# Semi-Supervised Learning With Graphs

Shannon Quinn

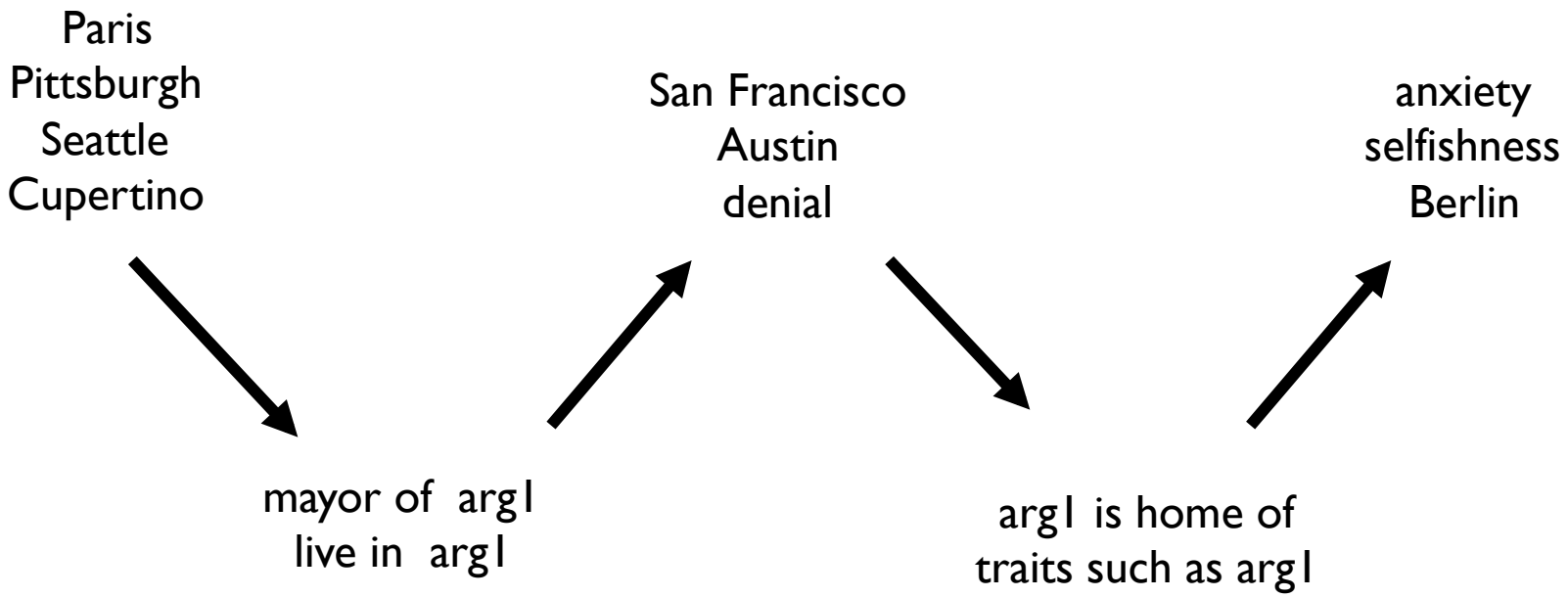
(with thanks to William Cohen at CMU)

# Semi-supervised learning

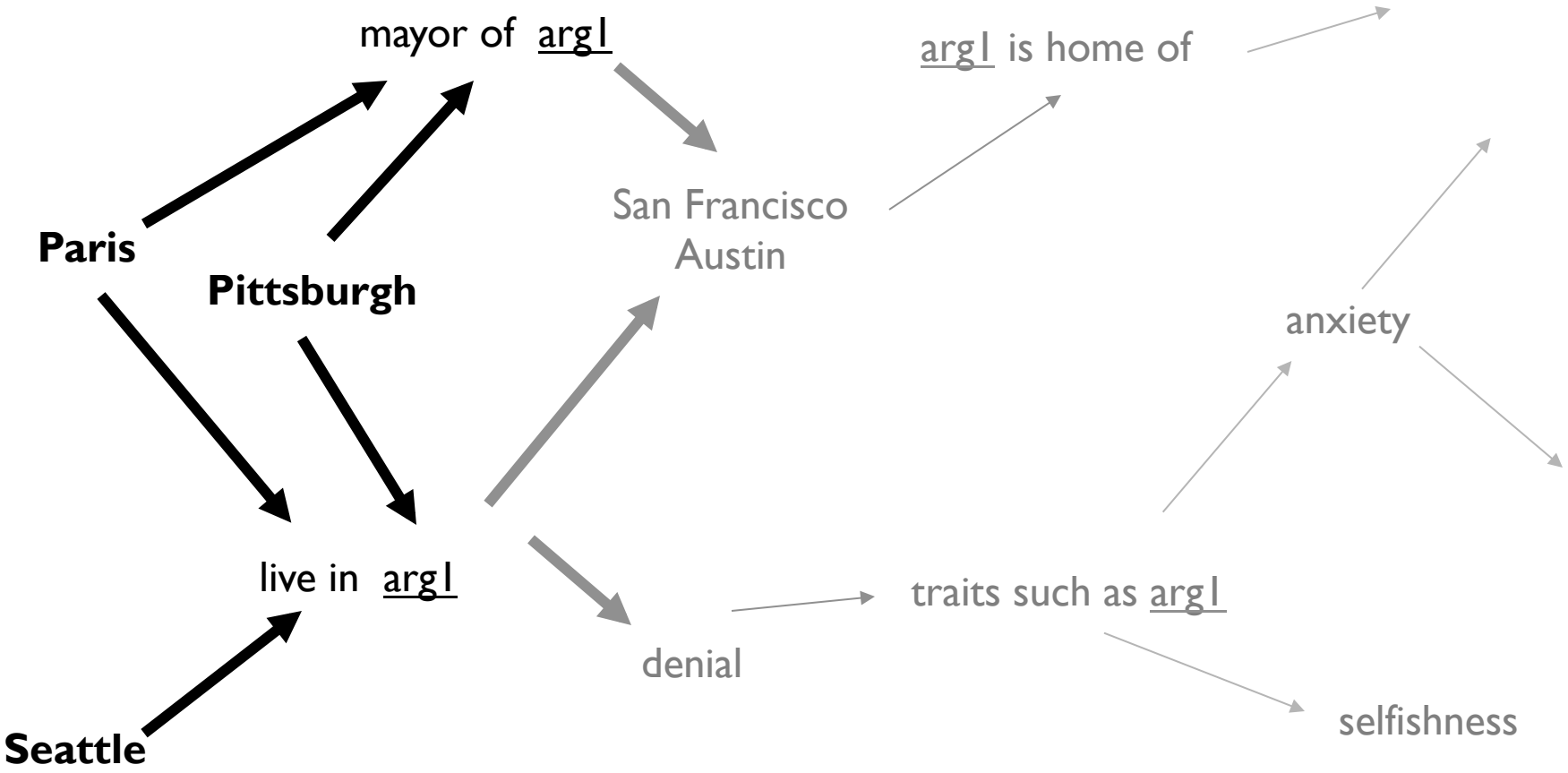
- A pool of labeled examples L
- A (usually larger) pool of unlabeled examples U
- Can you improve accuracy somehow using U?

