Scalable Affiliation Recommendation using Auxiliary Networks

VISHVAS VASUKI
NAGARAJAN NATARAJAN
ZHENGDONG LU
BERKANT SAVAS
and
INDERJIT DHILLON
University of Texas at Austin

Social network analysis has attracted increasing attention in recent years. In many social networks, besides friendship links amongst users, the phenomenon of users associating themselves with groups or communities is common. Thus, two networks exist simultaneously: the friendship network among users, and the affiliation network between users and groups. In this paper, we tackle the affiliation recommendation problem, where the task is to predict or suggest new affiliations between users and communities, given the current state of the friendship and affiliation networks. More generally, affiliations need not be community affiliations—they can be a user's taste, so affiliation recommendation algorithms have applications beyond community recommendation. In this paper, we show that information from the friendship network can indeed be fruitfully exploited in making affiliation recommendations. Using a simple way of combining these networks, we suggest two models of user-community affinity for the purpose of making affiliation recommendations: one based on graph proximity, and another using latent factors to model users and communities. We explore the affiliation recommendation algorithms suggested by these models and evaluate these algorithms on two real world networks—Orkut and Youtube. In doing so, we motivate and propose a way of evaluating recommenders, by measuring how good the top 50 recommendations are for the average user, and demonstrate the importance of choosing the right evaluation strategy. The algorithms suggested by the graph proximity model turn out to be the most effective. We also introduce scalable versions of these algorithms, and demonstrate their effectiveness. This use of link prediction techniques for the purpose of affiliation recommendation is, to our knowledge, novel.

Categories and Subject Descriptors: H.2.8 [**Database applications - Data mining**]:

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Scalability

A preliminary version of this paper was presented at RecSys '10, the Third ACM Conference on Recommender Systems [Vasuki et al. 2010].
Author's address: Department of Computer Science and ICES, University of Texas at Austin.

1. INTRODUCTION

There has been an explosion in the number of online social networks and their active members. This wealth of information in the social networks has driven prolific work on the analysis of the networks, understanding the processes that explain the evolution of the networks, modeling the spread of behavior through the networks, predicting their future state and so on.

Users of a social network tend to affiliate with communities. In some social networks, the groups are identified more by the preferences of the members of the social network than by direct declaration: e.g. the genre of movies that a set of Netflix customers tend to patronize more often. Online social networks like Facebook, Orkut and LiveJournal are more interesting examples because the affiliation networks here are explicitly established by the members of the network. Thus, two networks exist simultaneously: the friendship network among users, and the affiliation network between users and groups.

The problem. Group formation and evolution in social networks [Backstrom et al. 2006], and co-evolution of social and affiliation networks [Zheleva et al. 2009] have been recently studied. One of the interesting challenges in social network analysis is the affiliation recommendation problem, where the task is to recommend communities to users. The fact that the social and affiliation networks "co-evolve" suggests that a better solution to the affiliation recommendation problem can be obtained if the friendship network is considered along with the affiliation network. This problem setting has applications beyond community recommendation. Affiliations, for example, can be interpreted in general as a user's taste for an item. Neither is it limited to social networks. For example, in biology, the friendship network can correspond to a network among genes, whereas the affiliation network can correspond to a network between genes and traits/diseases, and the affiliation recommendation problem can be viewed as one of identifying genes affecting the expression of a disease.

Contributions. We consider how one can model the interplay between users and communities in both networks simultaneously. An ideal unifying model would not only explain the current state of the networks, but also help in predicting future relationships among the nodes. Using a simple way of combining these networks, we suggest and explore two ways of modeling the networks for the purpose of making affiliation recommendations. The *graph proximity model* is based on estimating the affinity between a user and a community by considering their proximity as nodes in a combined graph, while the *latent factors model* is based on the proposition that community affiliations arise from interactions of *user factors* and *group factors*. Each of these network models suggests affiliation recommendation algorithms. In order to make large-scale recommendations using algorithms suggested by the graph proximity model, we present two approaches *common subspace approximation* and *clustered low rank approximation*.

We evaluate a set of algorithms on social networks from Orkut consisting of 9,123 users and 75,546 communities, and Youtube consisting of 16,575 users and 21,326 communities. We propose a way of evaluating affiliation recommendations, by measuring how good the top 50 recommendations per user are, and demonstrate the

importance of designing the right evaluation strategy. Of the algorithms proposed, those suggested by the graph proximity model turn out to be the most effective and efficient. This use of link prediction techniques for the purpose of affiliation recommendation is, to our knowledge, novel. We show that information in the friendship network can be used effectively for affiliation recommendation. We also observe that the benefit we derive from the social network in affiliation recommendation is strongly contingent on how the problem is modeled and what algorithms are used.

Overview. We now provide a brief overview of the organization of the paper. In Section 2, we consider a network formed by merging the friendship and affiliation networks and introduce two models of the behaviour of nodes in this network: the *graph proximity model*, and the *latent factors model*, and explore recommendation algorithms that arise from these models. Section 3 details and two scalable models for large scale group recommendation. In Section 4, we consider how the proposed models and algorithms relate to prior work. In Section 5 we describe our evaluation strategy and present experimental results to evaluate the various algorithms for affiliation recommendation using real world networks. Finally, in Section 6, we conclude with a summary of our findings and give a brief discussion on future lines of research.

2. MODELS

In this section, we first establish the notation used, and then pose the affiliation recommendation problem as a *ranking* problem. Then we describe a natural way of combining the friendship and affiliation networks into a single graph. In the subsections that follow, we describe affiliation recommendation approaches based on the *graph proximity model* and those based on the *latent factors model*.

Notation. We use N_u to denote the total number of users in the affiliation network and N_g to denote the total number of communities in the affiliation network. $\mathbf{A} \in \mathbb{R}^{N_u \times N_g}$ denotes the user $\times$ group adjacency matrix of affiliation network $\mathcal{A}$. $\mathbf{S} \in \mathbb{R}^{N_u \times N_u}$ denotes the user $\times$ user adjacency matrix of friendship network $\mathcal{S}$. We will use $\mathbf{A}_{i,j}$ and $[\cdot]_{i,j}$ to denote the i, j entry of the matrix $\mathbf{A}$ or the corresponding matrix expression within the brackets. Other notation will be introduced as needed.

Ranking problem. We are considering the task to recommend affiliations to a given user. This problem can be posed as a problem of ranking various affiliations in the order of the user's interest in joining them. The methods we describe here to solve this problem rely on assigning scores to various affiliations in order to rank them. The task of an affiliation recommendation algorithm can be viewed as one of generating an $N_u \times N_g$ score matrix.

2.1 Prediction on the Combined Graph

Consider the adjacency matrices $\mathbf{A}$ and $\mathbf{S}$. For now, we assume $\mathbf{S}$ to be symmetric (or equivalently $\mathcal{S}$ to be undirected), although a non-symmetric extension to our model can be easily obtained. Clearly, $\mathbf{S}$ corresponds to an undirected graph among users and $\begin{bmatrix} 0 & \mathbf{A} \\ \mathbf{A}^T & 0 \end{bmatrix}$ corresponds to an undirected bipartite graph between users and groups. Despite the heterogeneity of the two types of "links", it is rather natural to

consider a graph $\mathcal{C}$ between all the users and groups, with the combined adjacency matrix

$$\mathbf{C} = \begin{bmatrix} \lambda\mathbf{S} & \mathbf{A} \\ \mathbf{A}^T & 0 \end{bmatrix}, \tag{1}$$

where the heterogeneity of two types of links is reduced to a single parameter $\lambda \geq 0$, that controls the ratio of the weight of friendship to the weight of group membership. Clearly when $\lambda = 0$, the user-user friendship ceases to play a role in this joint graph, and it simply degenerates to the bipartite affiliation graph.

As in the case of prediction with a single graph, the prediction on $\mathcal{C}$ can be carried out from the following two perspectives.

(1) **Graph Proximity Model:** We assume that the entire graph is known, and that the prediction of new or unobserved links is based on an estimated proximity in $\mathbf{C}$ between nodes.
(2) **Latent Factors Model:** We model the graph as a matrix, some of whose zero entries are actually ones, while modelling all entries as the product of latent factors.

