Improving Recommender Systems by Incorporating Social Contextual Information

HAO MA, TOM CHAO ZHOU, MICHAEL R. LYU, and IRWIN KING,
The Chinese University of Hong Kong

9

Due to their potential commercial value and the associated great research challenges, *recommender systems* have been extensively studied by both academia and industry recently. However, the data sparsity problem of the involved user-item matrix seriously affects the recommendation quality. Many existing approaches to recommender systems cannot easily deal with users who have made very few ratings. In view of the exponential growth of information generated by online users, social contextual information analysis is becoming important for many Web applications. In this article, we propose a factor analysis approach based on probabilistic matrix factorization to alleviate the data sparsity and poor prediction accuracy problems by incorporating social contextual information, such as social networks and social tags. The complexity analysis indicates that our approach can be applied to very large datasets since it scales linearly with the number of observations. Moreover, the experimental results show that our method performs much better than the state-of-the-art approaches, especially in the circumstance that users have made few ratings.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; J.4 [**Computer Applications**]: Social and Behavioral Sciences

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Recommender systems, collaborative filtering, social network, matrix factorization, tags

ACM Reference Format:
Ma, H., Zhou, T. C., Lyu, M. R. and King, I. 2011. Improving recommender systems by incorporating social contextual information. ACM Trans. Inf. Syst. 29, 2, Article 9 (April 2011), 23 pages.
DOI = 10.1145/1961209.1961212 http://doi.acm.org/10.1145/1961209.1961212

1. INTRODUCTION

Recommender systems are becoming increasingly indispensable nowadays since they focus on solving the information overload problem, by providing users with more proactive and personalized information services. Typically, recommender systems are based on *collaborative filtering*, which is a technique that automatically predicts the interest of an active user by collecting rating information from other similar users or items. The underlying assumption of collaborative filtering is that the active user will prefer those items which other similar users prefer [Ma et al. 2007]. Based on this simple but effective intuition, collaborative filtering has been widely employed

The work described in this article was fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK4154/10E and CUHK4152/10E) and a Google Focused Grant Project under "Mobile 2014".
Authors' address: H. Ma, T. C. Zhou, M. R. Lyu, and I. King, Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong S.A.R.; emails: {hma, czhou, lyu, king}@cse.cuhk.edu.hk.

DOI 10.1145/1961209.1961212 http://doi.acm.org/10.1145/1961209.1961212

in some large, well-known commercial systems, including product recommendation at Amazon[1], movie recommendation at Netflix[2], etc. Due to the potential commercial value and the great research challenges, recommendation techniques have drawn much attention in data mining [Bell et al. 2007; Koren 2008], information retrieval [Banerjee and Ramanathan 2008; Das et al. 2007; Herlocker et al. 2004; Huang et al. 2004; Koren et al. 2008; Zhou et al. 2008], and machine learning [Marlin 2004; Rennie and Srebro 2005; Salakhutdinov and Mnih 2008a,b; Salakhutdinov et al. 2007; Zhu et al. 2008] communities. Recommendation algorithms suggesting personalized recommendations greatly increase the likelihood of customers making their purchases online.

A number of algorithms have been proposed to improve both the recommendation quality and the scalability problems. These collaborative filtering algorithms can be divided into two main categories: neighborhood-based and model-based approaches [Breese et al. 1998a; Sarwar et al. 2001]. Different methods make different assumptions. The neighborhood-based recommendation algorithms assume that those who agreed in the past tend to agree again in the future. They usually fall into two classes: user-based approaches [Breese et al. 1998a; Herlocker et al. 1999] and item-based approaches [Deshpande and Karypis 2004; Sarwar et al. 2001]. To predict a rating for an item from a user, user-based methods find other similar users and leverage their ratings to the item for prediction, while item-based methods use the ratings to other similar items from the user instead [Cao et al. 2008]. In addition to the neighborhood-based approach, the model-based approaches employ the observed user-item ratings to train a predefined model. Algorithms in this category include clustering methods [Kohrs and Merialdo 1999], Bayesian model [Zhang and Koren 2007], aspect model [Hofmann 2004], etc.

However, despite their success in the industry, neighborhood-based methods and model-based methods all suffer the data sparsity problem. The density of available ratings in commercial recommender systems is often less than 1% [Sarwar et al. 2001] or even much less. In such circumstances, neighborhood-based [Jin et al. 2004; Linden et al. 2003; Ma et al. 2007] collaborative filtering algorithms fail to find similar users, since the methods of computing similarities, such as the Pearson Correlation Coefficient (PCC) or the cosine method, assume that two users have rated at least some items in common. Moreover, almost all of model-based [Hofmann 2003, 2004; Salakhutdinov and Mnih 2008b; Si and Jin 2003] collaborative filtering algorithms cannot handle users who rated only a few items.