# Semi-Supervised Bootstrapped Learning/Self-training

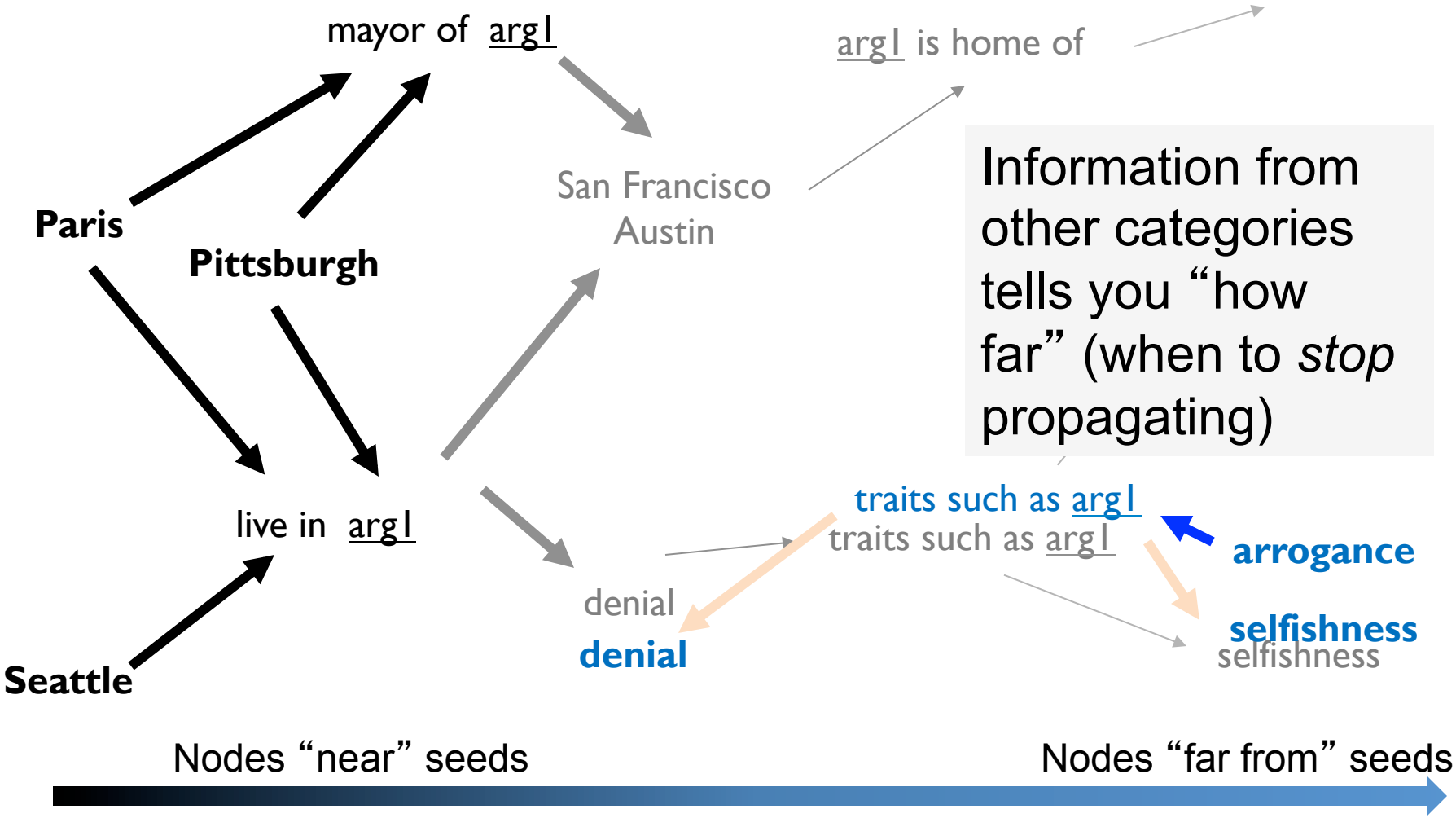
Extract cities:



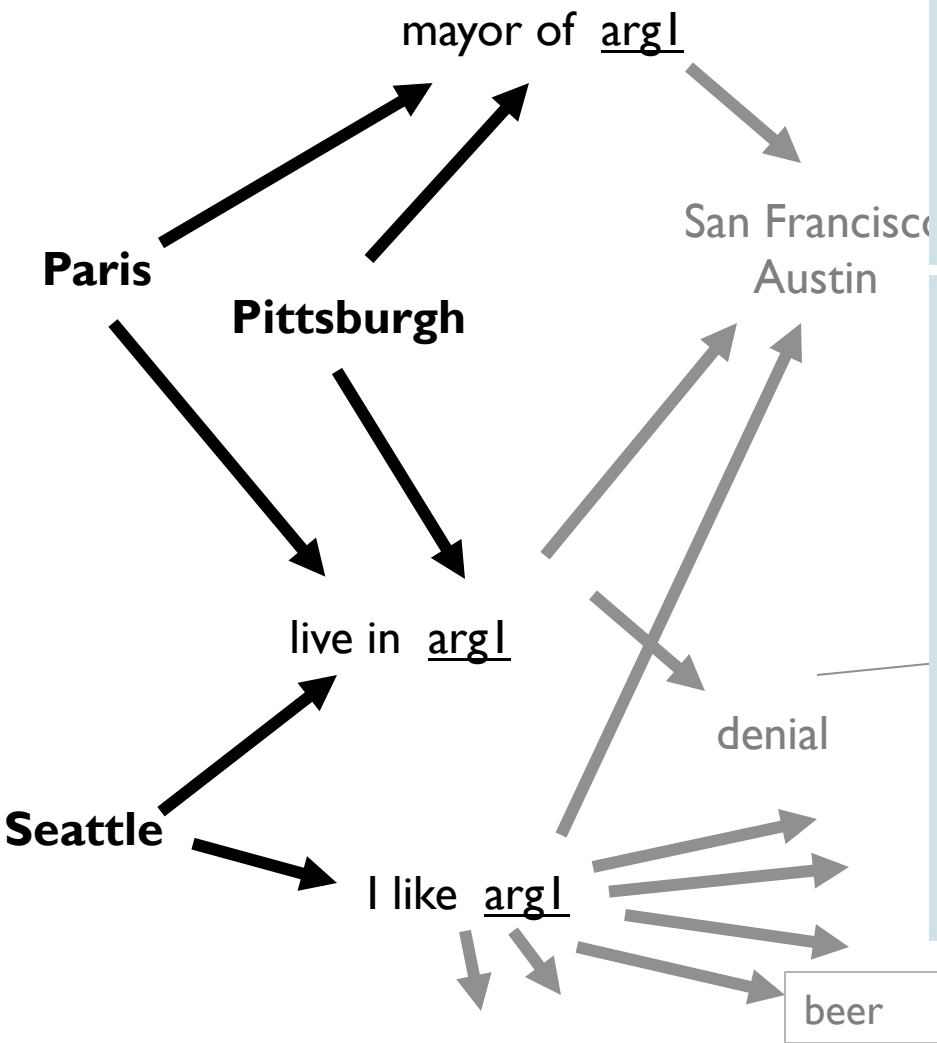
# Semi-Supervised Bootstrapped Learning via Label Propagation



