ROC curve is that deciding the number of hidden neurons in layers is a very important part of deciding your overall network architecture. Though these layers do not directly interact with the external environment these layers have a tremendous influence on the final output. Both the number of hidden layers and number of neurons in each of these hidden layers must be considered. Using too few neurons in the hidden layers will result in something called under fitting. Under fitting occurs when there are too few neurons in the hidden layers to adequately detect

the signals in a complicated data set. Also too many neurons in the hidden layers can result in over fitting. Over fitting occurs when the neural network has so much information processing capacity that the limited amount of information contained in the training set is not enough to train all of the neurons in the hidden layers

As from the ROC theory, the any classifier that appears in the upper left regions performs best and worst performance lies in lower right region. So we can see that the ELM performs best when the number of hidden neurons equals 25. Also experiment with 5 neurons gives worst performance and increasing the number of neurons from 25 onwards does not show any improvement but degrades the classification rate. Therefore we proved that the number of hidden neurons must be balanced to avoid under fitting and overfitting.

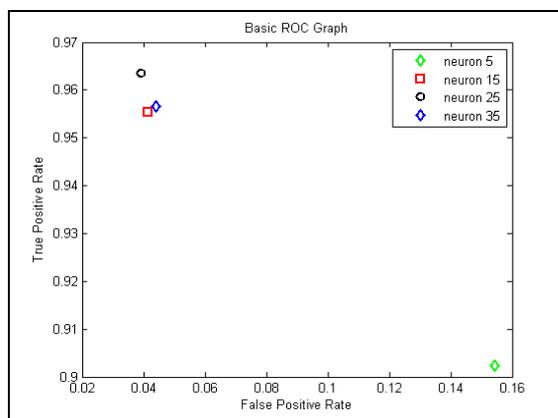


Figure 1: ROC Discrete Classifier Graph of Wisconsin Breast Dataset

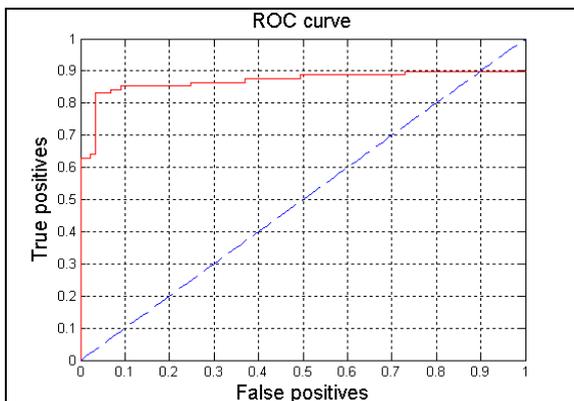


Figure 2: ROC Continuous Classifier Graph of Wisconsin Breast Dataset

VII. CONCLUSION

In this paper, we evaluate the classification rate of different machine learning techniques and the results shows that ELM outperforms all other algorithms with highest classification rate. The results in Table 4 which shows the best ELM performance proves that it has better generalization and can be used in high dimensional applications in future. For activation function, we proposed algebraic sigmoid function which

provides better classification than others. Further experiment on the number of neurons concludes that ELM works best when number of hidden neurons is within appropriate limit. Also the small number of hidden neurons shows that ELM has a very compact network as compared to traditional ANN. Also the learning speed of ELM is extremely fast and is completed in seconds or even less. From the experiment we concluded that ELM is 10 times faster than SVM and is imperceptible to the problems of local minima and over fitting.

ACKNOWLEDGMENTS

The authors would like to thank MEDai Inc an Elsevier Company, Mansoura University, Egypt, and KFUPM with KACST, Saudi Arabia for their encourage and support.

REFERENCES

- [1] T. Li, C. Zhang, and M. Ogihara, "A Comparative Study of Feature Selection and Multiclass Classification Methods for Tissue Classification Based on Gene Expression," *Bioinformatics*, vol. 20, no. 15, pp. 2429-2437, 2004
- [2] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.-H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J.P. Mesirov, T. Poggio, W. Gerald, M. Loda, E.S. Lander, and T.R. Golub, "Multiclass Cancer Diagnosis Using Tumor Gene Expression Signatures," *Proc. Nat'l Academy Sciences, USA*, vol. 98, no. 26, pp. 15149-15154, 2002.
- [3] A. Statnikov, C.F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy, "A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis," *Bioinformatics*, vol. 21, no. 5, pp. 631-643, 2005.
- [4] C.-H. Yeang, S. Ramaswamy, P. Tamayo, S. Mukherjee, R.M. Rifkin, M. Angelo, M. Reich, E. Lander, J. Mesirov, and T. Golub, "Molecular Classification of Multiple Tumor Types," *Bioinformatics*, vol. 17, pp. S316-S322, 2001.
- [5] R. Linder, D. Dew, H. Sudhoff, D. Theegarten, K. Remberger, S.J. Poppl, and M. Wagner, "The 'Subsequent Artificial Neural Network' (SANN) Approach Might Bring More Classificatory Power to ANN-Based DNA Microarray Analyses," *Bioinformatics*, vol. 20, no. 18, pp. 3544-3552, 2004.
- [6] M. Ringner, C. Peterson, and J. Khan, "Analyzing Array Data Using Supervised Methods," *Pharmacogenomics*, vol. 3, no. 3, pp. 403-415, 2002.
- [7] J. Khan, J.S. Wei, M. Ringner, L.H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C.R. Antonescu, C. Peterson, and S. Meltzer, "Classification and Diagnostic Prediction of Cancers Using Gene Expression Profiling and Artificial Neural Networks," *Nature Medicine*, vol. 7, no. 6, pp. 673-679, 2001.
- [8] J.W. Lee, J.B. Lee, M. Park, and S.H. Song, "An Extensive Comparison of Recent Classification Tools Applied to Microarray Data," *Computational Statistics and Data Analysis*, vol. 48, pp. 869-885, 2005.

- [9] G.-B. Huang and H. A. Babri, "Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions," *IEEE Transactions on Neural Networks*, vol. 9, no. 1, pp. 224-229, 1998.
- [10] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks," *Proc. Int'l Joint Conf. Neural Networks (IJCNN '04)*, July 2004.
- [11] G.-B. Huang and C.-K. Siew, "Extreme Learning Machine: RBF Network Case," *Proc. Eighth Int'l Conf. Control, Automation, Robotics, and Vision (ICARCV '04)*, Dec. 2004.
- [12] G.-B. Huang and C.-K. Siew, "Extreme Learning Machine with Randomly Assigned RBF Kernels," *Int'l J. Information Technology*, vol. 11, no. 1, 2005.
- [14] G.-B. Huang, "Learning capability and storage capacity of two-hidden layer feedforward networks," *IEEE Trans. on Neural Networks*, vol. 14, no. 2, pp. 274-281, 2003.
- G.-B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme Learning Machine: Theory and Applications," *Neurocomputing*, 2006.
- [15] K.Z. Mao, G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, P. Saratchandran, N. Sundararajan, "Can threshold networks be trained directly", *IEEE Trans. Circuits Syst. II* 53 (3) (2006) 187-191.
- [16] J.S Jiang, H. Zhang "Full Bayesian Network Classifiers," *Proceedings of the 23 rd International Conference on Machine Learning*, Pittsburgh, PA, 2006
- [17] Provost, F. J., & Domingos, P. (2003) "Tree induction for probability-based ranking" *Machine Learning* 52(3), 199-215.
- [18] D. Serre, *Matrices: Theory and Applications*. Springer-Verlag, 2002.

Parallel Synchronization of View Definitions Based on Clustering Techniques

Inès Hilali Jaghdam¹, Jalel Akaichi¹
 ines.hilali@gmail.com, jalel.akaichi@isg.rnu.tn

¹SOIE, ISG University of Tunis, 41 Avenue de la Liberté Bouchoucha 2000, Bardo, TUNISIA

Abstract -Data Warehouses are complex systems built by gathering data from several data sources and integrating them into one deposit. Data sources schemas and models are continuously evolving which may affect an important number of view definitions built upon them. Hence there is a necessity of view synchronization to guarantee data warehouses efficiency. Our proposal allows the synchronization of affected view definitions by combining biological sequences coding and clustering methods that perform view definitions regrouping into separated classes according to similar schema changes.

KeyWords: Schema Changes, View Definitions Evolving, Sequences Clustering.

1 Introduction

Data warehouses are huge and complex systems consisting of many components which integrate highly-aggregated data [1] from various information sources in one deposit.

There are many types of evolution in the data warehouse environment. That implies that they will update not only their data but also their schemas [2], without any concern of how this may affect the data warehouse and the view definitions defined upon them. As consequence, the whole data warehouse is evolving and the maintenance has been necessary so several view maintenance approaches have been developed such as view maintenance, view adaptation and view synchronization which aims to rewrite views definitions after that sources schema have been changed.

The problem is that traditional sequential approaches in previous works take care of only one single schema change that limits probably their performance and makes much time in the execution process and entail generally the whole system to be incoherent by ignoring correlations between current schemas changes and affected views. Also those approaches present a great difficulty related to schema changes detection.

