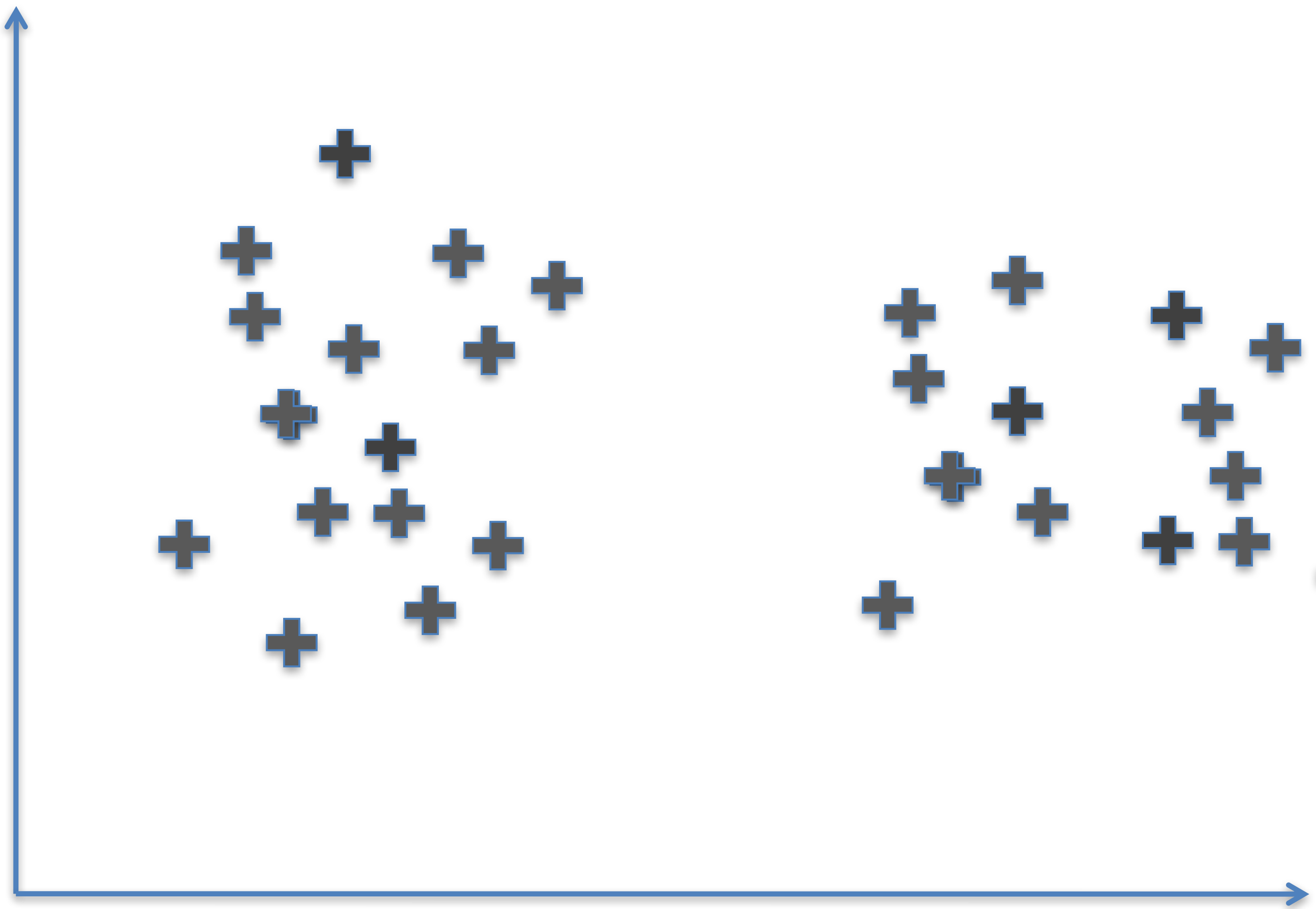
INTRO TO SEMI-SUPERVISED LEARNING (SSL)

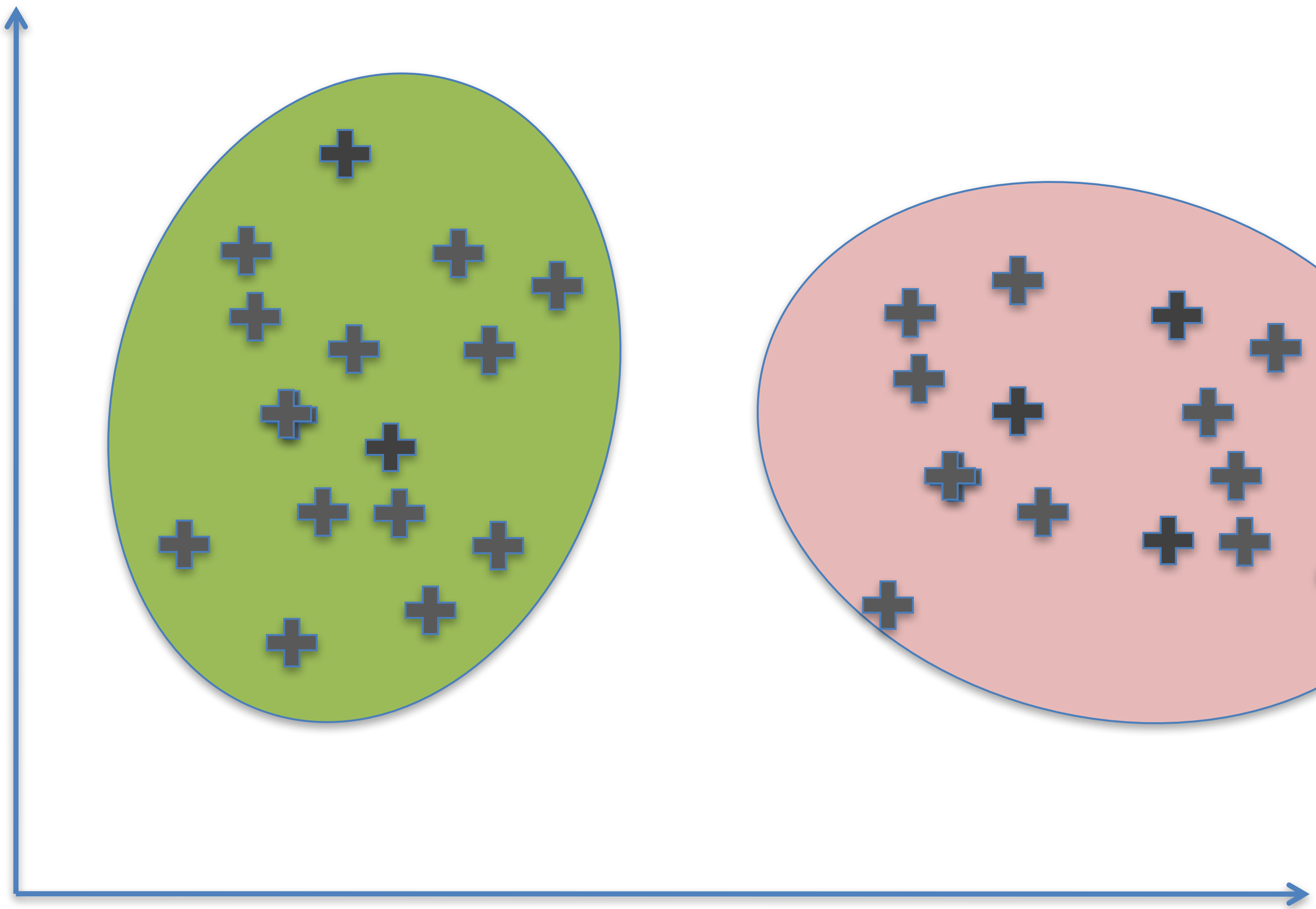
Semi-supervised learning

- Given:
  - A pool of labeled examples L
  - A (usually larger) pool of unlabeled examples U
- Option 1 for using L and U :
  - Ignore U and use supervised learning on L
- Option 2:
  - Ignore labels in L+U and use k-means, etc find clusters; then label each cluster using L
- Question:
  - Can you use both L and U to do better?

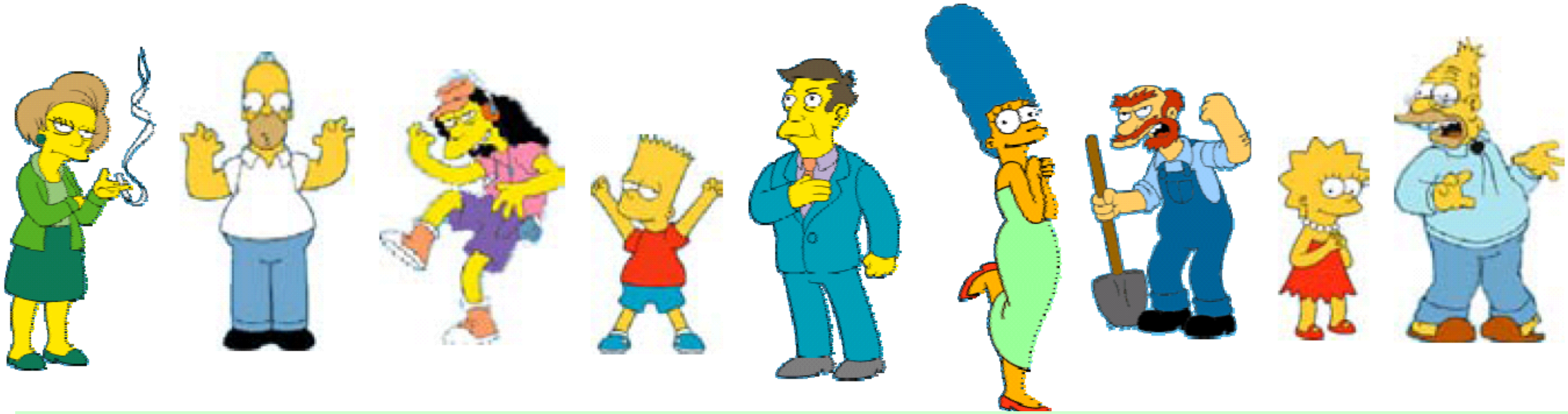
SSL is Somewhere Between Clustering and Supervised Learning

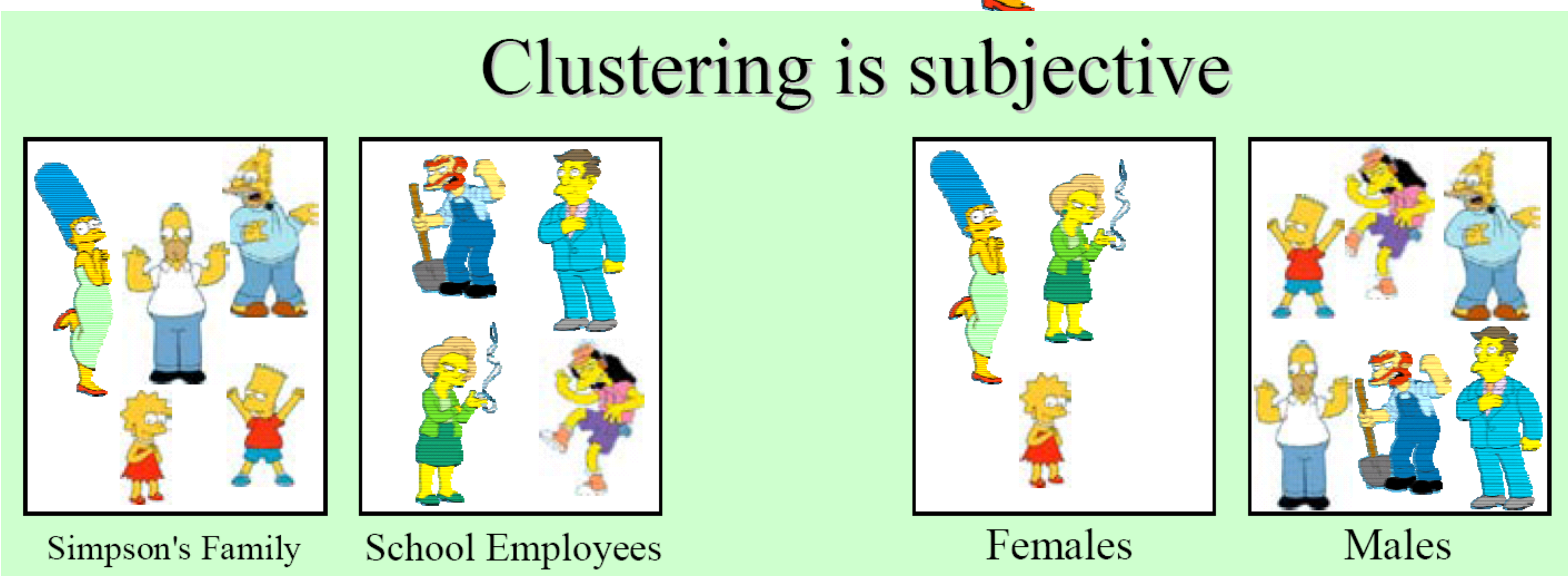


SSL is Between Clustering and SL



What is a natural grouping among these objects?