Based on the preceding analysis, in order to improve recommendation quality, we need to solve the data sparsity problem. Actually, thanks to the popularity of Web 2.0 applications, recommender systems are now associated with various kinds of social context information, including users' social trust network, tags issued by users or associated with items, etc. These contextual information contain abundant additional information about the interests of users or properties of items, hence providing a huge opportunity to improve the recommendation quality. For example, in users' social trust network, users tend to share their interests with the friends they trust. In reality, we always turn to friends we trust for movie, music, or book recommendations, and our tastes and characters can be easily affected by the company we keep.

However, traditional recommender systems assume that users are independent and identically distributed. This assumption ignores the social trust relationships among the users. But the fact is, offline, social recommendation is an everyday occurrence.

[1]http://www.amazon.com
[2]http://www.netflix.com

For example, when you ask a trusted friend for a recommendation of a movie to watch or a good restaurant to dine, you are essentially soliciting a verbal social recommendation. In Sinha and Swearingen [2001], the authors have demonstrated that, given a choice between recommendations from trusted friends and those from recommender systems, in terms of quality and usefulness, trusted friends' recommendations are preferred, even though the recommendations given by the recommender systems have a high novelty factor. Trusted friends are seen as more qualified to make good and useful recommendations compared to traditional recommender systems [Bedi et al. 2007]. From this point of view, traditional recommender systems that ignore the social network structure of users may no longer be suitable.

In order to alleviate the data sparsity problem and improve the recommendation quality, in this article, we design a general framework to make recommendations by incorporating social contextual information, such as users' social trust network [Ma et al. 2008], tags issued by users, tags associated with items, etc.

To achieve this goal, our framework integrates social contextual information and the user-item rating matrix, based on a probabilistic factor analysis. We connect these different data resources through the shared user latent feature space (or item latent feature space), that is, the user latent feature space in the social contextual information is the same as in the user-item rating matrix. By performing factor analysis based on probabilistic matrix factorization, the low-rank user latent feature space and item latent feature space are learned in order to make recommendations. The experimental results on the Epinions[3] and Movielens[4] datasets show that our method outperforms the state-of-the-art collaborative filtering algorithms, especially when active users have very few ratings. Moreover, the complexity analysis indicates that our approach can be applied to very large datasets since it scales linearly with the number of observations.

2. RELATED WORK

Our work is related to two research fields, recommender systems and social tag analysis.

2.1. Recommender Systems

In this section, we review several major approaches for recommender systems, especially for collaborative filtering. Two types of collaborative filtering approaches are widely studied: neighborhood-based and model-based.

The neighborhood-based approaches are the most popular prediction methods and are widely adopted in commercial collaborative filtering systems [Linden et al. 2003; Resnick et al. 1994]. The most analyzed examples of neighborhood-based collaborative filtering include user-based approaches [Breese et al. 1998b; Herlocker et al. 1999; Jin et al. 2004] and item-based approaches [Deshpande and Karypis 2004; Linden et al. 2003; Sarwar et al. 2001]. User-based approaches predict the ratings of active users based on the ratings of their similar users, and item-based approaches predict the ratings of active users based on the computed information of items similar to those chosen by the active user. User-based and item-based approaches often use the PCC (Pearson Correlation Coefficient) algorithm [Resnick et al. 1994] and the VSS (Vector Space Similarity) algorithm [Breese et al. 1998b] as the similarity computation

[3]http://www.epinions.com
[4]http://www.grouplens.org/node/73

methods. PCC-based collaborative filtering generally can achieve higher performance than the other popular algorithm VSS, since it considers the differences of user rating style.

In the model-based approaches, training datasets are used to train a predefined model. Examples of model-based approaches include the clustering model [Kohrs and Merialdo 1999], aspect models [Hofmann 2003, 2004; Si and Jin 2003], and the latent factor model [Canny 2002]. Kohrs and Merialdo [1999] presented an algorithm for collaborative filtering based on hierarchical clustering, which tried to balance robustness and accuracy of predictions, especially when few data were available. Hofmann [2003] proposed an algorithm based on a generalization of probabilistic latent semantic analysis to continuous-valued response variables. Recently, several matrix factorization methods [Rennie and Srebro 2005; Salakhutdinov and Mnih 2008a, 2008b; Srebro and Jaakkola 2003] have been proposed for collaborative filtering. These methods all focus on fitting the user-item rating matrix using low-rank approximations, and use it to make further predictions. The premise behind a low-dimensional factor model is that there is only a small number of factors influencing preferences, and that a user's preference vector is determined by how each factor applies to that user.

