Semi-Supervised Learning
With Graphs

Shannon Quinn

(with thanks to William Cohen at CMU)
Semi-supervised learning

- A pool of labeled examples L
- A (usually larger) pool of unlabeled examples U
- Can you improve accuracy somehow using U?
Semi-Supervised Bootstrapped Learning/Self-training

Extract cities:

- Paris
- Pittsburgh
- Seattle
- Cupertino
- San Francisco
- Austin
- denial
- anxiety
- selfishness
- Berlin

- mayor of arg1
- live in arg1
- arg1 is home of traits such as arg1
Semi-Supervised Bootstrapped Learning via Label Propagation

Paris

live in

San Francisco
Austin

traits such as

anxiety

mayor of

Pittsburgh

is home of

Seattle

denial

traits such as

selfishness
Semi-Supervised Bootstrapped Learning via Label Propagation

Paris live in San Francisco

Pittsburgh

San Francisco Austin

Seattle

Nodes “near” seeds

Information from other categories tells you “how far” (when to stop propagating)

mayor of arg1

arg1 is home of

traits such as arg1

denial

denial

arrogance

selfishness

Nodes “far from” seeds
Semi-Supervised Learning as Label Propagation on a (Bipartite) Graph

Propagation methods: “personalized PageRank” (aka damped PageRank, random-walk-with-reset)

- Propagate labels to nearby nodes
- X is “near” Y if there is a high probability of reaching X from Y with a random walk where each step is either (a) move to a random neighbor or (b) jump back to start node Y, if you’re at an NP node
  - rewards multiple paths
  - penalizes long paths
  - penalizes high-fanout paths

I like beer

Paris
live in
San Francisco
Austin

Seattle

Pittsburgh

mayor of

denial
Semi-Supervised Classification of Network Data Using Very Few Labels

Frank Lin
Carnegie Mellon University, Pittsburgh, Pennsylvania
Email: frank@cs.cmu.edu

William W. Cohen
Carnegie Mellon University, Pittsburgh, Pennsylvania
Email: wcohen@cs.cmu.edu

ASONAM-2010 (Advances in Social Networks Analysis and Mining)
Network Datasets with Known Classes

• UBMCMCBlog
• AGBlog
• MSPBlog
• Cora
• Citeseer
Given: A graph $G = (V, E)$, corresponding to nodes in $G$ are instances $X$, composed of unlabeled instances $X^U$ and labeled instances $X^L$ with corresponding labels $Y^L$, and a damping factor $d$. 

Returns: Labels $Y^U$ for unlabeled nodes $X^U$.

For each class $c$
1. Set $u_i = 1$, $\forall Y_i^L = c$
2. Normalize $u$ such that $\|u\|_1 = 1$
3. Set $R_c = \text{RandomWalk}(G, u, d)$

For each instance $i$
- Set $X^U_i = \arg\max_c(R_{ci})$

Seed selection
1. order by PageRank, degree, or randomly
2. go down list until you have at least $k$ examples/class

Fig. 1. The MultiRankWalk algorithm.

RWR - fixpoint of:
$$r = (1 - d)u + dWr$$
CoEM/HF/wvRN

- One definition [MacSkassy & Provost, JMLR 2007]:

**Definition.** Given \( v_i \in V^U \), the weighted-vote relational-neighbor classifier (wvRN) estimates \( P(x_i | \mathcal{N}_i) \) as the (weighted) mean of the class-membership probabilities of the entities in \( \mathcal{N}_i \):

\[
P(x_i = c | \mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = c | \mathcal{N}_j),
\]
CoEM/HF/wvRN

- Another definition in [X. Zhu, Z. Ghahramani, and J. Lafferty, ICML 2003]
  - A harmonic field – the score of each node in the graph is the harmonic, or linearly weighted, average of its neighbors’ scores (harmonic field, HF)
Another justification of the same algorithm....

... start with co-training with a naïve Bayes learner

- **Inputs:** An initial collection of labeled documents and one of unlabeled documents.

- Loop while there exist documents without class labels:
  - Build classifier A using the A portion of each document.
  - Build classifier B using the B portion of each document.
  - For each class C, pick the unlabeled document about which classifier A is most confident that its class label is C and add it to the collection of labeled documents.
  - For each class C, pick the unlabeled document about which classifier B is most confident that its class label is C and add it to the collection of labeled documents.

- **Output:** Two classifiers, A and B, that predict class labels for new documents. These predictions can be combined by multiplying together and then renormalizing their class probability scores.

---

Table 1: The co-training algorithm described in Section 3.3.
Notations

\( \hat{Y}_{v,l} \) : score of estimated label \( l \) on node \( v \)

\( Y_{v,l} \) : score of seed label \( l \) on node \( v \)

\( R_{v,l} \) : regularization target for label \( l \) on node \( v \)

\( S \) : seed node indicator (diagonal matrix)

\( W_{uv} \) : weight of edge \((u, v)\) in the graph
LP-ZGL (Zhu et al., ICML 2003)

\[
\text{arg min}_{\hat{Y}} \sum_{l=1}^{m} W_{uv} (\hat{Y}_{ul} - \hat{Y}_{vl})^2 = \sum_{l=1}^{m} \hat{Y}_l^T L \hat{Y}_l
\]

such that

\[
Y_{ul} = \hat{Y}_{ul}, \quad \forall S_{uu} = 1
\]

Smooth

Match Seeds (hard)

• Smoothness
  – two nodes connected by an edge with high weight should be assigned similar labels

• Solution satisfies harmonic property
Modified Adsorption (MAD)  
[Talukdar and Crammer, ECML 2009]

$$\arg\min_{\hat{Y}} \sum_{l=1}^{m+1} \left[ \| S\hat{Y}_l - SY_l \|^2 + \mu_1 \sum_{u,v} M_{uv}(\hat{Y}_{ul} - \hat{Y}_{vl})^2 + \mu_2 \| \hat{Y}_l - R_l \|^2 \right]$$