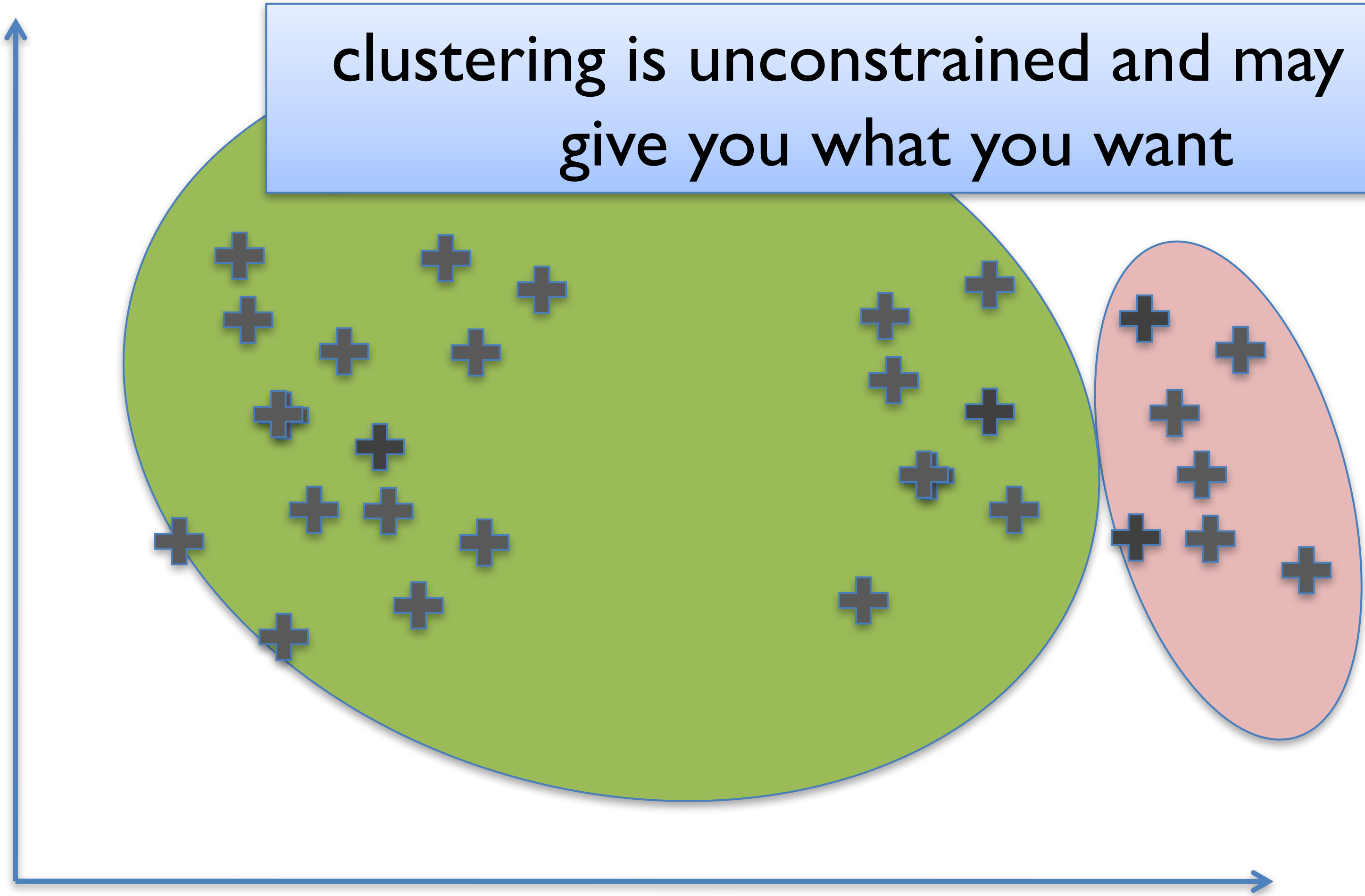




slides: Bhavana Dalvi

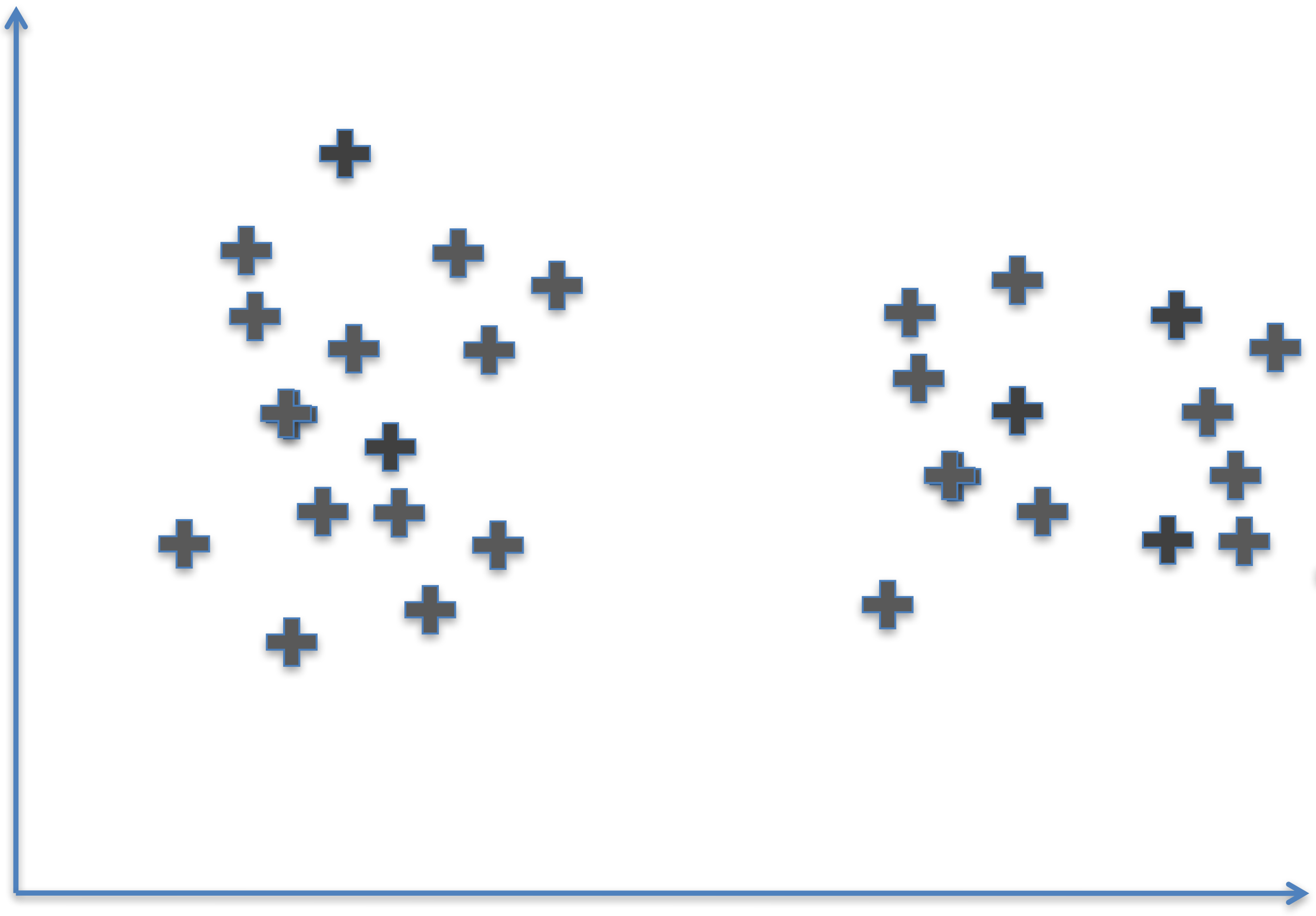
SSL is Between Clustering and SL

clustering is unconstrained and may not give you what you want

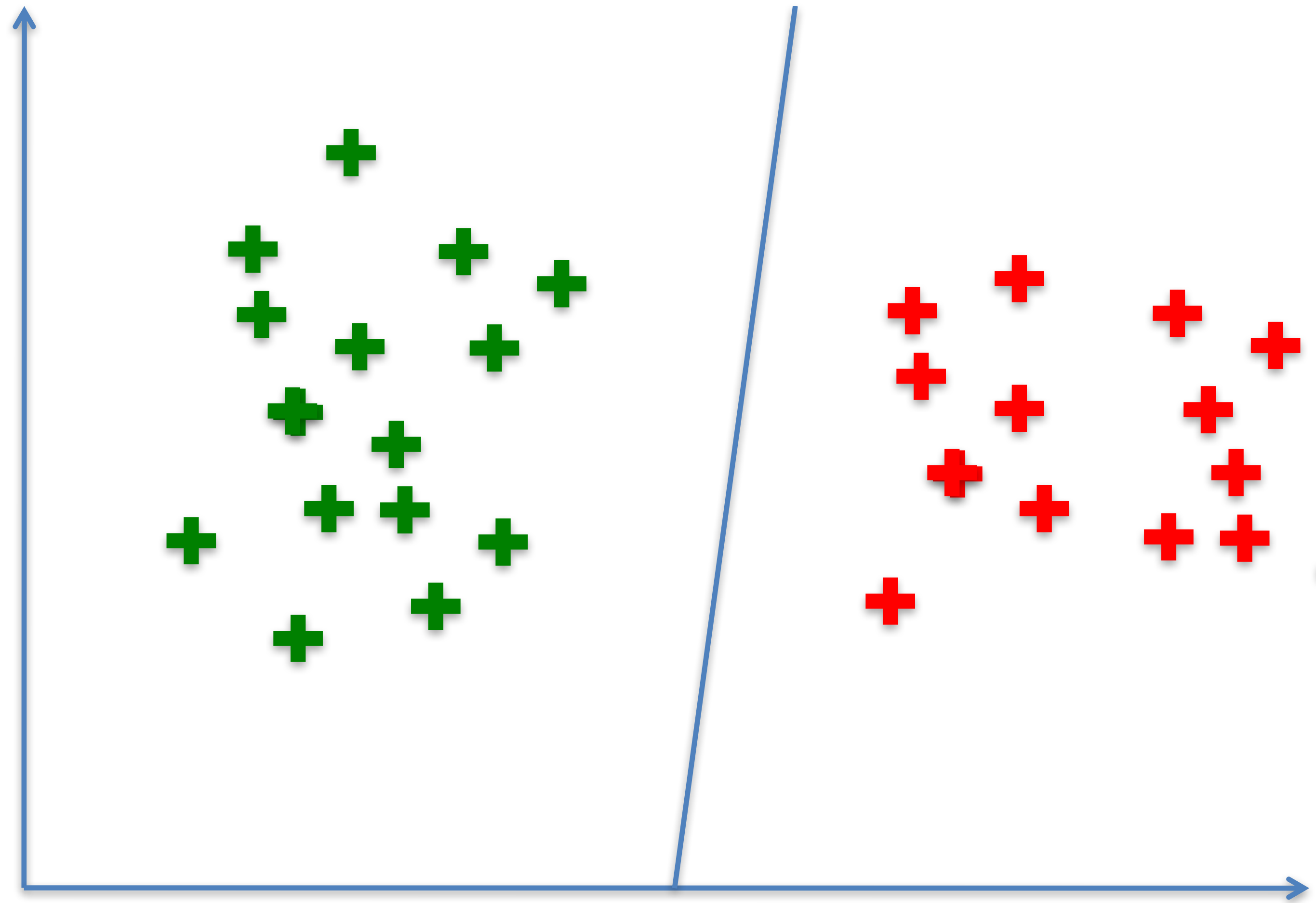


maybe this clustering is as good as the other

SSL is Between Clustering and SL

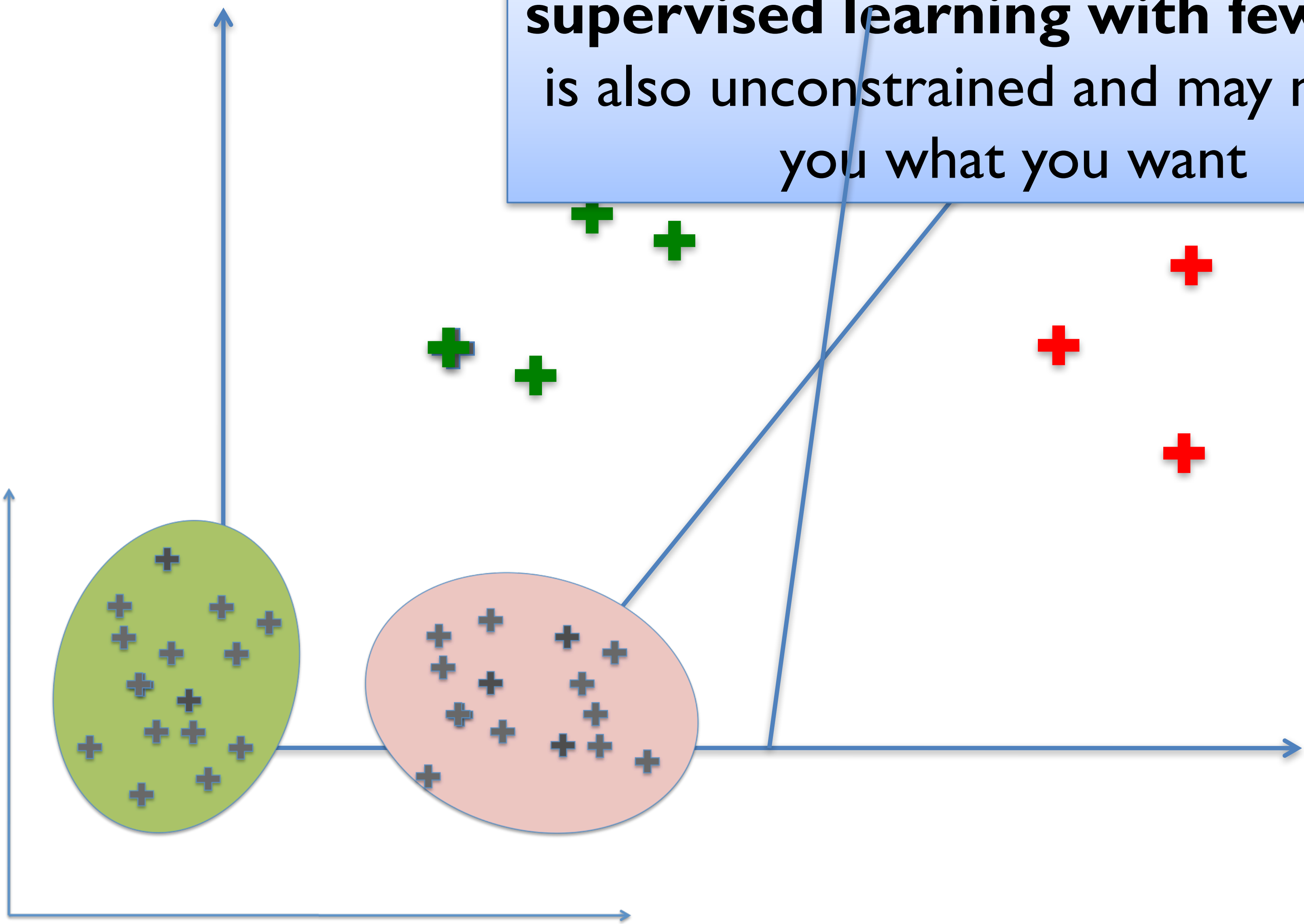


SSL is Between Clustering and SL

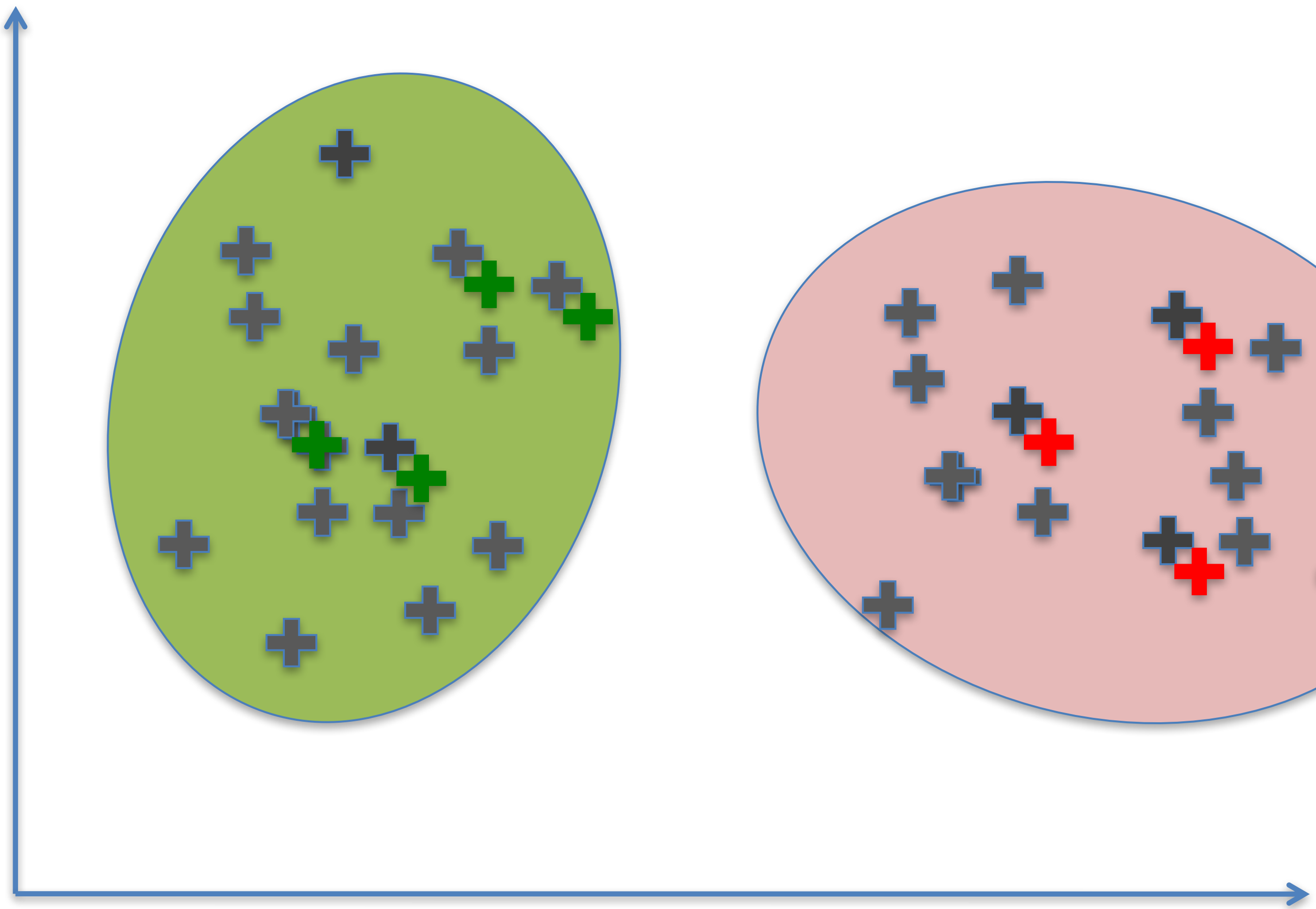


SSL is Between Clustering and SL

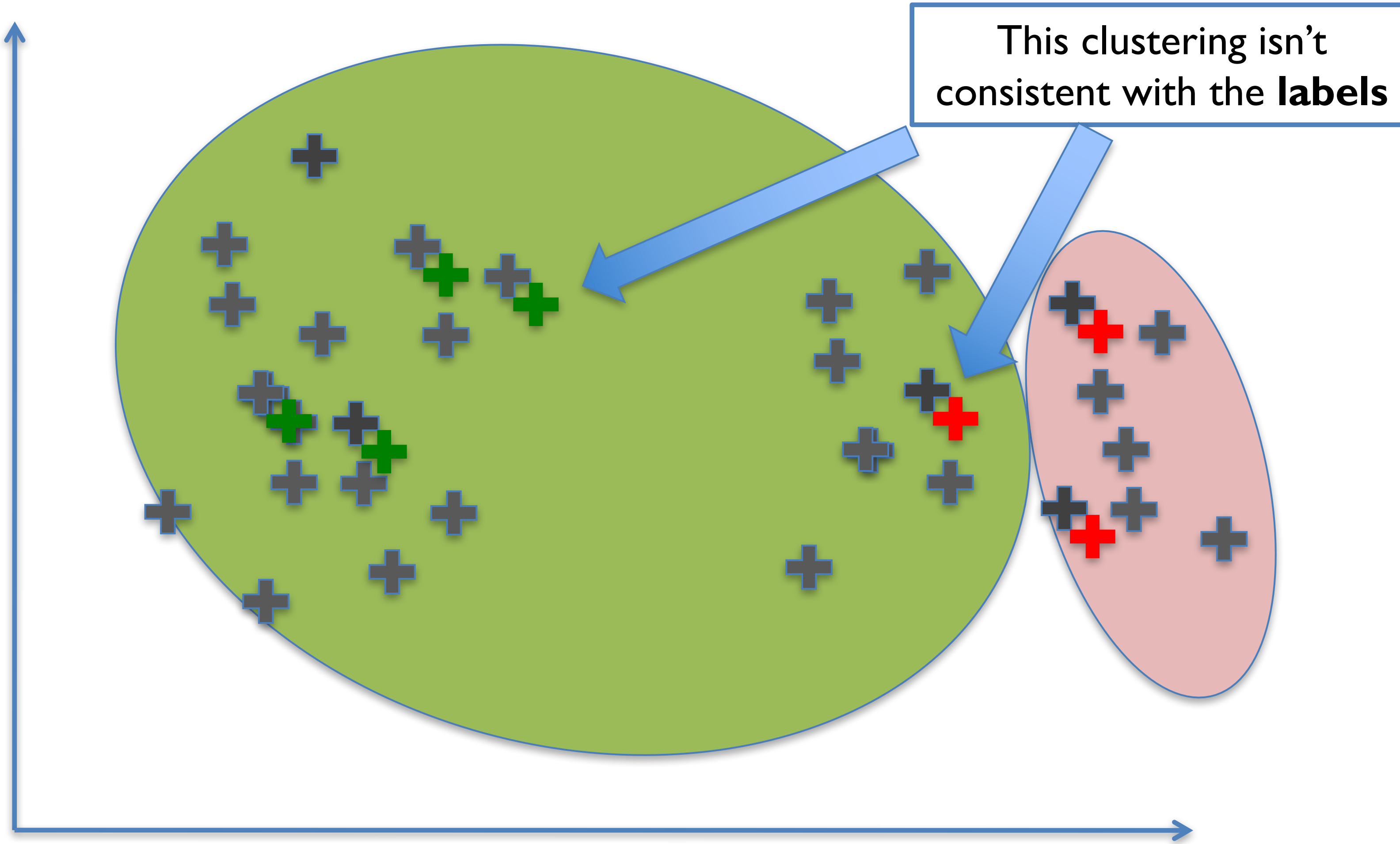
supervised learning with few labels is also unconstrained and may not give you what you want