# Semi-Supervised Bootstrapped Learning via Label Propagation



# 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

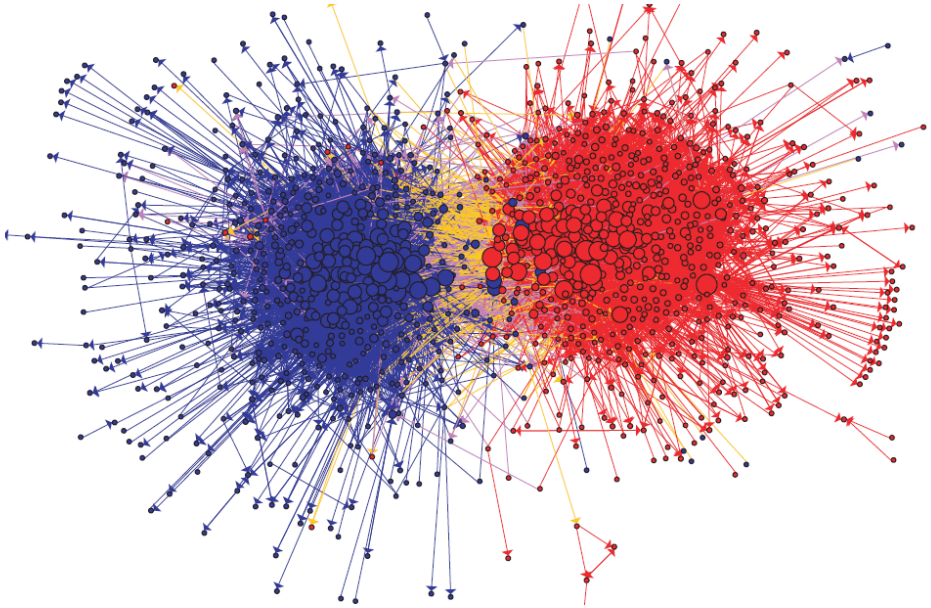
# Semi-Supervised Classification of Network Data Using Very Few Labels

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

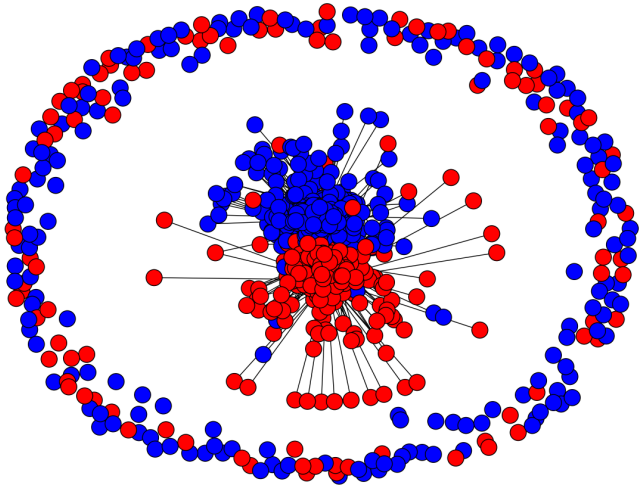
William W. Cohen
Carnegie Mellon University, Pittsburgh, Pennsylvania
Email: wcohen@cs.cmu.edu

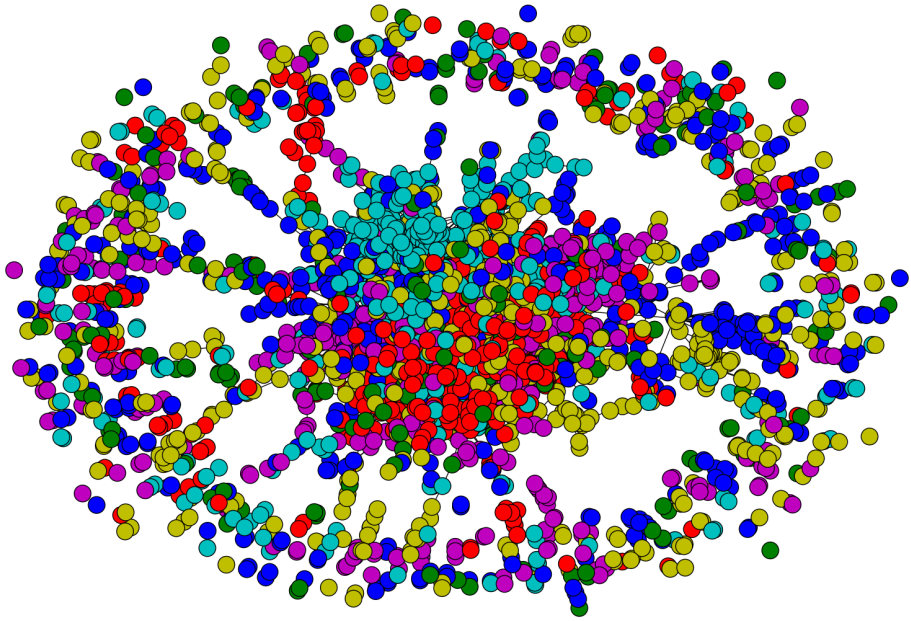
ASONAM-2010 (Advances in Social Networks Analysis and Mining)

# Network Datasets with Known Classes



- UBMCBlog
- 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 $\mathbf{u}_i \leftarrow 1, \forall Y_i^L = c$
2) Normalize $\mathbf{u}$ such that $||\mathbf{u}||_1 = 1$
3) Set $R_c \leftarrow RandomWalk(G, \mathbf{u}, d)$

**For each instance** $i$

- Set $X_i^U \leftarrow argmax_c(R_{ci})$

RWR - fixpoint of:

$$\mathbf{r} = (1 - d)\mathbf{u} + dW\mathbf{r}$$

Fig. 1. The MultiRankWalk algorithm.

Seed selection

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

# CoEM/HF/wvRN

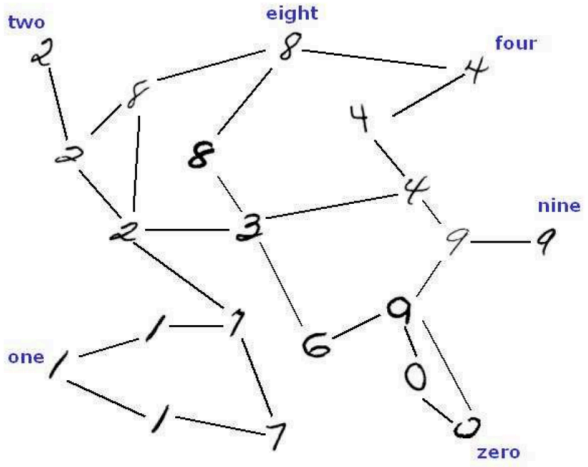
- One definition [MacSkassy & Provost, JMLR 2007]:…

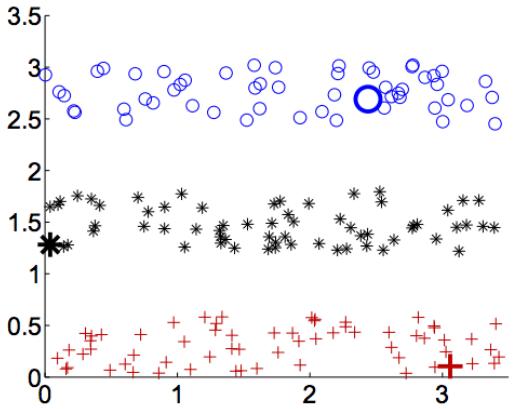
**Definition**. Given $v_i \in \mathbf{V}^U$, the weighted-vote relational-neighbor classifier (wvRN) estimates $P(x_i|\mathcal{N}_i)$ as the (weighted) mean of the class-membership probabilities of the entities in $\mathcal{N}_i$:

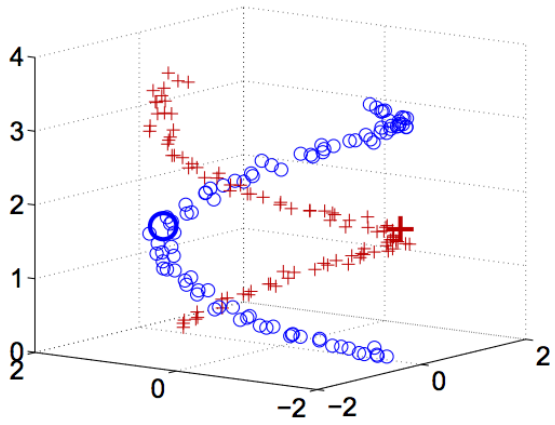
$$P(x_i = c|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = c|\mathcal{N}_j),$$

