Semi-Supervised Learning With Graphs

William Cohen

Review – Graph Algorithms so far….

- PageRank and how to scale it up

HW6

- Personalized PageRank/Random Walk with Restart and
  - how to implement it
  - how to use it for extracting part of a graph

We *might* come back to this more

- Other uses for graphs?
  - not so much

You can also look at the **March 19 lecture** from the **spring 2015** version of this class.

Main topics today

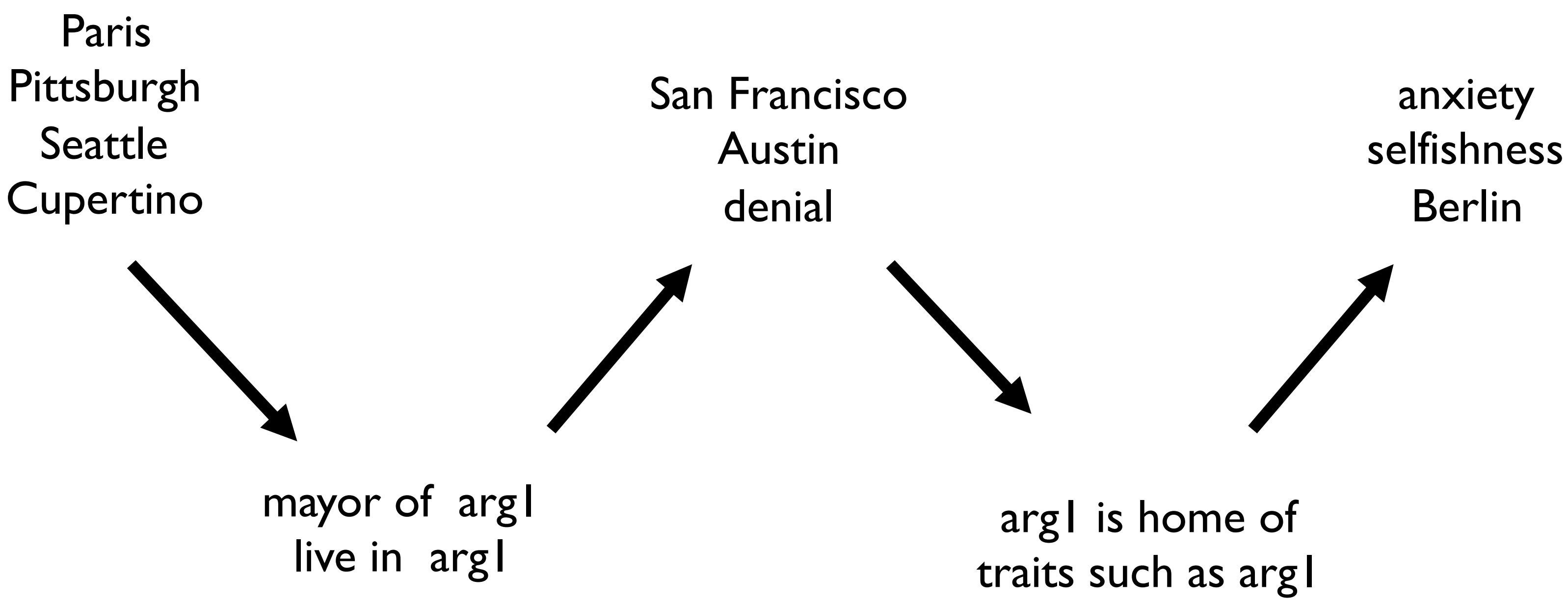
- Scalable semi-supervised learning on graphs
  - SSL with RWR
  - SSL with coEM/wvRN/HF
- Scalable unsupervised learning on graphs
  - Power iteration clustering
  - …

Semi-supervised learning

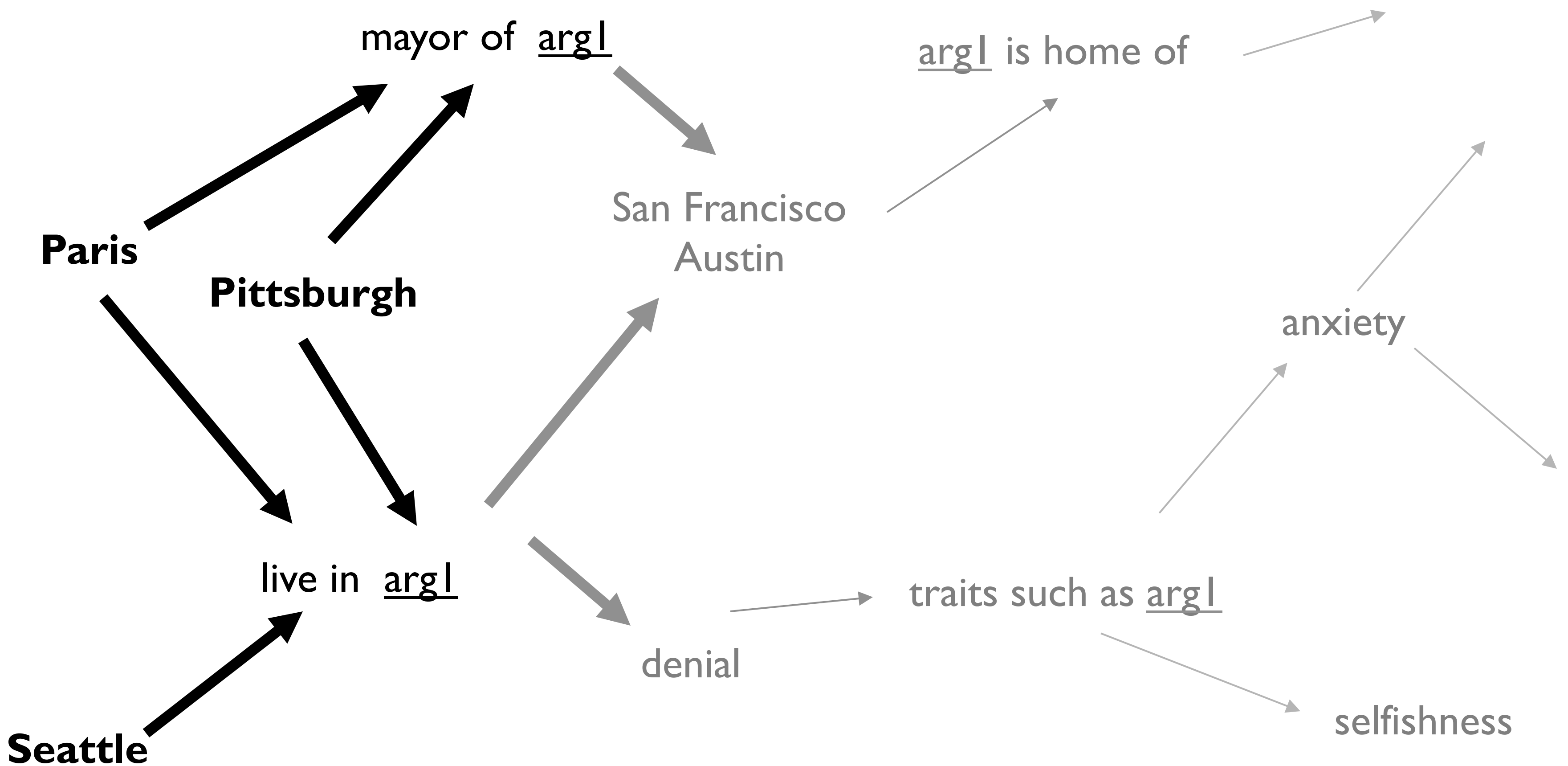
- A pool of labeled examples L
- A (usually larger) pool of unlabeled examples U
- Can you improve accuracy somehow using U?

Semi-Supervised Bootstrapped Learning/Self-training

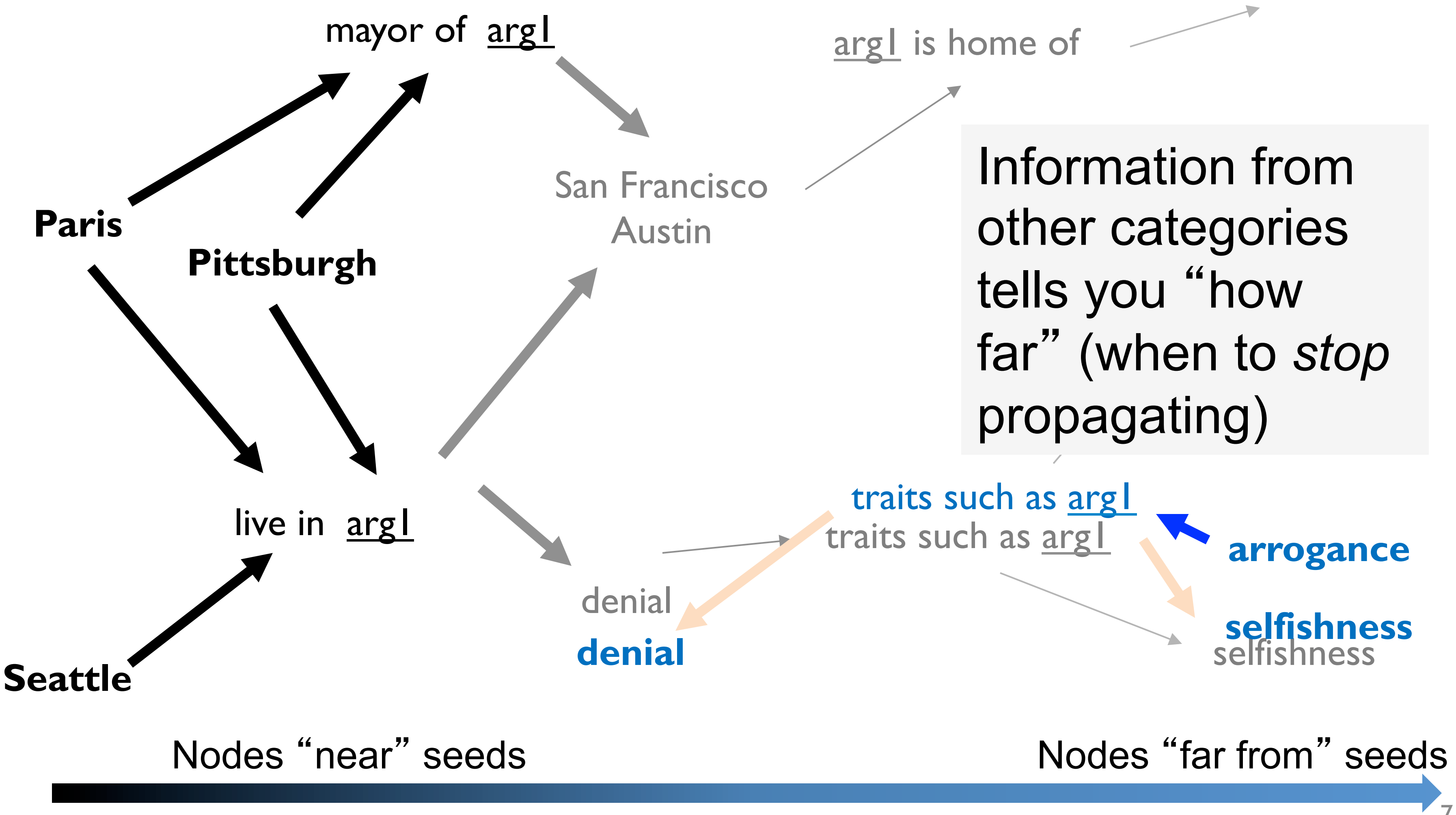
Extract cities:



Semi-Supervised Bootstrapped Learning via Label Propagation



Semi-Supervised Bootstrapped Learning via Label Propagation



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

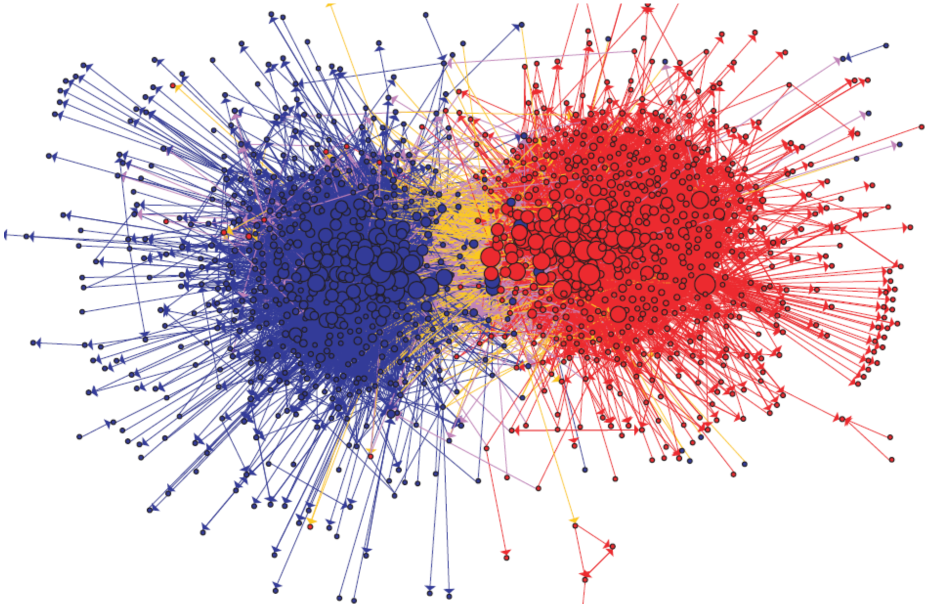
Semi-Supervised Classification of Network Data Using Very Few Labels

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

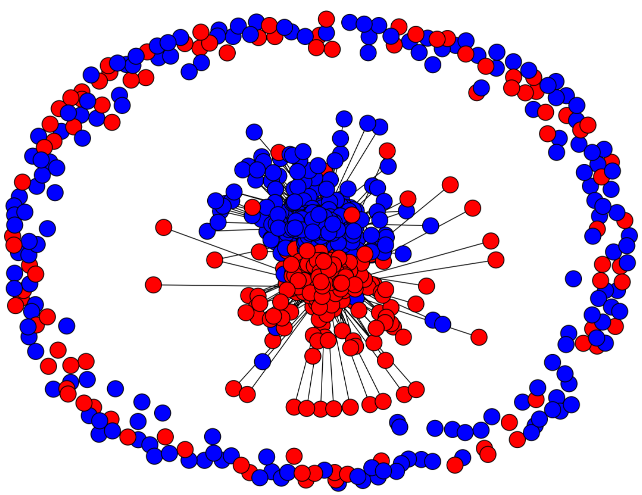
William W. Cohen
Carnegie Mellon University, Pittsburgh, Pennsylvania
Email: wcohen@cs.cmu.edu

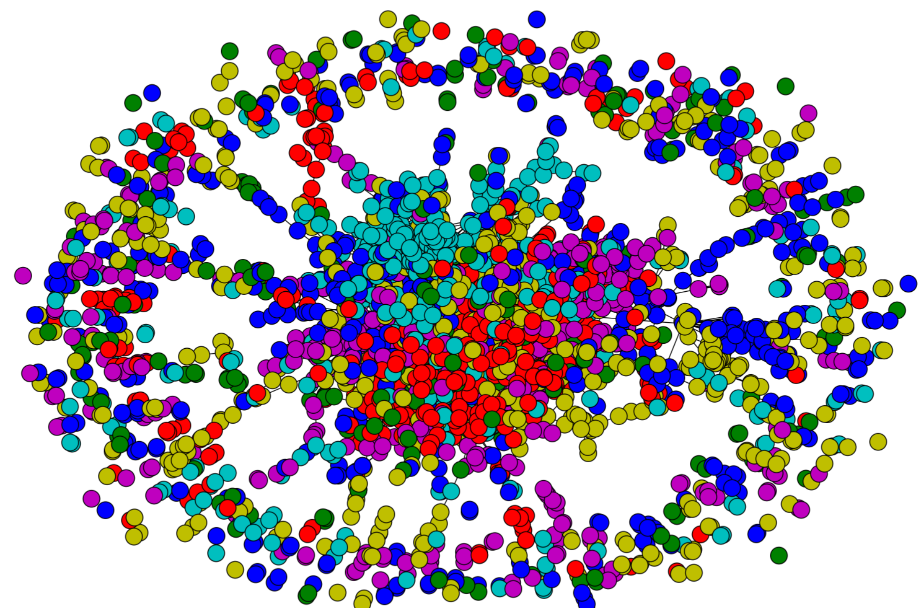
ASONAM-2010 (Advances in Social Networks Analysis and Mining)

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

Fig. 1. The MultiRankWalk algorithm.

RWR - fixpoint of:
$\mathbf{r} = (1 - d)\mathbf{u} + dW\mathbf{r}$

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

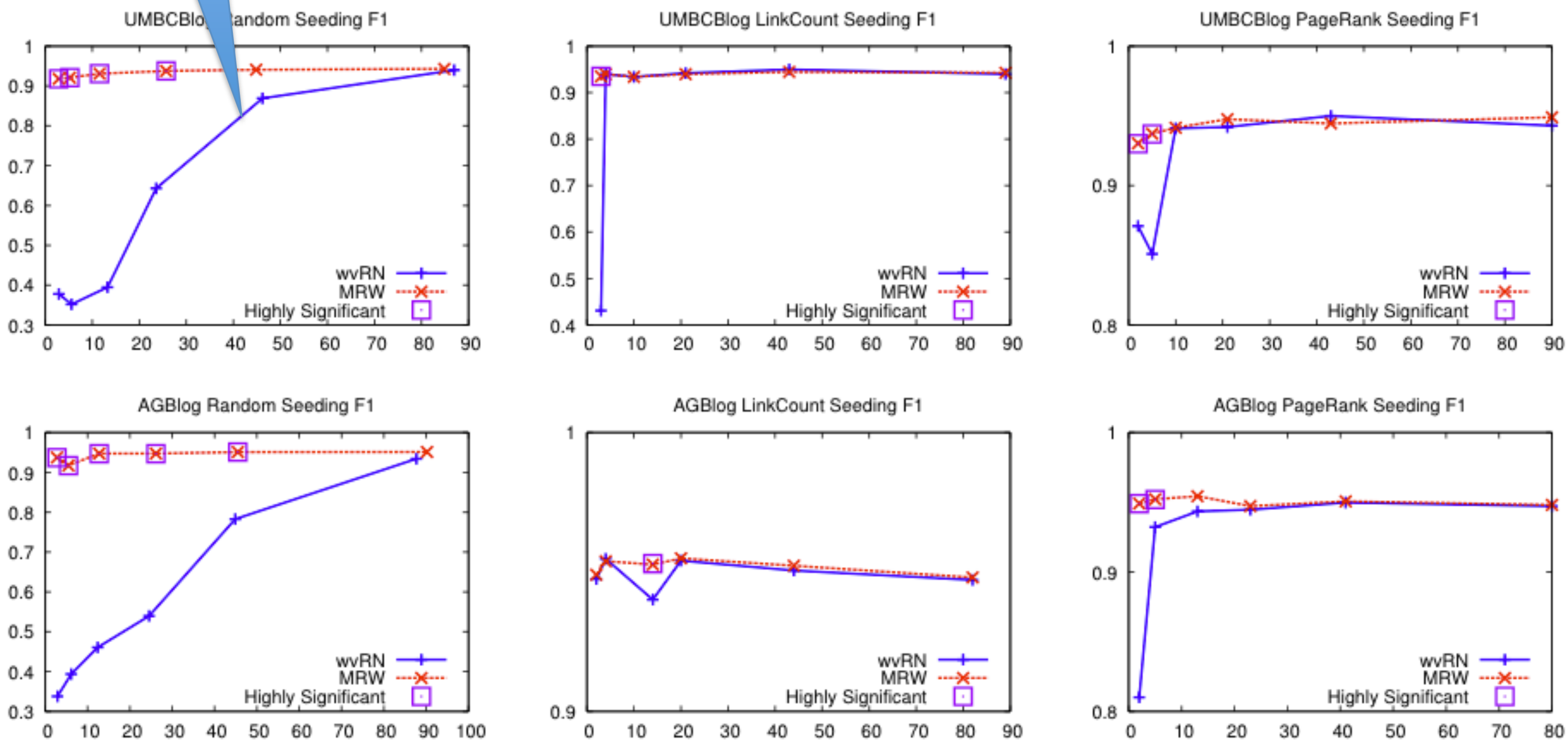
We'll discuss this soon….