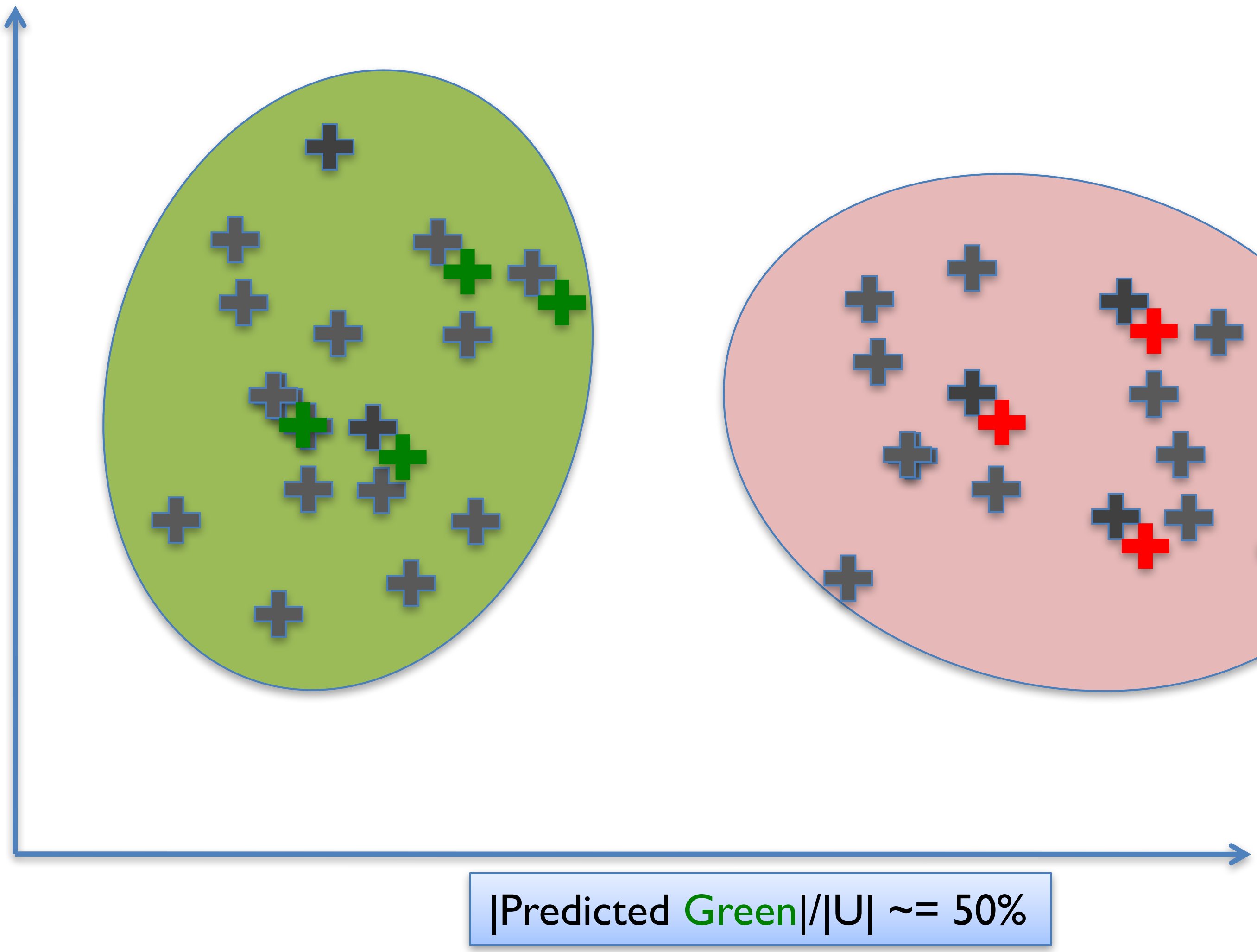
SSL is Between Clustering and SL



SSL is Between Clustering and SL



SSL is Between Clustering and SL

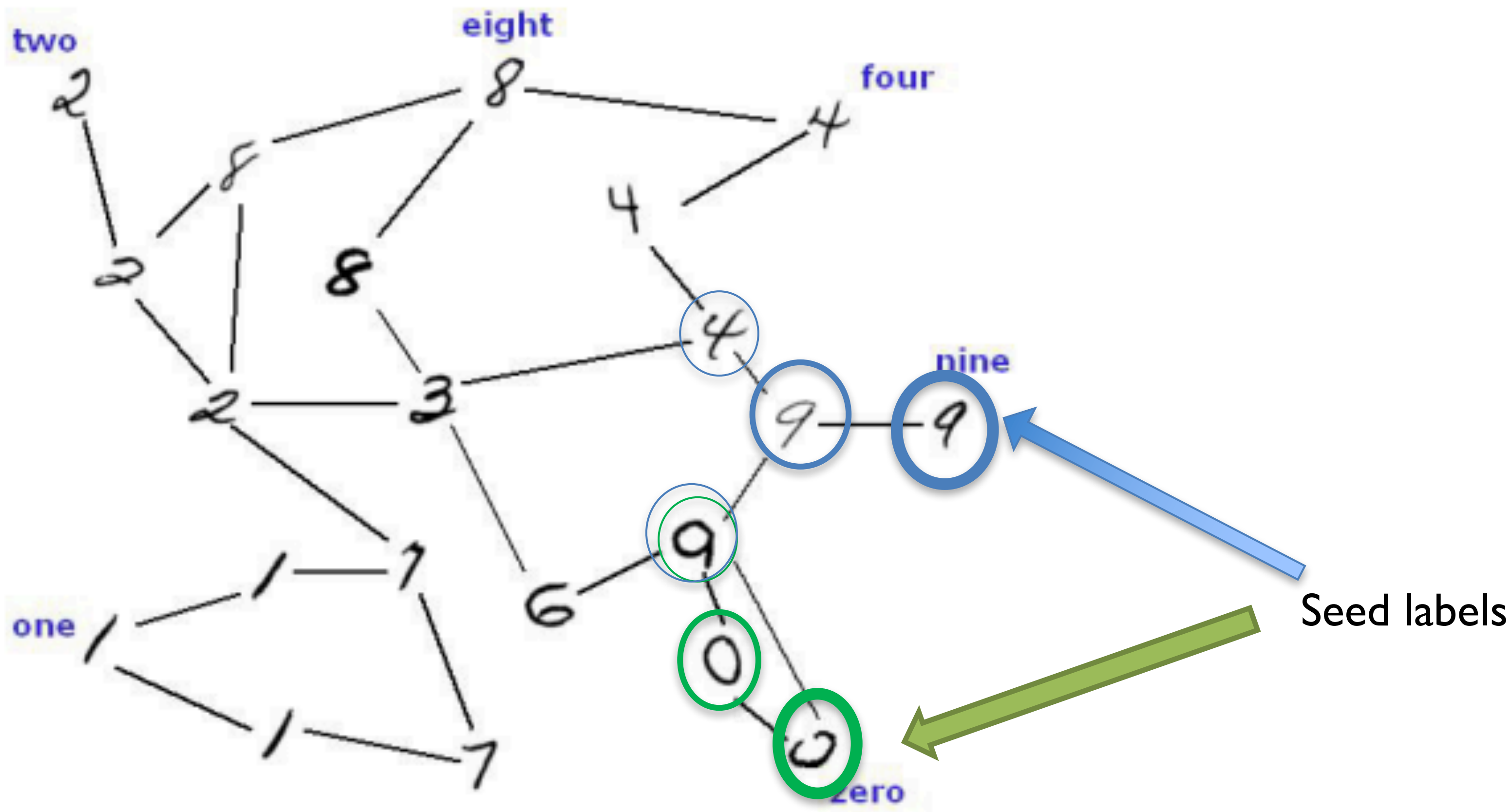


SSL in Action: The NELL System

Type of SSL

- Margin-based: transductive SVM
  - Logistic regression with entropic regularization
- Generative: seeded k-means
- Nearest-neighbor like: graph-based SSL
  - Label propagation

SSL via "Label Propagation"



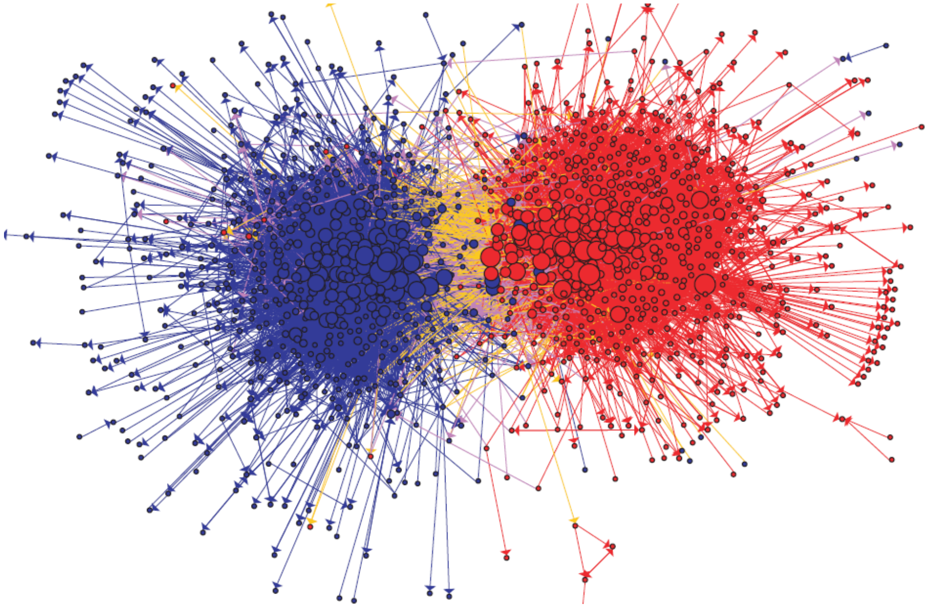
Semi-Supervised Classification of Network Data Using Very Few Labels

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

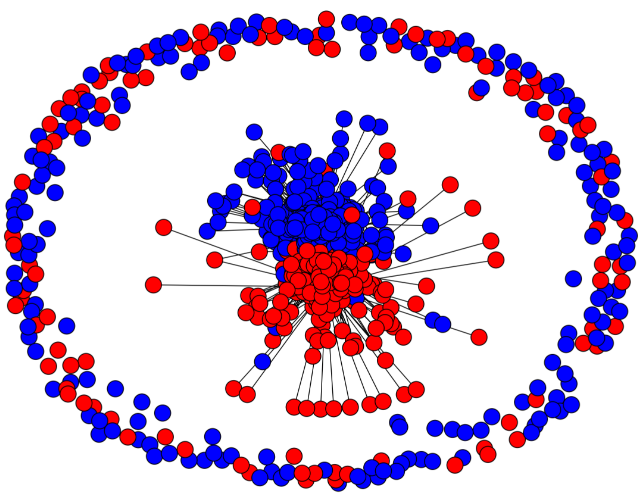
William W. Cohen
Carnegie Mellon University, Pittsburgh, Pennsylvania
Email: wcohen@cs.cmu.edu

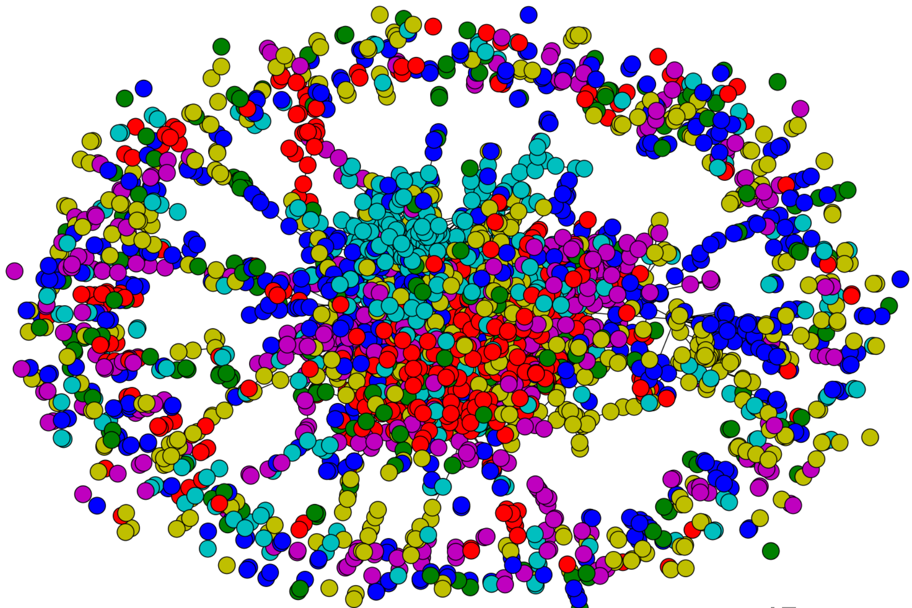
ASONAM-2010 (Advances in Social Networks Analysis and Mining)