Many works have been devoted to the synchronization problem / task.

For example, in [3 and 4] authors examine the problem of how maintaining the view after a view rewriting is taking place by studying under which conditions this view synchronization can take place without requiring access to base relations, i.e., the self-maintainability issue however this last didn't have success.

But works such as [5, 6, 7, 8, 9 and 10] are dealing with rewriting queries into equivalent ones using underlying views but obtained views do not assure precisely the same functionality.

Two others approaches are related to view synchronization task from the notion of relaxation query extent similar to E-SQL. In fact in [11 and 12] authors established an SQL extension called C-SQL (Cooperative SQL), which relaxes the

strictness of SQL-where-conditions. In the same context, in [13] authors discuss another SQL extension called S-SQL (Schema SQL) that can query data but also schema such as sets of attribute and relation names. The problem was that both these languages do not maintain the real extent of the new view.

Authors in DWQ (Data Warehouse Quality) Esprit Project [14 and 15] address problems related to the quality of data warehouses, they investigate the issues of data warehouse evolution. In fact, they use a meta repository in support of data warehouse evolution, they describe a process model for the capture of all changes made to any component into the meta repository. But, the DWQ project does not address really the problem of generating non-equivalent view rewritings over evolving warehouses.

An additional related work is which was done on TSE (Transparent Schema Evolution) technology by [16 and 17], and which uses view technology to handle schema changes transparently. But the TSE project does not deal with the view evolution problem.

Finally we conclude that there are many works concerning view synchronization problem, but they are presenting some limits such as much execution time, incoherence in the system process and difficulty in schema changes detection. In fact schemas changes are evolutions that can affect views components such as the delete of a relation respectively an attribute, the name- change of a relation respectively an attribute and the add of a relation respectively an attribute. Note that since adding a relation or an attribute will not affect the existing view definition so the view synchronizer will not take them into consideration, hence he will limit at two first changes.

As our knowledge, there are not view synchronization works or approaches that are treating the bioinformatics domain, so our approach is considering the first attempt in the view synchronization domain that uses bioinformatics tools and performs the synchronisation of affected views in parallel way called Parallel Synchronisation System.

The outline of this paper is as follows section 2 comprises methods concerning encoding and clustering that perform the coding of views into sequences then their clustering into common classes. Section 3 describes with details our proposed Parallel Synchronization System based on the obtained clusters in previous section. Finally section 4 concludes our paper and plans for futures works.

2 Encoding methods and views clustering methods

In this section we will present important methods used to code affected view definitions with their attributes and their relations into sequences in order to facilitate their clustering via appropriate regrouping algorithms and comparing for their optimal obtained results.

2.1 Encoding methods

A view definition is a derived relation defined in terms of base relations from various information sources. It defines so a function from a set of base tables to a derived one, this function is typically recomputed every time the view is referenced. Views are defined by the SELECT-FROM-WHERE SQL syntax with a conjunction of primitive clauses in the WHERE clause. The SELECT clause is made up of attribute components, the FROM clause contains relations components and the WHERE-Conditions clause stores conditions found in the WHERE statements of the data warehouse.

A view definition is so composed by common key words written by italic capital letters: {*CREATE VIEW, AS, SELECT, FROM, WHERE, AND*}, also the 'View-Name', which is specified by the view definer and the view components which are the list of attributes, the list of relations and the list of where-conditions.

```
CREATE VIEW V-name AS
SELECT {AS1, AS2, AS3, ..., ASp}
FROM {R1, R2, R3, ..., Rn}
WHERE {((AW1) Θ (AW2)) or ((AW1) Θ (value)) }
AND {((AW1) Θ (AW2)) or ((AW1) Θ (value)) };
```

Figure 1: Formal view definition representation.

In the context of sequences clustering, the use of data mining techniques requires always that biological sequences must be presented in a relational format and not in their original format as strings before applying clustering tools. So we propose the following process of biological data mining:

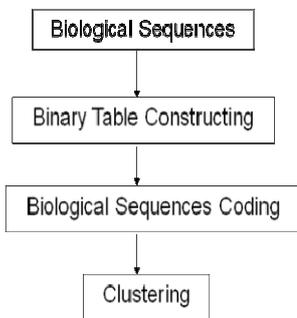


Figure 2: Transformation of biological sequences into binary table.

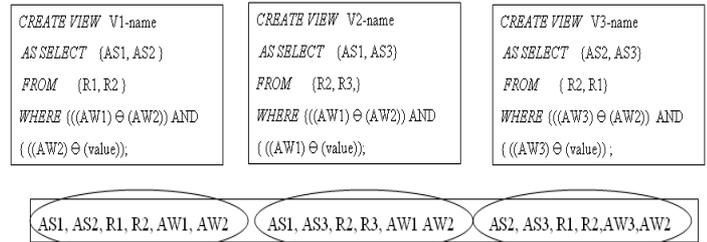
According to this model, our work will be organized like this:
-Step 1: extract all Attributes existing in the set of affected views definitions.
-Step 2: construct the binary table where lines represent view definitions and columns the set of all Attributes extracted in step1: the presence or the absence of one attribute in one sequence are expressed by 1 respectively 0. This binary table is called context and is considering the input for the next step.
-Step 3: the use of this table to produce a relational file in the ARFF¹ format used by the WEKA² environment [18].
 The ARFF format is very simple and it contains three tags, which all begin with the @ symbol.

¹ ARFF: Attribute Relation Format File.

² WEKA: Waikato Environment for Knowledge Analysis.

- @relation <name>, where <name> is the name of the relation that we wish use.
- @attributes <attribute-name> <type>, where <type> is the type of attributes in the relation.
- @data: where each line is a set of values separated by commas that represent a single instance.

Previous steps can be illustrated by the following diagram.



	AS1	AS2	AS3	R1	R2	R3	AW1	AW2	AW3
V1	1	1	0	1	1	0	1	1	0
V2	1	0	1	0	1	1	1	1	0
V3	0	1	1	1	1	0	0	1	1

Figure 3: Encoding steps.

```
@relation view
@attribute AS1 {1,0}
@attribute AS2 {1,0}
@attribute AS3 {1,0}
@attribute R1 {1,0}
@attribute R2 {1,0}
@attribute R3 {1,0}
@attribute AW1 {1,0}
@attribute AW2 {1,0}
@attribute AW3 {1,0}
@data
1, 1, 0, 1, 1, 0, 1, 1, 0
1, 0, 1, 0, 1, 1, 1, 1, 0
0, 1, 1, 1, 1, 0, 0, 1, 1
```

Figure 4: ARFF format.

2.2 Views clustering methods

Views definitions can be similar in their attributes, their relations, etc. Similarity can exist so in the SELECT clause and/or FROM clause and /or the WHERE clause. Many views can be completely similar and undergo the same schema changes, but the synchronization process treat them separately, hence the necessity of clustering before synchronization.

Also clustering is so efficient because instead of find appropriate substitution components for each affected view, we will try to find appropriate substitution components for the entire class that contains similar affected views.

The implementation of three previous steps is done by the language C. Then when obtaining the ARFF format we propose then the experimentation of clustering methods.

In literature, there are many kinds of clustering algorithms, for example the Exclusive Clustering where data are grouped in an exclusive way, so that if a certain datum belongs to a definite cluster then it could not be included in another cluster (K-means). Also the Probabilistic Clustering which uses a completely probabilistic approach (Mixture of Gaussian).

2.2.1 K-means

K-means is the clustering algorithm the most used. It minimizes the variance inside classes by minimizing the sum of distances between each data and the center of its cluster. The result is a set of compact and clearly separated clusters. Thus K-means is a simple clustering method that, when appropriate, shows optimal results. Its implementation assumes all attributes to be independent and normally distributed. When using the free K-means in the WEKA classifier, there are some instances which are incorrectly clustered compared to initial existing clustering. Consequently we will base on the proportion of those incorrectly clustered instances to choose the best clustering parameters.

2.2.2 EM: Expectation –Maximization (Mixture of Gaussians)

The algorithm which is used in practice to find the mixture of Gaussians that can model the data set is called Expectation-Maximization [19]. EM alternates between performing an expectation (E) step, which computes an expectation of the likelihood, and maximization (M) step, which computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found on the E step. The parameters found on the M step are then used to begin another E step, and the process is repeated.

2.3 Experimental study

In this section, we describe data which are the subject of our experimental study. We examine so each clustering algorithm alone then we compare them to the original clustering existing in the StatLib data base [20]. This data base contains many classes each one contains some views with certain attributes. Really the data base contains a great number of views but we are limiting to only five classes each with many views. Our classes are: archeology, biology, demography, environment and nature.

In this data base we tried to suppress the redundancy i.e., a view must belong to just one class no more. The number of views in each class is as follows archeology (2 views), biology (6 views), demography (2 views), environment (8 views) and nature (3 views).