Results – Blog data

Random

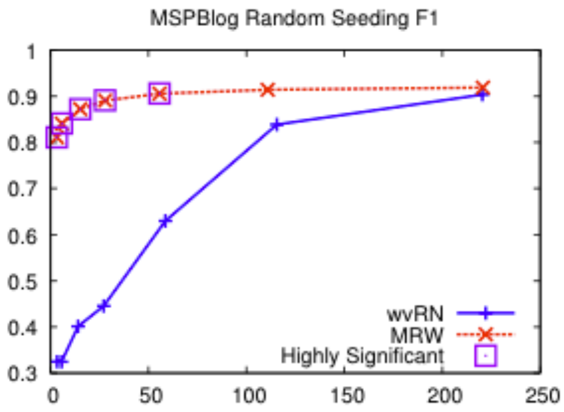
Degree

PageRank

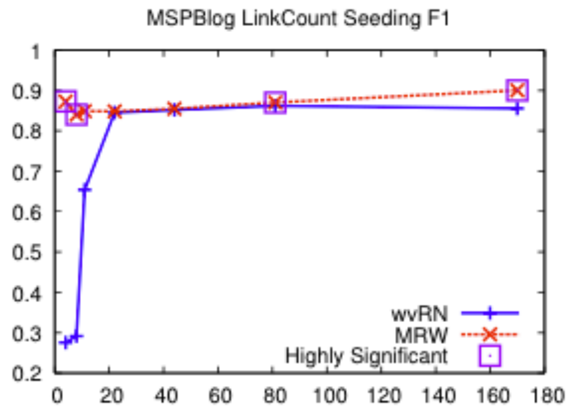


Results – More blog data

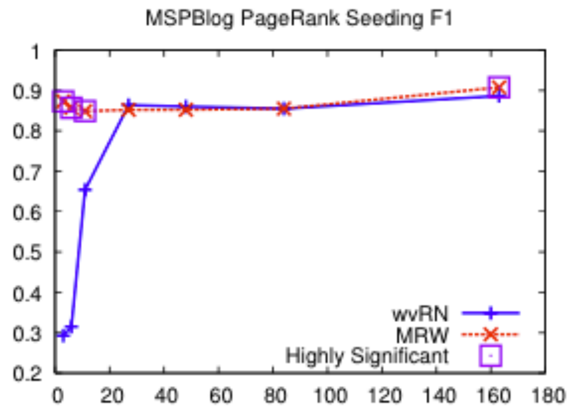
Random



Degree

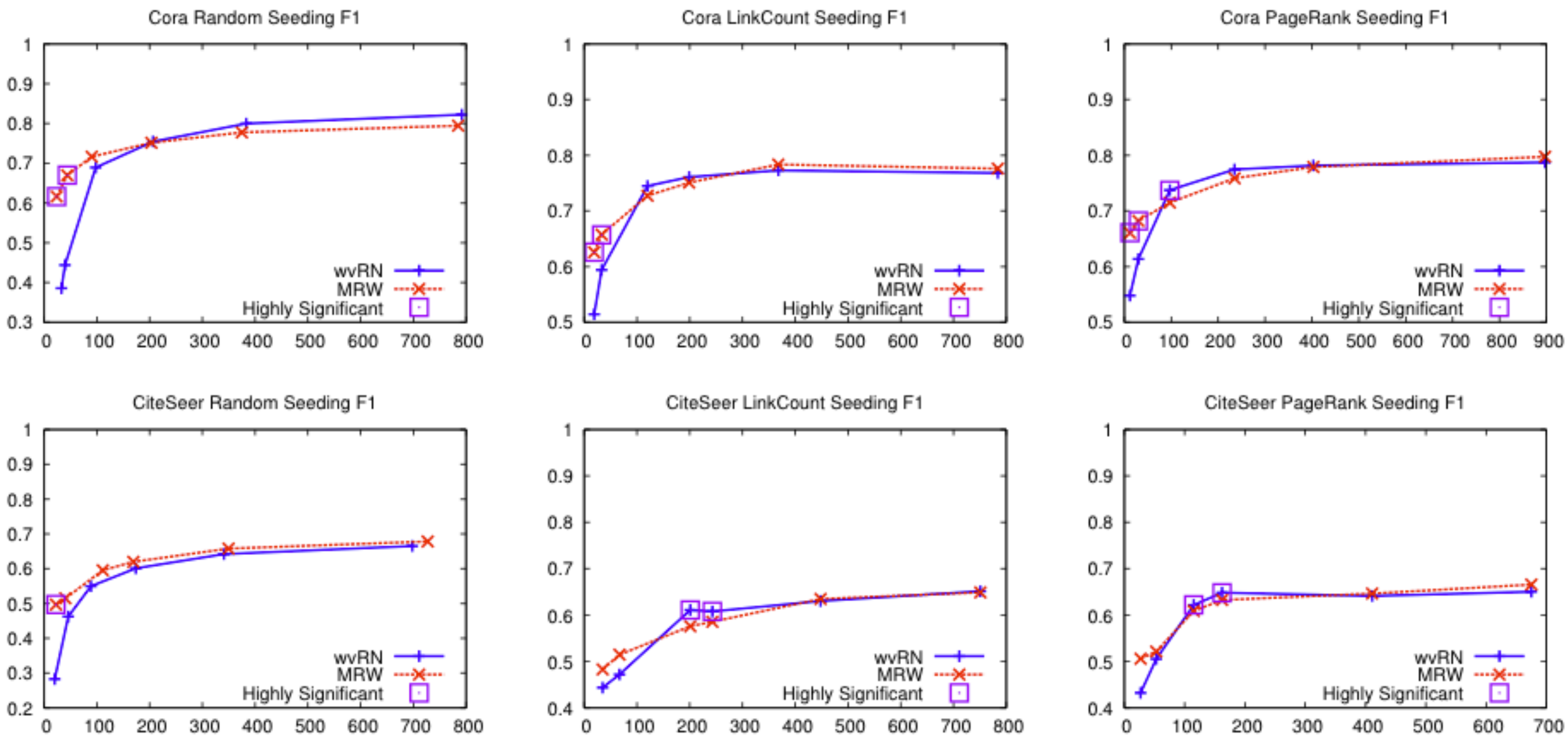


PageRank

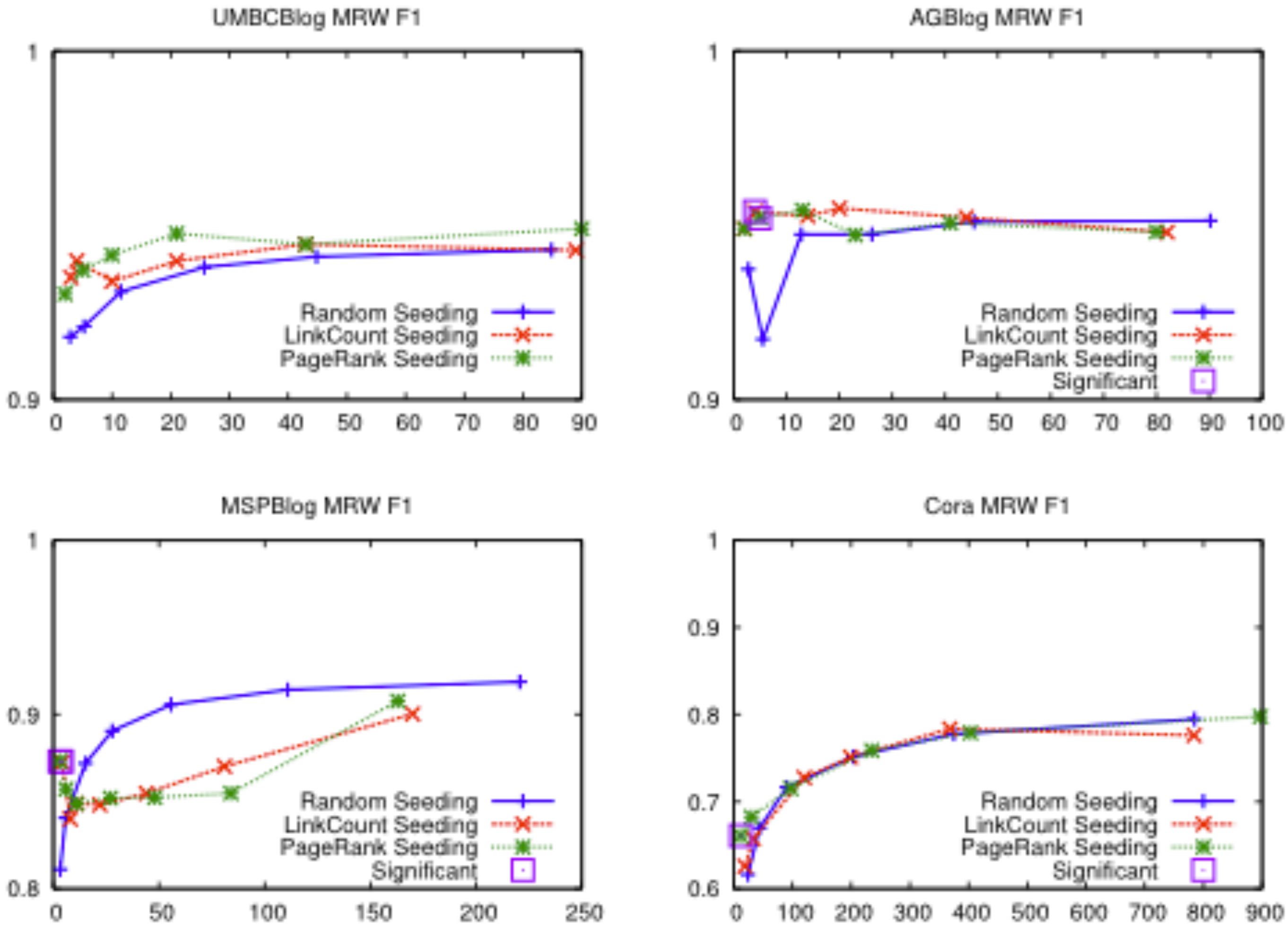


Results – Citation data

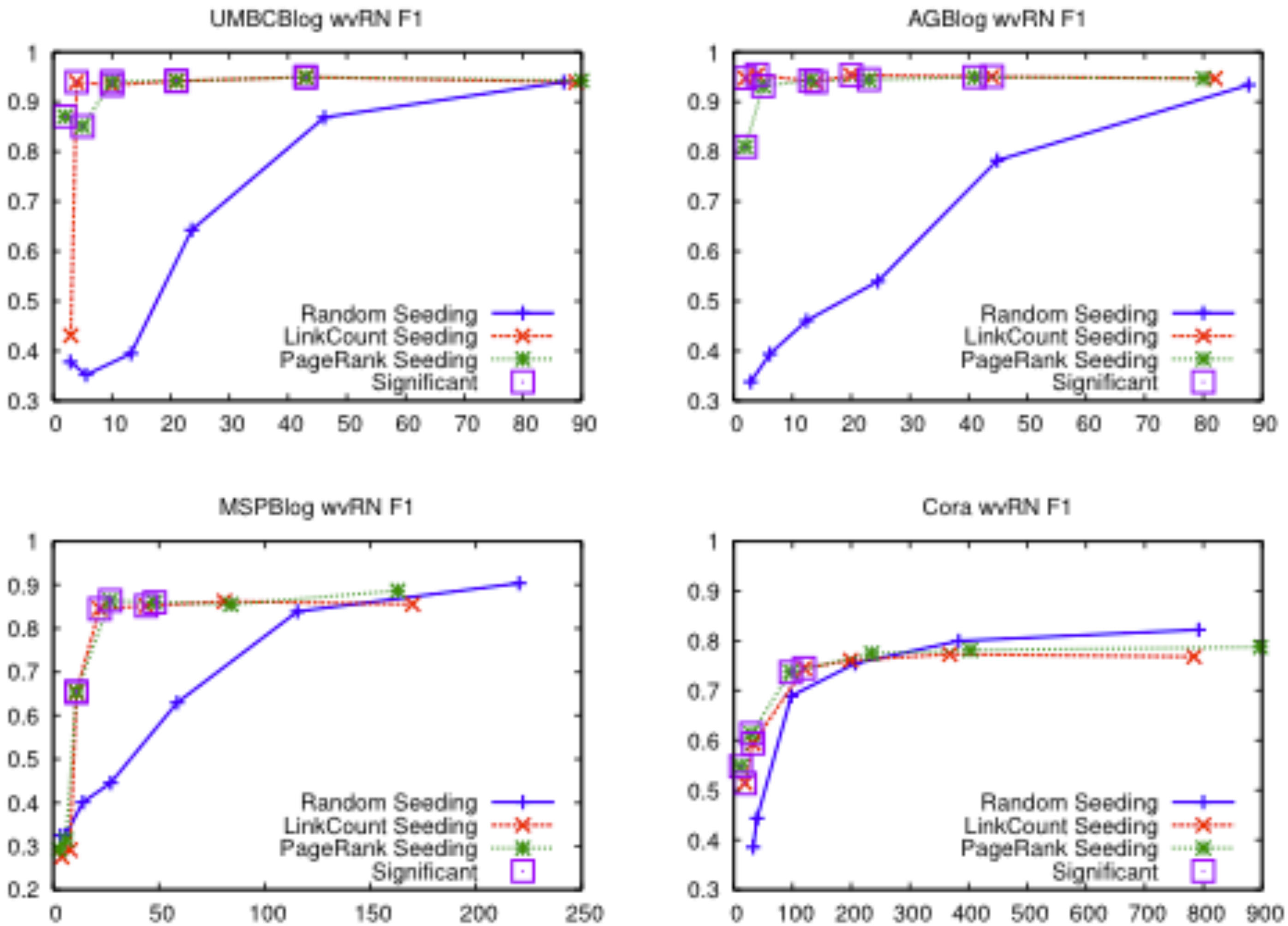
Random Degree PageRank



Seeding – MultiRankWalk



Seeding – HF/wvRN



What is HF aka coEM aka wvRN?

CoEM/HF/wvRN

- One definition [MacKassey & 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 $\mathcal{N}_i$:

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

Another definition: A *harmonic field* – the score of each node in the graph is the harmonic (linearly weighted) average of its neighbors' scores;

[X. Zhu, Z. Ghahramani, and J. Lafferty, ICML 2003]

CoEM/wvRN/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.

CoEM/wvRN/HF

- One algorithm with several justifications....
- One is to start with co-training with a naïve Bayes learner

- And compare to an EM version of naïve Bayes
  - E: soft-classify unlabeled examples with NB classifier
  - M: re-train classifier with soft-labeled examples

Algorithm	# Labeled	# Unlabeled	Error
Naive Bayes	788	–0–	3.3%
Co-training	12	776	5.4%
EM	12	776	4.3%
Naive Bayes	12	–0–	13.0%

CoEM/wvRN/HF

- A second experiment
  - each + example: concatenate features from two documents, one of class A+, one of class B+
  - each - example: concatenate features from two documents, one of class A-, one of class B-
  - features are prefixed with "A", "B" ➔ disjoint

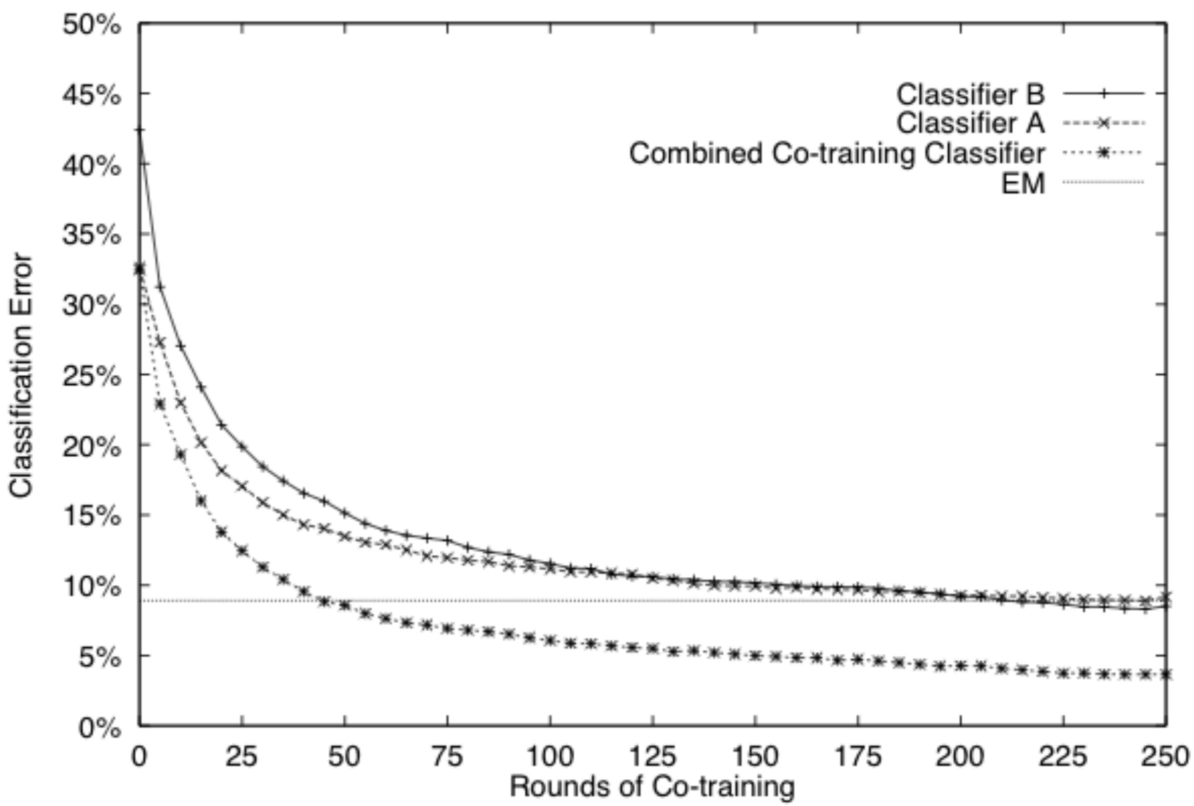
Table 3: The setup of the News 2x2 dataset. This data has class-conditional independence and redundancy between its two feature sets.