Network Datasets with Known Classes

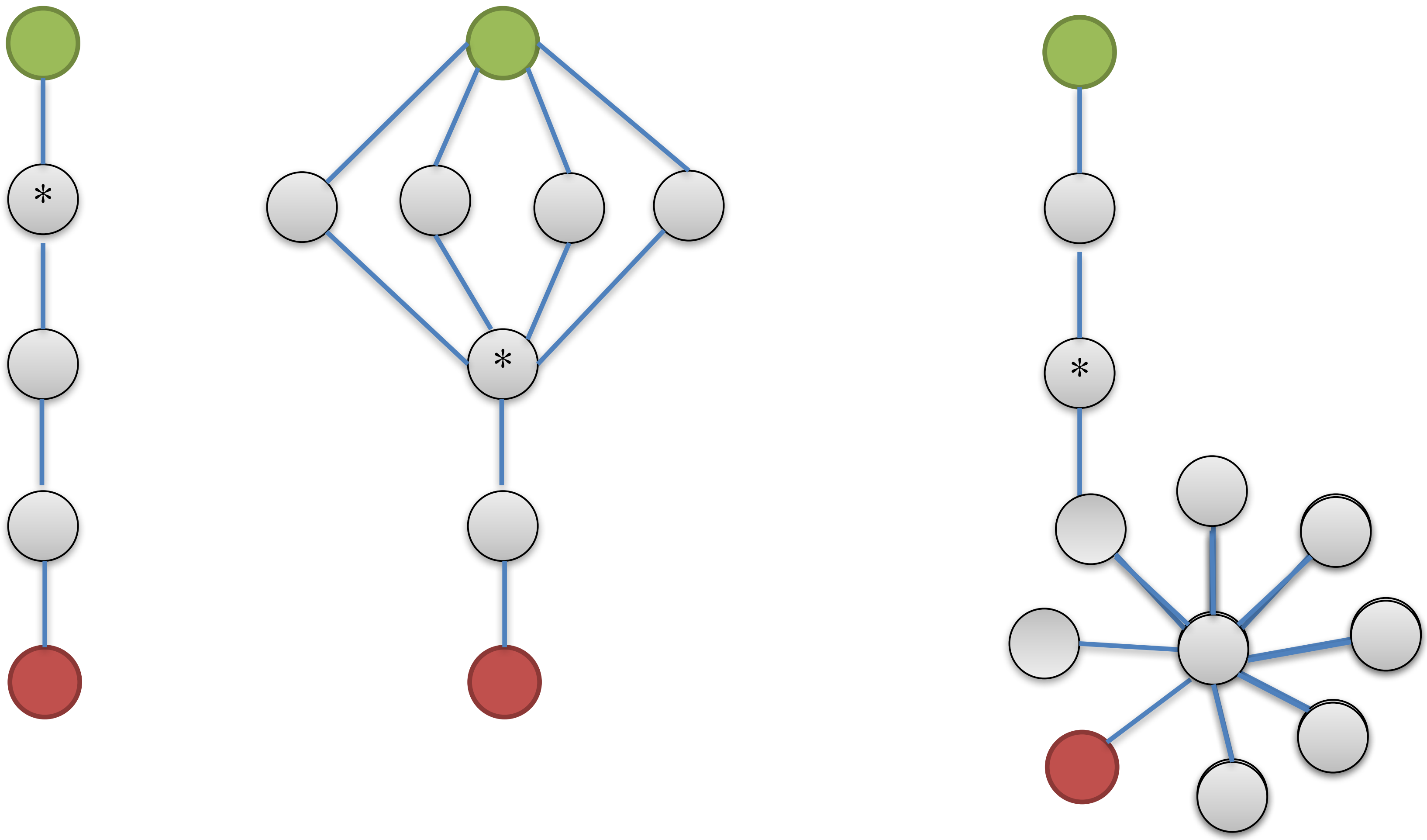


- UBMCBlog
- AGBlog
- MSPBlog
- Cora
- Citeseer





Some intuition



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

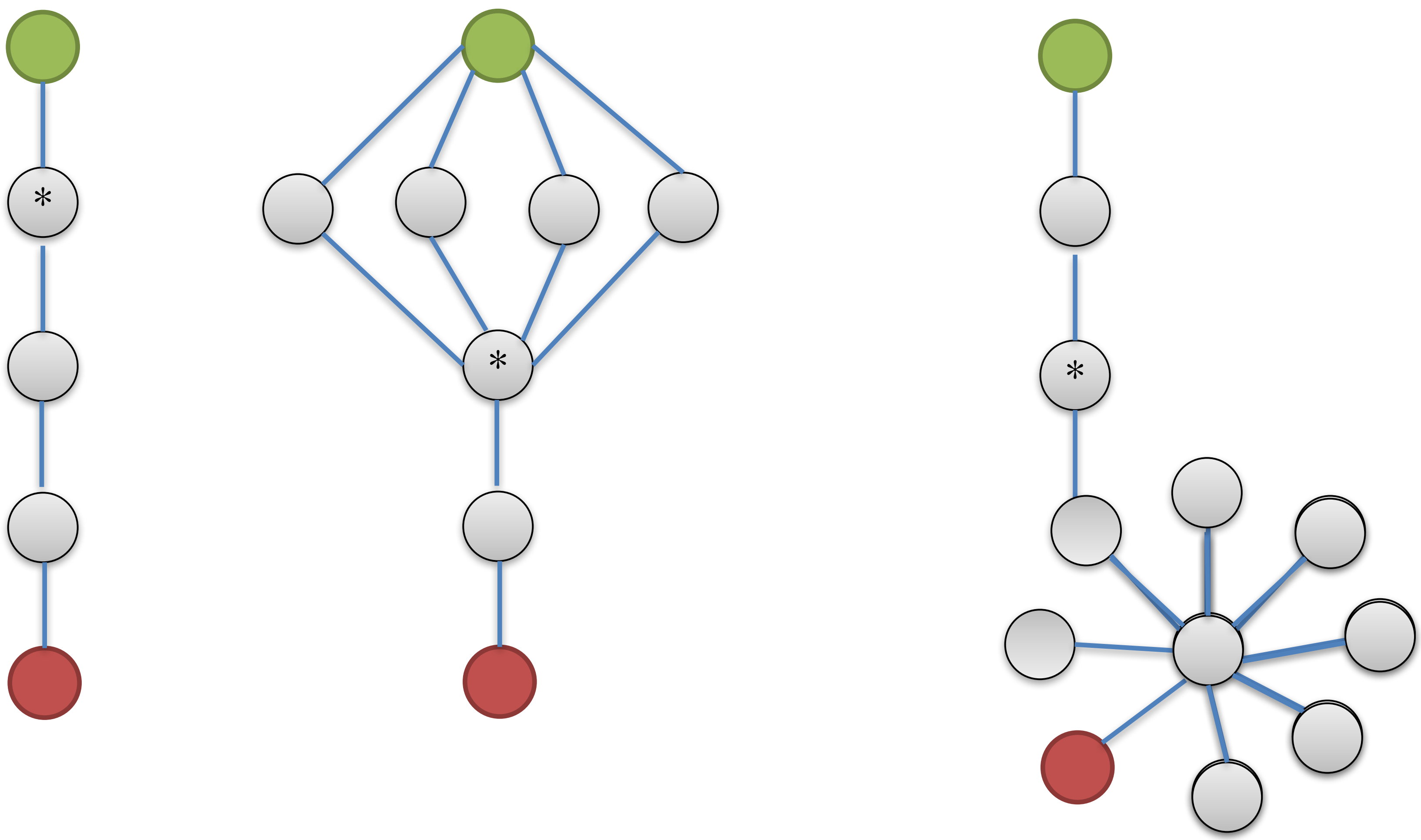
Fig. 1. The MultiRankWalk algorithm.

u is uniform over the seeds for class *c*

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

Some intuition



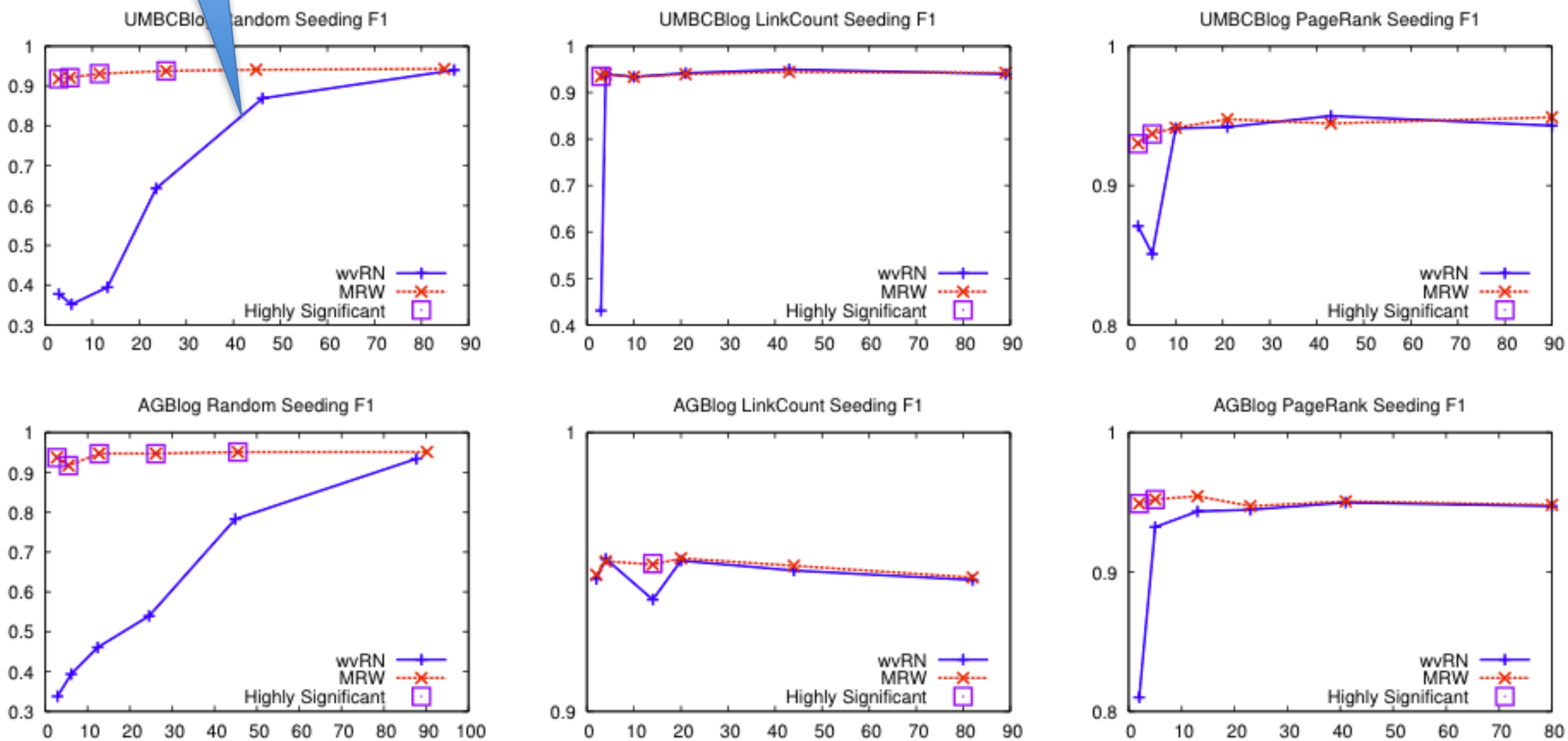
We'll discuss this soon….

