Fast and Accurate Low Rank Approximation of Massive Graphs

Inderjit S. Dhillon
University of Texas at Austin

IPAM
Aug 25, 2010

Joint work with Berkant Savas
and Zhengdong Lu, Nagarajan Natarajan, Wei Tang, Vishvas Vasuki,
Tae Won Cho, Vacha Dave, Han Hee Song, Lili Qiu and Yin Zhang

Outline

- **Some Problems in Network Analysis**
  - Link prediction problem
  - Link prediction with multiple sources of information
  - Affiliation recommendation problem
- **Tools and Methods**
  - Path based methods for link prediction
  - Low rank approximations
  - Graph clustering
  - Clustered low rank approximation
- **Experimental Results**
- **Summary and Conclusions**

Networks in Applications

- Web graphs
- Collaboration and citation networks
- Social networks (Facebook, MySpace, LiveJournal, ...)
- Call graph networks (SKT, AT&T)
- Networks in Bioinformatics
- ...

Graphs

- Graph $G = (\mathcal{V}, \mathcal{E})$,
  $\mathcal{V}$ is a set of vertices, and
  $\mathcal{E}$ is a set of edges between vertices.
- The adjacency matrix $A = [a_{ij}]$ of G is given by

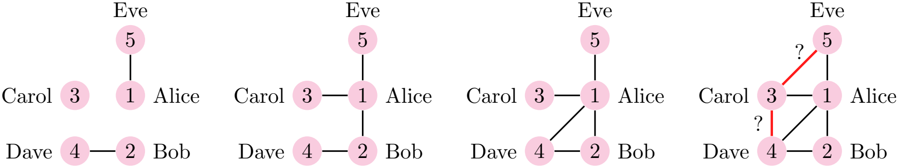
$$\begin{aligned} a_{ij} &= w_{ij} && \text{if there is an edge between vertex } i \text{ and } j \\ a_{ij} &= 0 && \text{otherwise} \end{aligned}$$

- Undirected graph then A is symmetric
- Directed graph then A is non-symmetric

Link Prediction Problem

Consider a social network that evolves with time,

$$\cdots \longrightarrow A^{(t-2)} \longrightarrow A^{(t-1)} \longrightarrow A^{(t)} \xrightarrow{?} A^{t+1}$$

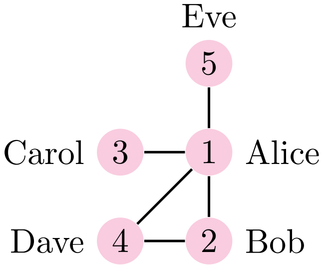


Q: Can we predict links that will form at time step $t+1$?

A: Many models exist, e.g. common neighbors, the Katz measure, etc.

Link prediction with multiple sources of information

In most cases, we have a social network between users

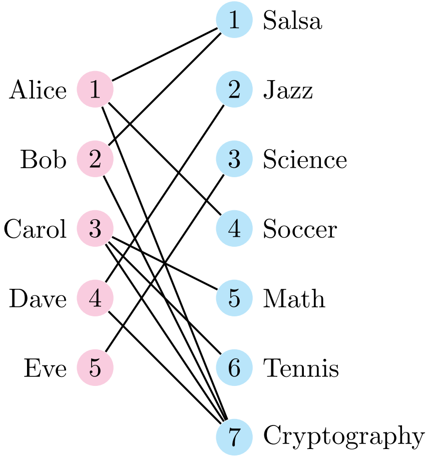


But also additional sources of information

- Other types of links between users, e.g. family ties, affiliations
- User profiles, demographic information
- Blog postings by users, movie ratings, etc
- ...