In the free classifier Weka, we try to experiment the clustering of these views by the K-means then the Expectation Maximisation while specifying the cluster mode to 'use training set' and numclusters to 5 which is equal to the number of classes in our StatLib data base , then we do the clustering. Surely there will be some instances or views that will be incorrectly clustered according to the original clustering existing in the initial data base. So the algorithm which has the weakest rate of incorrectly clustered instances is the better one. Knowing that our objective is to do the clustering and obtain the wished classes of the views in order to be used in the next section.

Clustering Algorithm	Incorrectly Clustered Instances
K-means	28,76%
EM	94,76%

Table 1: The variation of the percentage of incorrectly clustered instances according to the clustering algorithm.

Explication of obtained results: with K-means we have obtained for each class the following values (Environment, 52%), (Biology, 19%), (Nature, 14%), (Archaeology, 10%) and (Demography, 5%).

And with Expectation-Maximisation we got the following results for each class (Environment, 90.48%), (Biology, 9.52%), (Nature, 0%), (Archaeology, 0%) and (Demography, 0%).

Or according to the clustering in the StatLib data base we have (Environment, 38.09%), (Biology, 28.57%), (Nature, 14.28%), (Archaeology, 9.52%) and (Demography, 9.52%).

So we calculate the sum of the percentage of incorrectly clustered instances (by calculating the sum of differences between the values obtained in the algorithm and the values in the data base), we obtain the results in the table 1.

So the K-means seems best than Expectation-Maximisation. Note that as mention previously that our objective was to obtain separated classes (without overlap) of affected views (i.e., views who underwent schemas changes).

3 Parallel Synchronization System (PSS)

Our proposed *Parallel Synchronization System* aims to synchronize in parallel way affected views definitions by scheduling more than one schema change at once time. The PSS will accede to the view space and detects affected views. Before beginning synchronisation, views were grouped into separated classes as mentioned in the previous section.

The PSS will take the classes of views that have undergone common schemas changes and look for suitable substitutions from the information space.

By opposition to sequential approaches which take care of one single schema change and this limits probably their performance.

3.1 PSS Design

To conceive our PSS, we use case diagrams to model its principal actors:

-The detector actors detect in parallel way numerous occurred schemas changes (SCs) distributed in many information sources, comparing the source schema at moment t and at moment t-1.

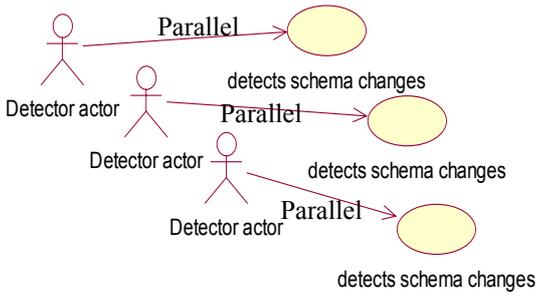


Figure 5: Use case diagram of the detector actor.

-The information space actor (or MKB actor) detects affected rules and knowledge, and updates the information space.

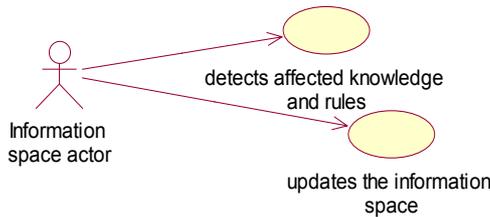


Figure 6: Use case diagram of the information space actor.

-The view space actor (or VKB actor) detects affected views stored in the view space.

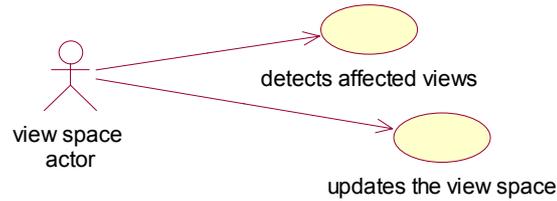


Figure 7: Use case diagram of the view space actor.

-The parallel view synchronizer actor determines the best legal rewriting for the affected views grouped in classes.

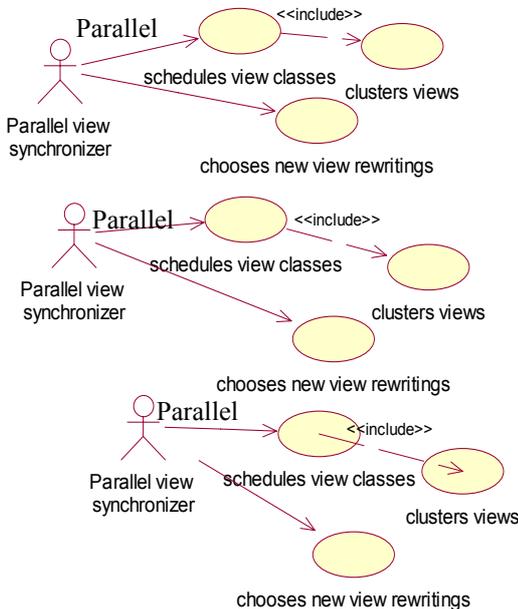


Figure 8: Parallel view synchronizer actor use case diagram.

3.2 Communication between PSS actors

PSS work is the result of communication between its actors. The detector actor transmits schemas changes to the information space and the view space. The information space actor detects the whole affected knowledge and sends them to the view synchronizer. The view space actor checks the view space to determine the set of view classes containing components which are affected by the same changes. Finally, it transmits affected views to the view synchronizer. The parallel view synchronizer receives affected rules from the information source and affected view classes from the view space, so it checks if it is possible to determine legal rewritings for the affected views in order to create new view definitions compatible with the current state of the information space. Therefore it refers to the user preferences incorporated into the E-SQL affected views. The view synchronizer transmits its results to the information space rules and to the user space view definitions, according to the new information space state. Hence the parallel view synchronizer schedules changes by classes and not by views, thing which is very important and permits to gain time by report to sequential approaches.

3.3 Parallel View Synchronizer Functioning

Previous centralized approaches opt for a sequential process where there is one view synchronizer that treats all schema changes (SCs) one by one. By consequence, it will be a list of affected views waiting their role to be synchronized.

So when the view synchronizer is repairing the first affected view, a new schema change (SC) is occurred and new affected view definitions are waiting to be repaired. Also it can be that the same schema change can affect many views at once time, and to repair each view it will be a new call for the same view synchronizer algorithm to do the same task.

In the parallel process [21] view definitions were grouped into similar classes according to common schema changes (SCs) that have undergone. So a class will contain affected view definitions waiting to be repaired by the same manner i.e. the view synchronizer algorithm will search for the same substitutions. Once it finds the appropriate substitutions it will affect them. Hence the view synchronizer is called only one time for each class, as consequence there is no waiting time for the view synchronizer.

The PSS permits besides less time for the execution process because the number of algorithm calls is limited by report to sequential process, in fact the number of calls is equal to the number of classes.

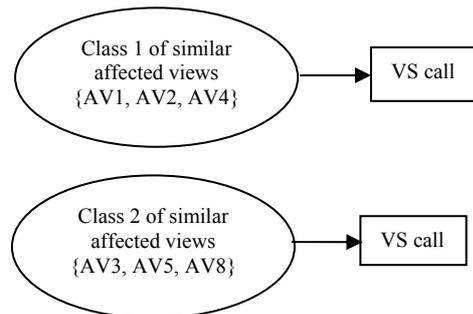


Figure 9: Parallel view synchronization process.

Parallel View Synchronizer

Begin

For each CC in ComponentChanges do in parallel

//CC: ComponentChange

For each Class in Classes do in parallel

For each View in Class do in parallel

If substitute (CC, CC') then

Begin

RewriteViews (CC, CC', AffectedClass);

UpdateMKB (CC, CC', AffectedMKB);

UpdateVKB (CC, CC', AffectedVKB);

End

Else

delete (CC, affectedMKB, AffcetedVKB);

End if;

End For;

End For;

End For;

End.

The algorithm starts by searching for substitutions for each schema change that affects similar views in one class, if good substitutions from the knowledge space are found, then affected view definitions will be rewritten, the user space will be updated according to new state of affected knowledge and the same for view space that will be modified according to new number of affected view definitions. Else if no appropriate substitutions are found the PSS proceeds by deleting the change, the affected view from classes and the affected knowledge from the information space.

4 Conclusions

In the end of this paper we have opted for a new approach in the view synchronization domain, which is the parallel view synchronization by opposition to previous sequential ones. So we had opted for first the coding of views definitions into genetic sequences then their clustering into common classes. Our future work will focus on processing correlations between affected view definitions and the implementation of our solution on a real parallel architectures.