Results – Blog data

Random

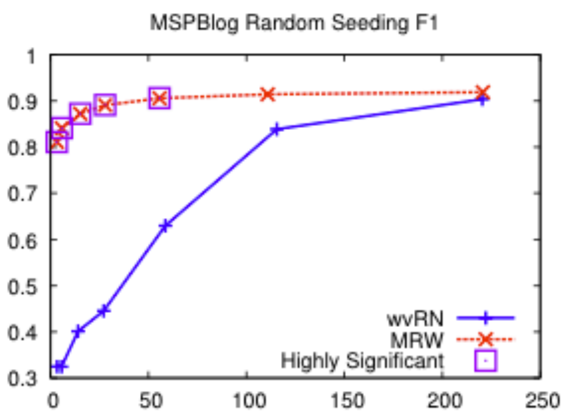
Degree

PageRank

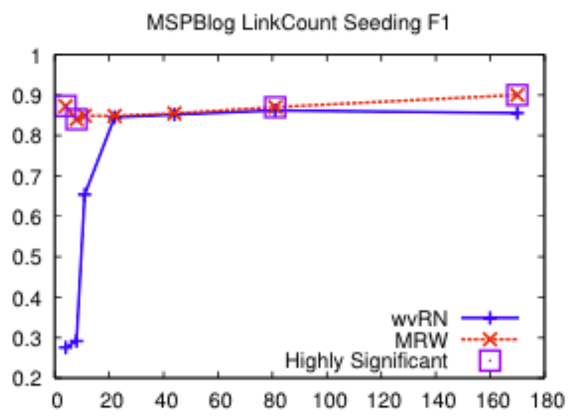


Results – More blog data

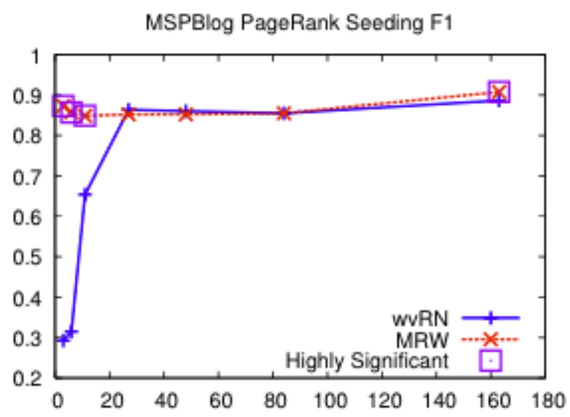
Random



Degree

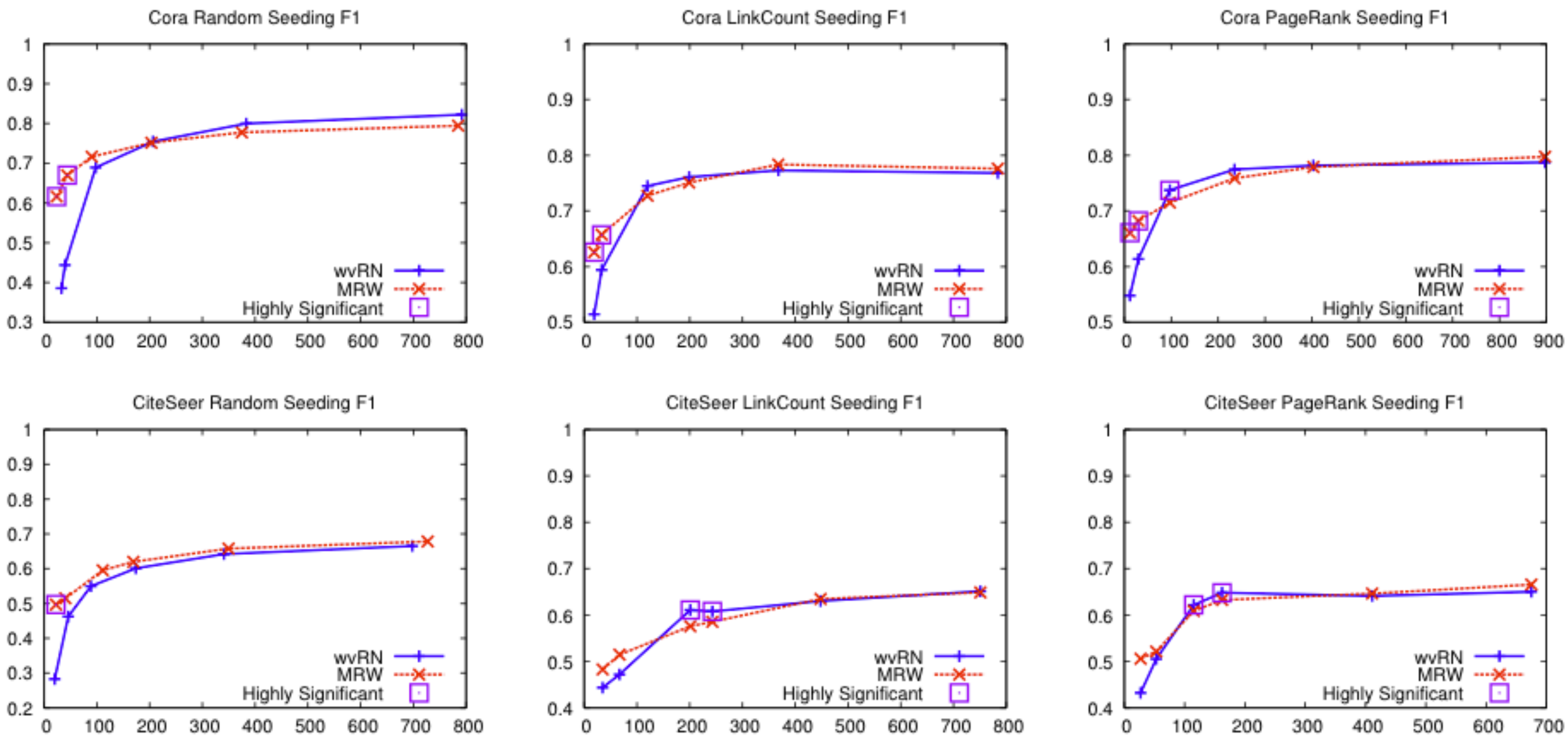


PageRank

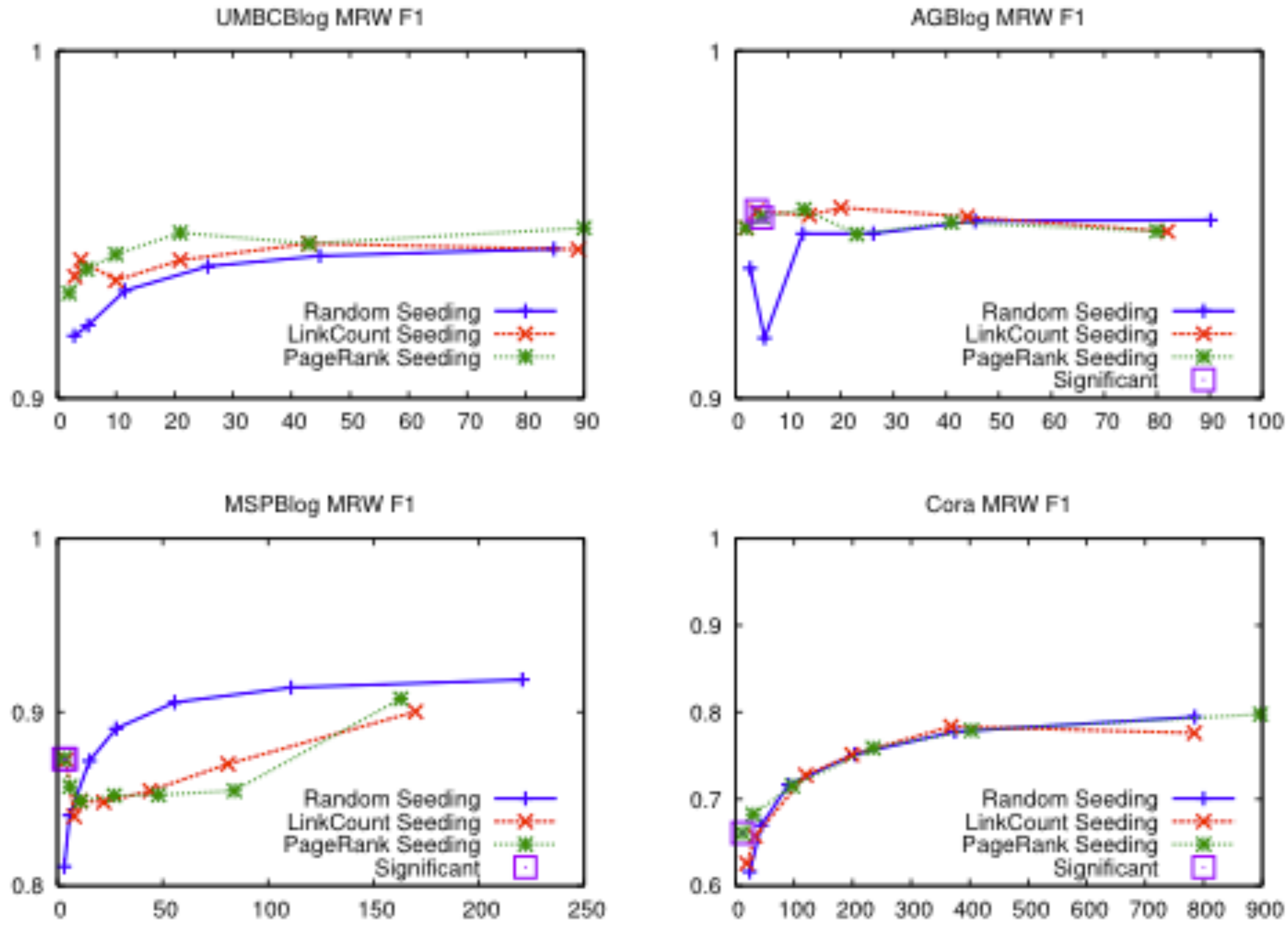


Results – Citation data

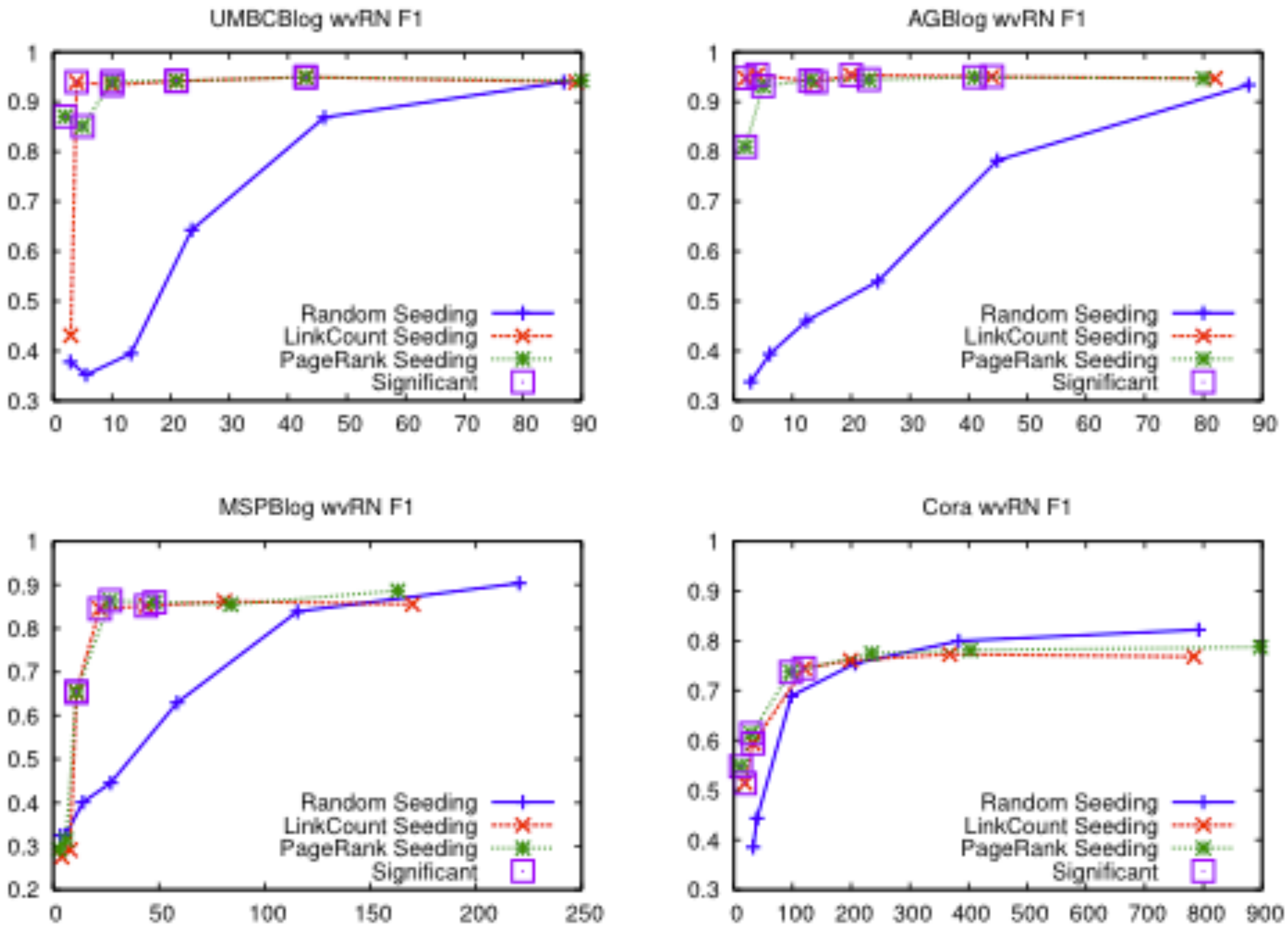
Random　　　　Degree　　　　PageRank



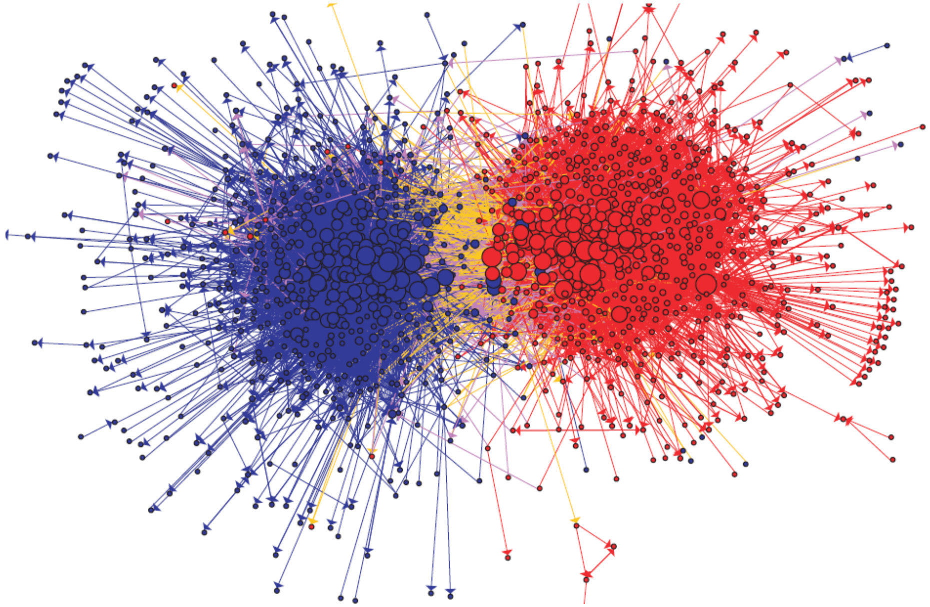
Seeding – MultiRankWalk



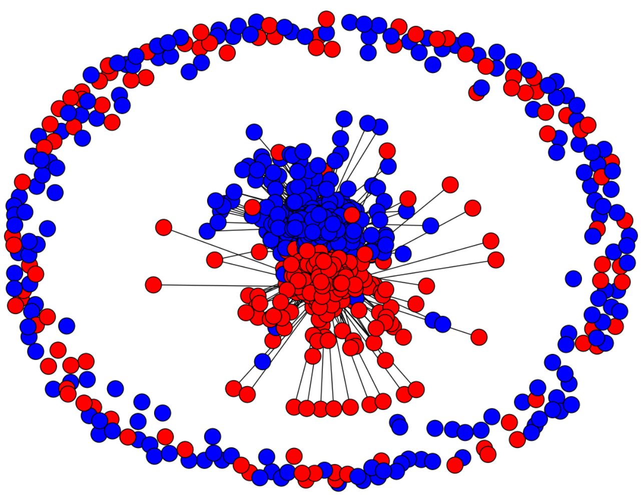
Seeding – HF/wvRN

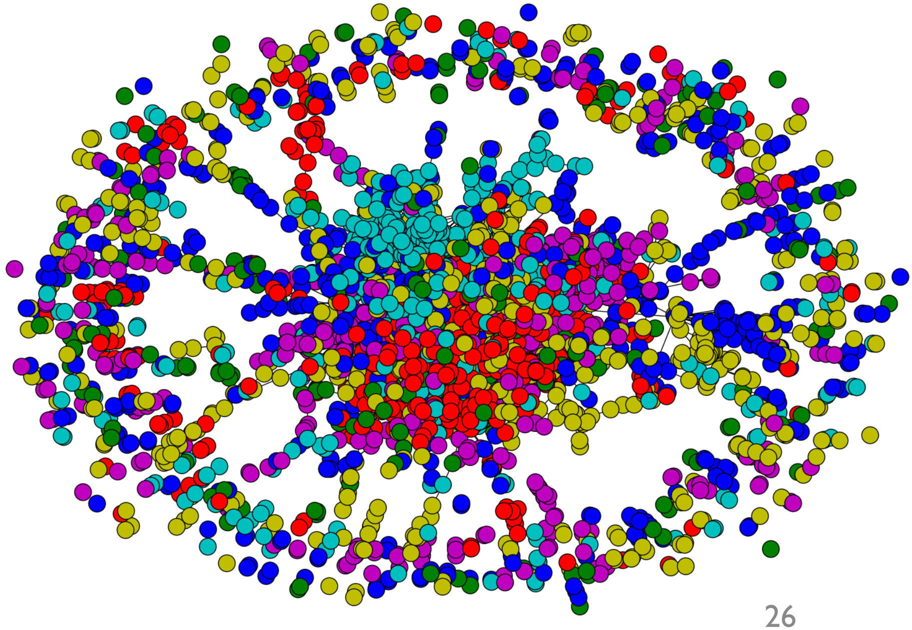


Back to Experiments: Network Datasets with Known Classes



- UBMCBlog
- AGBlog
- MSPBlog
- Cora
- Citeseer





MultiRankWalk vs wvRN/HF/CoEM

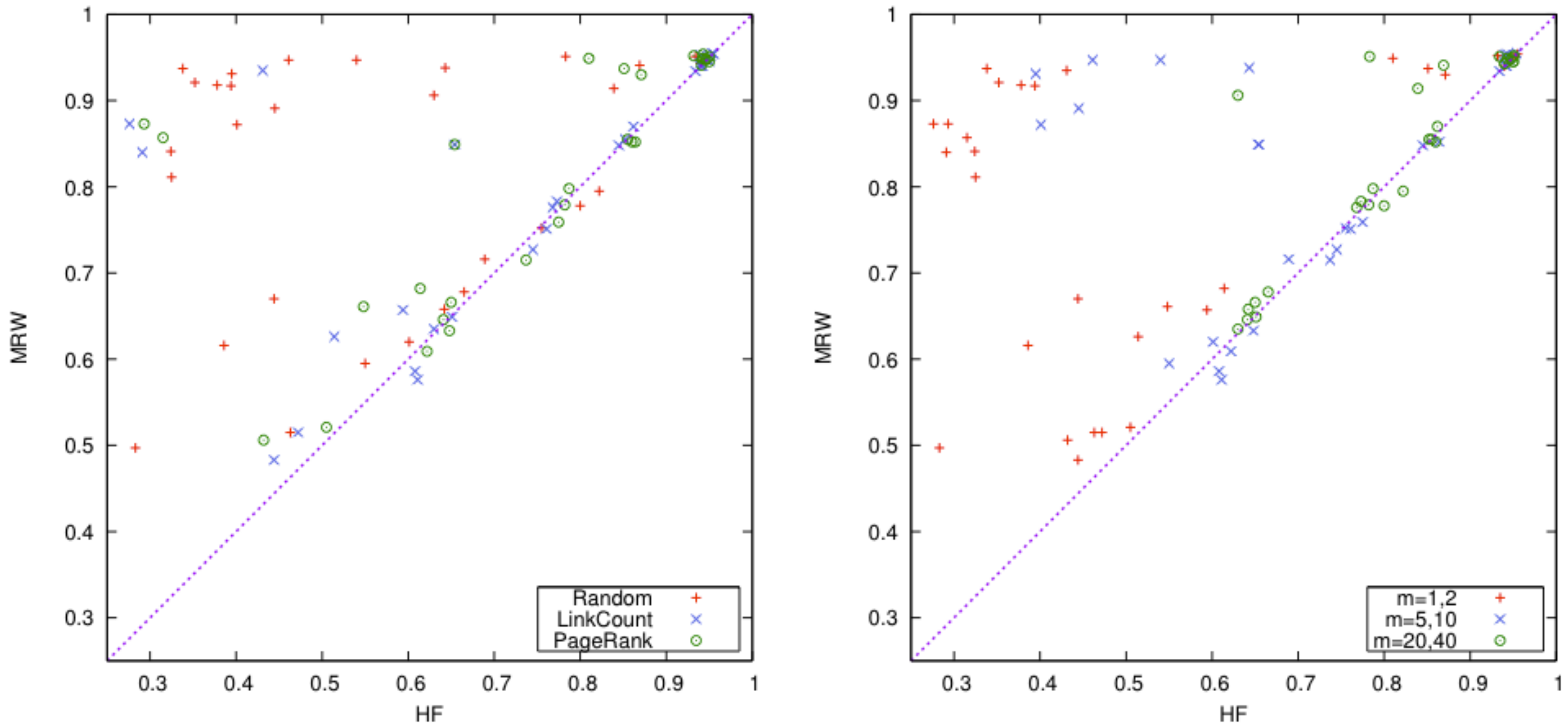


Figure 2.6: Scatter plots of HF F1 score versus MRW F1 score. The left plot marks different seeding preferences and the right plot marks varying amount of training labels determined by m.

How well does MWR work?

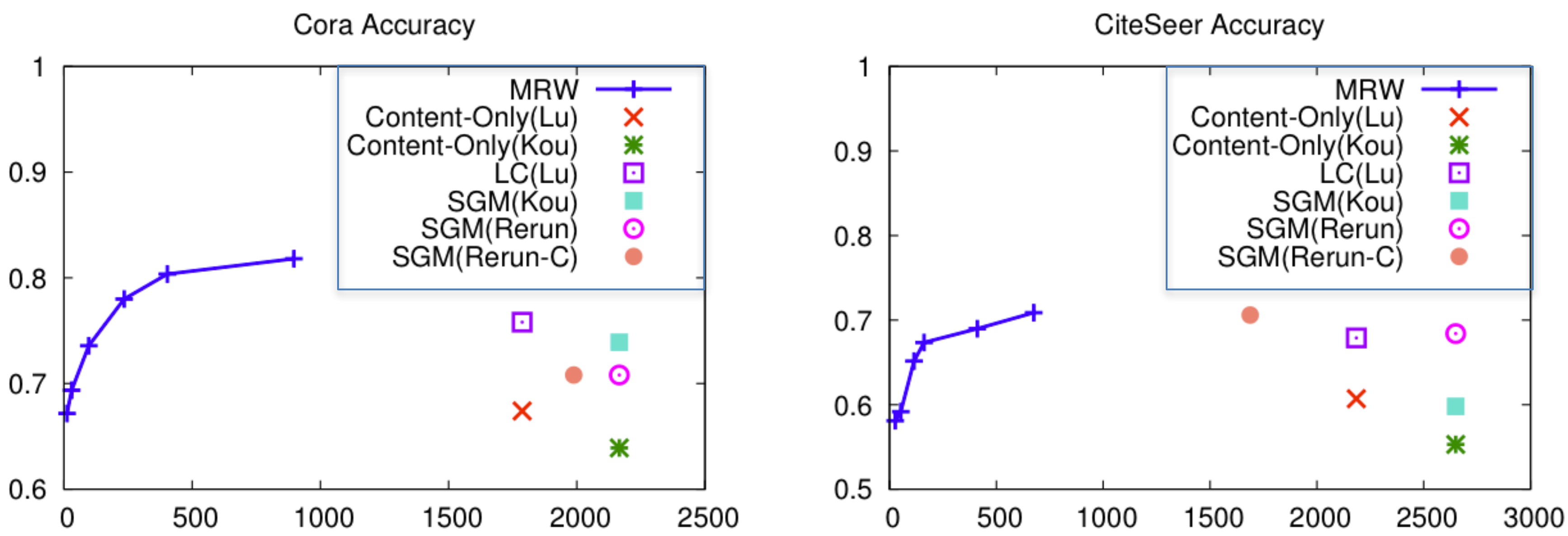


Fig. 5. Citation datasets results compared to supervised relational learning methods. The x-axis indicates number of labeled instances and y-axis indicates labeling accuracy.