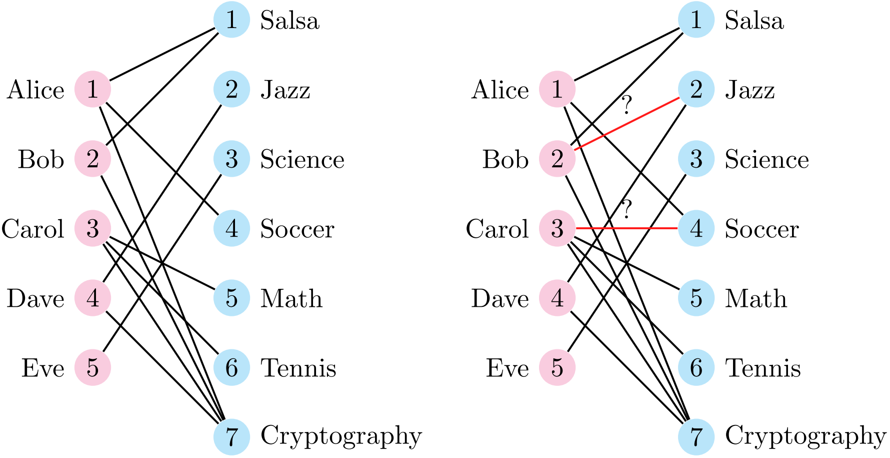
Affiliation recommendation/prediction

Consider a users $\times$ affiliations network



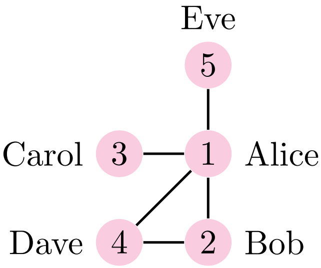
Affiliation recommendation

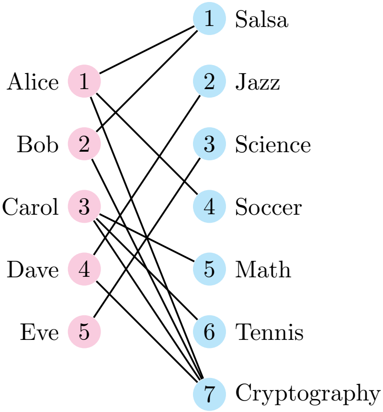
We want to suggest groups/affiliations to users



Affiliation recommendation given a social network

Suppose we additionally have a social network:





Can the social network help to improve affiliation recommendations?

Challenges

- Huge number of users and/or affiliations, e.g. Facebook, MySpace, Orkut, LiveJournal
- Multiple sources of information
- Need scalable algorithms and “accurate” predictions

Tools and Methods

Path based methods for link prediction

- No. of common neighbors can be computed as:

$$P_{\text{cn}} = A^2$$

  Yields no. of paths of length 2 between any two vertices

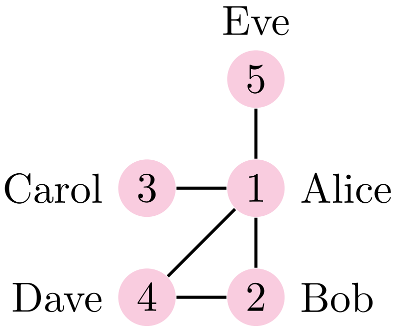
- Katz measure and truncated Katz measure

$$P_{\text{katz}} = \beta A + \beta^2 A^2 + \beta^3 A^3 + \cdots = \sum_{i=1}^{\infty} \beta^i A^i = (I - \beta A)^{-1} - I$$

$$P_{\text{t-katz}} = \beta A + \beta^2 A^2 + \cdots + \beta^k A^k = \sum_{i=1}^{k} \beta^i A^i$$

  - A^3 gives no. of paths of length 3, A^4 is no. of paths of length 4, & so on
  - Small β penalizes paths with larger lengths
  - Large β amplifies paths with larger lengths

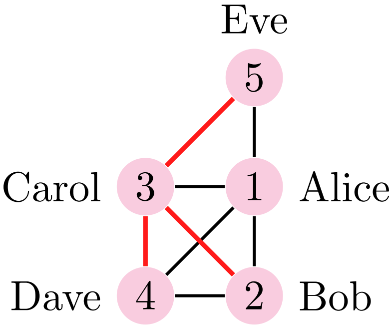
A social network and its adjacency matrix



$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Link prediction based on common neighbors

- No. of common neighbors is given by A^2
- Carol: common neighbor predicts links to Bob, Dave and Eve (in red).

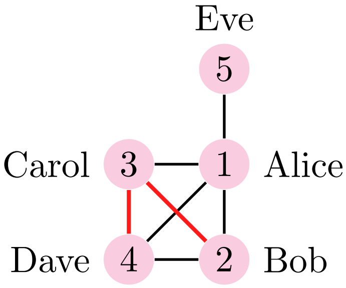


$$A^2 = \begin{bmatrix} 4 & 1 & 0 & 1 & 0 \\ 1 & 2 & \color{red}{1} & 1 & \color{red}{1} \\ 0 & \color{red}{1} & 1 & \color{red}{1} & \color{red}{1} \\ 1 & 1 & \color{red}{1} & 2 & \color{red}{1} \\ 0 & \color{red}{1} & \color{red}{1} & \color{red}{1} & 1 \end{bmatrix}$$

Link prediction based on Katz measure

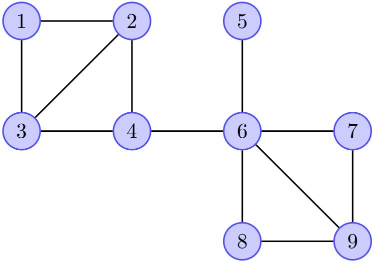
$$K(A,\beta) = \beta A + \beta^2 A^2 + \beta^3 A^3 + \cdots = (I - \beta A)^{-1} - I$$

- With $\beta = 0.4$
- Top two predictions for Carol are to Bob and Dave (in red)
- Katz score between Carol & Eve is smaller than other scores (in green)



$$K(A, 0.4) = \begin{bmatrix} * & * & * & * & * \\ * & * & 1.8 & * & 1.8 \\ * & 1.8 & * & 1.8 & 1.1 \\ * & * & 1.8 & * & 1.8 \\ * & 1.8 & 1.1 & 1.8 & * \end{bmatrix}$$

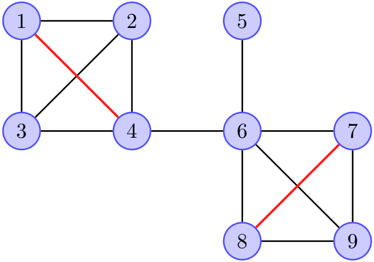
After a few time steps....