# 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)



# CoEM/HF/wvRN

- 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

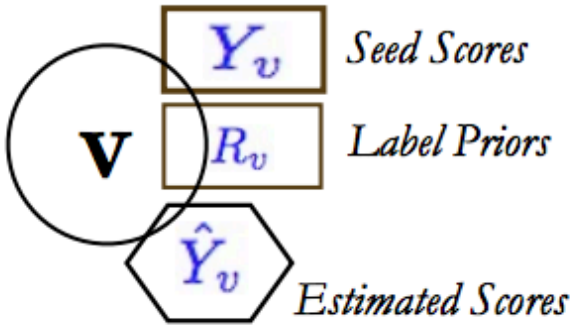
$\hat{Y}_{v,l}$ : score of estimated label l on node v

$Y_{v,l}$ : 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)

Smooth

$$\arg\min_{\hat{Y}} \boxed{\sum_{l=1}^{m} W_{uv}(\hat{Y}_{ul} - \hat{Y}_{vl})^2} = \sum_{l=1}^{m} \hat{Y}_l^T L \hat{Y}_l$$

Graph Laplacian
L = D - W (PSD)

such that $\boxed{Y_{ul} = \hat{Y}_{ul},\ \forall S_{uu} = 1}$

Match Seeds (hard)

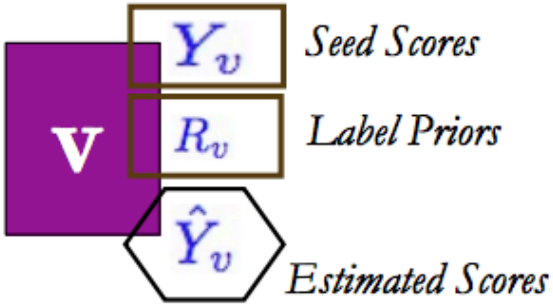
- Smoothness
  - two nodes connected by an edge with high weight should be assigned similar labels
- Solution satisfies harmonic property

# Modified Adsorption (MAD)

[Talukdar and Crammer, ECML 2009]

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

- $m$ labels, $+1$ dummy label
- $\boldsymbol{M} = \boldsymbol{W}^{\prime\top} + \boldsymbol{W}^{\prime}$ is the symmetrized weight matrix
- $\hat{\boldsymbol{Y}}_{vl}$: weight of label $l$ on node $v$
- $\boldsymbol{Y}_{vl}$: seed weight for label $l$ on node $v$
- $\boldsymbol{S}$: diagonal matrix, nonzero for seed nodes
- $\boldsymbol{R}_{vl}$: regularization target for label $l$ on node $v$



# Modified Adsorption (MAD)

[Talukdar and Crammer, ECML 2009]

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

How to do this minimization?
First, differentiate to find min is at

$$\left(\mu_1 \mathbf{S} + \mu_2 \mathbf{L} + \mu_3 \mathbf{I}\right) \hat{\mathbf{Y}}_l = \left(\mu_1 \mathbf{S}\mathbf{Y}_l + \mu_3 \mathbf{R}_l\right) .$$

**Jacobi method**:

- To solve Ax=b for x
- Iterate:

$$\mathbf{x}^{(k+1)} = D^{-1}(\mathbf{b} - R\mathbf{x}^{(k)}).$$

- … or:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right), \quad i = 1, 2, \ldots, n.$$

Inputs $\boldsymbol{Y}, \boldsymbol{R} : |V| \times (|L|+1)$, $\boldsymbol{W} : |V| \times |V|$, $\boldsymbol{S} : |V| \times |V|$ diagonal
$\hat{\boldsymbol{Y}} \leftarrow \boldsymbol{Y}$
$\boldsymbol{M} = \boldsymbol{W}' + \boldsymbol{W}^{\intercal}$
$Z_v \leftarrow \boldsymbol{S}_{vv} + \mu_1 \sum_{u \neq v} \boldsymbol{M}_{vu} + \mu_2 \quad \forall v \in V$
**repeat**
  **for all** $v \in V$ **do**
    $\hat{\boldsymbol{Y}}_v \leftarrow \frac{1}{Z_v} \left( (\boldsymbol{S}\boldsymbol{Y})_v + \mu_1 \boldsymbol{M}_{v\cdot} \hat{\boldsymbol{Y}} + \mu_2 \boldsymbol{R}_v \right)$
  **end for**
**until** convergence

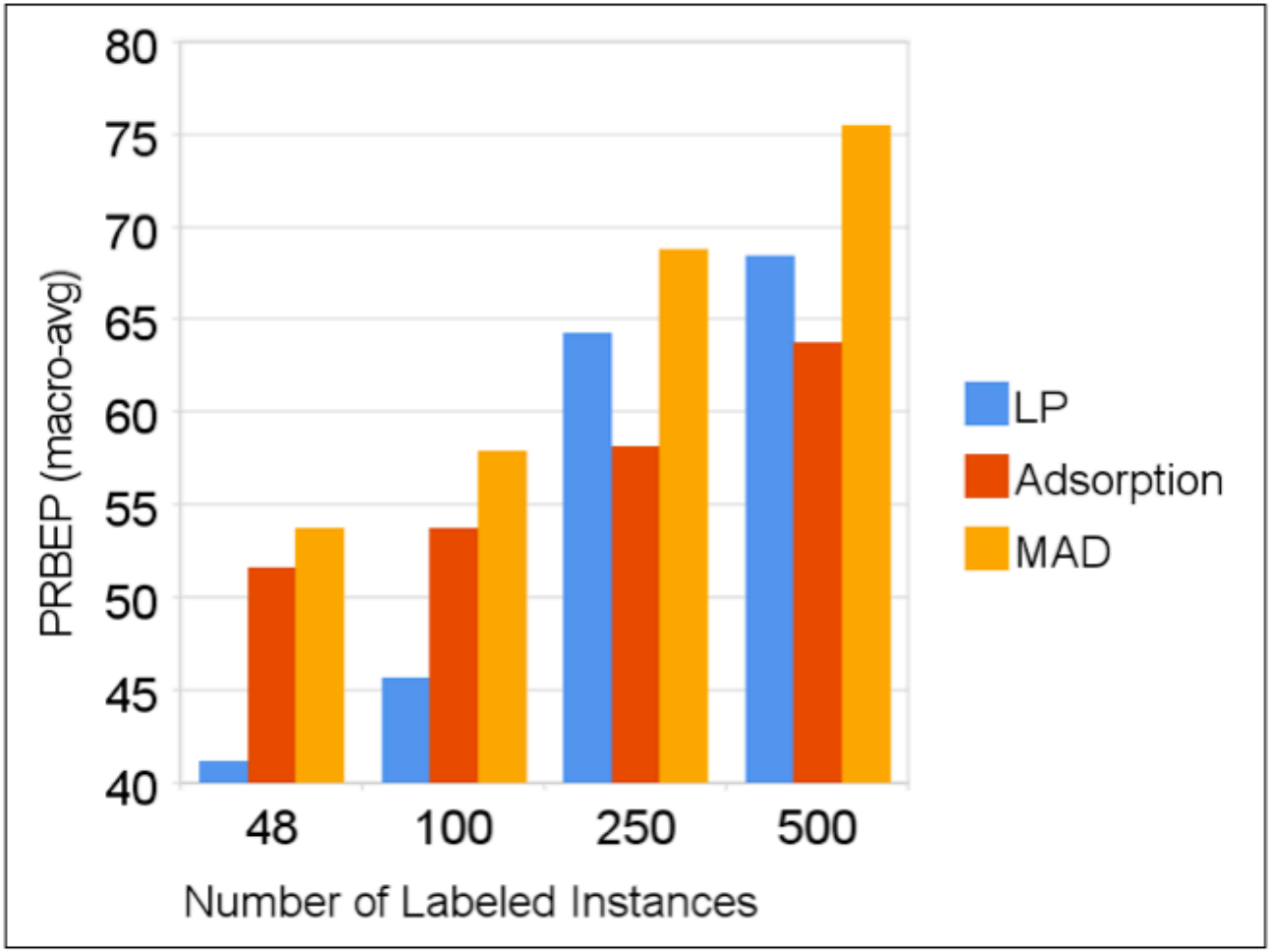
- Extends Adsorption with well-defined optimization
- Importance of a node can be discounted
- Easily Parallelizable: Scalable