5 References

- [1] C. Quix. "Repository Support for Data Warehouse Evolution". Proceedings of the International Workshop on Design and Management of DataWarehouses (DMDW'99). Heidelberg, Germany, 14. - 15.6. 1999.
- [2] Y. Zhuge, H. Garcia-Molina, J. Hamner, and J. Widom. "View Maintenance in a Warehousing Environment". In Proceeding of SIGMOD, pages 316-327, May 1995.
- [3] M. Boyd and al. "AutoMed: A BAV Data Integration System for Heterogeneous Data Sources". In Advanced Information Systems Engineering 16th International Conference, CAiSE 2004, Riga, Latvia, June 7-11,2004, Proceedings. Springer-Verlag, 2004.
- [4] A. Gupta, H.V. Jagadish, and I.S. Mumick. "Data Integration Using Self-Maintainable Views". Proc. Int'l Conf. Extending Database Technology (EDBT), pp. 140-144, 1996.
- [5] M. Mohania and G. Dong. "Algorithms for Adapting Materialized Views in Data Warehouses". Proc. Int'l Symp. Cooperative Database Systems for Advanced Applications, December. 1996.
- [6] A. Levy, I.S. Mumick, Y. Sagiv, and O. Shmueli. "Equivalence, Query Reachability and Satisfiability in Datalog

Extensions". Proc.12th ACM SIGACT-SIGMOD-SIGART Symp. Principles of Database Systems, pp. 109-122, May 1993.

[7] A. Levy and Y. Sagiv. "Constraints and Redundancy in Datalog". Proc. 11th ACM SIGACT-SIGMOD-SIGART Symp. Principles of Database Systems, pp. 67-80, June 1992.

[8] A.Y. Levy, A. Rajaraman, and J.D. Ullman. "Answering Queries Using Limited External Processors". Proc. 15th ACM Symp. Principals of Database Systems (pods), pp. 227-237, June 1996.

[9] A. Rajaraman, Y. Sagiv, and J.D. Ullman. "Answering Queries Using Templates With Binding Patterns." Proc. ACM Symp. Principles of Database Systems, pp. 105-112, May 1995.

[10] A.Y. Levy, A.O. Mendelzon, and Y. Sagiv. "Answering Queries Using Views". Proc. ACM Symp. Principles of Database Systems, pp. 95-104, May 1995.

[11] A. Rajaraman and J.D. Ullman. "Integrating Information by Outerjoins and Full Disjunctions". Proc. ACM Symp. Principles of Database Systems, pp. 238-248, 1996.

[12] W.W. Chu, M.A. Merzbacher, and L. Berkovich. "The Design and Implementation of CoBase", SIGMOD Record, vol. 22, no. 2, pp. 517-522, June 1993.

[13] W.W. Chu, H. Yang, K. Chiang, M. Minock, G. Chow, and C.Larson. "CoBase: A Scalable and Extensible Cooperative Information System". Intelligent Information Systems (JIIS), vol. 6, nos. 2/3, pp. 223-259, 1996.

[14] O. Etzioni and D. Weld. "A Softbot-Based Interface to the Internet". Comm. ACM, vol. 37, no. 7, pp. 72-76, July 1994.

[15] M. Jark, M.A. Jeusfeld, C. Quix, and P. Vassil-iadis. "Architecture and Quality in Data Warehouses: An Extended Repository Approach". Information System, vol.24 no 3, pp. 229-253, 1999.

[16] Y. Vassiliou and M. Jarke. "Data Warehouse Quality: a review of the DWQ project". Proc. Second Conf. Information Quality, 1997.

[17] Y.G. Ra and E.A. Rundensteiner. „A Transparent OO Schema Change Approach Using View Schema Evolution". IEEE Int'l Conf. Data Eng., pp. 165-172, Mar. 1995.

[18] W. I.H. & Eibe F. (2005). „Data mining: Practical machine learning tools and techniques". 2nd Edition. Morgan Kaufman, San Francisco, 2005.

[19] A.P. Dempster, N.M. Laird, and D.B. Rubin. "Maximum Likelihood from Incomplete Data via the EM algorithm". Journal of the Royal Statistical Society, Series B, vol. 39, 1:1-38, 1977.

[20] StatLib Data, software and News from the Statistics

Community.

<http://lib.stat.cmu.edu/modules.php?op=modload&name=Downloads&file=index>.

[21] Bosakova-Ardenska and N. Vasily. "Parallel algorithms of the scanning mask method for primary images processing". International conference on computer science systems and technologies_ CompSys Tech'2004.

The Analysis of the Relationships between the Real Vehicle Information and the Fuel Consumption

Jongwoo Choi, Daesub Yoon, Kyoungho Kim, and Hyunsuk Kim
Electronics and Telecommunications Research Institute

Abstract—This paper describes the process to analyze the vehicle information for the economical driving pattern in the car industry. The analyses are based on the real driving information. The VDMS applied in the postal trucks collects the driving information to analyze. There are various driving factors influenced the fuel consumption. We verified the relationships between the driving factors and the fuel consumption. And we will apply the various data mining models to analyze the vehicle information and make the economical driving grades by the results of the analyses.

I. INTRODUCTION

THE driving pattern affects the fuel economy and the CO₂ emission in the car industry. The efficient driving pattern reduces the fuel consumption and cuts back on the emissions of the greenhouse gases. There were the guidelines for the drivers to make the economical driving. The guidelines recommend the several ways to consume the less fuel, such as the choosing the efficient driving routes, the keeping the economical driving velocity, the restraining the sudden accelerations and the sudden decelerations, and the using the fuel cut driving, etc. The drivers can improve their driving patterns by acting upon the guidelines. The general drivers want to reduce the cost as the fuel consumption and to maintain their cars to be stable for a long time, whereas there are many drivers who don't care their driving patterns in the car industry, like taxies and trucks etc. Because the drivers are not the owners of the cars and the fuel is supported by the company, their driving patterns can't be controlled. Some drivers made private use of the cars, and the other drivers take a rest activating the air conditioner for a long time. These driving patterns incur a loss of the fuel cost to the company. But there is not any solution for company to manage the vehicles and the drivers, yet.

In this paper, we introduce the our system, the VDMS(Vehicle and Driver Management System) to manage the vehicles and the drivers for the economical driving. The VDMS collects the driving information from the cars during the driving. We analyze the information to make the criterion

This work was supported by the IT R&D program of MKE/IITA [2007-S025-02, Vehicle & Driver Management System Technology Development].

J. Choi is with the Electronics and Telecommunications Research Institute, Daejeon, Korea (e-mail: jwchoi@etri.re.kr).

D. Yoon is with the Electronics and Telecommunications Research Institute, Daejeon, Korea (e-mail: eyetracker@etri.re.kr).

K. Kim is with the Electronics and Telecommunications Research Institute, Daejeon, Korea (e-mail: kkh@etri.re.kr).

H. Kim is with the Electronics and Telecommunications Research Institute, Daejeon, Korea (e-mail: hyskim@etri.re.kr).

for the economical driving pattern. First, we assumed the important driving factors related with the economical driving. We will verify the relationships between the driving factors and the fuel consumption. And we will apply the various data mining models to analyze the driving factors and make the economical driving grades by the results of the analyses. The VDMS is applied in the postal trucks center and our analyses are based on the real driving information.

II. THE VEHICLE AND DRIVER MANAGEMENT SYSTEM

The VDMS is the system to manage the vehicles and drivers for the economical driving in the car industry, like taxies and trucks, etc. The VDMS consists of the car information collector, the driving information device, and the statistical information server. The car information collector is plugged into the OBD-II port to be connected the car network. The car information collector gathers the car status information, the car driving information, and the car diagnosis information per second. The car information is delivered to the driving information device. The driving information device receives the car information from the car information collector. The driving information device provides driving information to the drivers and guides the driver to make the economical driving. The driving information device sends the car information to the statistical information server through the wireless network. The statistical information server manages the information about the vehicles and the drivers. The supervisor of the company uses the statistical information server to confirm the status of the vehicles and the drivers.

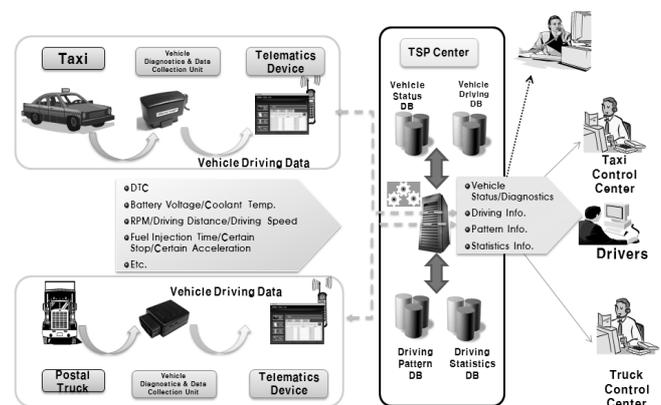


Fig. 1. The VDMS collects the car information and manages the vehicles and the drivers. The VDMS as the telematics application can be integrated with the various industry, taxies, trucks, and insurance, etc.

The statistical information server provides the useful information, the economical driving grades, safe driving grades, daily driving reports, and the diagnosis list, etc.

III. THE ANALYSIS OF THE VEHICLE INFORMATION

A. The Economical Driving Factors

We assume the economical driving factors by the rough observation of the vehicle information and the related work. The EPA(Environment Protection Agency) providing the fuel economy information had considered the various factors related with the fuel economy[1]. The table 1 represents the economical factors.