All the preceding methods for recommender systems, however, are based on the assumption that users are independent and identically distributed, and ignore the social activities between users, which is not consistent with the reality that we normally ask friends for recommendations.

In the most recent research conducted in Singla and Richardson [2008], by analyzing the who talks to whom social network on the MSN instant messenger[5] over 10 million people with their related search records on the Live Search Engine[6], Singla and Richardson [2008] revealed that people who chat with each other (using instant messaging) are more likely to share interests (their Web searches are the same or topically similar). Therefore, to improve recommendation accuracy, in modern recommender systems, both the social network structure and user-item rating matrix should be taken into consideration.

Based on this intuition, many researchers have recently started to analyze trust-based recommender systems. In Massa and Avesani [2004], a trust-aware collaborative filtering method for recommender systems is proposed. In this work, the collaborative filtering process is informed by the reputation of users, which is computed by propagating trust. Trust values are computed in addition to similarity measures between users. The experiments on a large real dataset show that this work increases the coverage (number of ratings that are predictable) while not reducing accuracy (the error of predictions). Bedi et al. [2007] proposed a trust-based recommender system for the semantic Web. This system runs on a server with the knowledge distributed over the network in the form of ontologies, and uses the web of trust to generate the recommendations. These methods are all neighborhood-based methods which employ only heuristic algorithms to generate recommendations. There are several problems with this approach, however. The relationship between the trust network and the user-item matrix has not been studied systematically. Moreover, these methods are not scalable to very large datasets since they may need to calculate the pairwise user similarities and pairwise user trust scores.

2.2. Social Tag Analysis

Recently, there has been plenty of research investigations on social tagging systems. Heymann et al. [2008], Li et al. [2008], and Sen et al. [2006] have shown that tags

[5]http://www.msn.com
[6]http://www.live.com

can represent users' judgments about Web content quite accurately, and also are good candidates to describe the resources. Heymann et al. [2008] studied social tag prediction, and found that tag-based association rules can produce very high-precision tag predictions. Song et al. [2008] advocated a two-stage framework to do real-time tag recommendation. Ramage et al. [2009] employed tags as a complementary data source to page text and anchor text for improving automatic clustering of Web pages. Schenkel et al. [2008] used tags as semantic expansions to help social search. However, in particular, little is known about whether we can utilize tagging information to help improve recommendation quality. Our method differs from the previous work because we leverage tagging information to improve recommendation quality by engaging a factor analysis approach based on probabilistic matrix factorization, and inherently keeps the flexibility of tagging.

2.3. Heterogeneous Data Mining

Our work is also related to heterogeneous data mining. In Long et al. [2006], the authors proposed a general model, the collective factorization on related matrices, for multitype relational data clustering. The proposed algorithm iteratively embeds each type of data objects into low-dimensional spaces and benefits from the interactions among the hidden structures of different types of data objects. Gao et al. [2005] proposed a coclustering method based on semidefinite programming on high-order heterogeneous data sources. However, different from previous work in heterogeneous data mining, our method is focusing on utilizing other data sources to improve recommender systems.

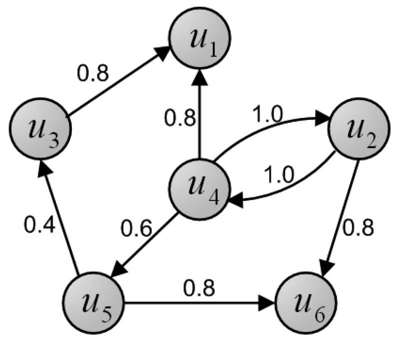
3. RECOMMENDATION FRAMEWORK

In this section, we first design a recommendation framework by consolidating a user-item rating matrix and users' social trust network in Section 3.1. Then in Section 3.2, we apply this framework to incorporating social tag information, which is another important source of social contextual information.

3.1. Recommendation with Social Trust Network

We first demonstrate our recommendation framework using a simple but illustrative toy example. Then we introduce the recommendation framework by factor analysis using probabilistic matrix factorization.