We will elaborate on the two perspectives in the following two subsections.

2.2 Graph Proximity Model

As described earlier, the affiliation network can be modelled by a graph. The graph proximity model assumes that the probability of there arising a link between two nodes in a graph is based on an estimated proximity between the two nodes. The proximity of two vertices can be calculated as the weighted sum of the number of paths connecting the two with varying lengths. This is the underlying mechanism of many link prediction models in the context of social network analysis. Consider the widely-used Katz measure [Katz 1953; Liben-Nowell and Kleinberg 2003] on the friendship network $\mathcal{S}$. The proximity is given by

$$\mathsf{Katz}(\mathbf{S};\beta) = \beta\mathbf{S} + \beta^2\mathbf{S}^2 + \beta^3\mathbf{S}^3 + \cdots = \sum_{i=1}^{\infty} \beta^i \mathbf{S}^i,$$

where the weights of paths decay exponentially with the length and β is a parameter to ensure convergence of the series.

A simple extension of Katz to the bipartite graph $\mathcal{A}$ is

$$\begin{aligned} \mathsf{extKatz}(\mathbf{A};\beta) &= (\beta\mathbf{A}\mathbf{A}^T + \beta^2(\mathbf{A}\mathbf{A}^T)^2 + \beta^3(\mathbf{A}\mathbf{A}^T)^3 + \cdots)\mathbf{A} \\ &= \mathsf{Katz}(\mathbf{A}\mathbf{A}^T;\beta)\mathbf{A}, \end{aligned} \tag{2}$$

where, in the co-occurrence matrix $\mathbf{A}\mathbf{A}^T$, two users i and j are considered connected if i and j are members in the same group, i.e. $[\mathbf{A}\mathbf{A}^T]_{i,j} > 0$. In this case, we consider the paths of the following types

$$\text{user } i \xrightarrow{\mathbf{A}\mathbf{A}^T} j \xrightarrow{\mathbf{A}\mathbf{A}^T} k \xrightarrow{\mathbf{A}} \text{group } n.$$

The intuition in considering $\mathbf{A}\mathbf{A}^T\mathbf{A}$ is if user i shares some community with user j, it is likely that i will join some other community j belongs to. The higher order terms can be interpreted in a similar way.

Consider the following Katz measure proximity matrix on the combined graph $\mathcal{C}$

$$\mathsf{Katz}(\mathbf{C};\beta) = \beta\mathbf{C} + \beta^2\mathbf{C}^2 + \beta^3\mathbf{C}^3 + \cdots = \sum_{i=1}^{\infty} \beta^i \mathbf{C}^i.$$

The user-community block[1] of $\mathsf{Katz}(\mathbf{C};\beta)$ is denoted $\mathsf{Katz}(\mathcal{C};\beta)_{12}$ and given by

$$\begin{aligned}\mathsf{Katz}(\mathcal{C};\beta)_{12} = \beta\mathbf{A} + \beta^2\lambda\mathbf{S}\mathbf{A} + \beta^3(\lambda^2\mathbf{S}^2\mathbf{A} + \mathbf{A}\mathbf{A}^T\mathbf{A}) + \\ \beta^4(\lambda^3\mathbf{S}^3\mathbf{A} + \lambda\mathbf{A}\mathbf{A}^T\mathbf{S}\mathbf{A} + \lambda\mathbf{S}\mathbf{A}\mathbf{A}^T\mathbf{A}) + \cdots . \end{aligned} \tag{3}$$

Equation (3) is another generalization of the Katz measure on the bipartite graph $\mathbf{A}$ by also considering paths involving the frienship graph $\mathcal{S}$, e.g.

$$\text{user } i \xrightarrow{\mathbf{S}} j \xrightarrow{\mathbf{A}} \text{group } n \text{ (in } \mathbf{C}^2)$$

and

$$\text{user } i \xrightarrow{\mathbf{S}} j \xrightarrow{\mathbf{A}\mathbf{A}^T} k \xrightarrow{\mathbf{A}} \text{group } n \text{ (in } \mathbf{C}^4).$$

Clearly we can use $\mathsf{Katz}(\mathcal{C};\beta)_{12}$ as a score matrix.

Computing the Katz measure is expensive as it either involves a summation of large number (infinite) of terms with matrix powers or the inverse of a potentially large matrix[2], where the resulting inverse is dense. We consider a truncated Katz measure, e.g. with the combined adjacency matrix $\mathbf{C}$ we have

$$\mathsf{tKatz}(\mathbf{C},\beta,k) = \sum_{i=1}^{k} \beta^i \mathbf{C}^i. \tag{4}$$

where the truncation parameter k is usually quite small. Also for the truncated Katz measure we will be interested in only the user-community block, which we will denote $\mathsf{tKatz}(\mathbf{C},\beta,k)_{12}$, e.g. with $k = 2$ we have

$$\mathsf{tKatz}(\mathbf{C},\beta,2)_{12} = \beta\mathbf{A} + \beta^2\lambda\mathbf{S}\mathbf{A}. \tag{5}$$

For computing the tKatz or extKatz measures for the graphs $\mathbf{C}$ or $\mathbf{A}$, respectively, a conservative estimate of the computational cost is $O(N_u \times \mathsf{nnz})$, where nnz is the number of non-zeros in $(\mathbf{A}\mathbf{A}^T)^k$.

2.3 Latent Factors Model

We will now describe the *latent factors model*. We will present the optimization problem it solves, and examine some of its properties. In this model, the zeros in the adjacency matrix of the affiliation network, $\mathbf{A}$ may be viewed as being unobserved entries with a huge prior belief in favor of them being actually zero. Every user i and community j are assumed to have a low dimensional representations $\mathbf{u}_i$, $\mathbf{g}_j$. The affinity of a user i to a community j is assumed to correspond to $\mathbf{u}_i^T\mathbf{g}_j$. In other words, users and communities with a high inner product are assumed to be connected to each other, i.e. $\mathbf{A}_{i,j} \approx \mathbf{u}_i^T\mathbf{g}_j$. In matrix form, we express this as

$$\mathbf{A} \approx \mathbf{U}^T\mathbf{G}, \tag{6}$$

[1]The user-community block is the 1-2 block matrix of $\mathsf{Katz}(\mathbf{C};\beta)$, that originates from the block partitioning of $\mathbf{C}$ in Equation (1).
[2]It can be shown that $\mathsf{Katz}(\mathbf{C};\beta) = (I - \beta\mathbf{C})^{-1} - I$.

where $\mathbf{U} \in \mathbb{R}^{d \times N_u}$ is the matrix of user factors and $\mathbf{G} \in \mathbb{R}^{d \times N_g}$ is the matrix of group factors. Note that $\text{rank}(\mathbf{U}) \leq d,\ \text{rank}(\mathbf{G}) \leq d,$ and $d \ll N_u, N_g$.

To get factors which account for both the observed entries in $\mathbf{A}$ as well as the interactions in $\mathbf{S}$ we consider the following combined network,

$$\mathbf{C}'(\lambda, \mathbf{D}) = \begin{bmatrix} \lambda\mathbf{S} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{D} \end{bmatrix}, \tag{7}$$

where $\mathbf{D}$ is a *derived* similarity between groups which is *not* observed. Note that $\mathbf{C} = \mathbf{C}'(\lambda, \mathbf{0})$. Let $\mathbf{C}' \approx \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ be the rank-d dominant spectral approximation, where the d columns of $\mathbf{V}$ are eigenvectors and $\mathbf{\Lambda}$ is a $d \times d$ diagonal matrix with the largest in magnitude eigenvalues. Partitioning the spectral approximation according to the block structure of $\mathbf{C}'$ we obatain

$$\mathbf{C}'(\lambda, \mathbf{D}) = \begin{bmatrix} \lambda\mathbf{S} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{D} \end{bmatrix} \approx \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} \mathbf{\Lambda} \begin{bmatrix} \mathbf{V}_1^T & \mathbf{V}_2^T \end{bmatrix} = \begin{bmatrix} \mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_1^T & \mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_2^T \\ \mathbf{V}_2\mathbf{\Lambda}\mathbf{V}_1^T & \mathbf{V}_2\mathbf{\Lambda}\mathbf{V}_2^T \end{bmatrix}.$$