TABLE I
THE ECONOMICAL DRIVING FACTORS

Factors	Description
1 Economical speed driving	60km/h ~ 80km/h (under 2500rpm) Under 2000cc : 60km/h Over 2000cc : 70km/h Over 3000cc : The best fuel economy at the speed, 80km/h
2 Steady driving	The frequent acceleration causes the increasing of the fuel consumption Steady driving using through inertia
3 Sudden acceleration	Increasing the fuel consumption Smooth control of the acceleration pedal Slow acceleration in the first 5 seconds
4 Sudden deceleration	Smooth control of the brake pedal.
5 Idling time	Deactivation during the stop
6 Fuel-cut driving	Put off the acceleration pedal on the way down Over the 1500 rpm

B. The Vehicle Information Preprocessing

The raw vehicle information is collected every second. The information includes the speed, rpm, battery voltage, coolant temperature, coordinates, direction, moving distance, DTC(Diagnosis Trouble Codes) and fuel consumption, etc. The raw vehicle information is collected from June.2008 to Feb.2009. But the information is not suitable to analyze because the raw information represents the instant status of the vehicle. The figure 3 represents the rough observation of the raw vehicle information. The figures are the graphs of the fuel consumption according to the speed and rpm. It is difficult to insist that the speed and the rpm have the relationships with the fuel consumption directly.

CAR_ID	DRIVER_ID	STATUS_INFO_DAY	STATUS_INFO_TIME	BATTERY	COOLANT_TEMPERATURE	LATITUDE	LONGITUDE	SPEED	DIRECTION	RPM	FUEL_JET	MOVE_DISTANCE
4521T5	4521	20090107	193253	28	79	36.179235	127.194092	17	14	1603	4	7
4521T5	4521	20090107	193254	28	79	36.179273	127.194099	23	10	1159	1	8
4521T5	4521	20090107	193255	28	79	36.179273	127.194099	27	10	1095	2	8
4521T5	4521	20090107	193256	28	79	36.179311	127.194109	28	12	1238	5	8
4521T5	4521	20090107	193257	28	79	36.179311	127.194109	31	12	1426	5	10
4521T5	4521	20090107	193258	28	79	36.179308	127.194126	35	6	1634	8	11
4521T5	4521	20090107	193259	28	79	36.17943	127.194117	40	359	1853	9	13
4521T5	4521	20090107	193300	28	79	36.179472	127.194095	46	348	2075	10	15
4521T5	4521	20090107	193301	28	79	36.179517	127.194048	52	335	2289	12	16
4521T5	4521	20090107	193302	28	79	36.179562	127.193901	58	325	2436	9	18
4521T5	4521	20090107	193303	28	79	36.179553	127.193842	63	323	2362	1	18
4521T5	4521	20090107	193304	28	79	36.179736	127.193776	65	323	2251	0	18
4521T5	4521	20090107	193305	28	79	36.179828	127.193715	64	328	2163	0	17
4521T5	4521	20090107	193306	28	79	36.179983	127.193597	61	328	1994	0	16
4521T5	4521	20090107	193307	28	79	36.179983	127.193597	58	328	1777	0	15
4521T5	4521	20090107	193308	28	79	36.1857	127.19354	53	328	1474	0	13

Fig. 2. The raw vehicle information is collected per second for 8 months. The information is collected per second.

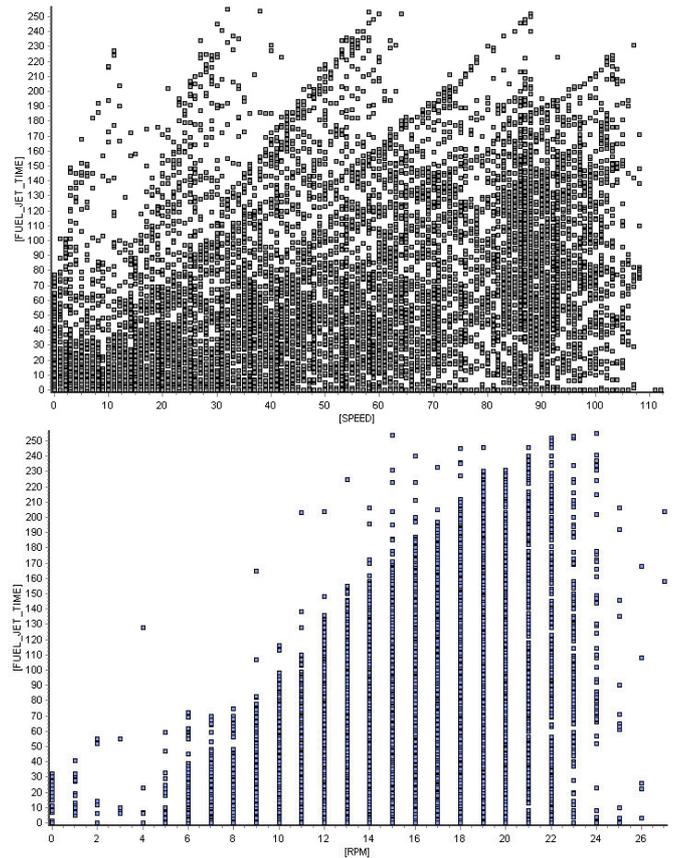


Fig. 3. These figures are the rough observations of the raw vehicle information. The graphs show the fuel consumption according to the speed, and the rpm. But it is difficult to analyze the vehicle information before the data preprocessing.

The purpose of the analysis is to verify the relationships between the economical driving factors and the fuel consumption. By the preprocessing, the information is cleared and ordered in the sequence of the driving periods.

C. The Relationships between the Economical Factors and the Fuel Economy

The vehicle information is not the discrete data but the continuous data. The results of the MLR analyses verify the relationship between the economical factors and the fuel consumption.

TABLE II
THE VARIABLES FOR THE MLR ANALYSES

Variables	Description
D_SPEED	{Economical speed driving time} / {Driving time}
D_STEADY_DRIVE_TIME	{Steady driving time} / {Driving time}
D_STARTS_NUM	{The number of sudden acceleration} / {Driving distance}
D_STOPS_NUM	{The number of sudden deceleration} / {Driving distance}
D_IDLE_TIME	{Idling time} / {Driving time}
D_FUEL_CUT_TIME	{Fuel-cut driving time} / {Driving time}
D_FUEL	{Driving distance} / {Fuel consumption}

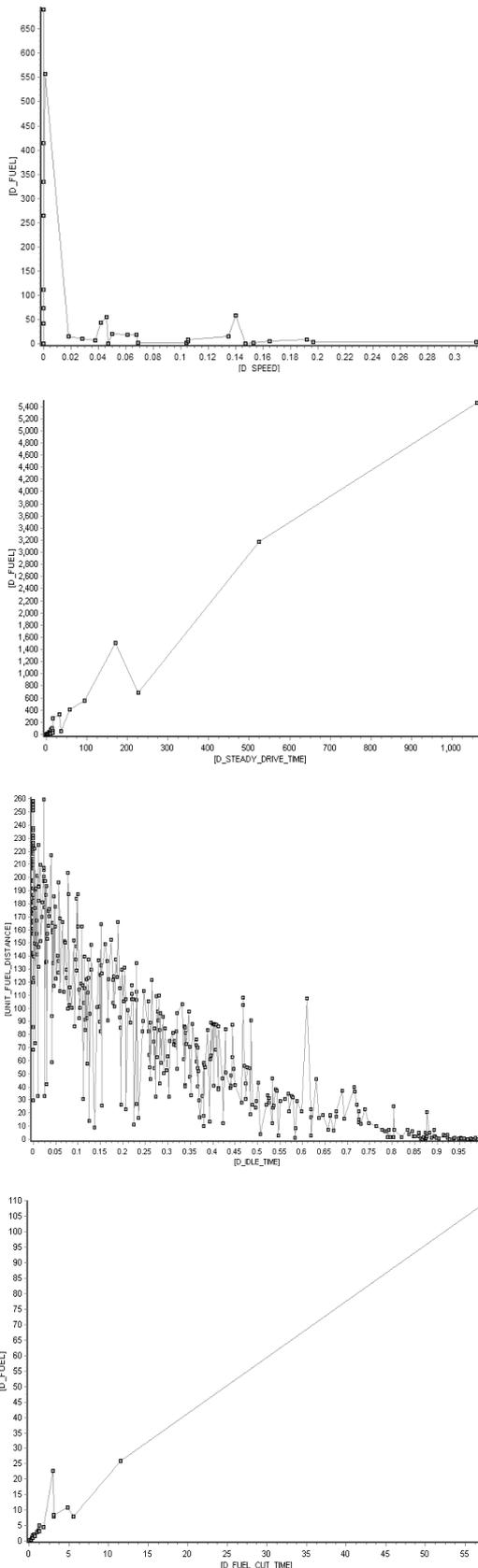


Fig. 4. These graphs show the relationships with the fuel consumption and the economical driving factors, in order of the economical speed driving, the steady driving, the idling time, and the fuel-cut time.

1) *Economical Speed Driving:*

The more the vehicle is driven in the economical speed range, the more fuel economy is obtained.

2) *Steady Driving:*

The more the vehicle is driven in the steady driving status, the more fuel economy is obtained.

3) *Sudden Acceleration:*

The more the vehicle is driven without the sudden acceleration, the more fuel economy is obtained.