Class	Feature Set A	Feature Set B
Pos	comp.os.ms-windows.misc	talk.politics.misc
Neg	comp.sys.ibm.pc.hardware	talk.politics.guns

CoEM/wvRN/HF

- A second experiment
  - each + example: concatenate features from two documents, one of class A+, one of class B+
  - each - example: concatenate features from two documents, one of class A-, one of class B-
  - features are prefixed with “A”, “B” ➔ disjoint
- NOW co-training outperforms EM

Algorithm	# Labeled	# Unlabeled	Error
Naive Bayes	1006	–0–	3.9%
Co-training	6	1000	3.7%
EM	6	1000	8.9%
Naive Bayes	6	–0–	34.0%



CoEM/wvRN/HF

- Co-training with a naïve Bayes learner

- *vs* an EM version of naïve Bayes
  - E: soft-classify unlabeled examples with NB classifier
  - M: re-train classifier with soft-labeled examples

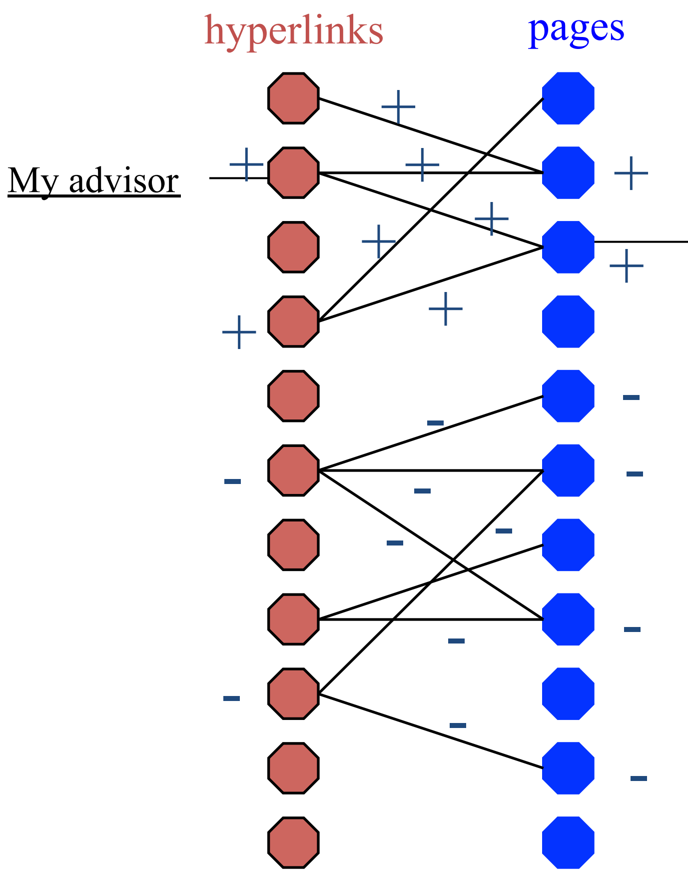
Method	Uses Feature Split? Yes	No
Incremental	co-training	self-training
Iterative	co-EM	EM

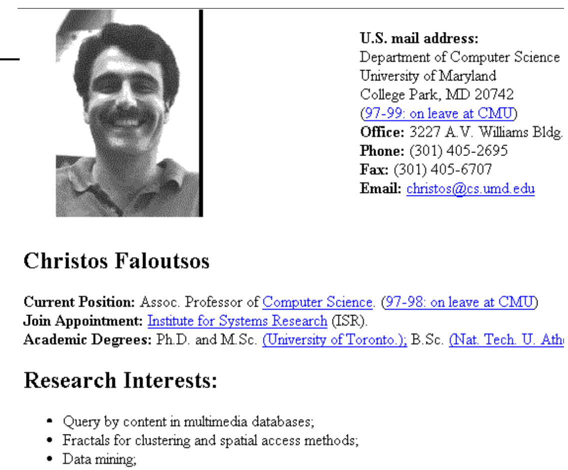
incremental hard assignments

iterative soft assignments

Method	Uses Random Feature Split? Yes	No
Incremental	5.5%	5.8%
Iterative	5.1%	8.9%

Co-Training Rote Learner





U.S. mail address:
Department of Computer Science
University of Maryland
College Park, MD 20742
(97-99: on leave at CMU)
Office: 3227 A.V. Williams Bldg.
Phone: (301) 405-2695
Fax: (301) 405-6707
Email: christos@cs.umd.edu

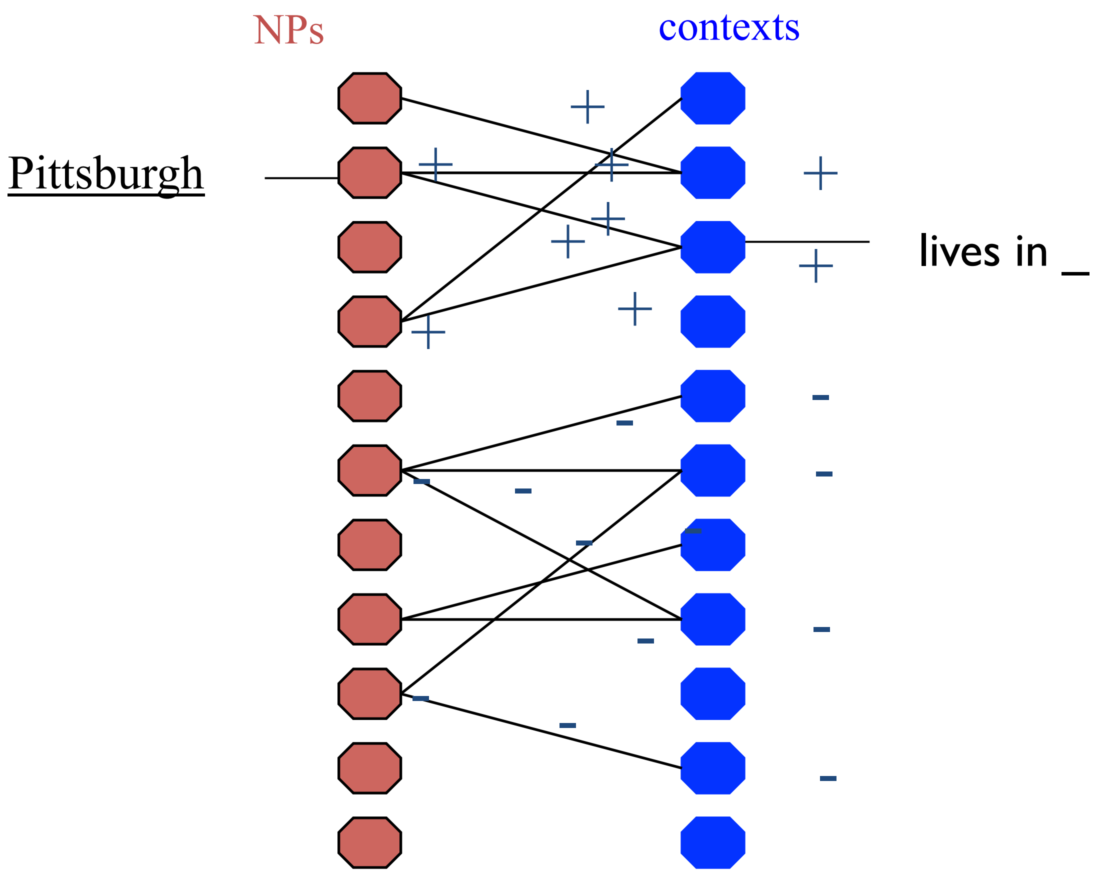
Christos Faloutsos

Current Position: Assoc. Professor of Computer Science. (97-98: on leave at CMU)
Join Appointment: Institute for Systems Research (ISR).
Academic Degrees: Ph.D. and M.Sc. (University of Toronto.); B.Sc. (Nat. Tech. U. Ath

Research Interests:

- Query by content in multimedia databases;
- Fractals for clustering and spatial access methods;
- Data mining;

Co-EM Rote Learner: equivalent to HF on a bipartite graph



What is HF aka coEM aka wvRN?

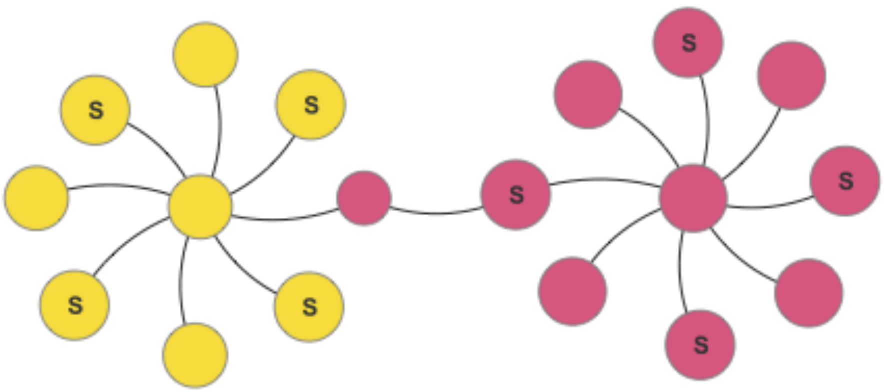
$$P(x_i = c|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = c|\mathcal{N}_j),$$

Algorithmically:

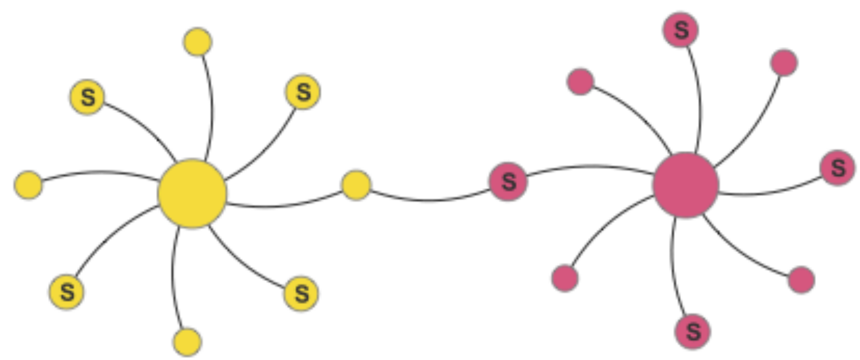
- HF propagates weights and then resets the seeds to their initial value
- MRW propagates weights and does not reset seeds