Parameter Sensitivity

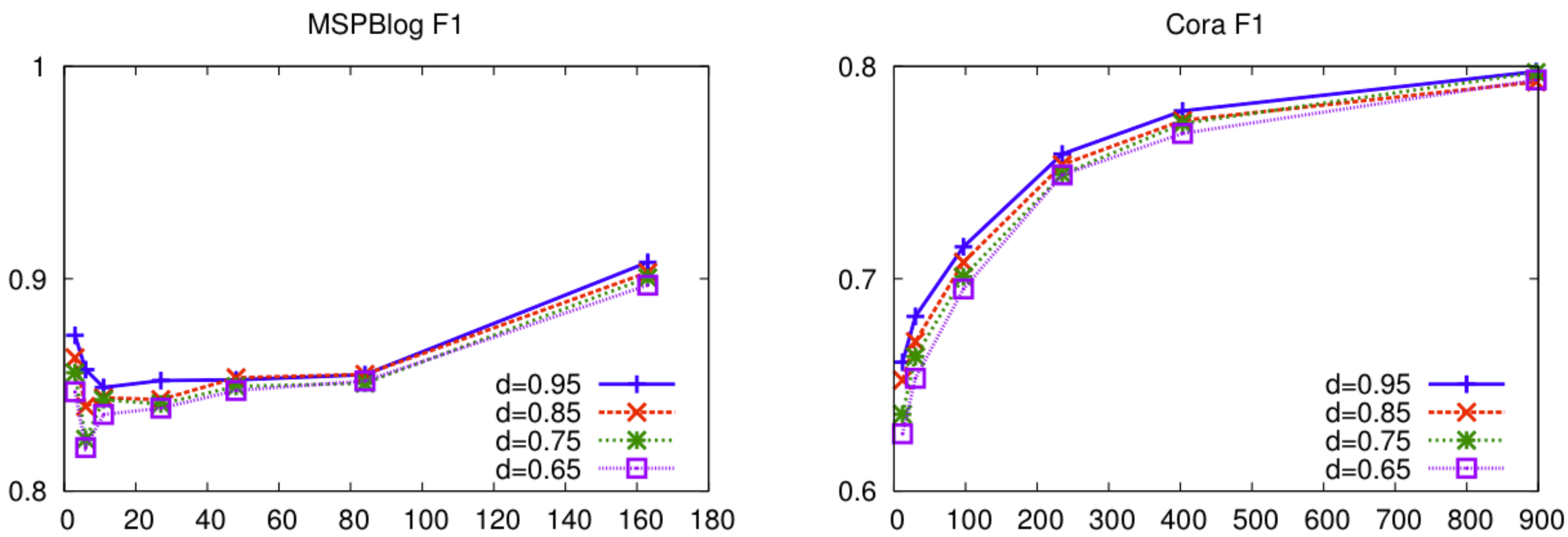


Fig. 7. Results on three datasets varying the damping factor. The x-axis indicates number of labeled instances and y-axis indicates labeling macro-averaged F1 score.

Harmonic Fields
aka coEM aka wvRN

CoEM/HF/wvRN

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

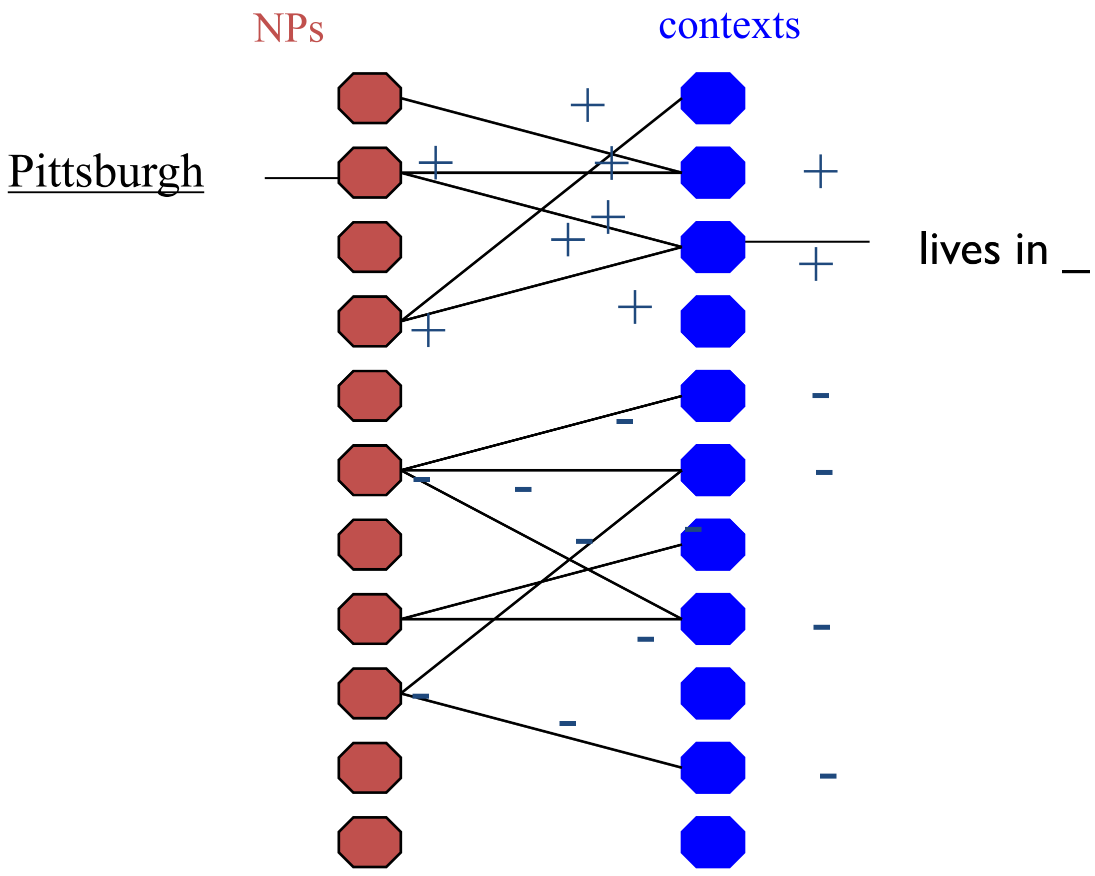
Definition. Given $v_i \in \mathbf{V}^U$, the weighted-vote relational-neighbor classifier (wvRN) estimates $P(x_i|\mathcal{N}_i)$ as the (weighted) mean of the class-membership probabilities of the entities in $\mathcal{N}_i$:

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

Another definition: A *harmonic field (HF)* – the score of each node in the graph is the harmonic (linearly weighted) average of its neighbors' scores --- also sometimes called LP-ZGL

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

Co-EM Learner: equivalent to HF on a bipartite graph (Ghani & Nigam, 2000)



The HF Algorithm

$\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^m, y^m)\}$ = labeled examples

$\{\mathbf{x}^{m+1}, \ldots, \mathbf{x}^{m+n}\}$ = unlabeled examples

$W[i, j]$ = graph = similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

Optimization problem: minimize

$$Loss = \sum_{i>m, j>m} W[i, j](\hat{y}_i - \hat{y}_j)^2$$

subject to constraint that all labeled examples are classified correctly

The HF Loss In Matrix Form

$W[i, j]$ = graph = similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is symmetric

$$
\begin{aligned}
Loss &= \sum_{i,j} w_{i,j}(\hat{y}_i - \hat{y}_j)^2 \\
&= \sum_{i,j} w_{i,j}\hat{y}_i^2 + \sum_{i,j} w_{i,j}\hat{y}_j^2 - 2\sum_{i,j} w_{i,j}\hat{y}_i\hat{y}_j \\
&= \sum_{i}(\sum_{j} w_{i,j})\hat{y}_i^2 + \sum_{j}(\sum_{i} w_{i,j})\hat{y}_j^2 - 2\sum_{i,j} w_{i,j}\hat{y}_i\hat{y}_j \\
&= \sum_{i} d_i\hat{y}_i^2 + \sum_{j} d_j\hat{y}_j^2 - 2\sum_{i,j} w_{i,j}\hat{y}_i\hat{y}_j \\
&= 2\sum_{i} d_i\hat{y}_i^2 - 2\sum_{i,j} w_{i,j}\hat{y}_i\hat{y}_j \\
&= 2(\sum_{i} d_i\hat{y}_i^2 - \sum_{i,j} w_{i,j}\hat{y}_i\hat{y}_j) \\
&= 2(\hat{\mathbf{y}}^T D\hat{\mathbf{y}} - \hat{\mathbf{y}}^T W\hat{\mathbf{y}}) \\
&= 2\hat{\mathbf{y}}^T(D - W)\hat{\mathbf{y}}
\end{aligned}
$$

The HF Algorithm

$\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^m, y^m)\}$ = labeled examples

$\{\mathbf{x}^{m+1}, \ldots, \mathbf{x}^{m+n}\}$ = unlabeled examples

$W[i, j]$ = graph = similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$S[i,i] = 1$ for all seed nodes $i<m+1$

Optimization problem: minimize $\hat{\mathbf{y}}^T(D - W)\hat{\mathbf{y}}$

subject to $S\hat{\mathbf{y}} = S\mathbf{y}$

The HF Algorithm

Optimization problem: minimize $\hat{\mathbf{y}}^T(D-W)\hat{\mathbf{y}}$

subject to $S\hat{\mathbf{y}} = S\mathbf{y}$

1. Let $\hat{\mathbf{y}}^0$ be any label assignment consistent with the seed labels.
2. For $t = 0, \ldots, T$:
   (a) For every unlabeled node $i > m$, let $\hat{y}_i^{t+1} = \frac{1}{d_i}\sum_j w_{i,j}\hat{y}_j^t$
   (b) For every labeled node $i \leq m$, let $\hat{y}_i^{t+1} = y_i$ (where y_i is the seed label for example i).

This converges quickly: on Frank's data usually 5-10 iterations was best (and more tends to overfit)

What is HF aka coEM aka wvRN?

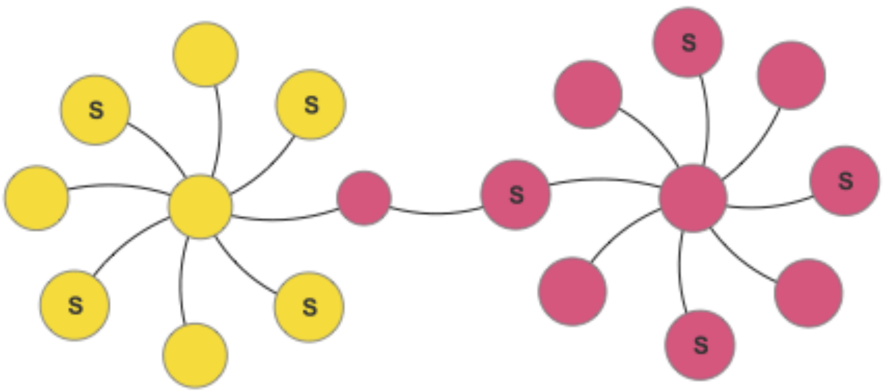
$$P(x_i = c|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = c|\mathcal{N}_j),$$

Algorithmically:

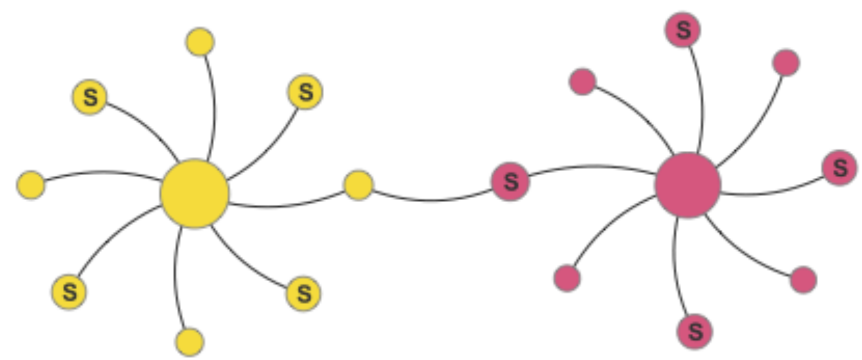
- HF propagates weights and then resets the seeds to their initial value
- MRW propagates weights and does not reset seeds

MultiRank Walk vs HF/wvRN/CoEM

Seeds are marked S

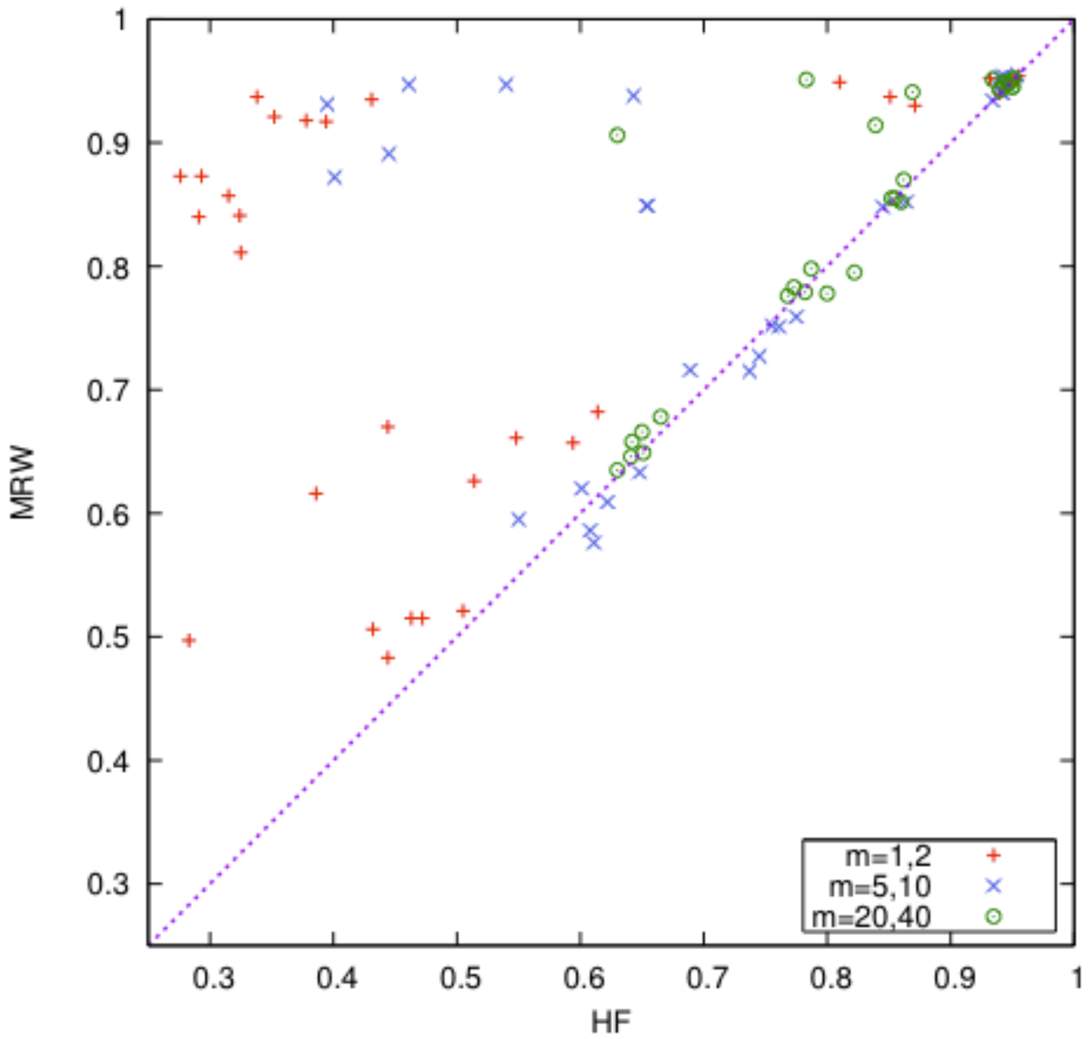


HF



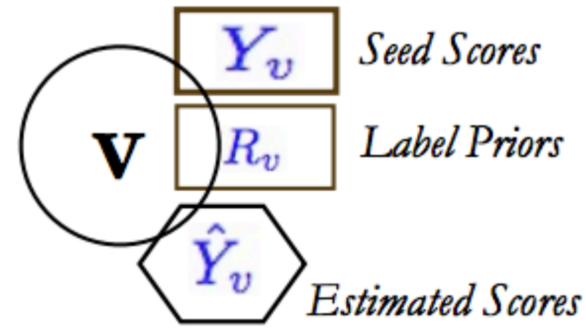
MRW

MultiRank Walk vs HF/wvRN/CoEM



SSL as optimization and Modified Adsorption 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

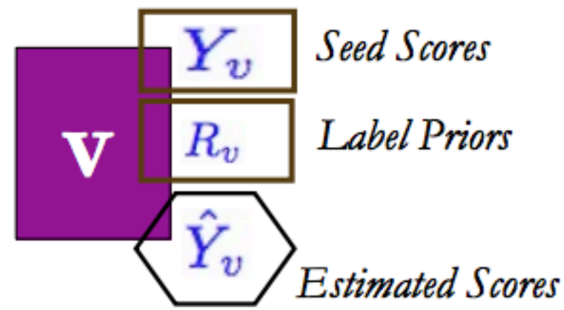
Modified Adsorption (MAD)