$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Common Neighbors

- All possible predictions in red, based on common neighbors.
- Possible recommendation: top two highest scores

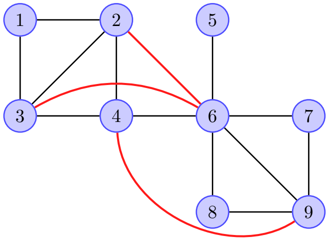


$$A^2 = \begin{bmatrix} 2 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 \\ 1 & 3 & 2 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 2 & 3 & 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 1 & 1 & 3 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 5 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 & 3 \end{bmatrix}$$

Katz measure

$$Katz(A,\beta) = \sum_{i=1}^{\infty} \beta^i A^i = (I - \beta A)^{-1} - I, \qquad \beta < |\lambda_i| \quad \forall\ i$$

- With $\beta = 1/3$ top three Katz scores are the $\color{red}{X}$'s



$$\begin{bmatrix}
* & * & * & * & * & * & * & * & * \\
* & * & * & * & * & \color{red}{X} & * & * & * \\
* & * & * & * & * & \color{red}{X} & * & * & * \\
* & * & * & * & * & * & * & * & \color{red}{X} \\
* & * & * & * & * & * & * & * & * \\
* & \color{red}{X} & \color{red}{X} & * & * & * & * & * & * \\
* & * & * & * & * & * & * & * & * \\
* & * & * & * & * & * & * & * & * \\
* & * & * & \color{red}{X} & * & * & * & * & *
\end{bmatrix}$$

Importance of low rank approximations

- Enables scalability
- Can filter out noise
- Introduces interpretable features...
- For example, with the spectral approximation $A \approx V \Lambda V^{\mathsf{T}}$,

$$A^k \approx (VDV^{\mathsf{T}})^k = (VDV^{\mathsf{T}})(VDV^{\mathsf{T}}) \cdots (VDV^{\mathsf{T}}) = VD^kV^{\mathsf{T}}$$

Link predictors with low rank approximations

- With spectral approximation,

$$A \approx V \Lambda V^{\mathsf{T}}$$

- Common neighbor,

$$P_{\text{cn}} = A^2, \qquad P_{\text{cn-lr}} = V \Lambda^2 V^{\mathsf{T}}$$

- Katz measure,

$$P_{\text{katz}} = (I - \beta A)^{-1} - I = \sum_{i=1}^{\infty} \beta^i A^i \qquad P_{\text{t-katz}} = \sum_{i=1}^{k} \beta^i A^i$$

$$P_{\text{katz-lr}} = V \left((I - \beta \Lambda)^{-1} - I \right) V^{\mathsf{T}} \qquad P_{\text{t-katz-lr}} = \sum_{i=1}^{k} \beta^i V \Lambda^i V^{\mathsf{T}}$$

Generalized Katz, evolving graphs and model learning

Replace β in truncated Katz with $\alpha_1, \ldots, \alpha_k$

$$P_{\text{t-katz}} = \sum_{i=1}^{k} \beta^i A^i \quad \Rightarrow \quad P_{\text{gt-katz}} = \sum_{i=1}^{k} \alpha_i A^i$$

Social network graphs are evolving....

- A adjacency matrix at time step t_1
- B adjacency matrix at time step $t_2 > t_1$
- Usually $B = A + \Delta$

Learn α_i from snapshots A and B

$$\min_{\alpha_i} \left\| \sum_{i=1}^{k} \alpha_i A^i - B \right\|_F \quad \text{or rather} \quad \min_{\alpha_i} \left\| \left(\sum_{i=1}^{k} \alpha_i A^i - B \right) \cdot M \right\|_F$$

where $M_{ij} = 0$ if $A_{ij} = 1$ and $B_{ij} = 1$, otherwise $M_{ij} = 1$

Link prediction with multiple sources(networks)

- Paths with edges in A:

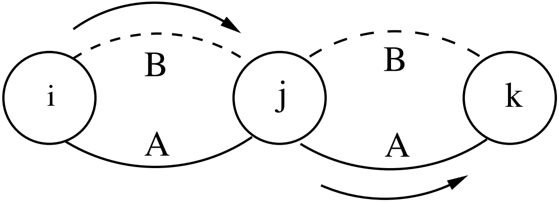
$$i \xrightarrow{A} j \xrightarrow{A} k, \qquad A^2$$

- Paths with edges in B:

$$i \xrightarrow{B} j \xrightarrow{B} k, \qquad B^2$$

- Hybrid paths with edges in A and B:

$$i \xrightarrow{B} j \xrightarrow{A} k, \qquad BA$$



Link prediction with multiple sources(networks)

- In general, utilizing single source and hybrid source paths of maximum length $l_{\max}$ involves terms from