4) *Sudden Deceleration:*

The more the vehicle is driven without the sudden deceleration, the more fuel economy is obtained.

5) *Idling Time:*

The more the vehicle is driven without the idling time, the more fuel economy is obtained.

6) *Fuel-cut Time:*

The more the vehicle is driven with the fuel cut driving time, the more fuel economy is obtained.

D. *The Analyses by Applying the Data Mining Models*

The real vehicle information is collected for about 8 months. But the data collection is complete very recently, the basic analysis of the vehicle information has completed. The basic analysis is the observation of the raw vehicle information, and the various data preprocessing. There are a lot of data mining models to analyze the vehicle information, the MLR(Multiple Linear Regression) analysis, the classifications, the clustering algorithms, etc. It is expected that the driving patterns will be found related with the fuel consumption and the emissions of the CO2.

IV. CONCLUSION

The vehicle information analysis is the important issue as the energy problem, and the environment protection. But the analysis of the real vehicle information is not done yet, because it is difficult to collect the real vehicle information. The system to manage the drivers and the vehicles is necessary to collect the real vehicle information

In this paper, we describe the VDMS to manage the vehicles and drivers. We applied VDMS in the postal trucks and collect the vehicle information. And we analyze the collected vehicle information for the economical and the ecological driving patterns. There are some factors influenced the fuel consumption. The factors are the economical speed driving, the steady driving, the sudden acceleration, the sudden deceleration, the idling time, and the fuel-cut time.

Because the vehicle information is collected recently, the information has been preprocessed and analyzed roughly without the various applications of the data mining models, yet. The relationships are found between the economical driving factors and the fuel consumption by the rough analyses.

We will analyze the real vehicle information by the applying the various data mining models. We expect that the

driving patterns will be extract related with the fuel consumption and the emissions of the CO₂. These analysis results will be utilized to manage the vehicles and the drives efficiently.

ACKNOWLEDGMENT

This work was supported by the IT R&D program of MKE/IITA [2007-S025-02, Vehicle & Driver Management System Technology Development].

REFERENCES

- [1] Fuel Economy Labeling of Motor Vehicle Revisions to Improve Calculation of Fuel Economy Estimates, Dec. 2006, EPA
- [2] Fuel Economy Guide 2008, U.S Department Energy
- [3] Summary of Fuel Economy Performance, Jan 15, 2008, U.S. Department Transportation
- [4] On-board Vehicle Data Stream Monitoring using MineFleet and Fast Resource Constrained Monitoring of Correlation Matrices, Hillol Kargupta, Agnik
- [5] <http://bpm.kenco.or.kr/transport>

A comparative study of Simulated Annealing and Variable Neighborhood Search for the Geographic Clustering Problem

M. B. Bernábe L.^{1,5}, J. E. Espinosa.², M. A Osorio L.³, J. R. Rodríguez⁴, R. R Aceves⁵

¹Fac. Cs. de la Computación, ²Fac. Cs. Físico Matemáticas, ³Fac. Ing. Química

B. Univ. Autónoma de Puebla México ,

⁴Departamento de Sistemas, Universidad Autónoma Metropolitana

⁵Programa de Posgrado en Ingeniería, Universidad Nacional Autónoma de México

Abstract—This paper describes a comparative study of two heuristic methodologies: Simulated Annealing and Variable Neighborhood Search. These methods have been used for solving the Geographical Clustering Problem (GCP), and compared according to their solution time, and the quality of the solutions obtained. The solution of this problem demands a zone classification process where each zone is made of objects that best fulfill the objective, usually the minimum accumulated distance from the objects to the centroid in each zone; informally this process is named the geometric compactness. This well known application is an NP hard combinatorial problem.

I. INTRODUCTION

THE geographic clustering problem (GCP) consists in the classification of objects in geographic units that fulfill a certain objective, mainly the geometric compactness [6,7,14]. The geographic units that have been considered correspond to AGEBS (Basic Geostatistic Areas) of the metropolitan zones in Toluca Valley MZTV [20].

The GCP problem belongs to the Territorial Design TD category and it is understood as the problem of grouping small geographic areas (basic areas) in greater geographical clusters called territories, in such a way that the acceptable grouping is the one that fulfills certain predetermined criteria [19]. The criteria or properties to fulfill in GCP problems depend on the space restrictions as continuity and geometric compactness [6,7,8,14]. The NP-hard condition of the GCP, implies solving a great number of geographic tasks that emphasizes the classification process directed toward the fulfillment of an objective [4].

Therefore, this problem is usually explained with a description oriented towards an optimization objective modeled mathematically as a cost function accompanied by the characteristics of the problem expressed as constraints. The NP nature of this problem, justify the utilization of a heuristic methodology to obtain a solution approximated to the optimal one [15]. The GCP is a special case of the classic clustering problem [7], but under the fulfillment of compactness, connectedness and/or homogeneity in some cases.

To solve the GCP, we developed our own partitioning algorithm that minimizes the distances between objects in

order to obtain compactness between the AGEBS [4]. However, the primary target of this research goes beyond revealing the solutions generated by the Variable Neighborhood Search VNS [9,10] and the Simulated Annealing SA [12]. In this sense, our main contribution is centered exactly in this point: we used the optimal solutions obtained with Simulated Annealing SA and compared them with the solutions generated with Variable Neighborhood Search VNS for the GCP, and used those results in the Box Behnken experimental design developed [13].

This document is organized in 6 sections. The introduction is presented in section 1, in section 2 we present the Mathematical model for the geographical clustering problem. Section 3 presents the solution to the GCP with Simulated Annealing and with Variable Neighborhood Search. Section 4 describes the results obtained for the Box Behnken design with VNS and section 5 presents the Box Behnken design with SA. Finally the conclusions are presented in section 6.

II. MATHEMATICAL MODEL FOR THE GEOGRAPHIC CLUSTERING PROBLEM

Many approaches have been used to solve the geographic clustering problem (GCP). The method utilized in this research to solve the AGEBS conglomerate design is similar to the method presented in [7], where the authors implemented a genetic algorithm for a similar zone design problem.

In the Geographic Clustering Problem solved here, the AGEBS are geographical units where each AGEBS is separated by different distances of non uniform geometric structure, because the AGEBS are spatial data [14] and its geographical localization is given by latitude and longitude, that made easier the calculation of the distances between them. The AGEBS are clustered in a way that the AGEBS composing such groups are very close geographically, in order to minimize the distances between them.

Basically, the strategy is to randomly choose AGEBS as centroids to identify the groups. Those AGEBS that are not centroids and have the shortest distance to a specific centroid-AGEBS are members of a group or a cluster. This informal idea is the definition of geometrical compactness.

The definition of compactness for geographic units is included in Definition 1:

Definition 1. Let $Z=\{1, 2, \dots, n\}$ be the set of n objects to classify; the objective is to divide Z in k groups G_1, G_2, \dots, G_k with $k < n$, such that:

- $\cup_{i=1,k} G_i = Z$
- $G_i \cap G_j = \emptyset \quad i \neq j$
- $|G_i| \geq 1 \quad i = 1, 2, \dots, k$

A group G_m with $|G_m| > 1$ is compact, if for every object $t \in G_m$ satisfies:

$$\text{Min}_{i \in G_m, i \neq t} d(t, i) < \text{Min}_{j \in Z-G_m} d(t, j)$$

A group G_m with $|G_m| = 1$ is compact only if its object t satisfies:

$$\text{Min}_{i \in Z - \{t\}} d(t, i) < \text{Min}_{j, l \in G_i, \forall j \neq m} d(j, l) \quad (1)$$

The neighborhood criterion between objects needed to achieve the compactness is given by the pairs of distances described in (1). Using this definition of compactness we will proceed to describe the model for the Geographic Clustering Problem (GCP).