Clearly, we have $\mathbf{A} \approx \mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_2^T$ and we can set $\mathbf{U}^T = \mathbf{V}_1\mathbf{\Lambda}$ and $\mathbf{G} = \mathbf{V}_2^T$ in model of Equation (6). The user factors in $\mathbf{U}$ and group factors in $\mathbf{G}$ contain information from the friendship network $\mathcal{S}$ as the computation of $\mathbf{V}_1$, $\mathbf{V}_2$ and $\mathbf{\Lambda}$ are influenced by the presense of $\mathbf{S}$ in $\mathbf{C}'$. We also note that the user and group factors are not unique. It is straightforward to show that the factors $\mathbf{V}_1$, $\mathbf{\Lambda}$ and $\mathbf{V}_2$ obtained by the spectral approximation of $\mathbf{C}'$ solve the following minimization problem

$$\underset{\mathbf{V}_1, \Lambda, \mathbf{V}_2}{\arg\min} \left(\|\mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_1^T - \lambda\mathbf{S}\|_F^2 + 2\|\mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_2^T - \mathbf{A}\|_F^2 + \|\mathbf{V}_2\mathbf{\Lambda}\mathbf{V}_2^T - \mathbf{D}\|_F^2 \right), \tag{8}$$

with the constraints $\text{rank}(\mathbf{V}_1) = \text{rank}(\mathbf{V}_2) = \text{rank}(\mathbf{\Lambda}) \leq d$. We will interpret the low rank approximation of $\mathbf{A}$

$$\mathbf{A} \approx \mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_2^T \tag{9}$$

as a score matrix.

Role of λ. Intuitively, in equation (8), λ controls the contribution of $\mathbf{S}$ in deciding the user factors. The larger the λ, the learned factor model describes the friendship network $\mathcal{S}$ better, and correspondingly the affiliation network is described less well.

Choice of D. The derived similarity between the communities $\mathbf{D}$ in the combined matrix $\mathbf{C}'$ can be approximated using proximity between communities in the graph corresponding to $\mathbf{C}'$. It follows that a potential choice is $\mathbf{D} = \mathbf{A}^T\mathbf{A}$, which is simply the number of users which any two communities share. One may also consider weighing the contribution of $\mathbf{D}$ with λ, which is the weight factor learnt for $\mathbf{S}$. We will see later that the experiments suggest that the choice of $\mathbf{D}$ is not very important, and that the information from $\mathbf{D}$, when derived from $\mathbf{A}$, is redundant.

3. SCALING TO LARGE DATA SETS

We have proposed two different approaches to group recommendation in the previous section. In this section we will consider the computational feasibility of our approaches when dealing with massive real world data sets. The latent factors model (Section 2.3) is in fact quite scalable due to the existence of highly efficient algorithms, e.g. ARPACK [Lehoucq et al. 1998] and PROPACK [Larsen 1998],

for computing the dominant eigenvectors and corresponding eigenvalues. Unfortunately, the graph proximity model (Section 2.2) is computationally formidable for large scale friendship networks $\mathcal{S}$ and affiliation networks $\mathcal{A}$. This difficulty lies in the computation of the power of $\mathbf{A}$ and $\mathbf{S}$ in $\mathsf{tKatz}(C;\beta,k)_{12}$. For example when $k=3$,

$$\mathsf{tKatz}(C;\beta,3)_{12} = \beta\mathbf{A} + \beta^2\lambda\mathbf{S}\mathbf{A} + \beta^3(\lambda^2\mathbf{S}^2\mathbf{A} + \mathbf{A}\mathbf{A}^T\mathbf{A}),$$

where both $\mathbf{A}\mathbf{A}^T$ and $\mathbf{S}^2$ could get fairly dense and hence the matrix multiplications $\mathbf{S}^2\mathbf{A}$ and $\mathbf{A}\mathbf{A}^T\mathbf{A}$ become expensive and the results even denser.

3.1 Common Subspace Approximation

The cost of matrix multiplication can be reduced if we first take low rank approximation of both $\mathbf{S}$ and $\mathbf{A}$. That is

$$\mathbf{S} \approx \mathbf{U}_\mathbf{S}\mathbf{\Lambda}_\mathbf{S}\mathbf{U}_\mathbf{S}^T, \qquad \mathbf{A} \approx \mathbf{U}_\mathbf{A}\mathbf{\Sigma}_\mathbf{A}\mathbf{V}_\mathbf{A}^T, \tag{10}$$

where $\mathbf{S}$ is approximated using the spectral factorization and $\mathbf{A}$ is approximated via the truncated singular value decomposition (SVD) [Golub and Van Loan 1996]. However, this simple approximation does not work well, since with very large $\mathbf{A}$ and $\mathbf{S}$, and small rank d in the approximations, $\mathbf{U}_\mathbf{A}$ and $\mathbf{U}_\mathbf{S}$ may be almost orthogonal to each other, i.e. entries of $\mathbf{U}_\mathbf{S}^T\mathbf{U}_\mathbf{A}$ are very small. Therefore the approximation

$$\mathbf{S}\mathbf{A} \approx \mathbf{U}_\mathbf{S}\mathbf{\Lambda}_\mathbf{S}(\mathbf{U}_\mathbf{S}^T\mathbf{U}_\mathbf{A})\mathbf{\Sigma}_\mathbf{A}\mathbf{V}_\mathbf{A}^T$$

may have almost all of its entries close to zero. One way to remedy this problem is to extend these low rank approximations so that we have a common subspace, that captures the dominant part of both $\mathbf{S}$ and $\mathbf{A}$. We found that the following heuristic works fairly well. We first form $[\mathbf{U}_\mathbf{S}\ \mathbf{U}_\mathbf{A}]$ and compute its QR factorization [Golub and Van Loan 1996], i.e. $\mathbf{Q}\mathbf{R} = [\mathbf{U}_\mathbf{S}\ \mathbf{U}_\mathbf{A}]$. Then the optimal (in least squares sense) rank-$2d$ approximations of $\mathbf{S}$ and $\mathbf{A}$ in terms of $\mathbf{Q}$ are given by

$$\mathbf{S} \approx \mathbf{Q}(\mathbf{Q}^T\mathbf{S}\mathbf{Q})\mathbf{Q}^T \equiv \mathbf{Q}\mathbf{D}_\mathbf{S}\mathbf{Q}^T, \qquad \mathbf{A} \approx \mathbf{Q}(\mathbf{Q}^T\mathbf{A}\mathbf{V})\mathbf{V}^T \equiv \mathbf{Q}\mathbf{D}_\mathbf{A}\mathbf{V}^T, \tag{11}$$

where $\mathbf{V}$ is an orthogonal matrix that spans the column space of $\mathbf{A}^T\mathbf{Q}$, e.g. the Q-part in the QR factorization of $\mathbf{A}^T\mathbf{Q}$. We also introduced the $2d \times 2d$ matrices $\mathbf{D}_\mathbf{S} = \mathbf{Q}^T\mathbf{S}\mathbf{Q}$ and $\mathbf{D}_\mathbf{A} = \mathbf{Q}^T\mathbf{A}\mathbf{V}$. Using the low rank approximations in (11) the user-community block of the tKatz measure becomes

$$\mathsf{tKatz}(C;\beta,3)_{12} \approx \beta\mathbf{Q}\mathbf{D}_\mathbf{S}\mathbf{Q}^T + \mathbf{Q}\left(\beta^2\lambda\mathbf{D}_\mathbf{S}\mathbf{D}_\mathbf{A} + \beta^3(\lambda^2\mathbf{D}_\mathbf{S}^2\mathbf{D}_\mathbf{A} + \mathbf{D}_\mathbf{A}\mathbf{D}_\mathbf{A}^T\mathbf{D}_\mathbf{A})\right)\mathbf{V}^T. \tag{12}$$

The intuition behind this heuristic is quite simple: we force the low-rank approximations of $\mathbf{A}$ and $\mathbf{S}$ in (11) to take the same subspace for users. This constraint certainly lowers the quality of individual approximation task, but better exploits the interaction between $\mathbf{S}$ and $\mathbf{A}$ when predicting future affiliations.