MultiRank Walk vs HF/wvRN/CoEM

Seeds are marked S

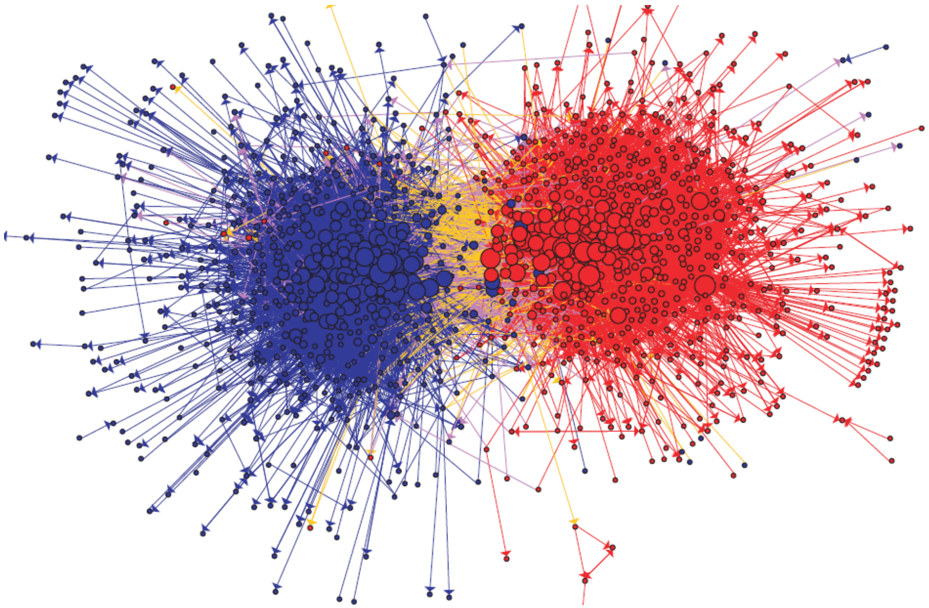


HF

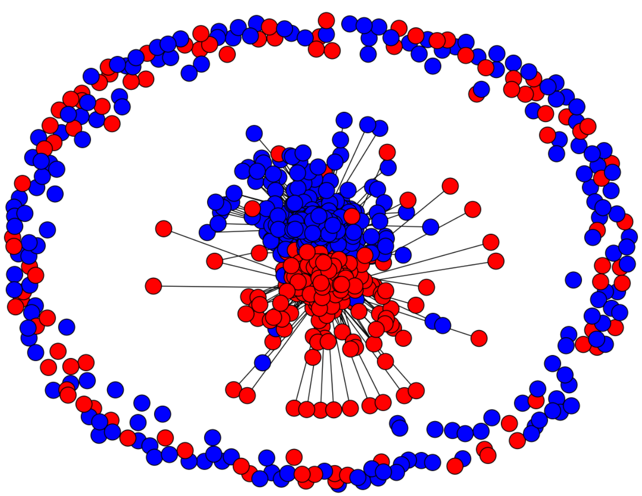


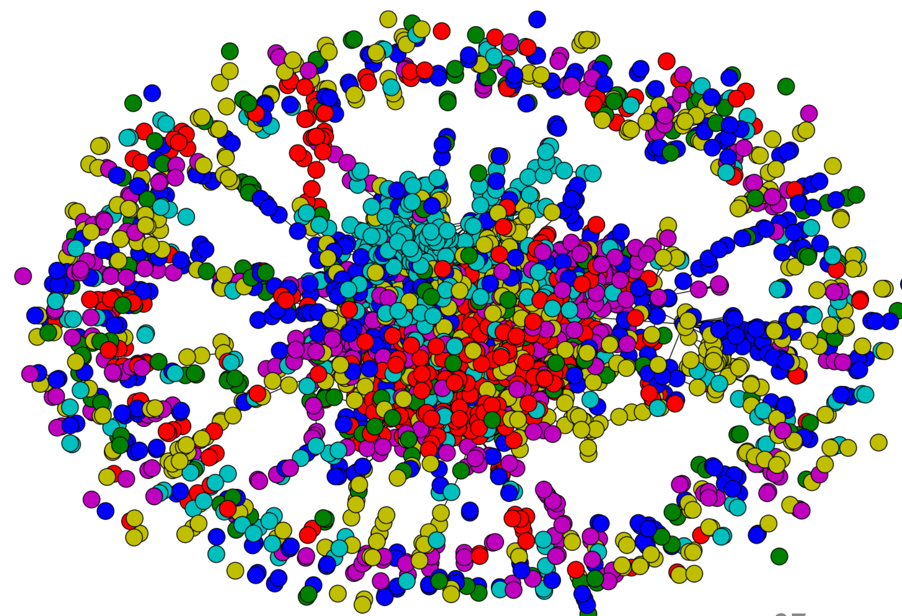
MRW

Back to Experiments: Network Datasets with Known Classes



- •UBMCBlog
- •AGBlog
- •MSPBlog
- •Cora
- •Citeseer





MultiRankWalk vs wvRN/HF/CoEM

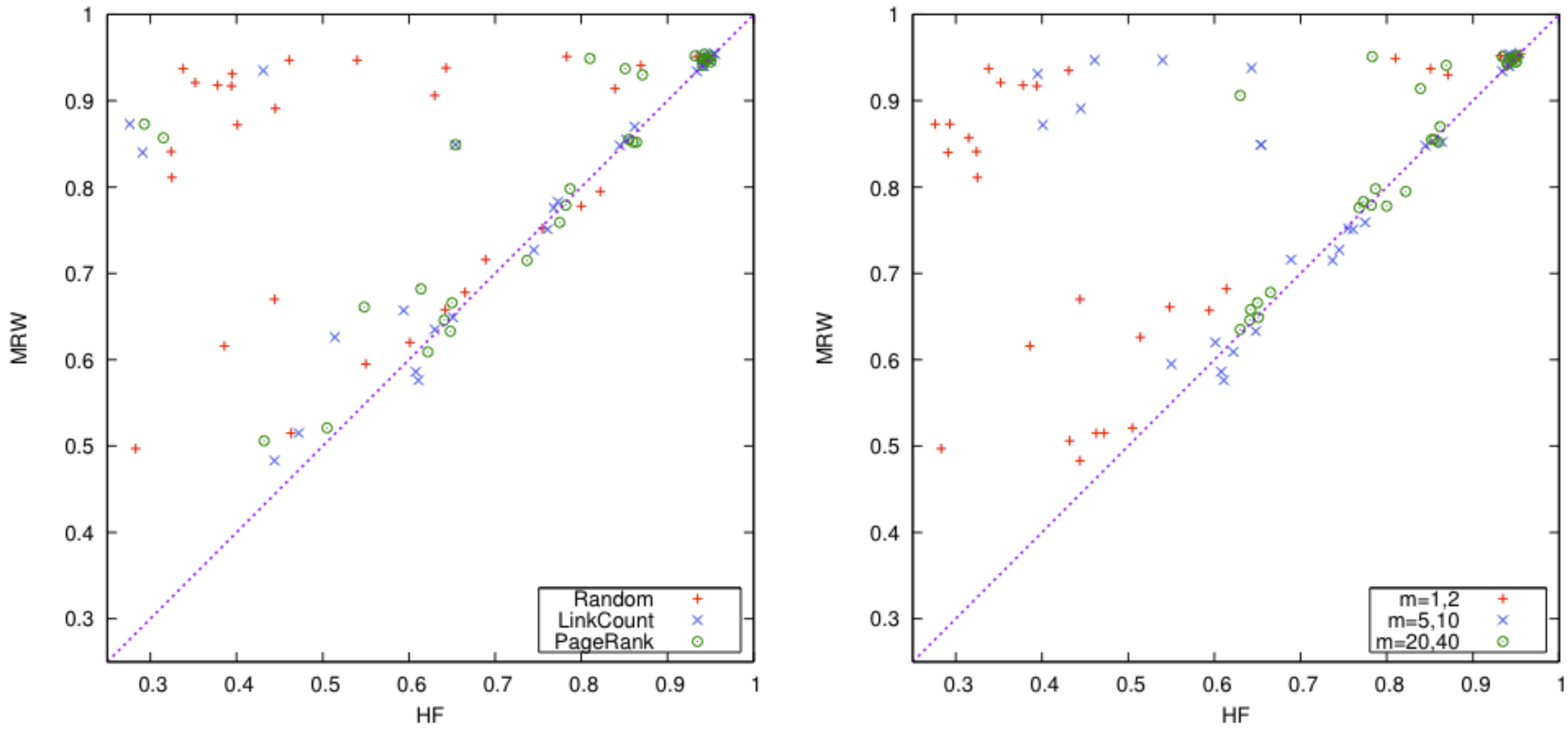


Figure 2.6: Scatter plots of HF F1 score versus MRW F1 score. The left plot marks different seeding preferences and the right plot marks varying amount of training labels determined by m.

How well does MWR work?

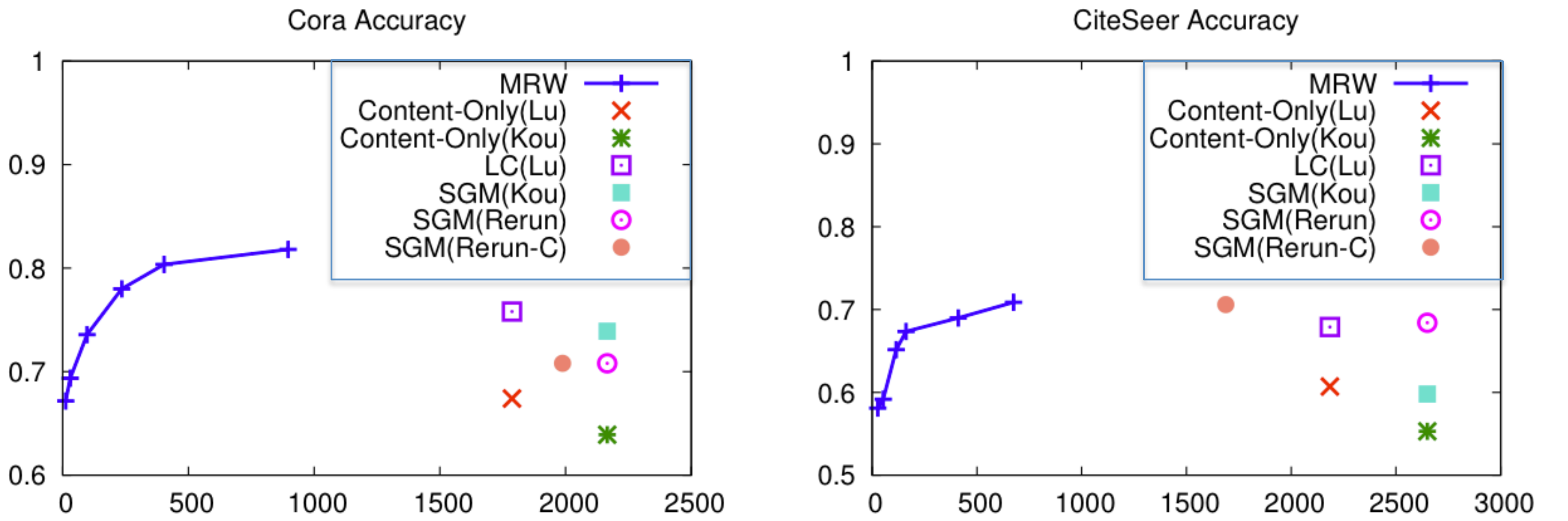


Fig. 5. Citation datasets results compared to supervised relational learning methods. The x-axis indicates number of labeled instances and y-axis indicates labeling accuracy.