A. Model for the Geographical Clustering Problem (GCP)

Data

UG= total of AGEBS

Let the initial set of n geographical units be

UG = $\{x_1, x_2, \dots, x_n\}$, where
 x_i is the i^{th} geographical unit (i =UG index)
 k is the number of the zone (group)

The following variables are defined to refer to the different groups:

Z_i is the set of geographical units that belong to the i^{th} zone
 n is the number of geographical units

C_i is the centroid

$d(i, j)$ is the euclidean distance from node i to node j (from one AGEBS to another)

Constraints

$Z_i \neq \emptyset$ for $i = 1, \dots, k$ (nonempty groups)

$Z_i \cap Z_j = \emptyset$ for $i \neq j$ (The same AGEBS cannot be in different groups)

$\cup_{i=1,k} G_i = U_g$ (The union of all the groups are all the AGEBS)

Objective Function Once the number of centroids (k) is decided (C_t with $t = 1, \dots, k$), the centroids will be randomly selected and the AGEBS will be assigned to the nearest centroids. Each AGEBS i is assigned to the nearest centroid C_t . Then, for each AGEBS i :

The objective function is the minimum of the sum of the distances between the centroids (for each k) and the AGEBS assigned to them. Each AGEBS is assigned to the closest centroid (C_t).

$$\text{Min}_{t=1, \dots, k} d(t, C_t)$$

For every k (where $k=1, n$) the sum of the distances from every AGEBS assigned to each centroid is calculated and the minimum is selected. Therefore the objective function can be written as:

$$\text{Min}_{k=1, \dots, nit} \left\{ \text{Min} \left\{ \sum_{t=1}^k \sum_{i \in c_t} d(i, c_t) \right\} \right\} \quad (2)$$

III. THE VARIABLE NEIGHBORHOOD SEARCH (VNS) AND THE SIMULATED ANNEALING (SA)

A. The Variable Neighborhood Search (VNS)

The Variable Neighborhood Search (VNS) metaheuristic, proposed by Hansen and Mladenovic [9,10] is based in the observation that local minima tend to cluster in one or more areas of the searching space. Therefore when a local optimum is found, one can get advantage of its contained information. For example, the value of several variables may be equal or close to their values in the global optimum. Looking for better solutions, VNS starts exploring, first the nearby neighborhoods of its current solution, and gradually the more distant ones. There is a current solution Sa and a neighborhood of order k associated to each iteration of VNS. Two steps are executed in every iteration: first, the generation of a neighbor solution of Sa , named $Sp \in N_k(Sa)$, and second, the application of a local search procedure on Sp , that leads to a new solution Sol . If Sol improves the current solution Sa , then the searching procedure will start now from G using $k = 1$. Otherwise, $k = k + 1$ and the procedure is repeated from Sa . The algorithm stops after a certain number of times that the complete exploration sequence $N_1; N_2; \dots; N_{kmax}$ is performed. The following algorithm shows how the solutions are obtained.

Two steps are executed in every iteration: first, the generation of a neighbor solution of Sa , named $Sp \in N_k(Sa)$, and second, the application of a local search procedure on Sp , that leads to a new solution Sol . If Sol improves the current solution Sa , then the searching procedure will start now from G using $k = 1$. Otherwise, $k = k + 1$ and the procedure is repeated from Sa . The algorithm stops after a certain number of times that the complete exploration sequence $N_1; N_2; \dots; N_{kmax}$ is performed. The following algorithm shows how the solutions are obtained.

Procedure Variable Neighborhood Search (VNS)

```

BEGIN
/* Nk : k = 1, ..., kmax, neighborhood structures */
/* Sa : current solution */
/* Sp : neighbor solution of Sa */
/* Sol: local optima solution */

REPEAT UNTIL (End) DO
  k ← 1
  REPEAT UNTIL (k ← kmax) DO
/* Generate neighbor Sp of the kth neighborhood of
  Sa (Sp ∈ Nk (Sa))*/

    Sp ← GetNeighbor (Sa, Nk);
    Sol ← LocalSearch (Sp);
    IF (Sol is better than Sa) THEN
      Sa ← Sol;
    ELSE
      k ← k + 1
    ENDIF
  ENDDO
END

```

Partitioning algorithms match the objective function (2) in order to minimize the distance of the objects to their centroids. The geographic clustering algorithm is following the same objective with the use of VNS.

The Local Search (LS) algorithm improves the searching of the current solution in its neighborhood. It can finish finding a better solution or reaching the maximum number of iterations. The maximum number of iterations avoids the algorithm cycling in the case that a better solution cannot be found.

B. Simulated Annealing (SA)

The SA algorithm is a neighbor-based search method characterized by an acceptance criterion for neighboring solutions that adapts itself during the running time. It uses a variable called temperature (T), that according to its value determines the degree in which worse neighboring solutions can be accepted. The temperature variable is initialized with a high value, called initial temperature T_i and is reduced with each iteration through a cooling temperature mechanism (alpha α) until a final temperature (T_f) is reached. During each iteration a concrete number of neighbors $L(T)$ is generated, which may be fixed for all the execution time or may change for each iteration. Each time a neighbor is generated, an acceptance criterion is applied to determine if it will substitute the current solution [12].

If the neighbor solution is better than the current one, it is automatically accepted, as a classic local search would do (LS). On the contrary, if it is worse, there still exists the possibility for the neighbor to substitute the current solution. This allows the algorithm to escape from local optima, where LS would get trapped. The higher the temperature, the more likely it is to accept worse solutions. In this way, the algorithm accepts solutions much worse than the current one

at the beginning of the execution (exploring) but not at the end (exploiting) [12].

The objective function previously described in (1) will be implemented with the following SA algorithm

Procedure Simulated Annealing (SA)

```

INPUT (To, α, L(t), Tf)
T ← To /*Initial value for the control parameter*/
Sact ← Generate initial solution
WHILE T ≥ Tf DO /* Stopping condition */
  BEGIN
    FOR cont ← 1 TO L(T) DO /*Cooling speed(T)*/
      BEGIN
        Scand ← Select solution N(Sact)
/* Creation of a new solution */
        δ ← cost(Scand) - cost(Sact)
/* Computation of cost difference */
        IF (U(0,1) < e(-δ/T) OR δ < 0) THEN
/* Application of acceptance criterion */
          Sact ← Scand
        END
      T ← α(T) /* Cooling mechanism */
    END
  {WRITE as solution the best of the visited Sact}

```

The clustering algorithm has been built exclusively for grouping geographical units with the addition of SA.

IV. EXPERIMENTAL BOX BENHKEN DESIGN FOR VNS

This section presents a Box Benhken BB [13] experimental design for VNS. The characteristics of this type of design make it easy to carry out experiments, defining adapted levels of the design parameters. Besides it is a rotary design that uses equal variance for all the experimentation points that have the same distance to the center of the design, and it is possible to make sequential experiments in the pruned regions, in order to study the individual effects of the control parameters and its combined effects.

The control variables for VNS are: neighborhood structures (NS), local search iterations (LS) and the number of groups (G).

The experiments were performed in a computer with an Intel Centrino® processor, speed of 1.4 Ghz 768MB in RAM memory and 80 GB of Hard Disk memory. We tested 171 variables in 473 AGEBS. Each AGEB includes data of 55 blocks, in average.

The parameter levels used in the experiment can be seen in Table I.

TABLE I
EXPERIMENTATION LEVELS FOR VNS

Parameter	High Level	Center Level	Low Level
LS	848	530	212
NS	640	400	160
G	24	18	12

In the Table II we showed the results for 15 runs with VNS. The nomenclature used is: NS (neighborhood structures), LS (Local Search), G (Groups).

TABLE II
TESTS FOR VARIABLE NEIGHBORHOOD SEARCH

R	G	LS	NS	FC
1	12	212	400	15.4535
2	24	212	400	10.9011
3	12	848	400	15.1598
4	24	848	400	10.8866
5	12	530	160	15.3206
6	24	530	160	11.0177
7	12	530	640	15.3221
8	24	530	640	10.8371
9	18	212	160	12.8541
10	18	848	160	12.4411
11	18	212	640	12.6957
12	18	848	640	12.4726
13	18	530	400	12.5597
14	18	530	400	12.5667

In test number 8, with 24 groups and parameters of LS = 530 and NS = 640, the cost function was 10.8371. This is the closest value to the optimal objective. The objective value obtained with PAM (Partitioning Around Medoids) [17] is 9.279. As a contrast, PAM managed to get that solution in 27 hours, while our VNS algorithm got it in 13 minutes, with 616529 iterations and 16 accepted solutions. The evolution of the iterations and the objective value can be seen in Fig. 1.

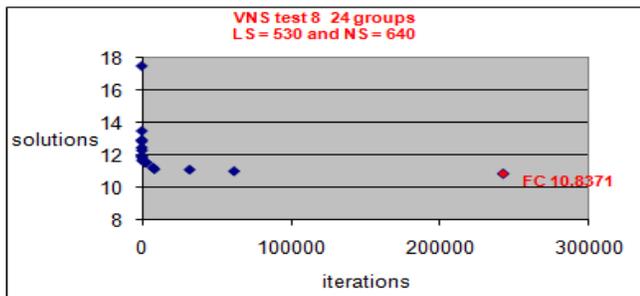


Fig. 1. Objective (Cost Function FC) vs. Number of Iterations for VNS. Run = 8, Groups = 24, LS = 530, NS = 640, FC = 10.8371

This instance has been chosen as a representative example of the experiment designed, because it was found that 24 groups is a turning point for our multivariate statistical study.

V. EXPERIMENTAL BOX BENHKEN DESIGN FOR SA

To obtain the instances evaluated with the SA heuristic, we developed a Box-Behnken design. The three levels determined for the experiment, are shown in Table 3.