3.2 Clustered Low Rank Approximation

An alternative approach related to the latent factor model is to compute a different (than the spectral) low rank approximation of the combined adjacency matrix $\mathbf{C}$. It is observed that many real world social networks naturally form a number of clusters (communities), where the links within the clusters are much denser than

the ones between clusters. We will generally referred to graphs with clear cluster structure as *clusterable*. In the *clustered low rank approximation*, [Savas and Dhillon 2010; Song et al. 2010] the cluster structure of the network is utilized in the low rank approximation. We will illustrate the procedure through a 2-cluster example. Assume that the nodes $\mathcal{V} = \{1, 2, \ldots, |\mathcal{V}|\}$ are clustered in two disjoint clusters, i.e. $\mathcal{V}_1 \cup \mathcal{V}_2 = \mathcal{V}$ and $\mathcal{V}_1 \cap \mathcal{V}_2 = \emptyset$. Then by permuting rows and columns of $\mathbf{C}$ according to the cluster memberships of the nodes we obtain

$$\mathbf{P}\mathbf{C}\mathbf{P}^T = \bar{\mathbf{C}} = \begin{bmatrix} \bar{\mathbf{C}}_{11} & \bar{\mathbf{C}}_{12} \\ \bar{\mathbf{C}}_{21} & \bar{\mathbf{C}}_{22} \end{bmatrix},$$

where $\mathbf{P}$ is a permutation matrix that reorders rows of $\mathbf{C}$ so that the first $|\mathcal{V}_1|$ rows correspond to nodes from the first cluster, and the remaining $|\mathcal{V}_2|$ rows correspond to nodes from the second cluster. The links between nodes from cluster one will then form the non-zeros of $\bar{\mathbf{C}}_{11}$ and the links between nodes from cluster two form the non-zeros of $\bar{\mathbf{C}}_{22}$. Then the non-zeros in $\bar{\mathbf{C}}_{12}$ and $\bar{\mathbf{C}}_{21}$ correspond to links between the two clusters. Assuming the graph is clusterable, the links/non-zeros of $\bar{\mathbf{C}}$ will be concentrated in the diagonal blocks $\bar{\mathbf{C}}_{11}$ and $\bar{\mathbf{C}}_{22}$. The amount of links/non-zeros in the off-diagonal blocks depend on how well the graph clusters, and usually this part is only a small fraction of all links. In a particular example, with five clusters on the Youtube data set, only about 10% of the links are between vertices from different clusters. After a clustering of the nodes is obtained, by e.g. using highly efficient multilevel algorithms [Dhillon et al. 2007; Karypis and Kumar 1998], a low rank approximation is computed of each diagonal block matrix of $\bar{\mathbf{C}}$. The cluster-wise low rank approximations are computed independently and may be obtained as the dominant spectral approximation, e.g. with two clusters we have

$$\bar{\mathbf{C}}_{11} \approx \bar{\mathbf{V}}_1 \bar{\mathbf{\Lambda}}_1 \bar{\mathbf{V}}_1^T \qquad \bar{\mathbf{C}}_{22} \approx \bar{\mathbf{V}}_2 \bar{\mathbf{\Lambda}}_2 \bar{\mathbf{V}}_2^T.$$

The two cluster-wise approximations are then used to obtain an approximation for the entire $\bar{\mathbf{C}}$,

$$\bar{\mathbf{C}} = \begin{bmatrix} \bar{\mathbf{C}}_{11} & \bar{\mathbf{C}}_{12} \\ \bar{\mathbf{C}}_{21} & \bar{\mathbf{C}}_{22} \end{bmatrix} \approx \begin{bmatrix} \bar{\mathbf{V}}_1 & 0 \\ 0 & \bar{\mathbf{V}}_2 \end{bmatrix} \begin{bmatrix} \bar{\mathbf{D}}_{11} & \bar{\mathbf{D}}_{12} \\ \bar{\mathbf{D}}_{21} & \bar{\mathbf{D}}_{22} \end{bmatrix} \begin{bmatrix} \bar{\mathbf{V}}_1 & 0 \\ 0 & \bar{\mathbf{V}}_2 \end{bmatrix}^T, \tag{13}$$

where $\bar{\mathbf{D}}_{ij} = \bar{\mathbf{V}}_i^T \bar{\mathbf{C}}_{ij} \bar{\mathbf{V}}_j$ and clearly if $i = j$ we have $\bar{\mathbf{D}}_{ii} = \bar{\mathbf{\Lambda}}_i$. If the rank of the approximation of each $\bar{\mathbf{C}}_{ii}$ is d we see that Equation (13) yields a rank-$2d$ approximation of $\bar{\mathbf{C}}$. An important observation is that the memory usage for the rank-$2d$ clustered low rank approximation is almost the same as the memory usage for a regular rank-d approximation. Recall that $\bar{\mathbf{V}}_i$ are "long-skinny", i.e. $d \ll N_u$, thus most of the memory in the low rank approximation is used by the eigenvectors. Although the cluster-wise low rank approximations do not involve off-diagonal blocks, e.g. $\bar{\mathbf{C}}_{12}$, information on these blocks in the approximation of $\bar{\mathbf{C}}$ is included due to the inclusion of $\bar{\mathbf{D}}_{12}$. Experiments show that with the same memory useage clustered low rank approximation is often much more accurate than the regular low rank approximation. In addition, the total computational time[3] for

[3]For timing analysis, the clustered low rank approximation includes a clustering step, a step with approximations of each diagonal block, and finally computing the off-diagonal blocks $\bar{\mathbf{D}}_{ij}$ with $i \neq j$. Timings are compared for cases where the number of parameters for the clustered and

the clusterd low rank approxmiationis is usually lower than the time for computing the truncated SVD for the entire matrix. Interested readers are referred to [Savas and Dhillon 2010] and [Song et al. 2010].

In summary, the clustered low rank approximation cleverly spends most of its resource (parameters) on modeling the links within clusters, and is therefore able to capture more information than regular SVD using the same number of parameters. The model in (13) may then be used both in the graph proximity model to compute various Katz measures and in the latent factors model.

4. RELATED WORK

Increasing attention to recommendation systems in general can be attributed primarily to commercial enterprises like Netflix and Amazon, where making good recommendations for the customers is important for business. A huge body of literature studies the problem of group recommendation, where the problem is to recommend items or products to a group of users in a friendship network. Affiliation recommendation for users of a friendship network, however, is relatively new and less studied.

4.1 Joint matrix factorization models

We now examine the relationship of the latent factors model proposed in this paper to a variety of other models proposed recently. Bader and Chew [Bader and Chew 2008], in the context of information retrieval, tackle the problem of applying LSA to multi-lingual corpora. In such corpora, one has access to term-similarity information along with term-document matrices corresponding to various languages. In order to derive low dimensional term and document factors which account for information from both these sources, they form a joint matrix similar to our combined adjacency matrix C and compute its SVD. However, unlike this work, [Bader and Chew 2008] does not deal with the item recommendation problem, and it does not view this joint matrix as arising out of a pair of networks.

We will now consider two other joint matrix factorization models: One class of models uses a probabilistic collaborative filtering to approach the problem, whereas another tackles the problem of combining information from multiple sources from the perspective of joint matrix factorization.

4.1.1 *Probabilistic Collaborative Filtering Models.* Collaborative filtering is a natural way to approach the affiliation recommendation problem. Typically collaborative filtering is applied to user-item preference problems. This is based on the simple idea that users with similar tastes behave similarly.

This approach has recently been applied to the affiliation recommendation problem by Chen et al. [Chen et al. 2009]. The authors examined the use of Latent Dirichlet Allocation (LDA) in affiliation recommendation. The LDA approach does not use information from the friendship network among users. So, here we briefly examine the relationship between the latent factors model we propose and this LDA based approach, while ignoring the friendship network aspects of our model.