Parameter Sensitivity

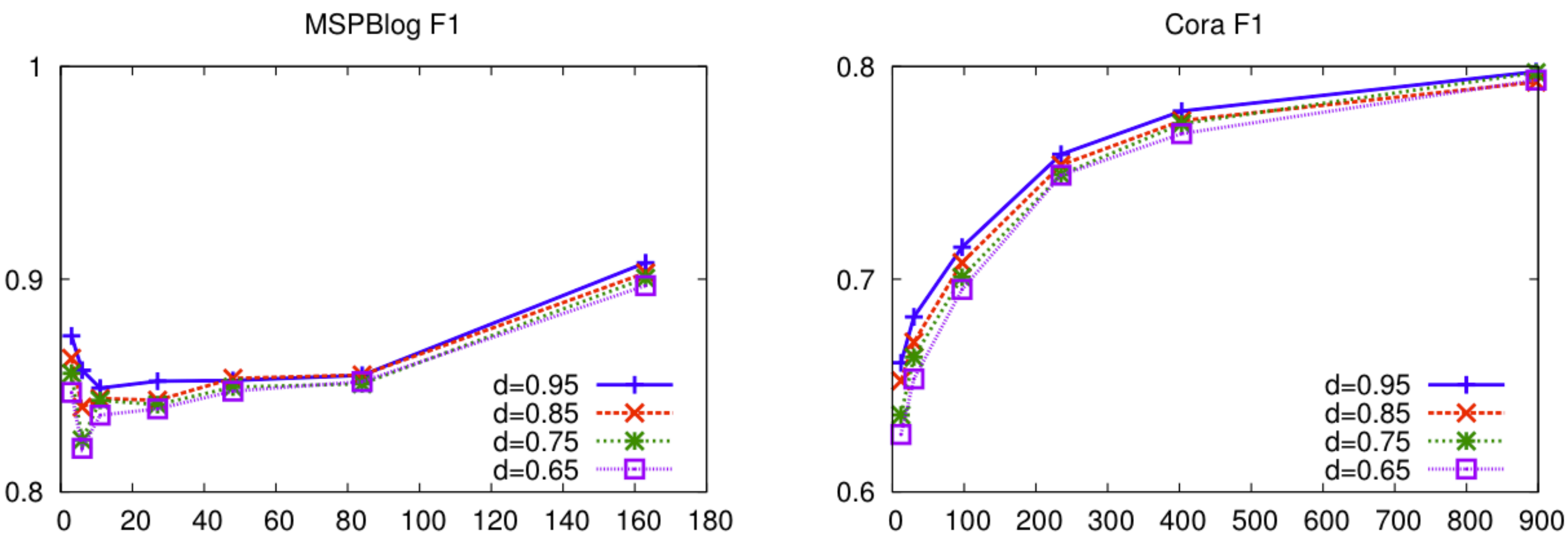


Fig. 7. Results on three datasets varying the damping factor. The x-axis indicates number of labeled instances and y-axis indicates labeling macro-averaged F1 score.

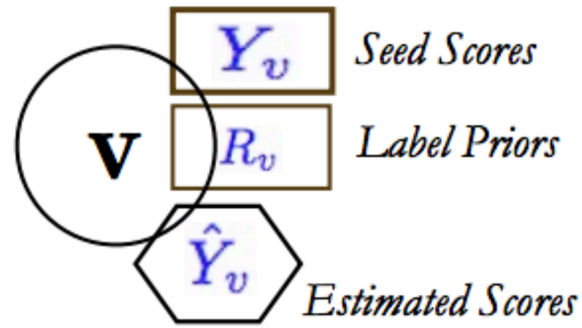
Semi-supervised learning

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

- These methods are different from EM
  - optimizes Pr(Data|Model)
- How do SSL learning methods (like label propagation) relate to optimization?

SSL as optimization slides from Partha Talukdar

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)

yet another name for HF/wvRN/coEM

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)

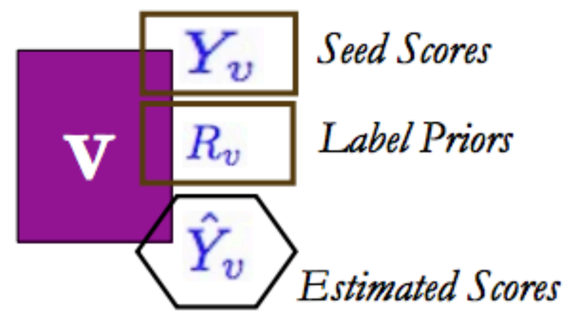
- Smoothness
  - two nodes connected by an edge with high weight should be assigned similar labels
- Solution satisfies harmonic property

Modified Adsorption (MAD)

[Talukdar and Crammer, ECML 2009]

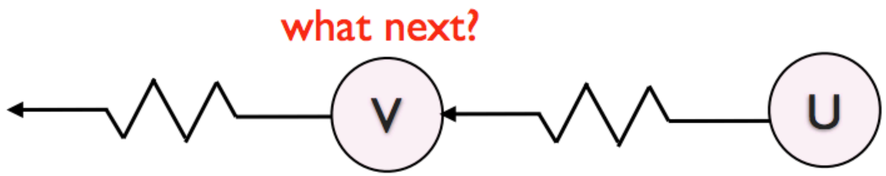
$$\arg\min_{\hat{\mathbf{Y}}} \sum_{l=1}^{m+1} \left[\overbrace{\|\mathbf{S}\hat{\mathbf{Y}}_l - \mathbf{S}\mathbf{Y}_l\|^2}^{\text{match seeds}} + \mu_1 \overbrace{\sum_{u,v} \mathbf{M}_{uv}(\hat{\mathbf{Y}}_{ul} - \hat{\mathbf{Y}}_{vl})^2}^{\text{smoothness}} + \overbrace{\mu_2\|\hat{\mathbf{Y}}_l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

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



- $M = W^{\dagger} + W'$ is the symmetrized weight matrix

Random Walk View



- Continue walk with prob. p_v^{cont}
- Assign V's seed label to U with prob. p_v^{inj}
- Abandon random walk with prob. p_v^{abnd}
  - assign U a dummy label

$$p_u^{cont} \times W_{uv}$$

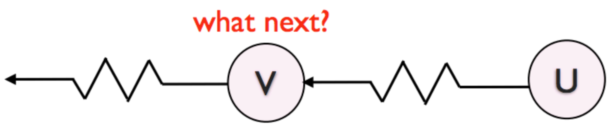
$$\sqrt{p_u^{inj}}$$

$\ldots_{u\top} = p_u^{abnd}$, and 0 for non-dummy labels

Dummy Label

- $\boldsymbol{M} = \boldsymbol{W}^{\dagger} + \boldsymbol{W}'$ is the symmetrized weight matrix

Random Walk View



- Continue walk with prob. $\mathbf{p}_{\mathbf{v}}^{\mathbf{cont}}$
- Assign V's seed label to U with prob. $\mathbf{p}_{\mathbf{v}}^{\mathbf{inj}}$
- Abandon random walk with prob. $\mathbf{p}_{\mathbf{v}}^{\mathbf{abnd}}$
  - assign U a dummy label

New Edge Weight

$$W'_{uv} = p_u^{cont} \times W_{uv}$$

$$S_{uu} = \sqrt{p_u^{inj}}$$

$R_{u\top} = p_u^{abnd}$, and 0 for non-dummy labels

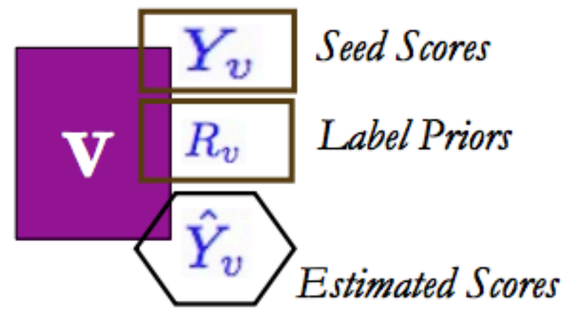
Dummy Label

Modified Adsorption (MAD)

[Talukdar and Crammer, ECML 2009]

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

- m labels, +1 dummy label
- $\boldsymbol{M} = \boldsymbol{W}^{\dagger} + \boldsymbol{W}'$ is the symmetrized weight matrix
- $\hat{\mathbf{Y}}_{vl}$: 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]

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

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

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

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

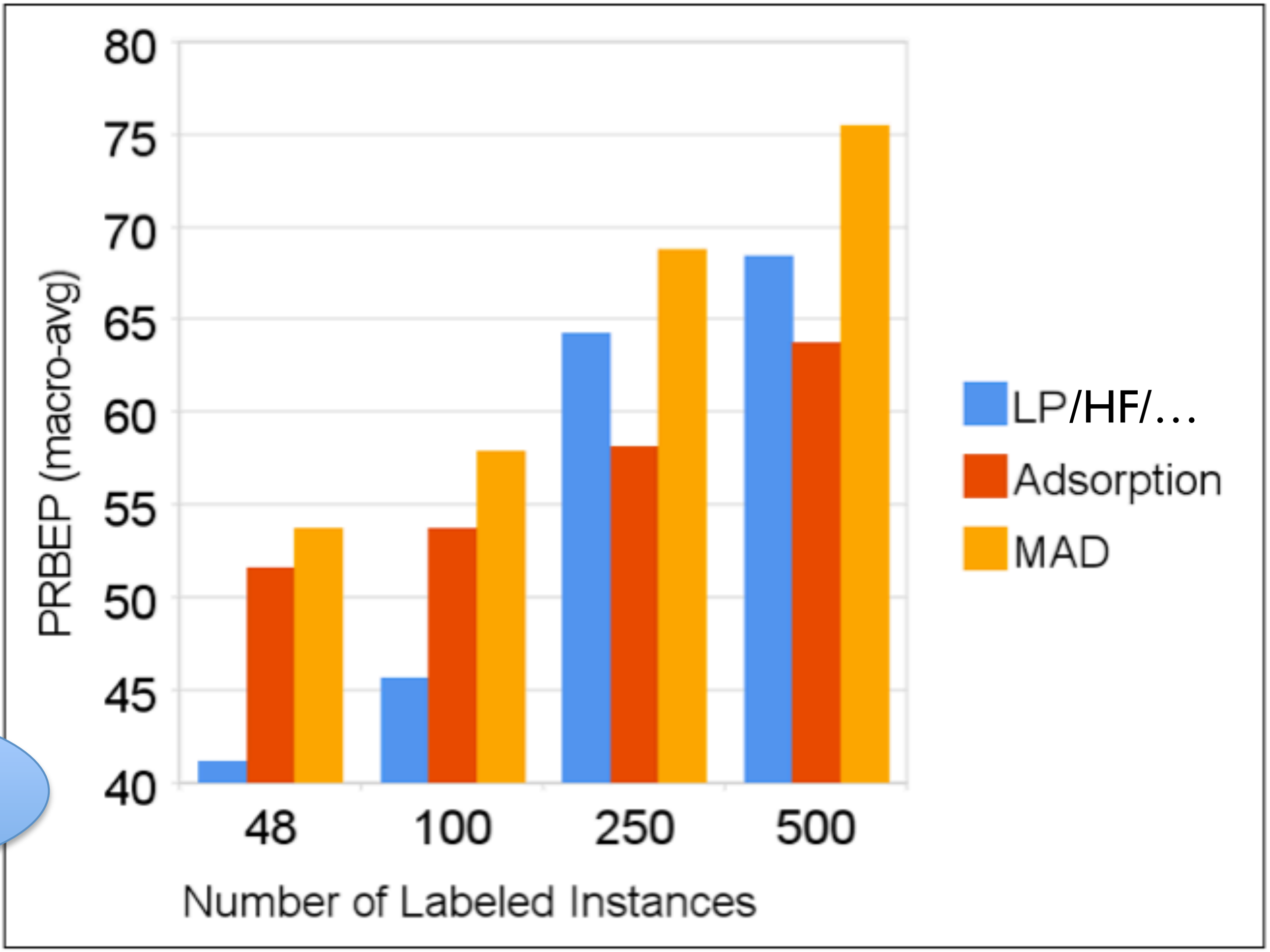
- Extends Adsorption with well-defined optimization
- Importance of a node can be discounted
- Easily Parallelizable: Scalable