# 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

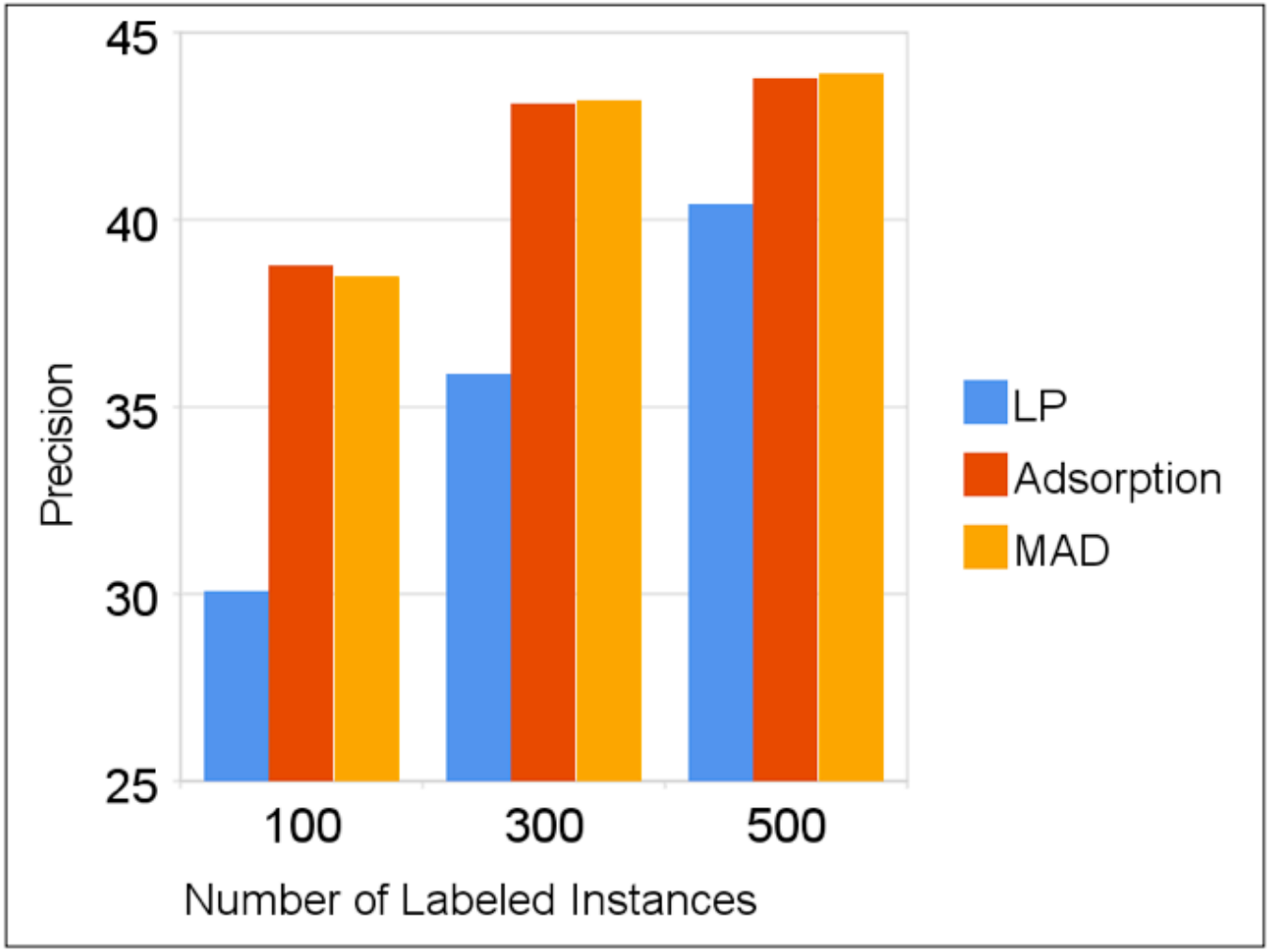
Code in Junto Label Propagation Toolkit
(includes Hadoop-based implementation)
http://code.google.com/p/junto/