Consider the objective (8) we are trying to minimize. In the proposed model, if

regular low rank approximations is the same.

we ignore the constraint that the user factors $\mathbf{U}$ do not result in too large a deviation from $\mathbf{S}$, we are essentially trying to find a low rank approximation to $\mathbf{A}$ in terms of the Frobenius norm. The solution which minimizes that objective is given by the SVD of $\mathbf{A}$. This is the Latent Semantic Analysis approach (LSA), which has long been exploited for similar problems in the area of information retrieval. pLSA, or probabilistic LSA [Hofmann 1999], instead proposes a statistical model for the process generating $\mathbf{A}$ and then learns the model parameters which are most likely to have generated the observations in $\mathbf{A}$. These parameters can then be used in finding a low rank approximation to $\mathbf{A}$, in terms of the KL divergence. It can thus be viewed as the probabilistic version of LSA. LDA, where Dirichlet priors are added to pLSA's generative model, can be viewed as the Bayesian version of pLSA. Thus, the LDA based approach to the affiliation recommendation problem may be viewed as trying to find a low rank approximation to $\mathbf{A}$, albeit from a probabilistic, Bayesian perspective, while ignoring information from $\mathbf{S}$.

Combinatorial collaborative filtering [Chen et al. 2008] is another work in the same vein. Unlike the LDA based approach, however, the probabilistic model of user-community relationships used in this work utilizes information not only from $\mathbf{A}$, but also from text descriptions of various communities. Next, we examine a couple of closely related matrix factorization models.

4.1.2 *Linked Matrix Factorization Models.* Tang et al have proposed Linked Matrix Factorization (LMF) [Tang et al. 2009] as a way of combining information from multiple graphs on the same set of entities, in order to make more accurate inferences. However, they tackle a different problem, namely, clustering. The link between their network model and ours is established by the objective functions that we optimize. The LMF model tries to simultaneously find a low rank approximation for the adjacency matrix of each network, using matrix factorization. Each such matrix factorization has a source-specific factor matrix, $\mathbf{\Lambda}^{(m)}$, and a factor matrix, $\mathbf{V}$, that is shared by all the sources. The objective function of LMF is effectively to minimize the quantity,

$$\sum_{i=1}^{M} \|\mathbf{A}^{(m)} - \mathbf{V}\mathbf{\Lambda}^{(m)}\mathbf{V}^T\|_F^2$$

Comparing this to (8), we see that $\mathbf{U}$, which represents the user factors, is shared by the two sources of information, i.e., $\mathbf{A}$ and $\mathbf{S}$. However, an important distinction is that we have two graphs which share only one set of entities in common, whereas in LMF, each source of information is a network on exactly the same set of users.

Singh and Gordon have proposed a model for relational learning called Collective Matrix Factorization [Singh and Gordon 2008]. They suggest a generalized framework for inferring relations, given a set of entities and observed relations among them. This model factors multiple source matrices simultaneously, and uses common factors for approximation whenever the same entity participates in multiple relations. It allows different loss functions for each matrix approximation, and combines the information from multiple relations using weights which reflect the relative importance for each relation. This essentially generalizes the idea of Linked Matrix Factorization. The Latent Factors Model proposed in this paper uses the parameter λ to determine the contribution of the friendship network $\mathbf{S}$ in generating

the user factors.

The above mentioned papers use optimization techniques based on the alternating least squares approach, we use SVD which efficiently solves the optimization problem posed by the latent factors model.

4.2 Co-evolution of social and affiliation networks

Researchers have studied the effects of friendship ties on affiliations in other contexts, like the growth and evolution of social networks [Backstrom et al. 2006], and spread of influence through a social network [Kempe et al. 2003; Chen et al. 2009]. They tend to model the dependence of a user joining a group on the number of friends the user already has in the group. Zheleva et al. [Zheleva et al. 2009] proposed a unified model for the generation of social and affiliation networks, and observed that the social network is one of the factors that influences the evolution of affiliation network. Our idea that friendship network combined with the affiliation network can be exploited in making affiliation recommendations is inspired by this line of research.

5. EXPERIMENTAL EVALUATION

We first introduce the data sets on which we conduct our experiments, and describe the experimental setup. We then describe our methodology for comparing the performance of various algorithms for the affiliation recommendation task. Finally, we present a series of experiments using methods presented in previous sections, and discuss the results.

5.1 Data

We use two popular online social networks for our experiments. These are *Orkut* and *Youtube*, and both are operated by Google. The users of both social networks explicitly identify themselves as belonging to some *communities* or *groups*. Thus, for each of the networks we have adjacency matrix $\mathbf{A}$ that identifies the memberships of the users in the groups and adjacency matrix $\mathbf{S}$ that identifies friendships among users. For our experiments, we used data gathered by [Mislove et al. 2007]. We compare the predictive ability of the algorithms using these large networks. In a few of the experiments we used the largest connected component, denoted with Okrut-1c and Youtube-1c, as path based measures, e.g. Katz, are decoupled and indepedent for different components. Some statistics for these networks are presented in Table I.

5.2 Experiment setup

For every user u and a group g, let $\mathbf{E}_u = \{(u, g) \mid \mathbf{A}_{u,g} = 1\}$ denote the affiliations of u, as observed in a given affiliation network $\mathbf{A}$. Invariably, in all the experiments, we set aside a subset of these affiliations $\mathbf{E}_u^{(t)} \subset \mathbf{E}_u$ as test data. We use $|\mathbf{E}_u^{(t)}| = 30\%|\mathbf{E}_u|$. The remaining affiliations $\mathbf{E}_u^{(tr)} = \mathbf{E}_u \setminus \mathbf{E}_u^{(t)}$ are used as training data for the recommendation algorithms.

All of our recommendation algorithms require "learning" parameters for some model of the affiliation process, and hence for the purposes of learning the parameters, we use a set of *validation* links $\mathbf{E}_u^{(v)} \subset \mathbf{E}_u^{(tr)}$ with $|\mathbf{E}_u^{(v)}| = 30\%|\mathbf{E}_u^{(tr)}|$. During

Feature	Orkut	Youtube	Orkut-1c	Youtube-1c
Number of users, N_u	9123	16575	9123	16535
Number of groups, N_g	75546	21326	20905	17672
Average number of groups per user	55.8	10.5	25	10.2
Minimum number of groups per user	4	4	0	2
Average number of users per group	6.7	8.1	10.9	9.6
Minimum number of users per group	1	1	2	1
Average number of friends per user	46	11.7	46	11.7

Table I. Some statistical properties of the data sets used in our experiments. The Orkut-1c and Youtube-1c are the largest connected components from the Orkut and Youtube networks, respectively.

the validation process, we compare different model parameters based on the number of correct edges among $25N_u$ recommendations[4] made using a model.

5.2.1 *Evaluation method.* We now describe our methodology for evaluating the performance of an affiliation recommendation algorithm. We first introduce notions of interest, such as *precision*, *recall*, *sensitivity*, *specificity*, *ROC* (receiver operating characteristic) and *AUC* (area under curve). We then describe the way in which we evaluate the performance of a recommendation algorithm based on its top 50 recommendations to the average user. We then demonstrate the importance of choosing the right evaluation method for the community recommendation task by showing that using a different, but less appropriate, evaluation strategy yields different results.

5.2.2 *Performance Measures.* Three commonly used measures of quality of solutions in information retrieval and classification tasks are *precision*, *recall* or *sensitivity* and *specificity*. Precision measures the exactness or fidelity of the prediction while sensitivity measures the completeness of the prediction. Suppose that the recommendation algorithm makes n recommendations to a user. Then, precision is defined as the ratio of the number of correctly identified positives (true positives) to n, and sensitivity is the ratio of the number of correctly identified positives to the total number of positives, i.e. $|\mathbf{E}_u^{(t)}|$. Specificity, on the other hand, measures the ability of the recommender to exclude uninteresting affiliations from the recommendations it makes. It is defined as the fraction of such "negative affiliations" correctly excluded from the recommendation. All three of these performance measures range from zero to one.

5.2.3 *ROC—Receiver Operating Characteristic.* Often, one is interested in evaluating the performance of a recommendation algorithm not for a single value of the number of recommendations n, but for the entire range of n. For a given recommendation algorithm and a user, sensitivity is a non-decreasing function of n. The relationship between the increase in sensitivity, as n increases, with the decrease in specificity is of interest in comparing the quality of recommendations. For a good recommendation, as n increases, sensitivity increases without a big drop in specificity. The ROC curve is a plot of the sensitivity vs $(1 - \text{specificity})$ for all

[4] $25N_u$ is chosen because, in Section 5.2.1, we argue that a predictive model should be judged based on the quality of its top few recommendations.

values of n. It is a common way of comparing the performance of classification algorithms over the entire range of n (or equivalently cutoff scores). The AUC curve (area under the ROC curve), is then used as a way to compare different classification algorithms: the greater the AUC, the better the algorithm's sensitivity vs $(1 - \text{specificity})$ tradeoff.