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/

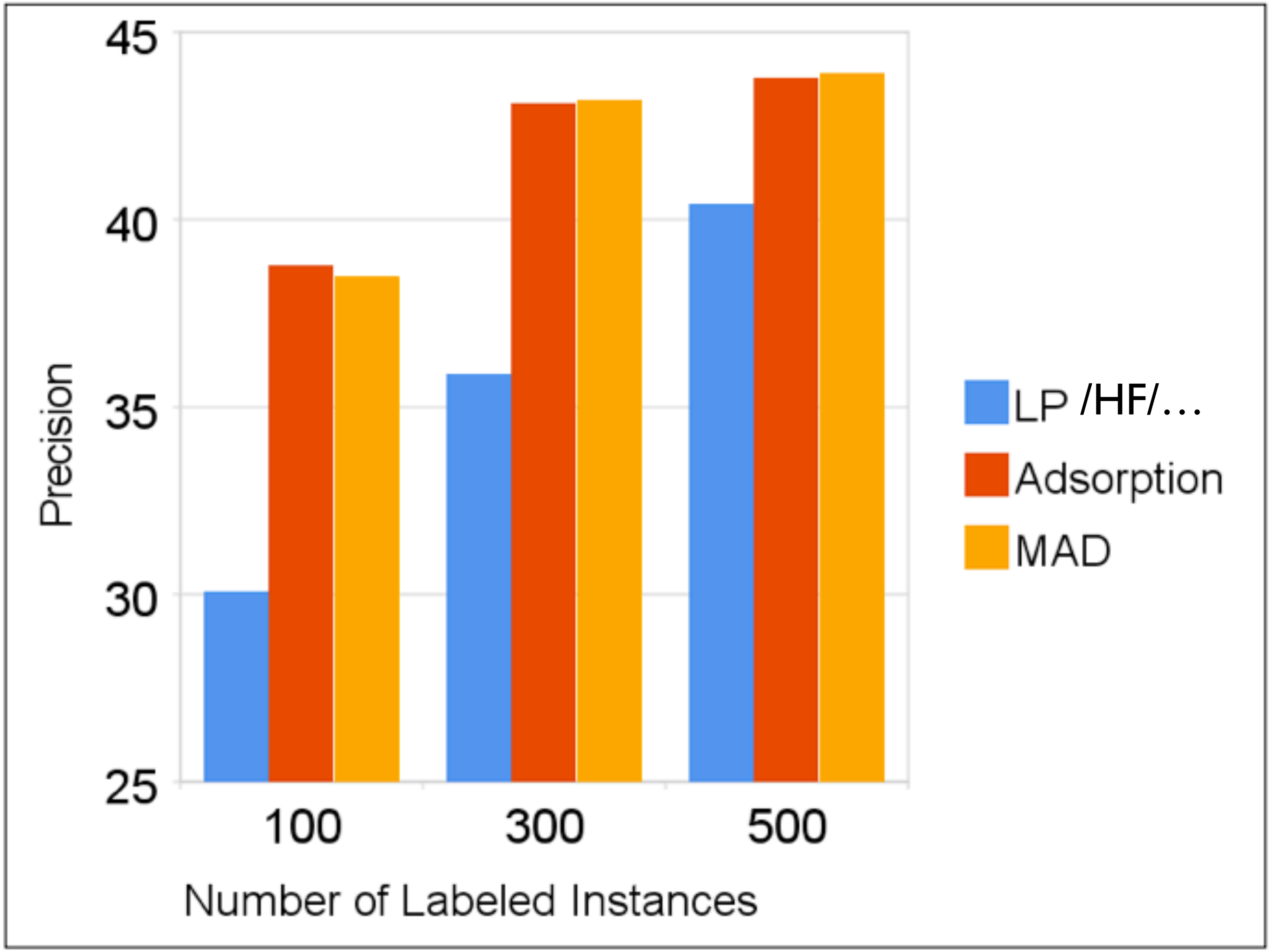
Text Classification



precision-recall break even point

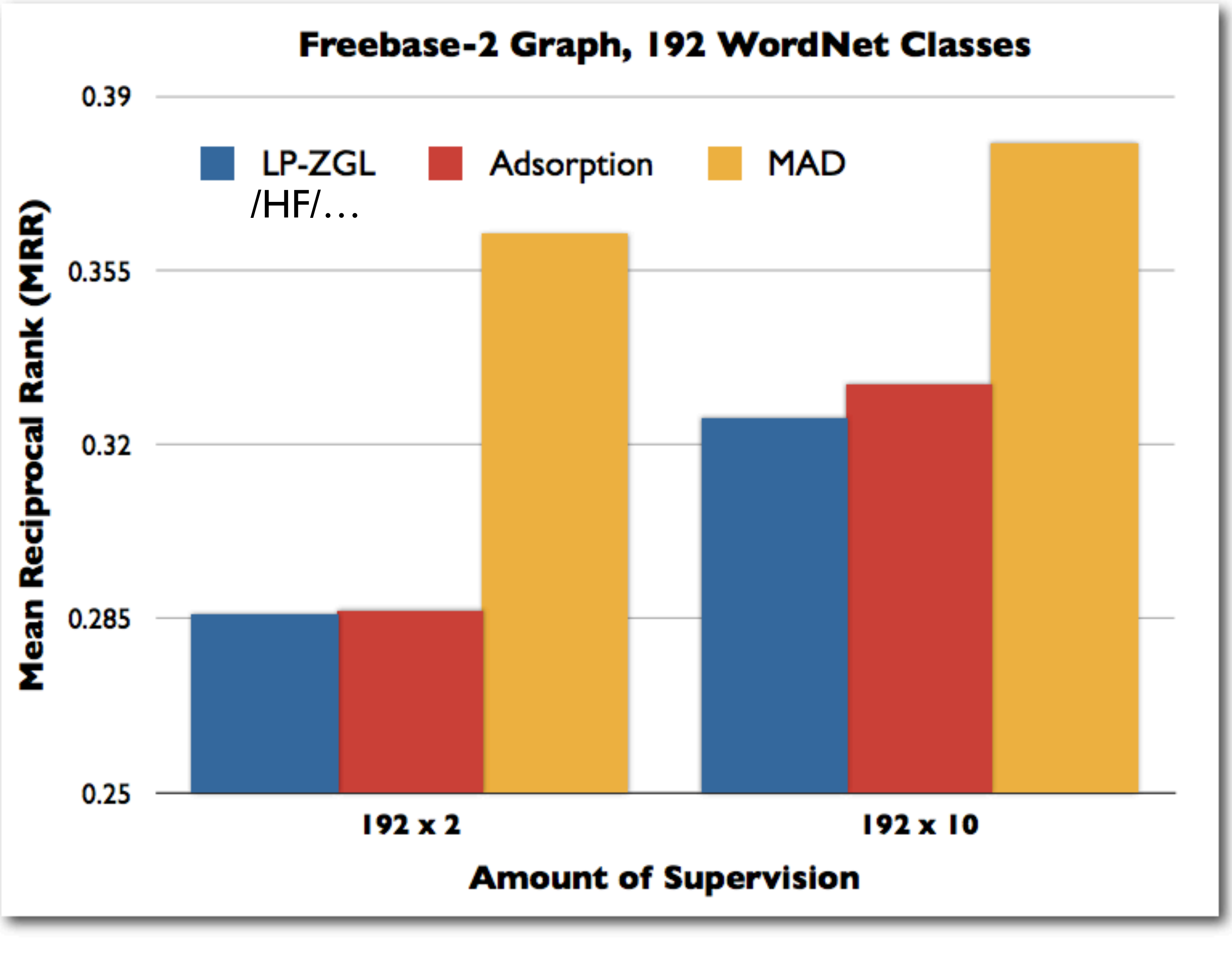
PRBEP (macro-averaged) on WebKB Dataset, 3148 test instances

Sentiment Classification



Precision on 3568 Sentiment test instances

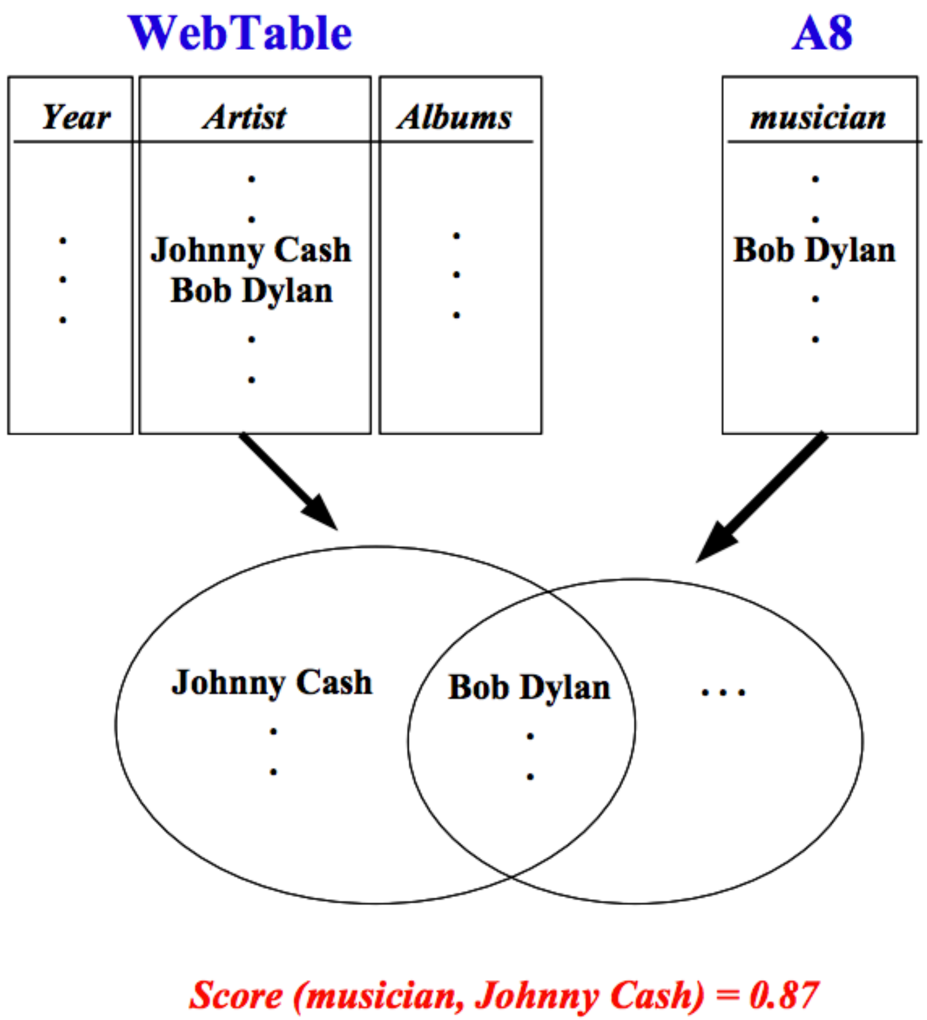
Class-Instance Acquisition



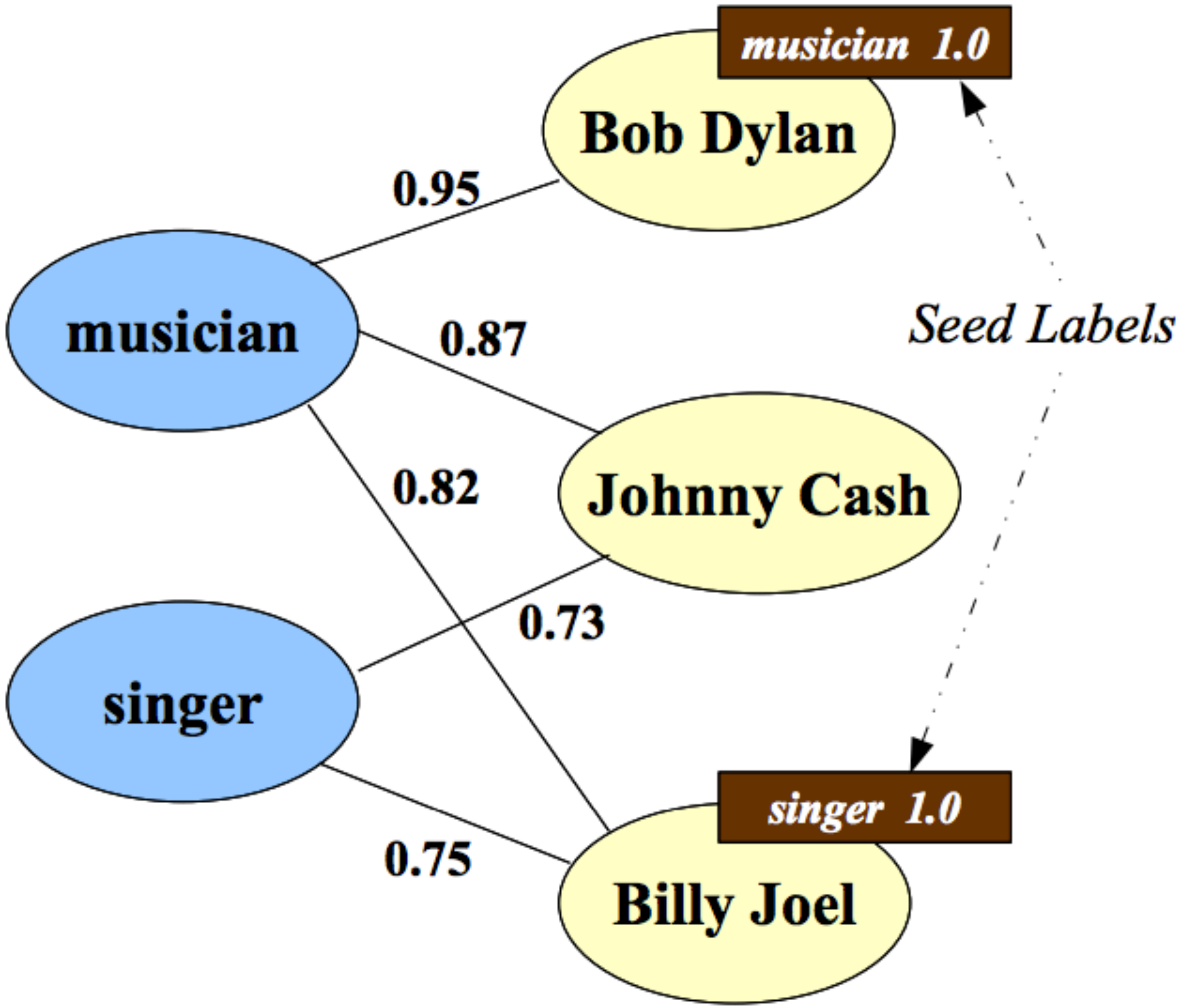
Graph with 303k nodes, 2.3m edges.