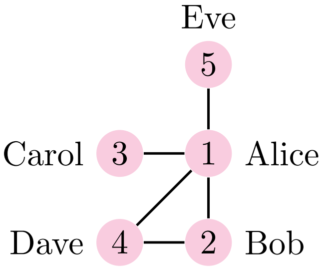
$$(I + A + B + C)^{l_{\max}}$$

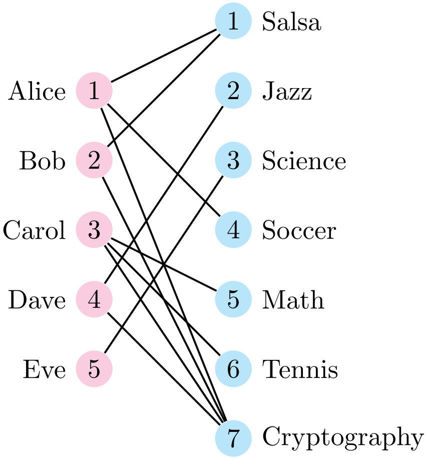
- Compare with truncated Katz
  - Linear combination in terms of $\{A,\ A^2,\ \cdots,\ A^{l_{\max}}\}$
  - Linear combination in terms of

$$\{A,\ B,\ C,\ A^2,\ B^2,\ C^2,\ AB,\ AC,\ BA,\ BC,\ CA,\ CB\} \qquad l_{\max} = 2$$

- Weights learned using "Hierarchical sparsity regularization"
- Utilizing multiple sources can improve link prediction
- Experiments with Arxiv and CiteSeer data
- Sources include:
  1. author $\times$ author collaboration network
  2. paper $\times$ paper citation network
  3. author $\times$ paper network
  4. author $\times$ author networks based on similarities of abstracts, titles and keywords.

Affiliation recommendation





Affiliation recommendation given a social network

A is the affiliation network, S is the social network

- Method 1: Low rank approximation of A
- Method 2: Low rank approximations of the combined matrix

$$C = \begin{bmatrix} \lambda S & A \\ A^T & 0 \end{bmatrix} \approx \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \Lambda \begin{bmatrix} V_1^T & V_2^T \end{bmatrix},$$

  where λ controls amount of influence of the social network.
- Predictions based on $V_1 \Lambda V_2^{\mathrm{T}}$

Affiliation recommendation given a social network

- Method 3: Katz or truncated Katz measure on the combined matrix

$$P_{\text{katz}} = \beta C + \beta^2 C^2 + \cdots$$

$$P_{\text{t-katz}} = \beta C + \beta^2 C^2 + \cdots + \beta^l C^l$$

  Predictions are based on the (1,2) block (corresponding to A in C), e.g.

$$P^{(1,2)}_{\text{t-katz}} = \beta A + \beta^2 \lambda S A + \beta^3 \lambda^2 (S^2 A + A A^{\mathsf{T}} A)$$

  We can use both low rank approach and Katz measure approach.

- Method 4: Common subspace to approximate A and S

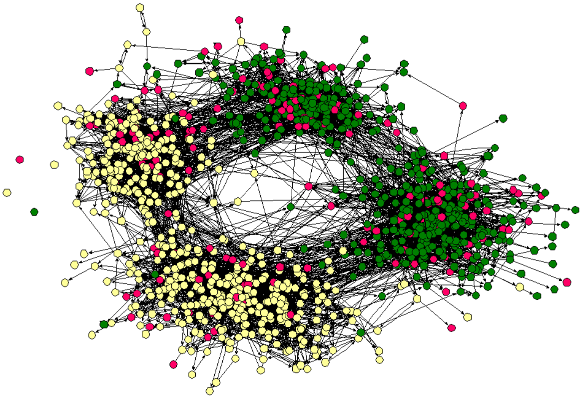
$$A \approx Q D_A V^{\mathsf{T}}, \qquad S \approx Q D_S Q^{\mathsf{T}}$$

  Q spans dominant subspaces of both A and S

Graph Clustering/Community Detection

Graph Partitioning/Clustering

- In many applications, goal is to partition/cluster nodes of a graph:

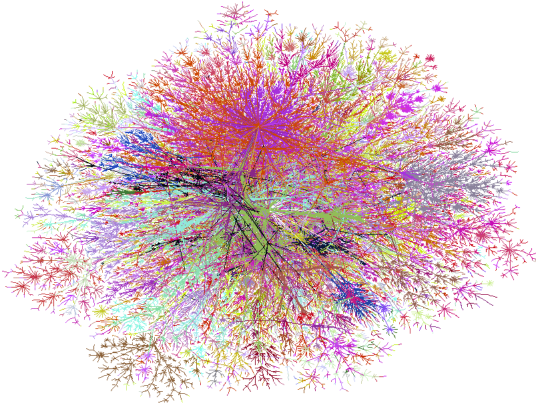


High School Friendship Network

[James Moody. American Journal of Sociology, 2001]

Graph Partitioning/Clustering

- In many applications, goal is to partition/cluster nodes of a graph:



The Internet

[The Internet Mapping Project, Hal Burch and Bill Cheswick, Lumeta Corp, 1999]

Graph clustering

- Cluster vertices $\mathcal{V}$ into c disjoint sets $\mathcal{V}_i$,

$$\mathcal{V} = \cup_{i=1}^{c} \mathcal{V}_i \quad \text{and} \quad \mathcal{V}_i \cap \mathcal{V}_j = \emptyset \qquad i \neq j$$