Consider a social network website, like Orkut or Facebook; or a vendor like Netflix which sells movies. They would be interested in making, let us say, five pages of recommendations to their users, but not much more than that. Certainly not a hundred. Also, irrespective of whether a user participates in five communities or seventy, the social networking website would probably want to make roughly the same number of recommendations per user. So, we choose to evaluate the recommendation algorithms we propose based on their top 50 recommendations. We do this by examining the portion of the ROC curve obtained by measuring the sensitivity and specificity the recommendation algorithm achieves for an average user at regular intervals between $n = 1$ and $n = 50$. To do this, for a given n between 1 and 50, we compute the sensitivity and specificity for every user in the network, and take the mean of these values to be the average sensitivity and average specificity for that n.[5] We then plot the average sensitivity vs $(1 - \text{average specificity})$ curve, as in Figure 2. Note that comparisons made using this method are statistically robust, as the sensitivities and specificities of recommendation algorithms are averaged over, e.g., 9500 users in Orkut and 16000 users in Youtube.

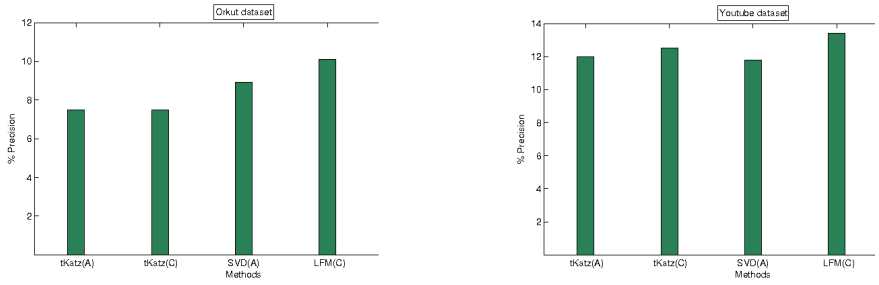
5.2.4 *"Global" vs "per-user" sensitivity.* Let $k(n_u)$ be the number of "good recommendations" made by a recommendation algorithm for a user u, when it makes n_u recommendations to that user. Then, "per user" sensitivity measure is defined as $N_u^{-1} \sum_u (k(n_u)/|\mathbf{E}_u^{(t)}|)$. In our experiments, we will use $n_u = 50\ \forall u$. Contrast this with finding the "global" sensitivity $k'(n)/(\sum_u |\mathbf{E}_u^{(t)}|)$, where $k'(n)$ denotes the number of "good recommendations" made by a given recommendation algorithm while making n predictions in total. For a fixed n, this "global" sensitivity is proportional to precision, and is a commonly used measure of performance of link prediction algorithms in the context of social network analysis. Note that, in this case, while $n = \sum_u n_u$, there is no guarantee that, for two given users u and v, $n_u = n_v$; indeed the recommendation algorithm, when asked to make n recommendations, may not make any recommendations at all for some users. Therefore, this measure of goodness of a recommendation algorithm is not equivalent to the "per user" sensitivity.

Judging identical algorithms on identical data sets, using these alternative evaluation methods can yield very different rankings of recommendation algorithms, as illustrated by comparing Figure 1 with Figure 2. Hence, the choice of an appropriate method for evaluating affiliation recommendations is an important one.

5.3 Results and Discussion

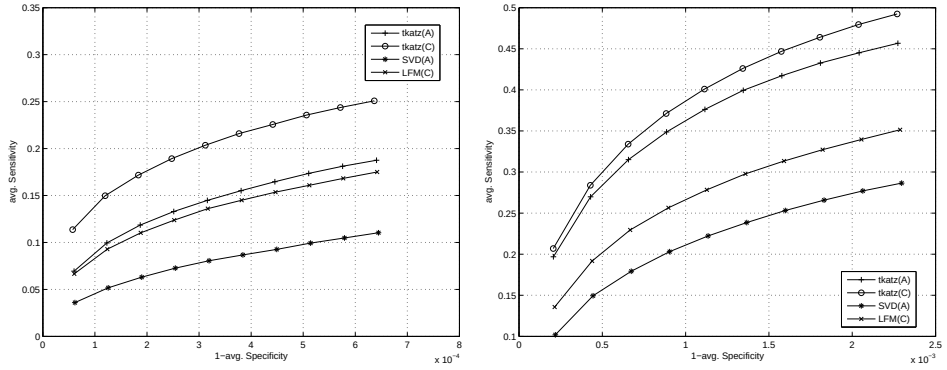
In this section, we report and analyse the performance of the various recommendation algorithms, based on the graph proximity model and latent factors model

[5]So, both average sensitivity and average specificity may be viewed as functions of the number of recommendations n.



(a) Orkut data set (b) Youtube data set

Fig. 1. In this figure, various recommendation algorithms are compared using the "global" sensitivity measure described in Section 5.2.4, where a total of $\sum_u E_u$ recommendations are made, with no guarantees about the number of recommendations made for any given user. According to this evaluation method, LFM(**C**) appears to be superior to tKatz(**C**). However, these results are different from those described in Figure 2, where recommendation algorithms are compared based on the goodness of the top 50 recommendations made for the average user.



(a) Orkut data set (b) Youtube data set

Fig. 2. Comparison of recommendation algorithms based on graph proximity and latent factors models, as described in Section 5.2.1. The leading portion of the ROC curve is shown, thus recommendation algorithms are compared based on the goodness of their top 50 recommendations. The graph proximity based predictors consistently outperform latent factors based predictors in the two data sets. See Section 5.3 for further discussion.

discussed in Sections 2 and 3. We compare the performance of the graph proximity methods with the latent factors methods on the average sensitivity and average specificity metrics introduced earlier, for a given number of recommendations in $\{5, 10, \cdots, 45, 50\}$. We study the performance of the scalable approximation methods proposed in Section 3, and compare these methods amongst themselves and with the methods proposed in Section 2. The list of experiments based on latent factor and graph proximity predictors are presented in Table II.

5.3.1 *Performance of recommenders based on the two models.* Consider the performance of the recommendation algorithms on the Orkut data set in Figure 2(a).

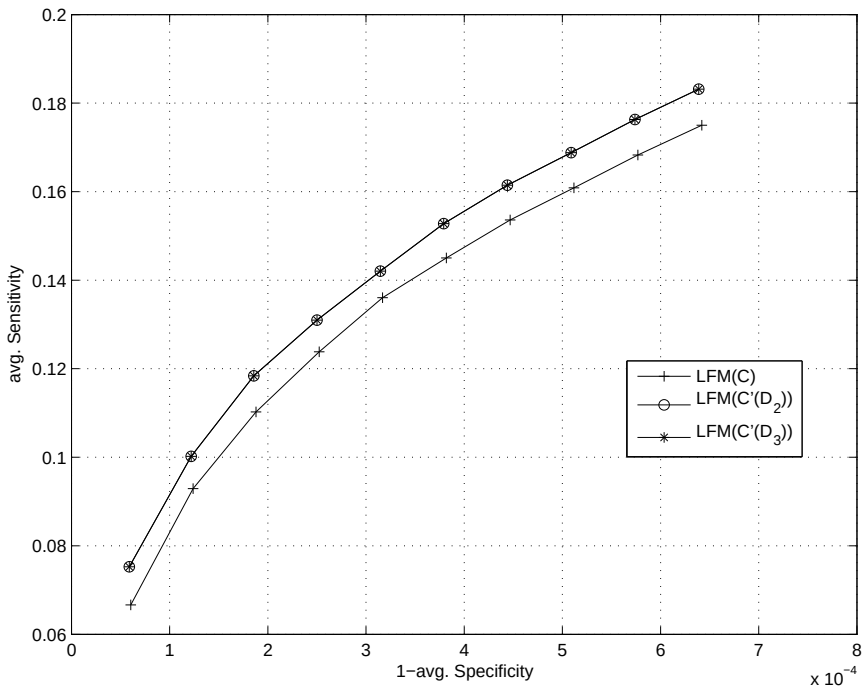


Fig. 3. Comparison of latent factors based algorithms for various choices of $\mathbf{D}$, for the Orkut data set: $\mathbf{D}_2 = \mathbf{A}^T\mathbf{A}, \mathbf{D}_3 = \lambda\mathbf{A}^T\mathbf{A}$.