[Talukdar and Crammer, ECML 2009]

match seeds — smoothness — prior

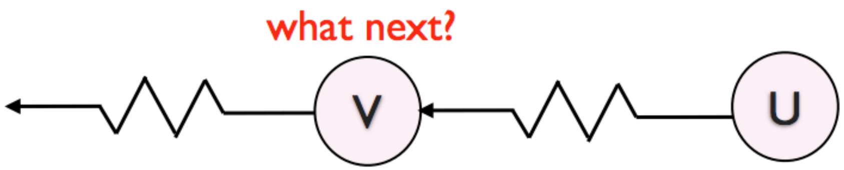
$$\arg\min_{\hat{\mathbf{Y}}} \sum_{l=1}^{m+1} \left[\|\boldsymbol{S}\hat{\mathbf{Y}}_l - \boldsymbol{S}\boldsymbol{Y}_l\|^2 + \mu_1 \sum_{u,v} \boldsymbol{M}_{uv} (\hat{\mathbf{Y}}_{ul} - \hat{\mathbf{Y}}_{vl})^2 + \mu_2 \|\hat{\mathbf{Y}}_l - \boldsymbol{R}_l\|^2 \right]$$

- 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



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

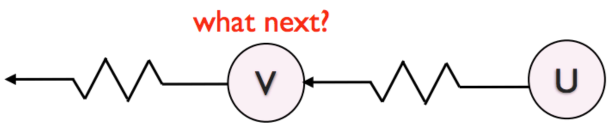
Adsorption SSL algorithm



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

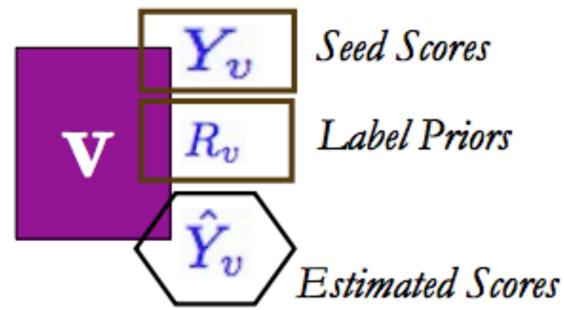
Dummy Label

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

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

The minimize with *Jacobi method* (which works for linear matrix equations like this one)

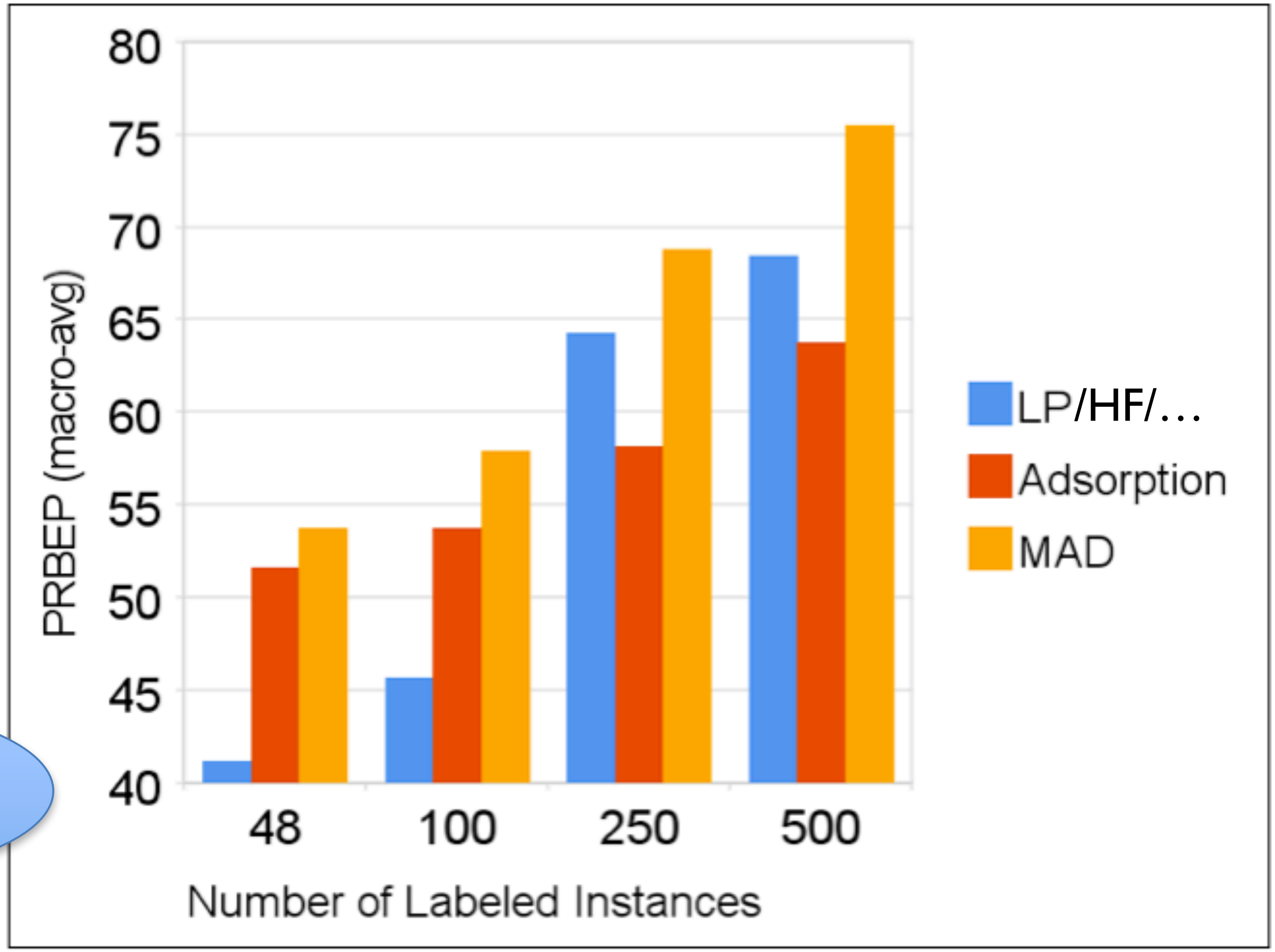
MapReduce Implementation of MAD

- 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/

k-NN graph

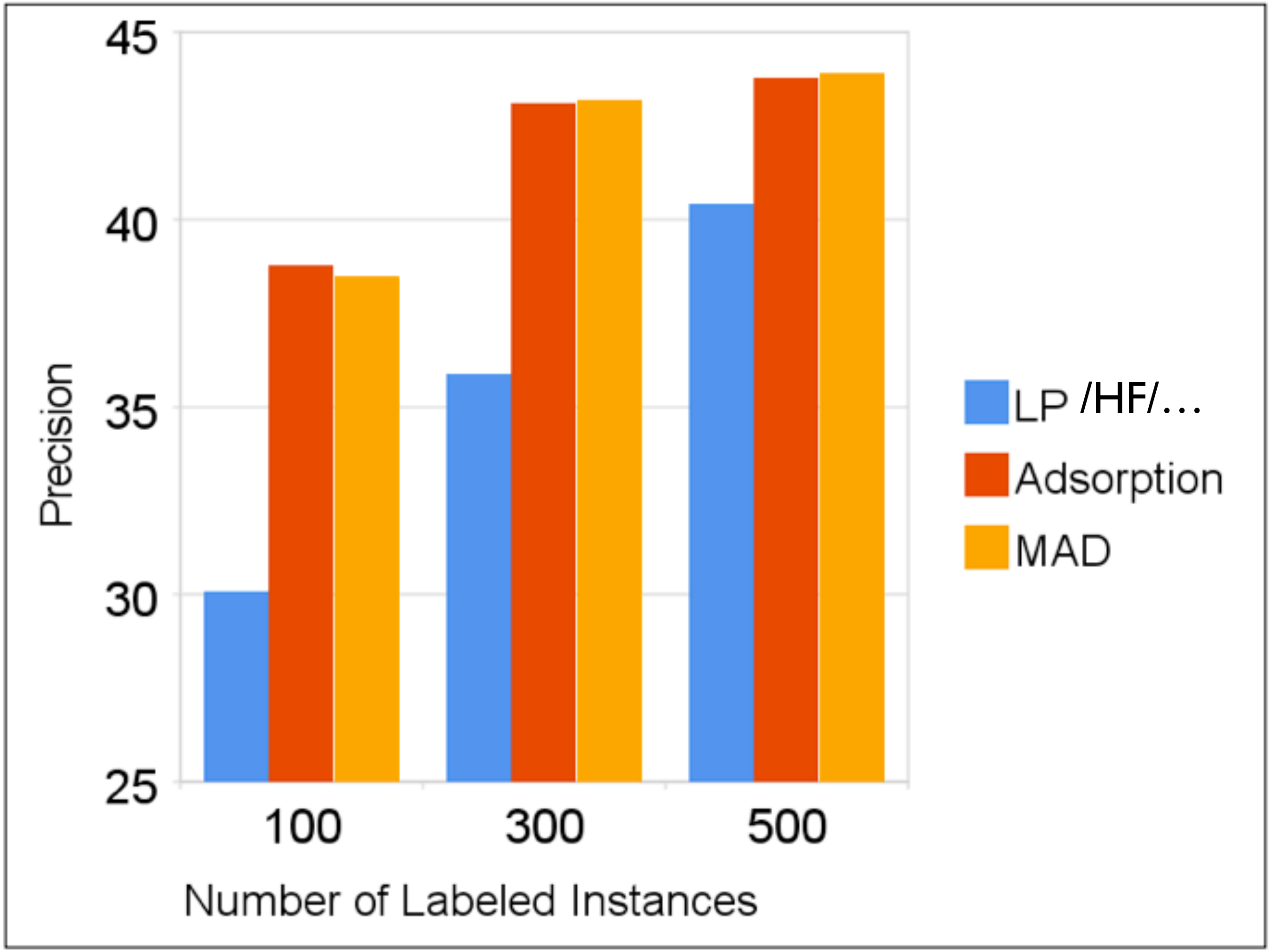
Text Classification



precision-recall break even point

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

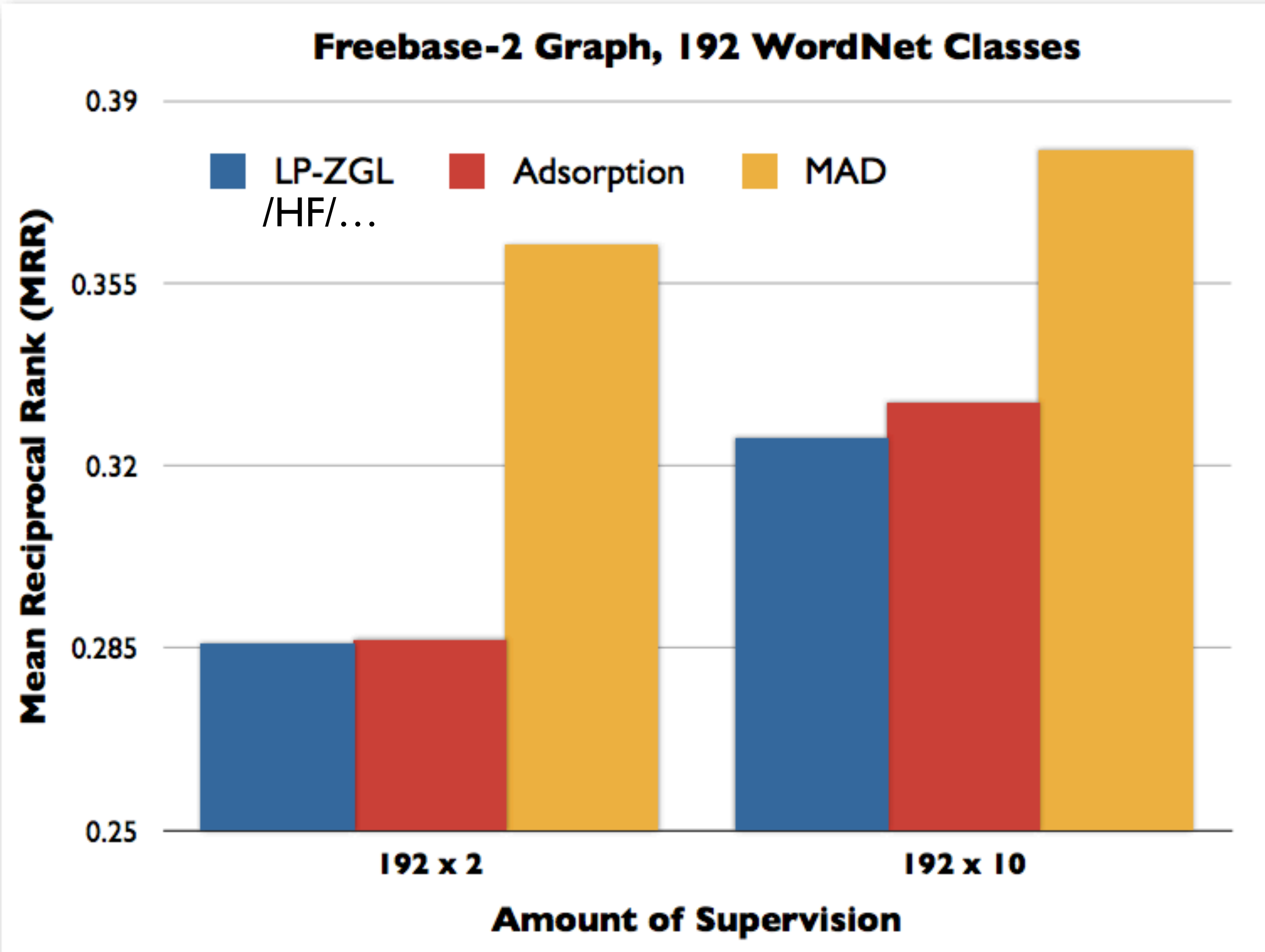
Sentiment Classification



Precision on 3568 Sentiment test instances

Class-Instance Acquisition

coupling graph

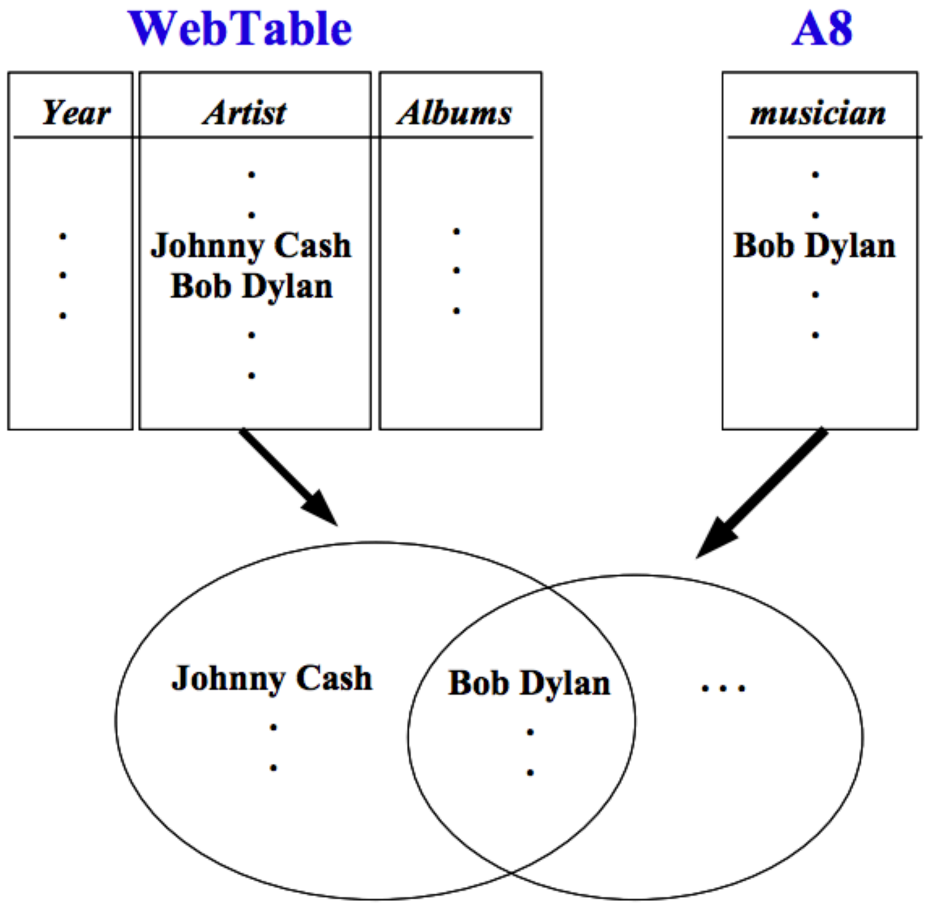


Graph with 303k nodes, 2.3m edges.