- Objective functions for clustering

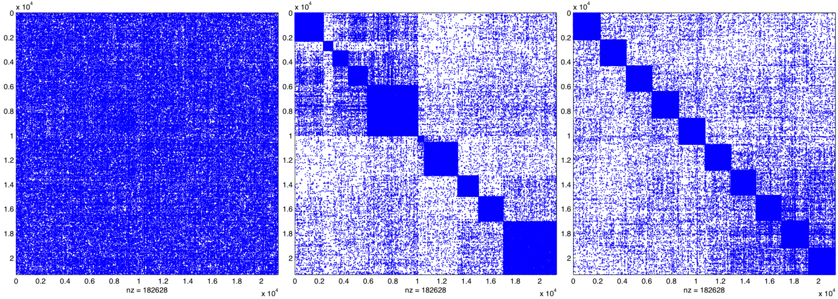
$$\text{RCut} = \min_{\mathcal{V}_1,\ldots,\mathcal{V}_c} \sum_{i=1}^{c} \frac{\text{links}(\mathcal{V}_i, \mathcal{V} n \mathcal{V}_i)}{|\mathcal{V}_i|} \qquad \text{(Chan et al. 1994)}$$

$$\text{NCut} = \min_{\mathcal{V}_1,\ldots,\mathcal{V}_c} \sum_{i=1}^{c} \frac{\text{links}(\mathcal{V}_i, \mathcal{V} n \mathcal{V}_i)}{\text{degree}(\mathcal{V}_i)} \qquad \text{(Shi and Malik 2000)}$$

$$\text{KLObj} = \min_{\mathcal{V}_1,\ldots,\mathcal{V}_c} \sum_{i=1}^{c} \frac{\text{links}(\mathcal{V}_i, \mathcal{V} n \mathcal{V}_i)}{|\mathcal{V}_i|} \quad \text{s.t.} \quad |\mathcal{V}_i| = |\mathcal{V}_j| \qquad \text{(Kernighan and Lin 1970)}$$

- Spectral clustering can be expensive for massive graphs
- Fast clustering algorithms without eigenvector computations
  - GRACLUS (Dhillon et al. 2007) and METIS (Karypis and Kumar 1999)

Clustering example: arXiv data graph **condMat**



# of clust	edges in clust	# of vertices	# of edges	relative error
10 (graclus)	79.8 %	21,363	182,628	45.0
10 (metis)	77.9 %	21,363	182,628	47.0

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1c} \\ \vdots & \ddots & \vdots \\ A_{c1} & \cdots & A_{cc} \end{bmatrix}$$

Motivation: main idea through an example

$$A = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}$$

Let $\sigma_1(A_{22}) < \sigma_k(A_{11})$ and $\bar{U}_k \bar{\Sigma}_k \bar{V}_k^{\mathsf{T}} \approx A_{11}$

$$A \approx U_k \Sigma_k V_k^{\mathsf{T}} = \begin{bmatrix} \bar{U}_k \bar{\Sigma}_k \bar{V}_k^{\mathsf{T}} & 0 \\ 0 & 0 \end{bmatrix}$$

No information from A_{22} is extracted.
Approximate each block: $\bar{U}_k \bar{\Sigma}_k \bar{V}_k^{\mathsf{T}} \approx A_{11}$ and $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^{\mathsf{T}} \approx A_{22}$

$$A \approx \begin{bmatrix} \bar{U}_k & 0 \\ 0 & \hat{U}_k \end{bmatrix} \begin{bmatrix} \bar{\Sigma}_k & 0 \\ 0 & \hat{\Sigma}_k \end{bmatrix} \begin{bmatrix} \bar{V}_k & 0 \\ 0 & \hat{V}_k \end{bmatrix}^{\mathsf{T}},$$

Main idea continued

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

Compute a low rank approximation of each diagonal block

$$U_i S_{ii} V_i^{\mathsf{T}} \approx A_{ii} \qquad U_i,\ V_i \text{ orthonormal}$$

Approximate now

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \approx \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix}^{\mathsf{T}}$$

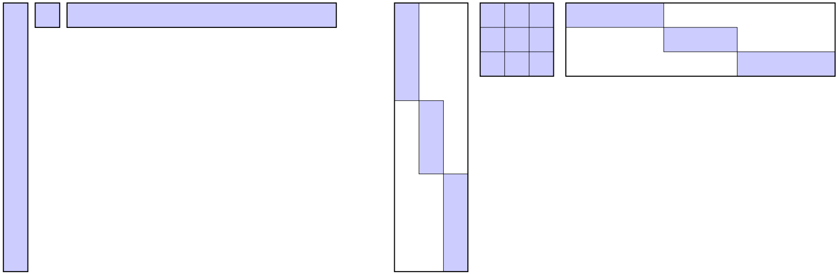
where

$$S_{ij} = U_i^{\mathsf{T}} A_{ij} V_j$$

Low rank vs clustered low-rank

Low rank: $A \approx U \Sigma V^{\mathsf{T}}$

Clustered low rank: $A \approx \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix}^{\mathsf{T}}$



- Observe $\mathrm{diag}(U_1, U_2, \cdots, U_c)$ and has the same memory usage as U
- Experiments show that most of U is contained in $\mathrm{diag}(U_1, U_2, \cdots, U_c)$
- Similarly of V and $\mathrm{diag}(V_1, V_2, \cdots, V_c)$