Assigning class labels to WebTable instances

from HTML tables on the web that are used for data, not formatting



from mining patterns like "musicians such as Bob Dylan"


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, …

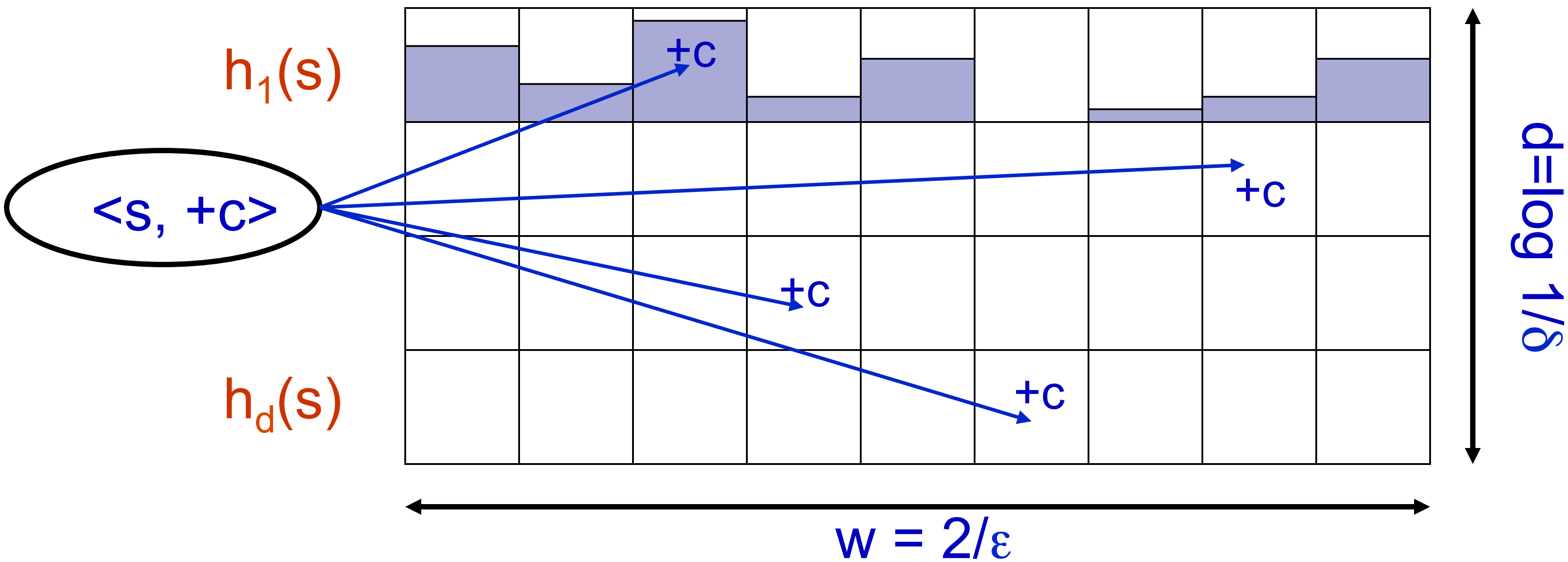
17

Total classes: **9081**

More recent work (AIStats 2014)

- Propagating labels requires usually small number of optimization passes
  - Basically like label propagation passes
- Each is linear in
  - the number of edges
  - and the number of labels being propagated
- Can you do better?
  - basic idea: store labels in a countmin sketch
  - which is basically an compact approximation of an object→double mapping

Flashback: CM Sketch Structure



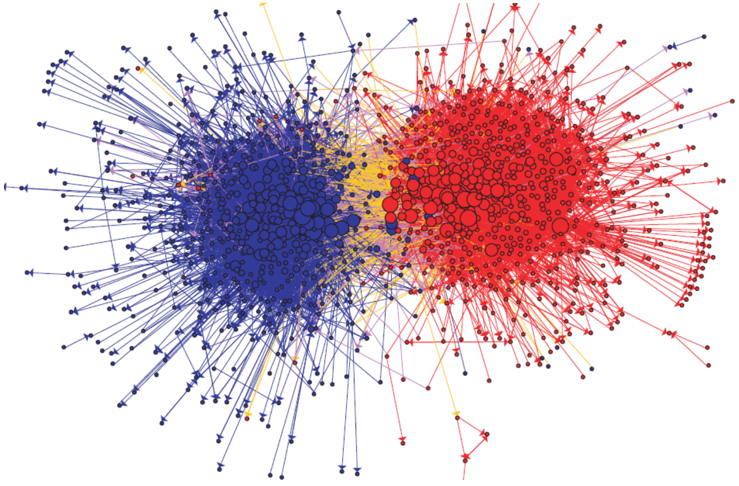
- Each string is mapped to one bucket per row
- Estimate A[j] by taking $\min_k \{ CM[k,h_k(j)] \}$
- Errors are always over-estimates
- Sizes: $d=\log 1/\delta$, $w=2/\varepsilon$ ➔ error is usually less than $\varepsilon||A||_1$

More recent work (AIStats 2014)

- Propagating labels requires usually small number of optimization passes
  - Basically like label propagation passes
- Each is linear in
  - the number of edges
  - ~~and the number of labels being propagated~~
  - the sketch size
  - sketches can be combined linearly without "unpacking" them: sketch($a\mathbf{v} + b\mathbf{w}$) = a*sketch($\mathbf{v}$) + b*sketch($\mathbf{w}$)
  - sketchs are good at storing *skewed distributions*

More recent work (AIStats 2014)

- Label distributions are often very skewed
  - sparse initial labels
  - community structure: labels from other subcommunities have small weight

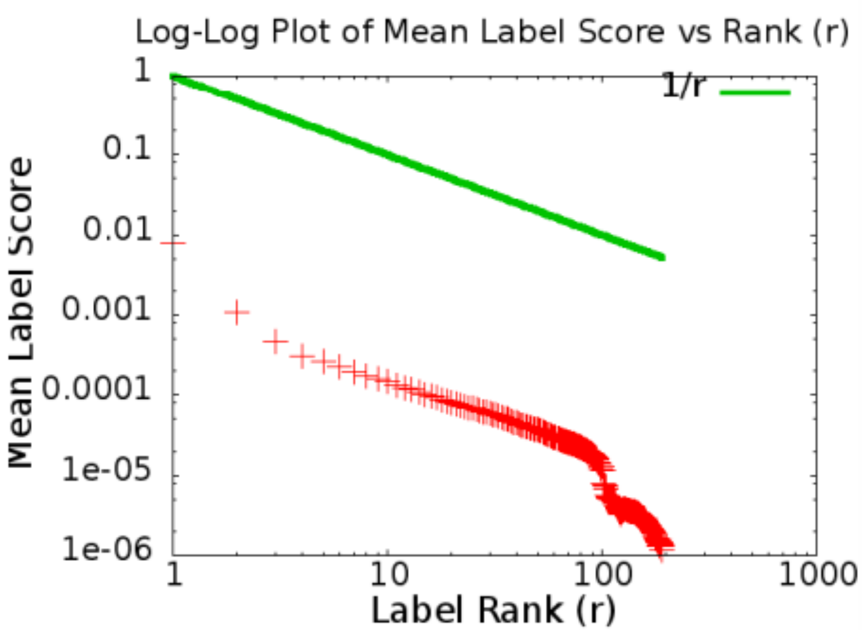


More recent work (AIStats 2014)

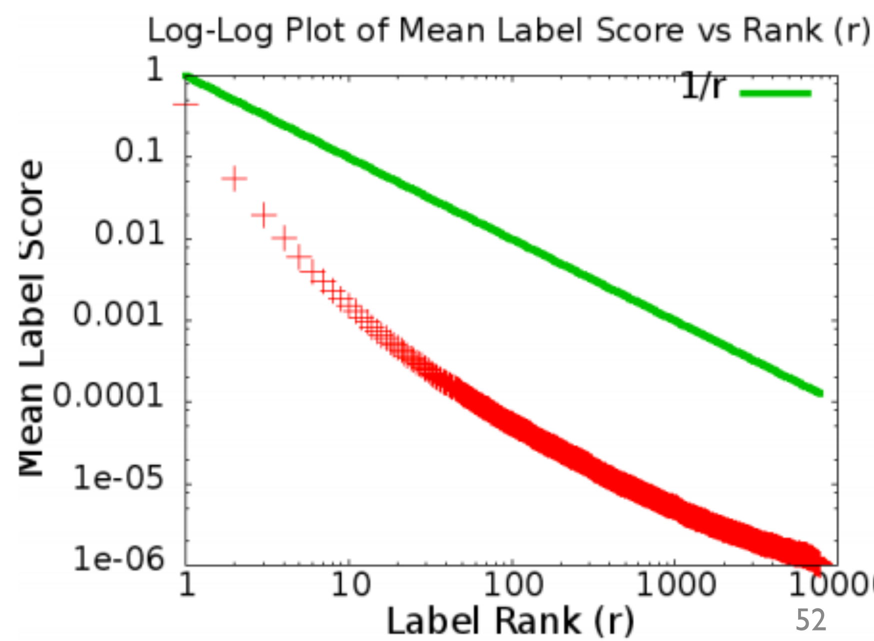
"self-injection": similarity computation

Name	Nodes (n)	Edges	Labels (m)	Seed Nodes	$k-$Sparsity	$\lceil \frac{ek}{\epsilon} \rceil$	$\lceil \ln \frac{m}{\delta} \rceil$
Freebase	301,638	1,155,001	192	1917	2	109	8
Flickr-10k	41,036	73,191	10,000	10,000	1	55	12
Flickr-1m	1,281,887	7,545,451	1,000,000	1,000,000	1	55	17

Freebase



Flick-10k



More recent work (AIStats 2014)

Name	Nodes (n)	Edges	Labels (m)	Seed Nodes	$k-$Sparsity	$\lceil \frac{ek}{\epsilon} \rceil$	$\lceil \ln \frac{m}{\delta} \rceil$
Freebase	301,638	1,155,001	192	1917	2	109	8
Flickr-10k	41,036	73,191	10,000	10,000	1	55	12
Flickr-1m	1,281,887	7,545,451	1,000,000	1,000,000	1	55	17

	Average Memory Usage (GB)	Total Runtime (s) [Speedup w.r.t. MAD-EXACT]	MRR
MAD-EXACT	3.54	516.63 [1.0]	0.28
MAD-SKETCH ($w = 109, d = 8$)	2.68	110.42 [4.7]	0.28
MAD-SKETCH ($w = 109, d = 3$)	1.37	54.45 [9.5]	0.29
MAD-SKETCH ($w = 20, d = 8$)	1.06	47.72 [10.8]	0.28
MAD-SKETCH ($w = 20, d = 3$)	1.12	48.03 [10.8]	0.23

Freebase

More recent work (AIStats 2014)

Name	Nodes (n)	Edges	Labels (m)	Seed Nodes	$k-$Sparsity	$\lceil \frac{ek}{\epsilon} \rceil$	$\lceil \ln \frac{m}{\delta} \rceil$
Freebase	301,638	1,155,001	192	1917	2	109	8
Flickr-10k	41,036	73,191	10,000	10,000	1	55	12
Flickr-1m	1,281,887	7,545,451	1,000,000	1,000,000	1	55	17