16

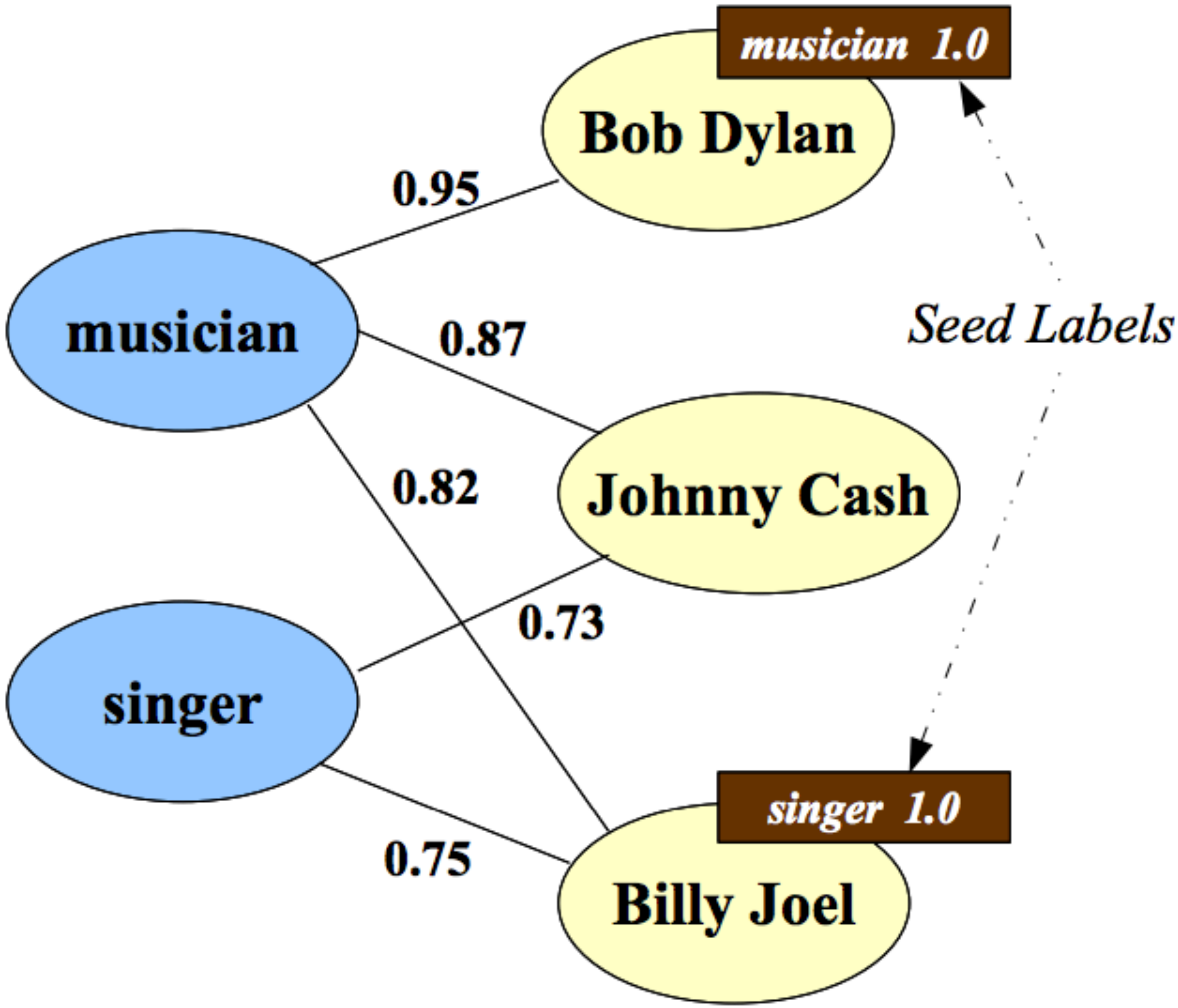
Assigning class labels to WebTable instances

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



from mining patterns like "musicians such as Bob Dylan"

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, …

17

Total classes: **9081**

Scaling up Graph SSL

Followup 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

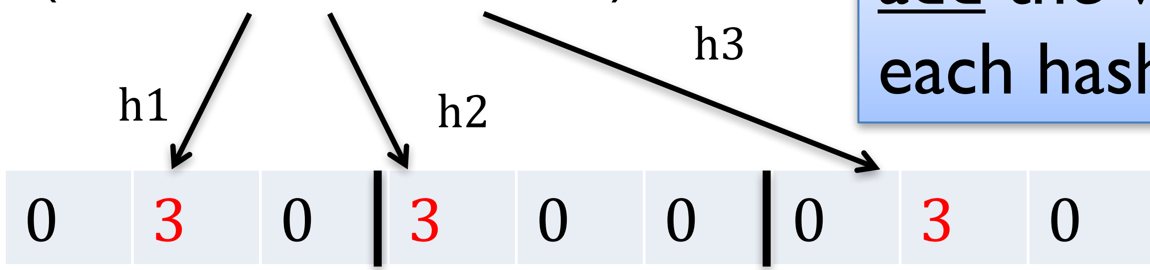
Count-min sketches

split a real vector into k ranges, one for each hash function

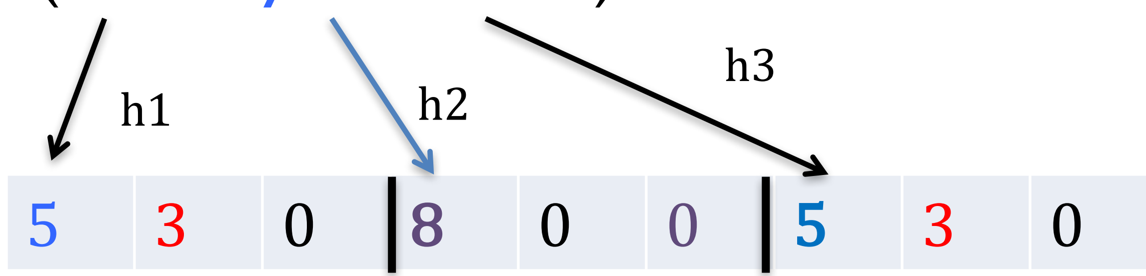
0	0	0	0	0	0	0	0	0

cm.inc("fred flintstone", 3):

add the value to each hash location



cm.inc("barney rubble",5):

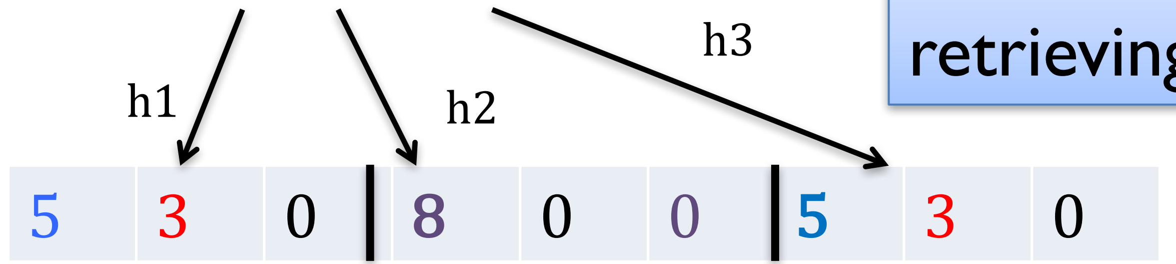


Count-min sketches

split a real vector into k ranges, one for each hash function

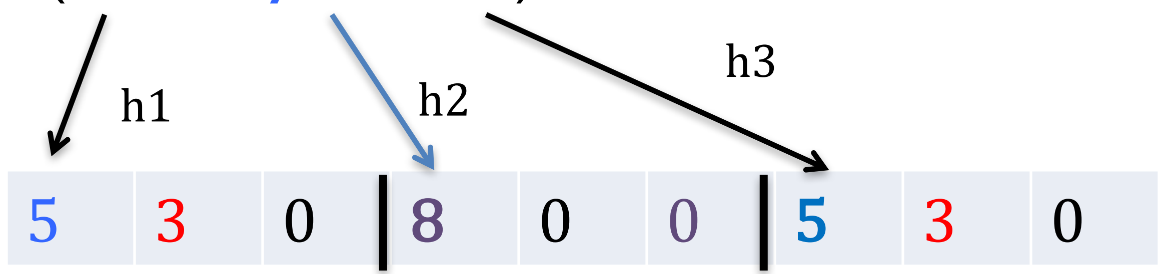
0	0	0	0	0	0	0	0	0

cm.get("fred flintstone"): 3



take min when retrieving a value

cm.get("barney rubble): 5

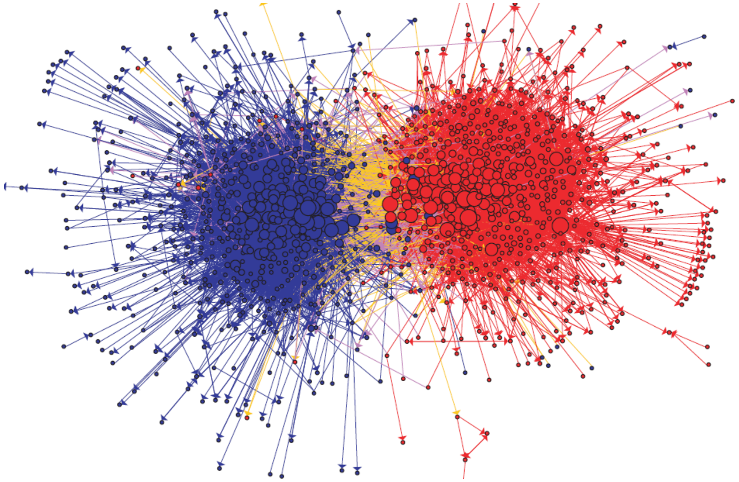


Followup work (AIStats 2014)

- 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*

Followup work (AIStats 2014)

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

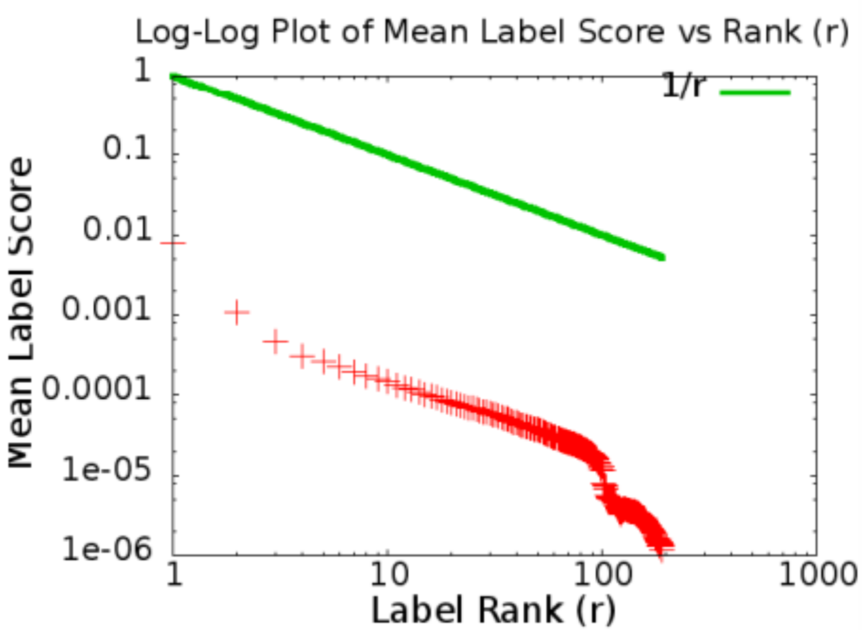


Followup work (AIStats 2014)

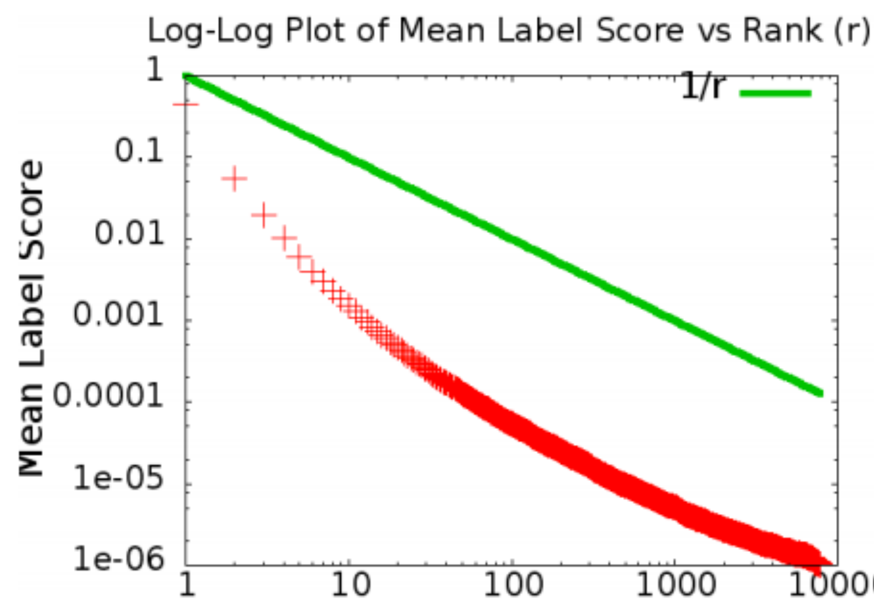
“self-injection”: similarity computation

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



Followup work (AIStats 2014)

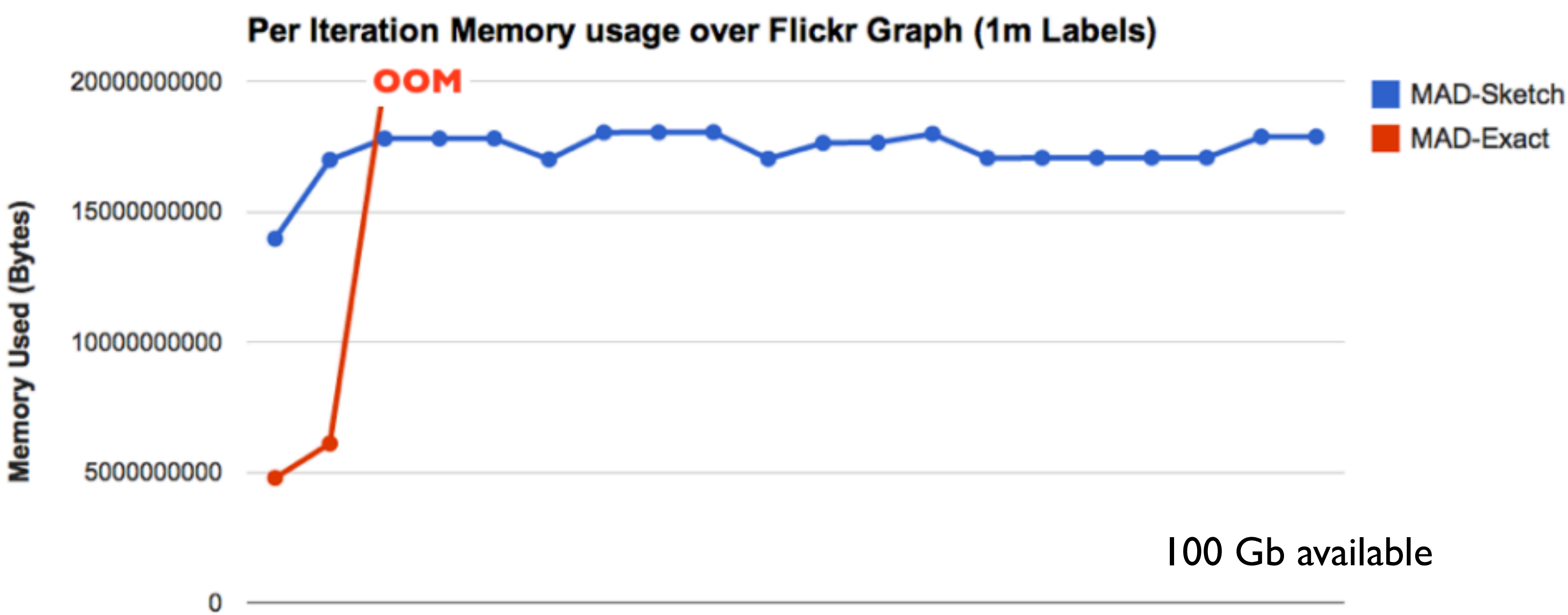
Name	Nodes (n)	Edges	Labels (m)	Seed Nodes	$k-$Sparsity	$\lceil \frac{ek}{\epsilon} \rceil$	$\lceil \ln \frac{m}{\delta} \rceil$
Freebase	301,638	1,155,001	192	1917	2	109	8
Flickr-10k	41,036	73,191	10,000	10,000	1	55	12
Flickr-1m	1,281,887	7,545,451	1,000,000	1,000,000	1	55	17

	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

Followup 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



Even more recent work

Large Scale Distributed Semi-Supervised Learning Using Streaming Approximation

Sujith Ravi
Google Inc., Mountain View, CA, USA
sravi@google.com

Qiming Diao[1]
Carnegie Mellon University, Pittsburgh, PA, USA
Singapore Mgt. University, Singapore
qiming.ustc@gmail.com

AIStats 2016

Differences: objective function

$$\mathcal{C}(\hat{\mathbf{Y}}) = \mu_1 \sum_{v \in V_l} s_{vv} ||\hat{\mathbf{Y}}_\mathbf{v} - \mathbf{Y}_\mathbf{v}||_2^2$$

$$+ \mu_2 \sum_{v \in V, u \in \mathcal{N}(v)} w_{vu} ||\hat{\mathbf{Y}}_\mathbf{v} - \hat{\mathbf{Y}}_\mathbf{u}||^2$$

$$+ \mu_3 \sum_{v \in V} ||\hat{\mathbf{Y}}_\mathbf{v} - \mathbf{U}||_2^2$$

$$s.t. \sum_{l=1}^{L} \hat{Y}_{vl} = 1, \forall v$$

seeds

smoothness

close to uniform label distribution

normalized predictions

Differences: scaling up

- Updates done in parallel with Pregel
- Replace count-min sketch with "streaming approach"
  - updates from neighbors are a "stream"
    - break stream into "sections"
      - maintain a list of $(y, Prob(y), \Delta)$
      - filter out labels at end of "section" if $Prob(y)+\Delta$ is small

Results with EXPANDER