Algorithm: Clustered low rank approximation

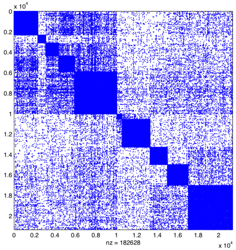
Input: An $m \times m$ adjacency matrix A of a graph, number of clusters c
Output: Clustered low rank approximation of A

1: Cluster the graph into c clusters
2: Compute a low rank approximation of each cluster

$$U_i S_i V_i \approx A_{ii}$$

3: Extend the cluster-wise approximations, into an approximation for the entire matrix A

$$S_{ij} = U_i^{\mathsf{T}} A_{ij} V_j$$



Different low rank approximations

- Deterministic methods — truncated SVD using ARPACK, PROPACK, SVDPACK.
- Stochastic methods

$$Y = A\Omega \qquad \text{or} \qquad Y = (AA^T)^q A\Omega$$

where q is small (1,2 or 3), and Ω is a random matrix (standard Gaussian). An approximation is obtained with

$$A \approx \hat{A} = P_Y A = Y(Y^T Y)^{-1} Y^T A$$

where P_Y is an orthogonal projection onto the range space of Y

Deterministic error bound—preliminaries

For any partitioning

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1c} \\ \vdots & \ddots & \vdots \\ A_{c1} & \cdots & A_{cc} \end{bmatrix}$$

and any $Y^{(i)}$ form

$$Y = \begin{bmatrix} Y^{(1)} & & \\ & \ddots & \\ & & Y^{(c)} \end{bmatrix} \qquad P_Y = \begin{bmatrix} P_{Y^{(1)}} & & \\ & \ddots & \\ & & P_{Y^{(c)}} \end{bmatrix}$$

P_Y and each $P_{Y^{(i)}}$ are orthogonal projectors

Deterministic clustered error bound

The approximation error is given by

$$\|A - \hat{A}\| = \|A - P_Y A\| = \|(I - P_Y)A\|$$

Theorem

$$\|(I - P_Y)A\|_F^2 \leq \sum_{i,j=1}^{c} \|(I - P_{Y^{(i)}})A_{ij}\|_F^2$$

Introduce variables

Assume we have c clusters

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1c} \\ \vdots & \ddots & \vdots \\ A_{c1} & \cdots & A_{cc} \end{bmatrix}$$

Generate

$$Y^{(i)} = A_{ii}\Omega^{(i)}$$
$$Y^{(i)} = (A_{ii}A_{ii}^{\top})^{q}A_{ii}\Omega^{(i)}$$

using random $\Omega^{(i)}$

$$Y = \begin{bmatrix} Y^{(1)} & & \\ & \ddots & \\ & & Y^{(c)} \end{bmatrix} \qquad P_Y = \begin{bmatrix} P_{Y^{(1)}} & & \\ & \ddots & \\ & & P_{Y^{(c)}} \end{bmatrix}$$

Stochastic error bounds in the clustered setting

Theorem

$$\mathbb{E}\|(I - P_Y)A\|_F \leq \sum_{i=1}^{c} \left(1 + \frac{k_i}{p_i + 1}\right)^{1/2} \|\Sigma_2^{(i)}\|_F + \sum_{i,j=1,\ i \neq j}^{c} \|A_{ij}\|_F^2$$

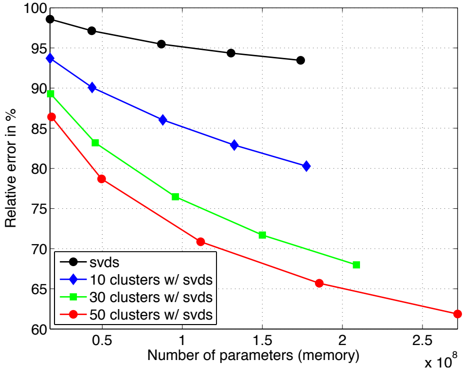
$$\mathbb{E}\|(I - P_Y)A\|_2 \leq \sum_{i=1}^{c} \left(\left(1 + \frac{\sqrt{k_i}}{\sqrt{p_i - 1}}\right)\|\Sigma_2^{(i)}\|_2 + \frac{e\sqrt{k_i + p_i}}{p_i}\|\Sigma_2^{(i)}\|_F\right) + \sum_{i,j=1,\ i \neq j}^{c} \|A_{ij}\|_2^2$$

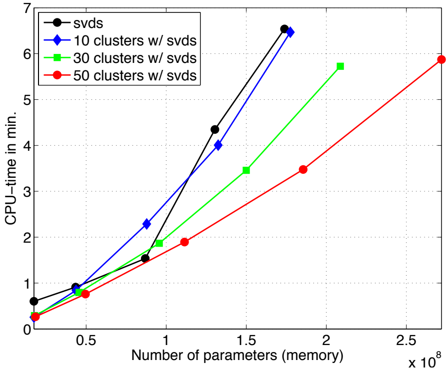
$$A_{ii} \approx U^{(i)}\Sigma^{(i)}(V^{(i)})^\mathsf{T} = U^{(i)} \begin{bmatrix} \Sigma_1^{(i)} & 0 \\ 0 & \Sigma_2^{(i)} \end{bmatrix} \begin{bmatrix} V_1^{(i)} & V_2^{(i)} \end{bmatrix}^\mathsf{T}$$

Experimental Results

Google web matrix—performance comparison

- 434,818 vertices; 3,419,124 edges
- Edges within clusters: 98.2%, 97.8%, 97.6%; graph is highly clusterable
- "Ranks" are $k = 20, 50, 100, 150, 200$.