# Text Classification



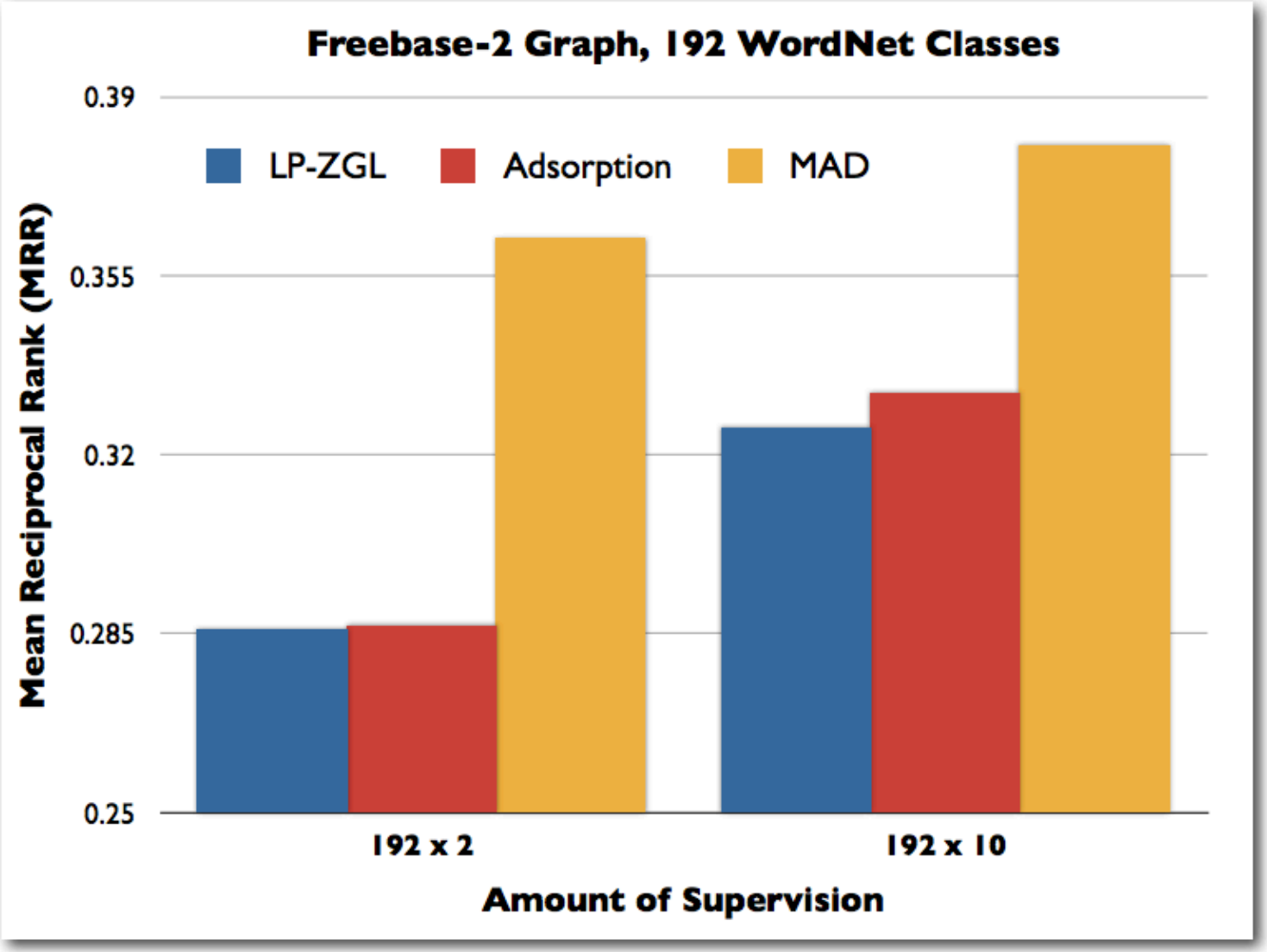

PRBEP (macro-averaged) on WebKB Dataset, 3148 test instances

# Sentiment Classification



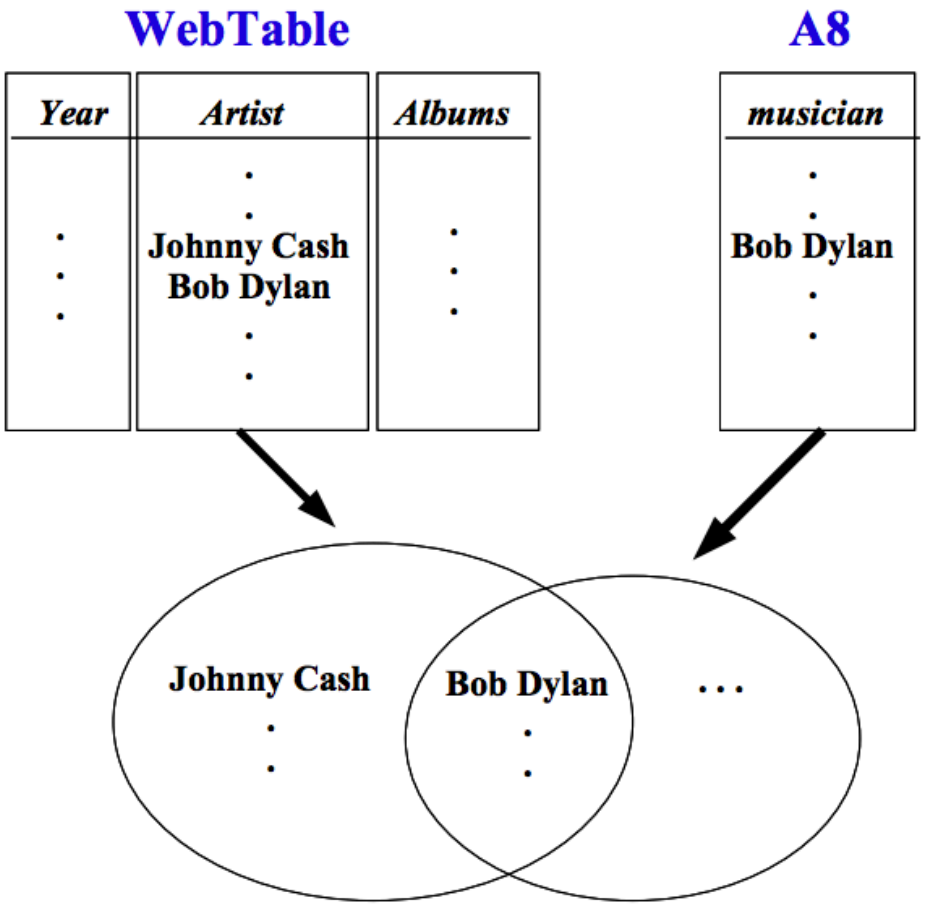

Precision on 3568 Sentiment test instances

# Class-Instance Acquisition

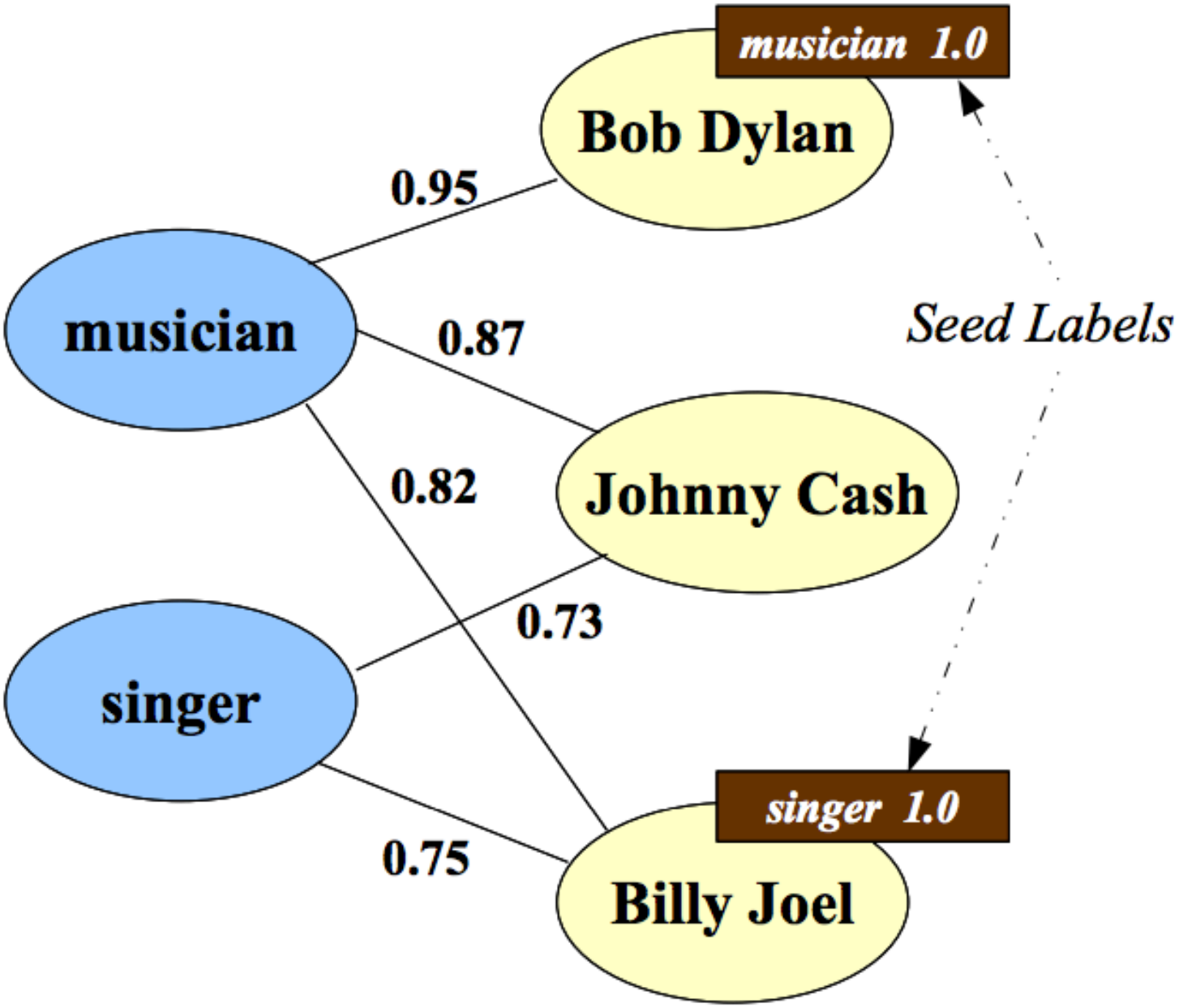



Graph with 303k nodes, 2.3m edges.