Experiment	Score Matrix
tKatz($\mathbf{A}$)	tKatz($\mathbf{A}\mathbf{A}^T, \beta, k)\mathbf{A}$
tKatz($\mathbf{C}$)	Equation (4)
SVD($\mathbf{A}$)	SVD($\mathbf{A}$)
LFM($\mathbf{C}$)	Equation (9)
LFM-c($\mathbf{C}$, c)	Equation (13) extended to c clusters
tKatzCS(d)	Equation (12)
tKatzLFM($\mathbf{C}$, d)	tKatz($\mathbf{C}_d$), where $\mathbf{C}_d$ is rank-d approximation of $\mathbf{C}$
tKatzLFM-c($\mathbf{C}$, c, d))	tKatz($\mathbf{C}_d$), where $\mathbf{C}_d$ is the clustered rank-d approximation of $\mathbf{C}$

Table II. List of experiments, with the score matrix used for ranking the user-group connections, based on latent factor and graph proximity models.

SVD($\mathbf{A}$) gives the lowest performance of all the methods. LFM($\mathbf{C}$) performs better than SVD($\mathbf{A}$), which is expected given that it uses information from the friendship network $\mathbf{S}$ in addition to the information from affiliation network $\mathbf{A}$. For the average user, the graph proximity model based methods significantly outperform latent factors based methods as observed in both plots of Figure 2. In particular, we see that tKatz($\mathbf{C}$) performs much better than tKatz($\mathbf{A}$), which in turn outperforms latent factor based methods. We see that the information in the friendship network indeed proves highly beneficial in making affiliation recommendations and graph proximity based methods exploit this information the most. A summary of perfor-

Algorithm	Orkut	Youtube
LFM($\mathbf{C}$)	$d = 50, \lambda = 0.8$	$d = 90, \lambda = 1$
LFM($\mathbf{C}'(\lambda \mathbf{A}^T\mathbf{A})$)	$d = 60, \lambda = 0.6$	$d = 70, \lambda = 1$
tKatz($\mathbf{A}$)	$\beta = 10^{-12}$	$\beta = 10^{-12}$
tKatz($\mathbf{C}$)	$\beta = 0.01, \lambda = 0.2$	$\beta = 0.1, \lambda = 0.4$

Table III. Best parameters learned by the recommendation algorithms using validation.

mances of the algorithms on the Youtube data set are shown in Figure 2 (b). We observe that the case for Youtube is similar to that of Orkut, in that graph proximity based algorithms significantly outperform latent factors based algorithms. In particular, tKatz($\mathbf{C}$) is highly successful compared to the other methods.

Another interesting comparison of latent factor methods based on the choice of $\mathbf{D}$ in constructing the combined network $\mathbf{C}'$ given in (7), is presented in Figure 3. We observe that the choice of $\mathbf{D}$ does not appear to make any significant difference in the performance of the recommendation algorithm. In the plot we use $\mathbf{D}_2 = \mathbf{A}^T\mathbf{A}$ and $\mathbf{D}_3 = \lambda\mathbf{A}^T\mathbf{A}$, where λ is also the weight associated with $\mathbf{S}$ in the combined graph. Even though, in case of the Orkut data set, we see that SVD($\mathbf{C}', \mathbf{D}$) performs slightly better than SVD($\mathbf{C}$), it appears as if scaling $\mathbf{D} \neq 0$ does not affect recommendation quality. In case of Youtube (plots not shown), our experiments indicate that $\mathbf{D}$ is not useful at all. The obvious choices for $\mathbf{D}$ do not seam to improve the performance compared with $\mathbf{D} = 0$.

The best parameters learned by the various algorithms are presented in Table III. Note that the best parameter $\beta = 10^{-12}$ implies that the calculated tKatz measure was effectively using the common neighbors method. In other words, users and communities connected by path lengths 5 or more[6] are not useful in making affiliation recommendations.

We see that the recommendation algorithms perform consistently across the two data sets, and the evaluations are robust as the specificities and sensitivities are averaged over 9000 users in Orkut and 16000 users in Youtube.

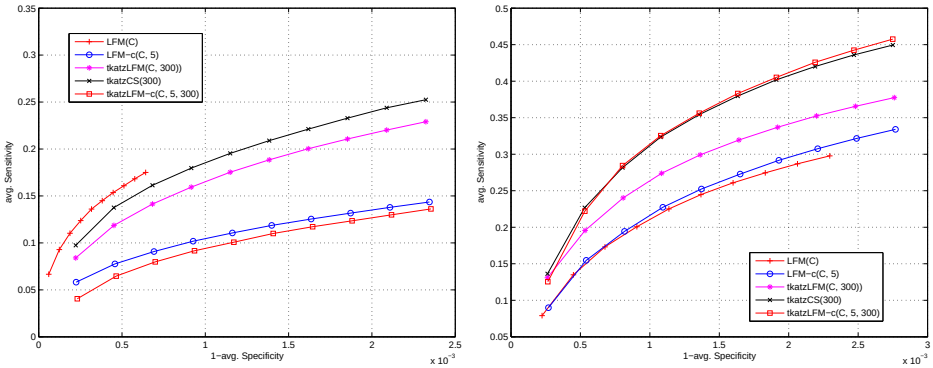
5.3.2 *Scalable approximations to tKatz(C).* From Figure 4, we observe that in general, combining ideas from the latent factor model with graph proximity model yields better performance than simply using the low rank approximations of $\mathbf{C}$. The exception to this is the case of recommendations made by LMF-c on the Orkut-1c data set, here tKatzLMF-c decreases the quality of the predictions made by LMF-c further.

In Figure 5, we observe that the recommendation quality of the scalable recommenders proposed in this paper is close to that of tKatz($\mathbf{C}$). In particular, we note that tKatzCS consistently performs very well on both data sets. Comparing the performance of tKatzLFM and tKatzLFM-c, we also note that the use of clustering is effective in improving performance, and this suggests that we can perhaps derive similar benefit out of considering a clustered version of the tKatzCS algorithm.

Finally, in Figure 6, we study the effects of change in the number of clusters and the number of factors used on the performance of algorithms which use graph clustering for the purpose of scalability. We observe that changing the number of

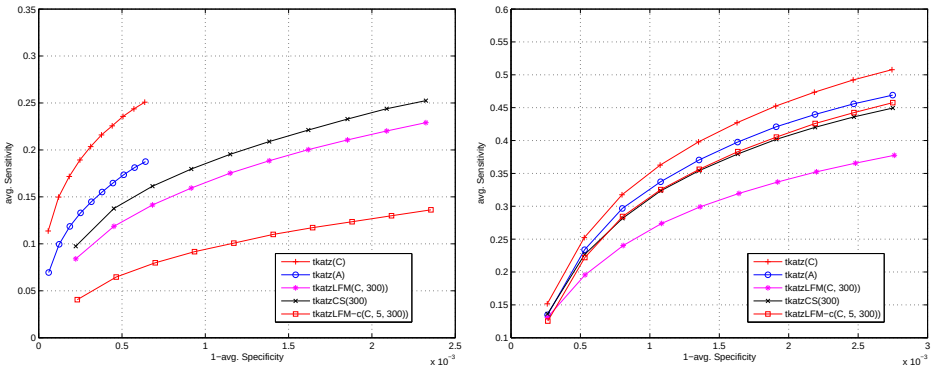
[6]Corresponding to 2nd and higher powers of $\beta\mathbf{A}\mathbf{A}^T$ in tKatz($\mathbf{A}$).

clusters does not make a significant difference in performance, whereas increasing the number of factors yields better performance. This is not surprising considering the fact that these algorithms can be viewed as computing approximations of tKatz($\mathbf{C}$).