LiveJournal graph

- 3,828,682 vertices; 65,825,429 edges
- Edges within clusters: 76.9%, 69.3%, 66.3%; graph not highly clusterable
- "Ranks" are $k = 20, 50, 100, 150, 200$.

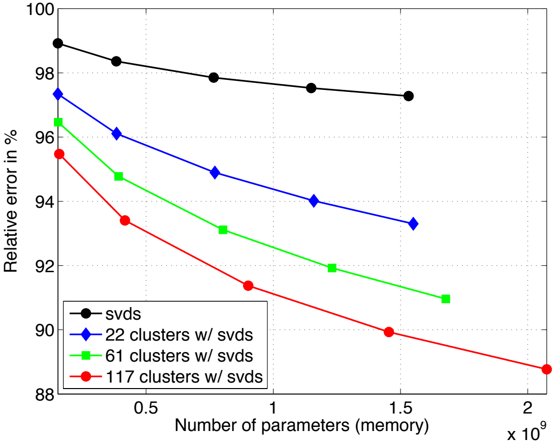
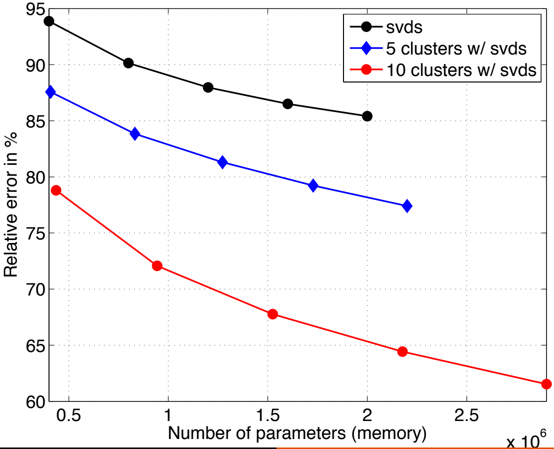


Image segmentation graph

Here we use the normalized adjacency matrix:

- 10,000 vertices; 1,091,910 edges
- Edges within clusters: 77.5%, 70.4%; graph not highly clusterable



Testbed of Social Networks

Network	Date	# nodes	# links	# added links	% added links
Flickr	4/14/2007	1,990,149	41,302,536	–	–
	4/25/2007	1,990,149	42,056,754	754,218	1.8%
	5/6/2007	1,990,149	42,879,714	822,960	1.9%
LiveJournal	02/16/2009	1,770,961	83,663,478	–	–
	03/4/2009	1,770,961	84,413,542	750,064	0.8%
	04/03/2009	1,770,961	85,713,766	1,300,224	1.5%
MySpace	12/11/2008	2,137,264	90,333,122	–	–
	1/11/2009	2,137,264	90,979,264	646,142	0.7%
	2/14/2009	2,137,264	91,648,716	669,452	0.7%

	# clusters	avg size	% intra links	% inter links
Flickr	18	110,563	71.8%	28.2%
LiveJournal	17	106,241	72.5%	27.5%
MySpace	17	125,721	51.9%	48.1%

Link prediction evaluation — have to sample

Introduce the sets

$$\begin{aligned} \mathcal{E} &= \{(i,j) \mid A_{ij} = 1\}, && \text{“old links”} \\ \mathcal{N} &= \{(i,j) \mid A_{ij} = 0 \ \&\ B_{ij} = 1\}, && \text{newly formed links} \\ \mathcal{Z} &= \{(i,j) \mid A_{ij} = 0 \ \&\ B_{ij} = 0\}, && \text{no links between vertices } i \text{ and } j \end{aligned}$$

Concrete example (MySpace): A is 2.1M $\times$ 2.1M matrix with

$$|\mathcal{E}| \approx 0.0020\%, \quad |\mathcal{N}| \approx 1.46 \cdot 10^{-5}\%, \quad |\mathcal{Z}| \approx 99.998\%$$

Training Randomly sample 100,000 links from $\mathcal{N}$ and 500,000 links from $\mathcal{Z}$, true ratio 1 to 6.8M

Testing Randomly sample testing subsets from $\mathcal{N}$ and $\mathcal{Z}$, compute corresponding scores, and predict links on highest scores.