# Assigning class labels to WebTable instances




*Score (musician, Johnny Cash) = 0.87*


musician 1.0
Bob Dylan
0.95
musician
0.87
Seed Labels
0.82
Johnny Cash
0.73
singer
singer 1.0
0.75
Billy Joel

# New (Class, Instance) Pairs Found

| Class | A few non-seed Instances found by Adsorption |
|---|---|
| Scientific Journals | Journal of Physics, Nature, Structural and Molecular Biology, Sciences Sociales et sante, Kidney and Blood Pressure Research, American Journal of Physiology-Cell Physiology, … |
| NFL Players | Tony Gonzales, Thabiti Davis, Taylor Stubblefield, Ron Dixon, Rodney Hannan, … |
| Book Publishers | Small Night Shade Books, House of Ansari Press, Highwater Books, Distributed Art Publishers, Cooper Canyon Press, … |

Total classes: **9081**

# From SemiSupervised to Unsupervised Learning

# Spectral Clustering: Graph = Matrix

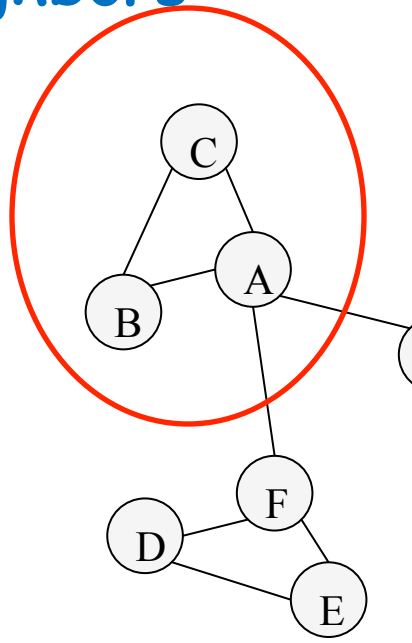
$M * v_1 = v_2$ "propogates weights from neighbors"

$M$ * $v_1$ = $v_2$

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| A | _ | 1 | 1 | | | 1 | | | | |
| B | 1 | _ | 1 | | | | | | | |
| C | 1 | 1 | _ | | | | | | | |
| D | | | | _ | 1 | 1 | | | | |
| E | | | | 1 | _ | 1 | | | | |
| F | | | | 1 | 1 | _ | | | | |
| G | | | | | | | _ | | 1 | 1 |
| H | | | | | | | | _ | 1 | 1 |
| I | | | | | | | 1 | 1 | _ | 1 |
| J | | | | | | | 1 | 1 | 1 | _ |

| | |
|---|---|
| A | 3 |
| B | 2 |
| C | 3 |
| D | |
| E | |
| F | |
| G | |
| H | |
| I | |
| J | |

| | |
|---|---|
| A | 5 |
| B | 6 |
| C | 5 |
| D | |
| E | |
| F | |
| G | |
| H | |
| I | |
| J | |

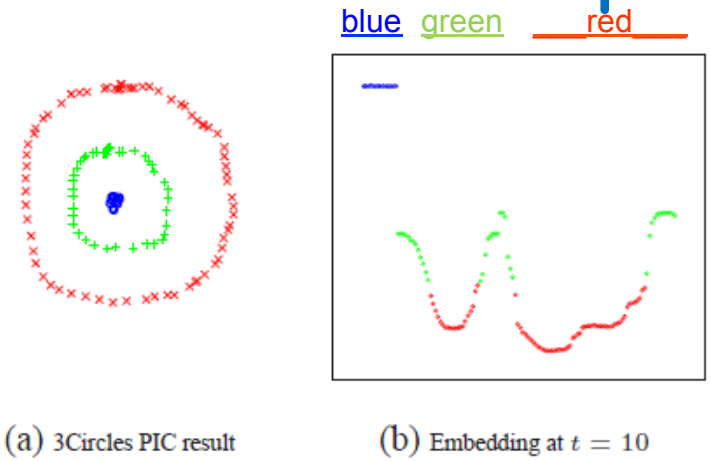


# Repeated averaging with neighbors as a clustering method

- 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 $v^2$, $v^3$, …

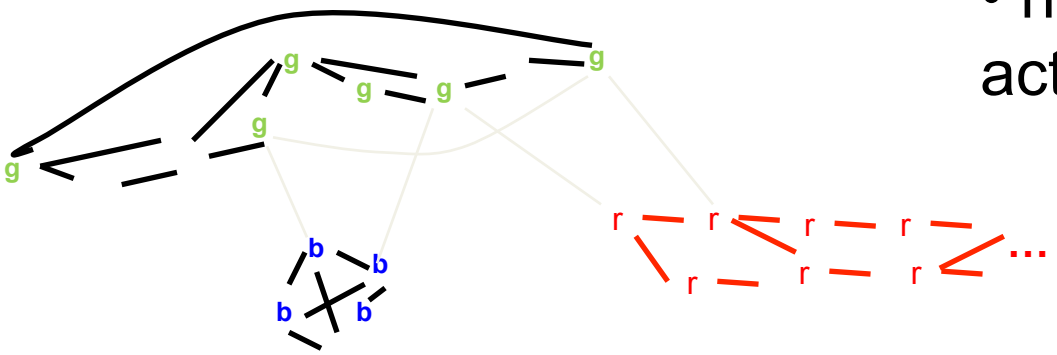
- Variants widely used for *semi-supervised* learning
  - clamping of labels for nodes with known labels
- Without clamping, will converge to constant $v^\dagger$
- What are the *dynamics* of this process?