- $m$ labels, +1 dummy label
- $M = W^\dagger + W'$ is the symmetrized weight matrix
- $\hat{Y}_{vl}$: weight of label $l$ on node $v$
- $Y_{vl}$: seed weight for label $l$ on node $v$
- $S$: diagonal matrix, nonzero for seed nodes
- $R_{vl}$: regularization target for label $l$ on node $v$
How to do this minimization?
First, differentiate to find min is at

\[(\mu_1 S + \mu_2 L + \mu_3 I) \hat{Y}_l = (\mu_1 SY_l + \mu_3 R_l).\]

**Jacobi method:**
- To solve $Ax=b$ for $x$
- Iterate:
  \[x^{(k+1)} = D^{-1}(b - R x^{(k)}).\]
- \[x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}\right), \quad i = 1, 2, \ldots, n.\]
Inputs $Y, R : |V| \times (|L| + 1)$, $W : |V| \times |V|$, $S : |V| \times |V|$ diagonal

$\hat{Y} \leftarrow Y$

$M = W' + W^\dagger$

$Z_v \leftarrow S_{vv} + \mu_1 \sum_{u \neq v} M_{vu} + \mu_2 \quad \forall v \in V$

repeat
  for all $v \in V$ do
    $\hat{Y}_v \leftarrow \frac{1}{Z_v} \left( (SY)_v + \mu_1 M_v. \hat{Y} + \mu_2 R_v \right)$
  end for
until convergence

- Extends Adsorption with well-defined optimization
- Importance of a node can be discounted
- Easily Parallelizable: Scalable
MapReduce Implementation of MAD

• Map
  – Each node send its current label assignments to its neighbors

• Reduce
  – Each node updates its own label assignment using messages received from neighbors, and its own information (e.g., seed labels, reg. penalties etc.)

• Repeat until convergence

Code in Junto Label Propagation Toolkit (includes Hadoop-based implementation)
http://code.google.com/p/junto/
Text Classification

PRBEP (macro-averaged) on WebKB Dataset, 3148 test instances
Sentiment Classification

Precision on 3568 Sentiment test instances
Class-Instance Acquisition

Freebase-2 Graph, 192 WordNet Classes

Graph with 303k nodes, 2.3m edges.
Assigning class labels to WebTable instances

<table>
<thead>
<tr>
<th>Year</th>
<th>Artist</th>
<th>Albums</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Johnny Cash</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bob Dylan</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>musician</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob Dylan</td>
</tr>
</tbody>
</table>

Score (musician, Johnny Cash) = 0.87
# New (Class, Instance) Pairs Found

<table>
<thead>
<tr>
<th>Class</th>
<th>A few non-seed Instances found by Adsorption</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFL Players</td>
<td>Tony Gonzales, Thabit Davis, Taylor Stubblefield, Ron Dixon, Rodney Hannan, …</td>
</tr>
</tbody>
</table>

Total classes: 9081
From SemiSupervised to Unsupervised Learning
Spectral Clustering: Graph = Matrix

\[ M \ast v_1 = v_2 \]

"propogates weights from neighbors"
Repeated averaging with neighbors as a clustering method

- Pick a vector \( v^0 \) (maybe at random)
- Compute \( v^1 = Wv^0 \)
  - i.e., replace \( v^0[x] \) with weighted average of \( v^0[y] \) for the neighbors \( y \) of \( x \)
- Plot \( v^1[x] \) for each \( x \)
- Repeat for \( v^2, v^3, \ldots \)

- Variants widely used for semi-supervised learning
  - clamping of labels for nodes with known labels
- Without clamping, will converge to constant \( v^\dagger \)
- What are the dynamics of this process?
Repeated averaging with neighbors on a sample problem...

- Create a graph, connecting all points in the 2-D initial space to all other points
  - Weighted by distance
- Run power iteration for 10 steps
- Plot node id x vs \( v^{10}(x) \)
  - Nodes are ordered by actual cluster number
Repeated averaging with neighbors on a sample problem...

(a) 3Circles PIC result
(b) Embedding at $t = 10$
(c) Embedding at $t = 50$
(d) Embedding at $t = 100$
Repeated averaging with neighbors on a sample problem...

(a) 3Circles PiC result
(b) Embedding at $t = 10$
(c) Embedding at $t = 50$
(d) Embedding at $t = 100$
(e) Embedding at $t = 200$
(f) Embedding at $t = 400$
Repeated averaging with neighbors on a sample problem...

(a) 3Circles PIC result
(b) Embedding at $t = 10$
(c) Embedding at $t = 50$
(d) Embedding at $t = 100$
(e) Embedding at $t = 200$
(f) Embedding at $t = 400$
(g) Embedding at $t = 600$
(h) Embedding at $t = 1000$

very small
PIC: Power Iteration Clustering
run power iteration (repeated averaging w/ neighbors)
with early stopping

1. Pick an initial vector $v^0$.
2. Set $v^{t+1} \leftarrow \frac{Wv^t}{\|Wv^t\|_1}$ and $\delta^{t+1} \leftarrow |v^{t+1} - v^t|$.
3. Increment $t$ and repeat above step until $|\delta^t - \delta^{t-1}| \approx 0$.
4. Use $k$-means to cluster points on $v^t$ and return clusters $C_1, C_2, ..., C_k$.

- $V^0$: random start, or “degree matrix” $D$, or ...
- Easy to implement and efficient
- Very easily parallelized
- Experimentally, often better than traditional spectral methods
- Surprising since the embedded space is 1-dimensional!