Predictions Generic predictor in factored form

$$P = \bar{V}\bar{D}\bar{V}^{\mathsf{T}}$$

we can compute P_{ij} for any ij

Performance of link predictors

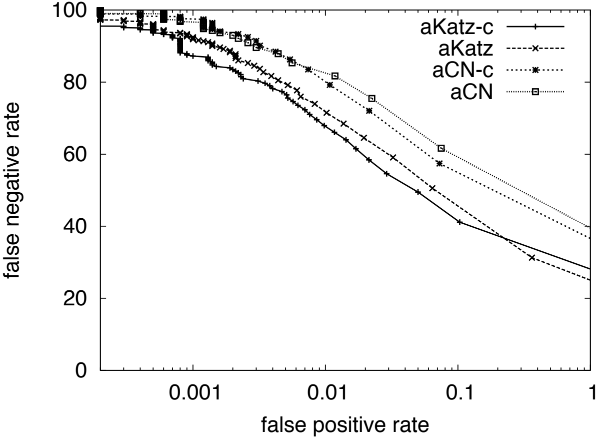
1. Compute scores for 100k positive & 500k negative links (600k total)
2. Sort the scores
3. For a given threshold consider all scores above this threshold as predictions of friendships
4. Plot false positive rate (FPR) vs false negative rate (FNR) for a range of threshold values

$$
\begin{aligned}
FPR &= \frac{\#\text{of incorrectly predicted friend links}}{\#\text{of non-friend pairs}} \\
FNR &= \frac{\#\text{of missed friend links}}{\#\text{of new friend links}}
\end{aligned}
$$

5. Each threshold value gives one point in the plot

Link prediction results on LiveJournal

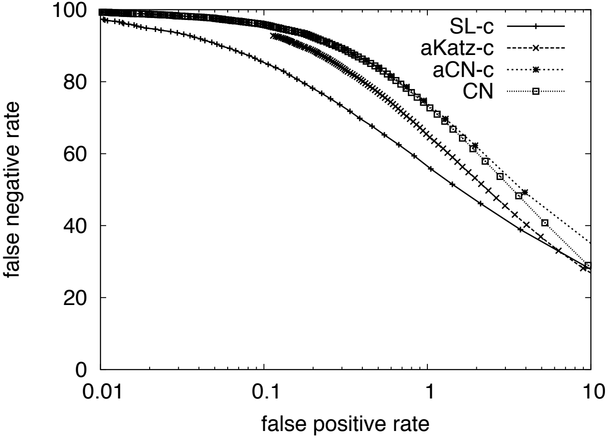
Lower-left better; "-c" indicates clustering is involved



Clustering improves both Katz measures and Common neighbors method

Link prediction results on Flickr

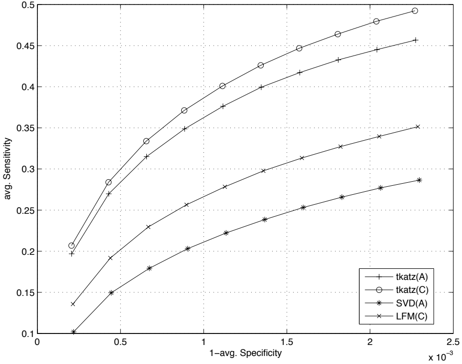
Lower-left better; "-c" indicates clustering is involved



Best method "SL-c" Spectral learning with clustering (not in this talk)

Affiliation recommendation on Youtube data

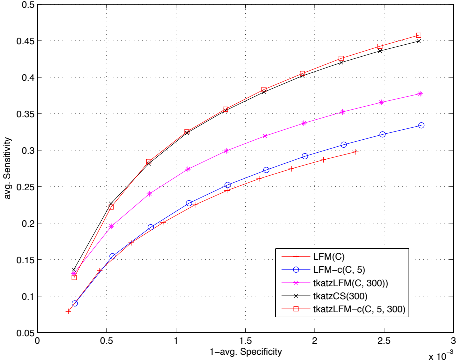
Upper-left is better



Clear improvement in performance by utilizing the social network (for both truncated Katz and latent factors model (LFM)).

Affiliation recommendation on Youtube data

Upper-left is better



Best two methods: truncated Katz measure using Common Subspaces and truncated Katz measure using latent factors model and clustering.

Clustering gives a substantial boost in performance (compare "pink" with "red square")

Summary and Conclusions

- Clustered low rank approximation
  - Improved quality of approximation with identical memory consumption faster than state-of-the-art (PROPACK, SVDPACK, ARPACK). Note: clustering time included.
- Prediction methods
  - Katz and CN using graph embedding
  - Generalized truncated Katz
- Utilizing multiple sources improves link prediction
- Utilizing social network improves affiliation recommendation
- Improved and scalable computation of proximity measures & link predictions with clustered low rank approximation
- Fruitful to combine clustering structure with low rank approximation framework

References

[1] B. Savas and I. S. Dhillon. *Fast and accurate low rank approximation of massive graphs*. Technical report TR–10–18. Department of Computer Science, The University of Austin, 2010.

[2] H. H. Song, B. Savas, T. W. Cho, V. Dave, Z. Lu, I. S. Dhillon, Y. Zhang, and L. Qiu. *Clustered embedding of massive online social networks*. In preparation.

[3] V. Vasuki, N. Natarajan, Z. Lu B. Savas and I. S. Dhillon. *Scalable Affiliation Recommendation using Auxiliary Networks*. Submitted for publication, 2010.

[4] Z. Lu, B. Savas, W. Tang, and I. S. Dhillon. *Supervised link prediction using multiple sources*. Submitted for publication, 2010.