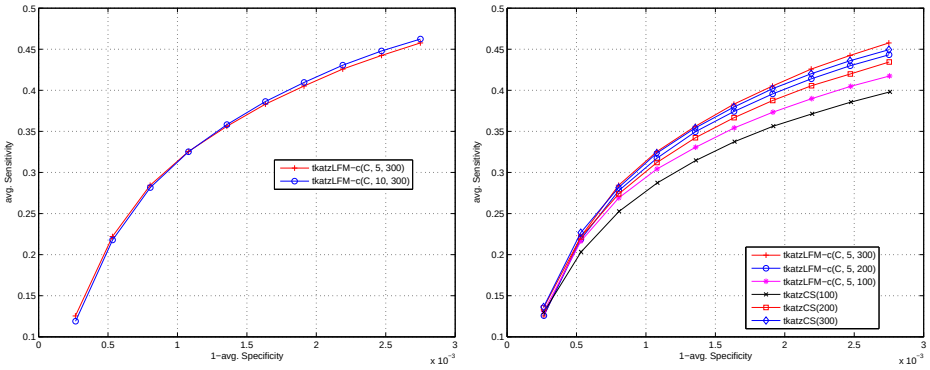
(a) Orkut-1c data set (b) Youtube-1c data set

Fig. 4. In this figure, we observe that, in general, combining ideas from the latent factor model with ideas from the graph proximity model yields better performance than simply using the low rank approximations of $\mathbf{C}$ in isolation. This is consistent with our observations in Figure 2. See Section 5.3 for further discussion.



(a) Orkut-1c data set (b) Youtube-1c data set

Fig. 5. Comparison of the performance of the proposed scalable graph proximity based methods. In this figure, we observe that the recommendation-quality of the scalable recommenders proposed in this paper is close to that of tKatz($\mathbf{C}$). In particular, we note that tKatzCS consistently performs very well on both data sets. Comparing the performance of tKatzLFM and tKatzLFM-c, we also note that the use of clustering is effective in improving performance.



(a) Effect of changing the number of clusters. (b) Effect of changing the number of factors.

Fig. 6. In this figure, we study the effects of change in the number of clusters and the number of factors used on the performance of algorithms which use graph clustering for the purpose of scalability. These experiments were conducted on the Youtube-1c data set. We observe that changing the number of clusters does not make a significant difference in performance, whereas increasing the number of factors yields better performance. This is not surprising considering the fact that these algorithms can be viewed as computing approximations of $\mathsf{tKatz}(\mathbf{C})$.

6. CONCLUSION AND FUTURE WORK

6.1 Summary and Conclusion

In this paper, we have tackled the affiliation recommendation problem, where the task is to recommend new affiliations to users, given the current state of the friendship and affiliation networks. We show that information from the friendship network can indeed be fruitfully exploited in making affiliation recommendations. This auxiliary source of information was hitherto not used in making community recommendations.

Using a simple way of combining these networks, we suggested two ways of modeling the networks for the purpose of making affiliation recommendations (Section 2). The first of these approached the problem from the graph proximity viewpoint, whereas the second modelled the interactions of users and groups in the two networks using latent factors derived from optimizing towards a joint matrix factorization objective. We studied the algorithms suggested by these models on real world networks (Section 5). We motivated and proposed a way of evaluating recommenders, by measuring how good the top 50 recommendations are, and demonstrated the importance of choosing the right evaluation strategy. Algorithms suggested by the graph proximity model turn out to be the most effective, based on experiments on large real world data sets. We also introduced scalable versions of these algorithms, and demonstrated their performance. These results show that the application of techniques from social network link prediction in affiliation and item recommendation is a promising one.

6.2 Future Work

There is the intriguing possibility of using an affiliation network for link prediction in the friendship network. Discovering techniques and models which do this effectively

seems to be a challenging research avenue. Our early experiments at doing this indicate that this is a much harder problem. The reasons for this are not yet clear, and this question seems fertile for further exploration.

Within the ambit of the affiliation recommendation problem itself, one may research the ways of fruitfully using even more sources of information. For example, [Chen et al. 2008] use information from textual description of communities along with the affiliation network to make affiliation recommendations. It might be useful to consider the social network together with this auxiliary information. Also, predictors based on latent factors model and the graph proximity model may be suited for different types of users, and creating a meta-predictor which combines predictions from both classes of predictors is another attractive research direction. Finally, from the perspective of making scalable recommendations, and considering the relative effectiveness of the common subspace approach to approximate $\mathbf{S}$ and $\mathbf{A}$, we may benefit from a clustered version of the truncated Katz measure when it is based on the common subspace approximations of the two networks.

7. ACKNOWLEDGEMENTS

We thank Prateek Jain for helpful discussions. Zhengdong Lu is supported by an ICES postdoctoral fellowship at UT Austin. Berkant Savas is supported by the Swedish Research Foundation international postdoctoral fellowship and ICES. This research was supported by NSF grants CCF-0916309 and IIS-0713142.

REFERENCES

Backstrom, L., Huttenlocher, D., Kleinberg, J., and Lan, X. 2006. Group formation in large social networks: membership, growth, and evolution. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 44–54.

Bader, B. W. and Chew, P. A. 2008. Enhancing multilingual latent semantic analysis with term alignment information. In *COLING '08: Proceedings of the 22nd International Conference on Computational Linguistics*. Association for Computational Linguistics, Morristown, NJ, USA, 49–56.

Chen, W., Wang, Y., and Yang, S. 2009. Efficient influence maximization in social networks. In *KDD 09': Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 199–208.

Chen, W.-Y., Chu, J.-C., Luan, J., Bai, H., Wang, Y., and Chang, E. Y. 2009. Collaborative filtering for orkut communities: discovery of user latent behavior. In *WWW '09: Proceedings of the 18th International Conference on World wide web*. ACM, New York, NY, USA, 681–690.

Chen, W.-Y., Zhang, D., and Chang, E. Y. 2008. Combinational collaborative filtering for personalized community recommendation. In *KDD '08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 115–123.

Dhillon, I. S., Guan, Y., and Kulis, B. 2007. Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Trans. Pattern Anal. Mach. Intell. 29,* 11, 1944–1957.

Golub, G. H. and Van Loan, C. F. 1996. *Matrix Computations*, Third ed. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD.

Hofmann, T. 1999. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual International ACM SIGIR Conference on Research and development in Information Retrieval*. ACM, New York, NY, USA, 50–57.

Karypis, G. and Kumar, V. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing 20,* 1, 359–392.

Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika 18*, 39–43.

Kempe, D., Kleinberg, J., and Tardos, E. 2003. Maximizing the spread of influence through a social network. In *KDD '03: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 137–146.

Larsen, R. 1998. Lanczos bidiagonalization with partial reorthogonalization. Tech. Rep. DAIMI PB-357, Department of Computer Science, Aarhus University.

Lehoucq, R., Sorensen, D., and Yang, C. 1998. *Arpack Users' Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia.

Liben-Nowell, D. and Kleinberg, J. 2003. The link prediction problem for social networks. In *CIKM '03: Proceedings of the Twelfth International Conference on Information and Knowledge Management*. ACM, New York, NY, USA, 556–559.

Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P., and Bhattacharjee, B. 2007. Measurement and analysis of online social networks. In *IMC '07: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*. ACM, New York, NY, USA, 29–42.

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

Singh, A. P. and Gordon, G. J. 2008. Relational learning via collective matrix factorization. In *KDD '08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 650–658.

Song, H. H., Savas, B., Cho, T. W., Dave, V., Lu, Z., Dhillon, I. Zhang, Y., and Qiu, L. 2010. Clustered embedding of massive online social networks. Tech. rep., Department of Computer Science, The University of Texas at Austin.

Tang, W., Lu, Z., and Dhillon, I. S. 2009. Clustering with multiple graphs. In *ICDM '09: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 1016–1021.

Vasuki, V., Natarajan, N., Lu, Z., and Dhillon, I. 2010. Affiliation recommendations using auxiliary friendship networks. In *RecSys '10: Proceedings of the Third ACM Conference on Recommender Systems*. Barcelona, Spain.

Zheleva, E., Sharara, H., and Getoor, L. 2009. Co-evolution of social and affiliation networks. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 1007–1016.