# Repeated averaging with neighbors on a sample problem…

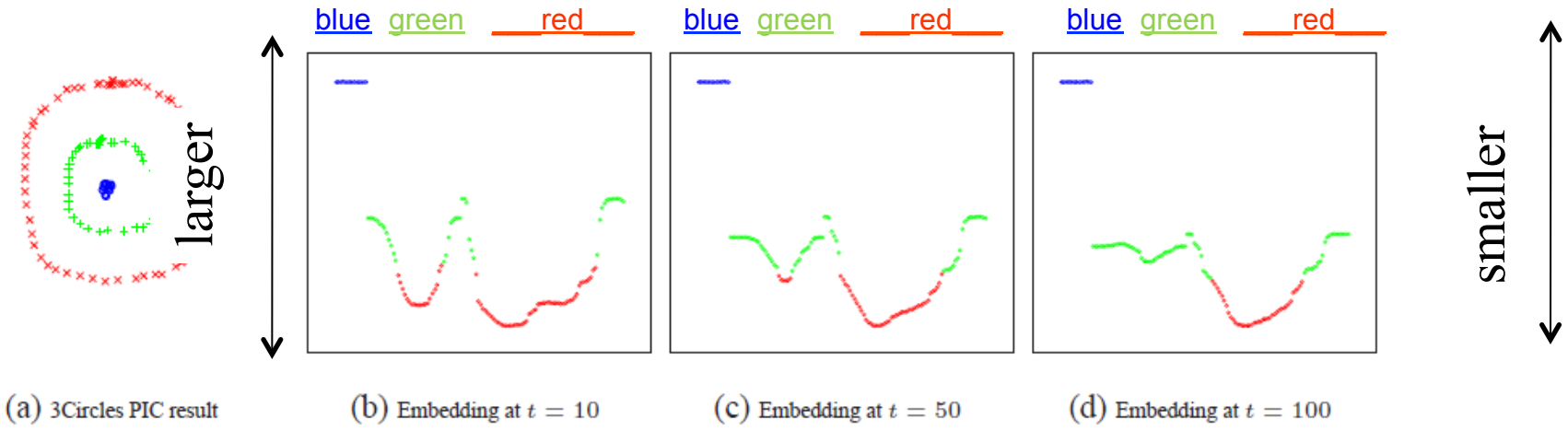



(a) 3Circles PIC result (b) Embedding at $t = 10$

- 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

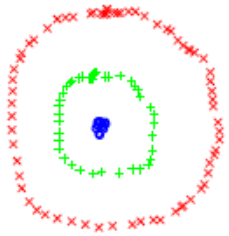


# Repeated averaging with neighbors on a sample problem…

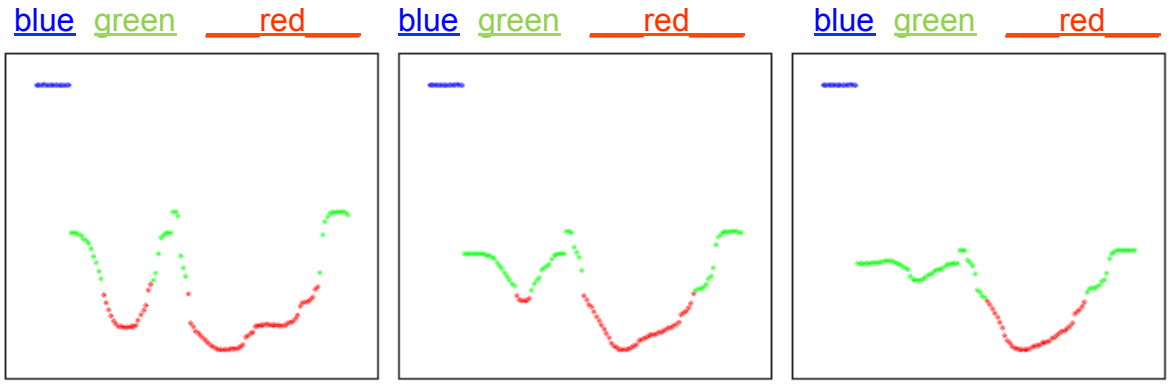



(a) 3Circles PIC result (b) Embedding at $t = 10$ (c) Embedding at $t = 50$ (d) Embedding at $t = 100$

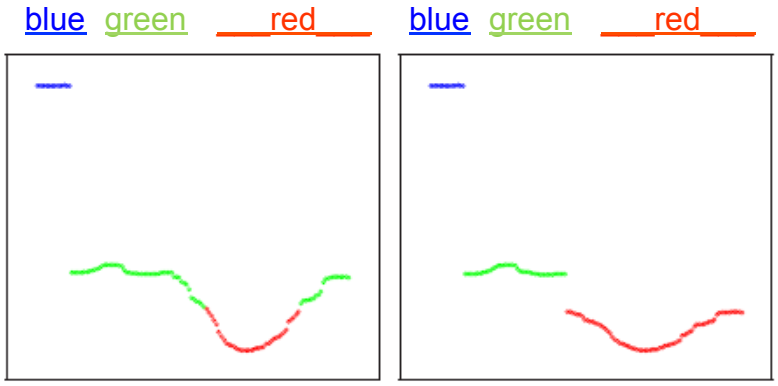
# Repeated averaging with neighbors on a sample problem…



(a) 3Circles PIC result

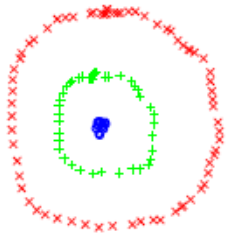



(b) Embedding at $t = 10$ (c) Embedding at $t = 50$ (d) Embedding at $t = 100$

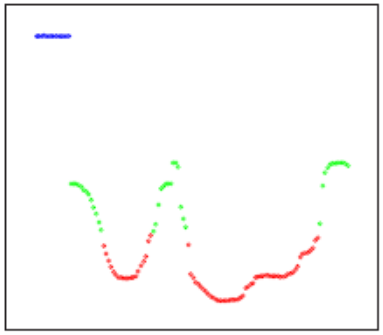



(e) Embedding at $t = 200$ (f) Embedding at $t = 400$

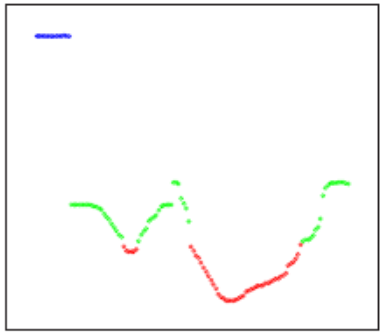
# Repeated averaging with neighbors on a sample problem…



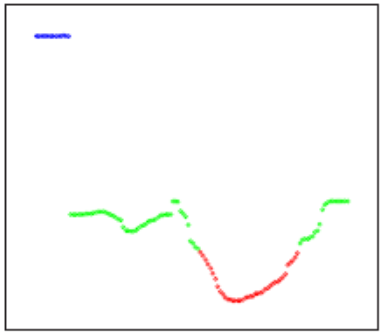
(a) 3Circles PIC result



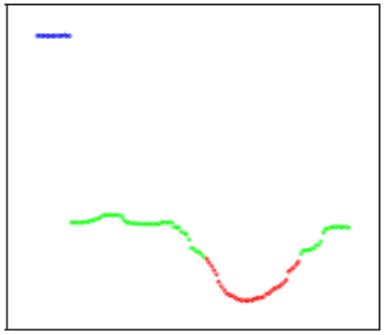
(b) Embedding at $t = 10$



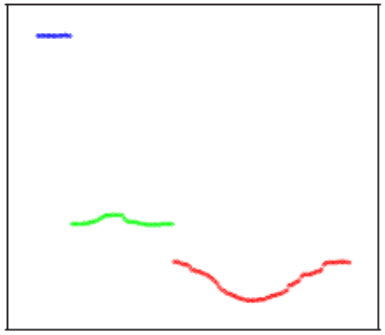
(c) Embedding at $t = 50$



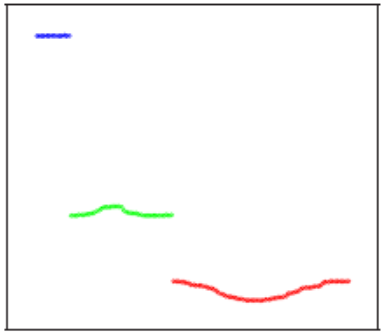
(d) Embedding at $t = 100$



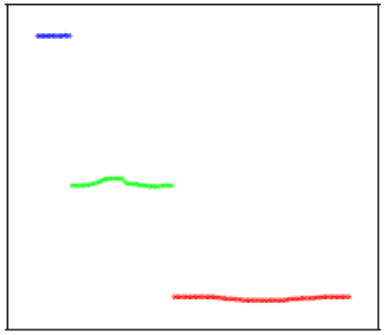
(e) Embedding at $t = 200$



(f) Embedding at $t = 400$



(g) Embedding at $t = 600$



(h) Embedding at $t = 1000$

# PIC: Power Iteration Clustering

## run power iteration (repeated averaging w/ neighbors) with early stopping

1. Pick an initial vector $\mathbf{v^0}$.
2. Set $\mathbf{v^{t+1}} \leftarrow \frac{W\mathbf{v^t}}{\|W\mathbf{v^t}\|_1}$ and $\delta^{t+1} \leftarrow |\mathbf{v^{t+1}} - \mathbf{v^t}|$.
3. Increment $t$ and repeat above step until $|\delta^t - \delta^{t-1}| \simeq \mathbf{0}$.
4. Use $k$-means to cluster points on $\mathbf{v^t}$ 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!