TABLE III
SIMULATED ANNEALING LEVELS

Parameter	High Level	Central Level	Low Level
T_i	5500	5250	5000
T_f	0.1	0.055	0.01
α	0.99	0.985	0.98
L(t)	5	4	3
Groups	24	18	12

These 3 levels originated 46 trials that are shown in Table 4. The nomenclature used in the Table III and Table IV is: R (run), T_i (Initial Temperature), T_f (Final Temperature), α (alpha), L_t (L(t)), G (Groups), FC (Objective Function)

TABLE IV
TESTS FOR SIMULATED ANNEALING

R	T_i	T_f	α	L_t	G	FC
1	5000	0.01	0.985	4	18	13.5279
2	5500	0.01	0.985	4	18	13.5878
3	5000	0.1	0.985	4	18	14.0342
4	5500	0.1	0.985	4	18	14.1222
5	5250	0.055	0.98	3	18	13.9166
6	5250	0.055	0.99	3	18	14.1286
7	5250	0.055	0.98	5	18	13.2346
8	5250	0.055	0.99	5	18	13.8935
9	5250	0.01	0.985	4	12	16.2161
10	5250	0.1	0.985	4	12	16.553
11	5250	0.01	0.985	4	24	11.5392
12	5250	0.1	0.985	4	24	12.0292
13	5000	0.055	0.98	4	18	16.3016
14	5500	0.055	0.98	4	18	14.1095
15	5000	0.055	0.99	4	18	13.9159
16	5500	0.055	0.99	4	18	13.9545
17	5250	0.055	0.985	3	12	15.6353
18	5250	0.055	0.985	5	12	16.0845
19	5250	0.055	0.985	3	24	12.3314
20	5250	0.055	0.985	5	24	11.6377
21	5250	0.01	0.98	4	18	13.5198
22	5250	0.1	0.98	4	18	14.304
23	5250	0.01	0.99	4	18	13.3445
24	5250	0.1	0.99	4	18	13.7725
25	5000	0.055	0.985	3	18	13.6595
26	5500	0.055	0.985	3	18	13.5348
27	5000	0.055	0.985	5	18	14.0258
28	5500	0.055	0.985	5	18	13.0667
29	5250	0.055	0.98	4	12	16.8496
30	5250	0.055	0.99	4	12	17.1076
31	5250	0.055	0.98	4	24	12.2146
32	5250	0.055	0.99	4	24	11.7276
33	5000	0.055	0.985	4	12	16.6959
34	5500	0.055	0.985	4	12	16.7826
35	5000	0.055	0.985	4	24	11.8841
36	5500	0.055	0.985	4	24	11.2403
37	5250	0.01	0.985	3	18	13.5575
38	5250	0.1	0.985	3	18	13.2107
39	5250	0.01	0.985	5	18	13.6996
40	5250	0.1	0.985	5	18	14.7604
41	5250	0.055	0.985	4	18	13.9274
42	5250	0.055	0.985	4	18	13.8217
43	5250	0.055	0.985	4	18	13.5833
44	5250	0.055	0.985	4	18	13.9886
45	5250	0.055	0.985	4	18	13.6392
46	5250	0.055	0.985	4	18	12.9008

In trial 36 there are 24 groups with the parameters: $T_i=5500$, $T_f=0.055$, $\alpha=0.985$, $L(t)=4$, and an objective function with a cost of 11.2403. This is the best result, the

closest to the optimum obtained with PAM that was of 9.279. In contrast the time required by PAM to generate the exact solution, was 37 hours, while the Simulated Annealing, with 3,049 iterations, 2,183 accepted solutions, reduced the computational time to only 2 seconds. The behavior of objective function with the number of iterations can be seen in Figure 2.

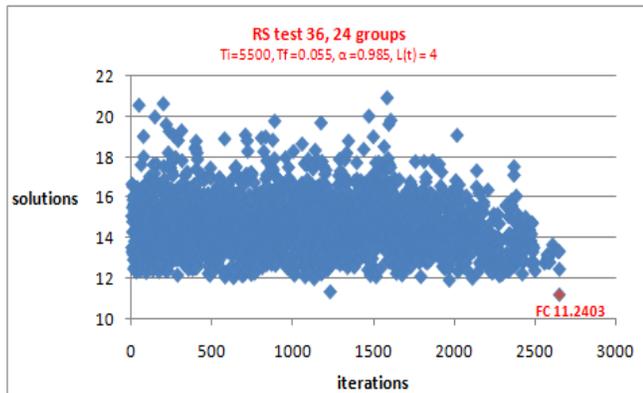


Fig. 2. Run 36, 24 groups, $T_i=5500$, $T_f=0.055$, $\alpha=0.985$, $L(t)=4$, $FC=11.2403$

VI. CONCLUSION

From the results obtained through all this work, we have found the VNS parameter values, used for solving the GCP that accompanied the best objective solutions.

- In general, the number of groups is directly proportional to the quality of the solutions.
- A value of NS close to 640 units, independently of the group size, will yield better values in the objective function.
- The best objective values were found for LS values between 848 and 530.

The SA parameter values, used for solving the GCP that accompanied the best objective solutions are: heuristic implemented for the geographical clustering problem are sensitive to different conditions.

- As in VNS, the number of groups is directly proportional to the quality of the solutions.
- With initial temperatures close to 5000 units, give a better convergence to the optimum regardless of the number of groups.
- A final temperature of 0.01 with $\alpha = 0.98$ gave the best minimum value for the objective function.

The experiment for both heuristics used the results obtained with the empirical combinations, where 24 proved to be a good number of groups. For this reason, Tables 1 and 3 used 24 groups.

The future work pretends to extend the experiment, increasing the parameters value, leading to generate more instances that permit the experiment to be more extensive.

We performed a two tails T-Student hypothesis test on the solutions gotten by both heuristics. We used results from tests number 8 and 36 of tables II and IV and repeated them, 10 times for VNS and RS, respectively. We used 13 degrees of freedom, and a marginal error of media μ and variance σ^2 . The statistical value to test of 37.7 showed that the null

hypothesis establishing that the quality of the solutions obtained with RS and VNS is the same, should be rejected. With the null hypothesis rejected, a new right-tailed T test where performed to test if the quality of the solutions with the two methodologies showed a statistical difference. Results verified the right-tailed alternative hypothesis about the difference in the quality of the solutions obtained with VNS and with RS. The statistical evidence allowed us to conclude that under the conditions of the experiment proposed in this paper the quality of solutions generated with VNS is better than the quality of the solutions obtained with RS.

REFERENCES

- [1] Barr, R. S., Golden, J.P., Resende, M. G., W.R. Stewart W.R.: Designing and Reporting on Computational Experiments with Heuristics Methods. Journal of Heuristics, Kluwer Academic Publisher (1995) 9–32.
- [2] Bernábe, L. B., López S.: Statistical Classificatory Analysis Applied to Population Zones. 8th. World Multiconference on Systemics, Cybernetics and Informatics (2004).
- [3] Bernábe, L. B., Ramírez R. J., Espinosa, R. J.: Evaluación de un algoritmo de recocido simulado con superficies de respuestas. Revista de Matemáticas Teoría y Aplicaciones, issn: 1409-2433 volume 16 number 1, (2009) 159-177
- [4] Duque, J. C., Ramos, R., Suriñach, J.: Supervised Regionalization Methods: A Survey. Inter-national Regional Science Review (2007) 195-220.
- [5] Fernando Bação, Victor Lobo, Marco Painho.: Applying genetic algorithms to zone design. Springer Verlag (2004)
- [6] Gordon, A. D.: A survey of constrained classification. Computational Statistics & Data Analysis (1996) 17–29.
- [7] Hansen P., Mladenovic, N.: Variable neighborhood search, Les Cahiers du GERAD (1996) 96-49.
- [8] Hansen P., Mladenovic, N.: Variable neighbourhood search. In Fred Glover and Gary A. Kochenberger, editors, Handbook of Metaheuristics, Kluwer (2003).
- [9] Kaufman, L., Rousseeuw, P.J.: 1987. Clustering by means of medoids. Statistical Data Analysis based on the L1 Norm, North-Holland, Amsterdam (1987) 405-416.
- [10] Kirkpatrick, C.D Gelatt, M.P Vecchi. 1983. Optimization by Simulated Annealing. SCIENCE. 220, N 4598
- [11] Montgomery D.: Design and Analysis of Experiments. Ed. Wiley 2^a Edition (1991)
- [12] Murtagh F.: A survey of algorithms for contiguity-constrained clustering and related problems. Computer Journal (1991) 82-88
- [13] Pizza, E., Murilo, A., Trejos, J.: Nuevas técnicas de particionamiento en clasificación automática. Revista de Matemáticas Teoría y Aplicaciones, issn: 1409-2433 (1999) 51-66.
- [14] Romero D., Burguete J., Martínez E., Velasco J.: Parcelación del territorio nacional: Un enfoque de optimización combinatoria para la construcción de marcos de muestreo en hogares. INEGI (2004)
- [15] Rousseeuw, P.J., Hubert M., Struyf A.: Clustering in an object-oriented environment. Journal of Statistical Software (1997) 02-10
- [16] Zamora, A. E.: Implementación de un algoritmo compacto y homogéneo para la clasificación de zonas geográficas AGEBS bajo una interfaz gráfica. Tesis de Ingeniería en Ciencias de la Computación BUAP FCC, Asesor B. Bernábe (2006)
- [17] Zoltners, A., Sinha, P.: 1983, Towards a unified territory alignment: A review and model. Management Science, (1983) 1237-1256
- [18] <http://www.inegi.gob.mx> Instituto Nacional de Estadística, Geografía e Informática (INEGI).