3.1.1. A Toy Example. Let us first consider the typical social trust network graph in Figure 1(a). There are 6 users in total (nodes, from u_1 to u_6) with 8 relations (edges) between users in this graph, and each relation is associated with a weight w_{ij} in the range $[0, 1]$ to specify how much user u_i knows or trusts user u_j. In an online social network Web site, the weight w_{ij} is often explicitly stated by user u_i. As illustrated in Figure 1(b), each user also rates some items (from i_1 to i_8) on a 5-point integer scale to express the extent of favor of each item. The problem we study in this article is how to predict the missing values of the user-item matrix effectively and efficiently by employing two different data sources. As mentioned in Section 1, motivated by the intuition that a user's social trust connections will affect this user's behaviors on the Web, we therefore factorize the social trust graph and user-item matrix simultaneously and seamlessly using $U^T Z$ and $U^T V$, where the shared low-dimensional matrix U denotes the user latent feature space, Z is the factor matrix in the social network graph, and V represents the low-dimensional item latent feature space. If we



(a) social network graph

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	5	2		3		4		
u_2	4	3			5			
u_3	4		2				2	4
u_4								
u_5	5	1	2		4	3		
u_6	4	3		2	4		3	5

(b) user-item matrix

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	5	2	**2.5**	3	**4.8**	4	**2.2**	**4.8**
u_2	4	3	**2.4**	**2.9**	5	**4.1**	**2.6**	**4.7**
u_3	4	**1.7**	2	**3.2**	**3.9**	**3.0**	2	4
u_4	**4.8**	**2.1**	**2.7**	**2.6**	**4.7**	**3.8**	**2.4**	**4.9**
u_5	5	1	2	**3.4**	4	3	**1.5**	**4.6**
u_6	4	3	**2.9**	2	4	**3.4**	3	5

(c) predicted user-item matrix

Fig. 1. Example for toy data.

use 5 dimensions to perform the matrix factorization for social recommendation, we obtain

$$U = \begin{bmatrix} 1.55 & 1.22 & 0.37 & 0.81 & 0.62 & -0.01 \\ 0.36 & 0.91 & 1.21 & 0.39 & 1.10 & 0.25 \\ 0.59 & 0.20 & 0.14 & 0.83 & 0.27 & 1.51 \\ 0.39 & 1.33 & -0.43 & 0.70 & -0.90 & 0.68 \\ 1.05 & 0.11 & 0.17 & 1.18 & 1.81 & 0.40 \end{bmatrix},$$

$$V = \begin{bmatrix} 1.00 & -0.05 & -0.24 & 0.26 & 1.28 & 0.54 & -0.31 & 0.52 \\ 0.19 & -0.86 & -0.72 & 0.05 & 0.68 & 0.02 & -0.61 & 0.70 \\ 0.49 & 0.09 & -0.05 & -0.62 & 0.12 & 0.08 & 0.02 & 1.60 \\ -0.40 & 0.70 & 0.27 & -0.27 & 0.99 & 0.44 & 0.39 & 0.74 \\ 1.49 & -1.00 & 0.06 & 0.05 & 0.23 & 0.01 & -0.36 & 0.80 \end{bmatrix},$$

where U_i and V_j are the column vectors and denote the latent feature vectors of user u_i and item v_j, respectively. Note that the solutions of U and V are not unique. Then we can predict the missing value w_{ij} in Figure 1(b) using $U_i^T V_j$ (before prediction, we need to first transfer the value of $U_i^T V_j$ using logistic function $g(x)$ and another mapping function $f(x)$, which will be introduced in Section 3.1.2 and Section 3.1.3, respectively). Therefore, all the missing values can be predicted using five-dimensional matrices U and V, as shown in Figure 1(c). Note that even though user u_4 does not rate any items, our approach still can predict reasonable ratings.

Since this example is a toy example, we cannot evaluate the accuracy of the prediction. However, the experimental analysis in Section 4 based on the Epinions dataset tests the effectiveness of our approach. In the following sections, we will present the details of how we conduct factor analysis for social recommendation using probabilistic matrix factorization.

3.1.2. Social Network Matrix Factorization. Suppose we have a directed social network graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the vertex set $\mathcal{V} = \{v_i\}_{i=1}^n$ represents all the users in a social network and the edge set $\mathcal{E}$ represents the relations between users. Let $C = \{c_{ik}\}$ denote the $m \times m$ matrix of $\mathcal{G}$, which is also called the social network matrix in this article. For a pair of vertices, v_i and v_k, let $c_{ik} \in (0, 1]$ denote the weight associated with an edge from v_i to v_k, and $c_{ik} = 0$, otherwise. The physical meaning of the weight c_{ik} can be interpreted as how much a user i trusts or knows user k in a social network. Note that C is an asymmetric matrix, since in a social network, especially in a trust-based social network, user i trusting k does not necessary indicate user k trusts i.

The idea of social network matrix factorization is to derive a high-quality l-dimensional feature representation U of users based on analyzing the social network graph $\mathcal{G}$. Let $U \in R^{l\times m}$ and $Z \in R^{l\times m}$ be the latent user and factor feature matrices, with column vectors U_i and Z_k representing user-specific and factor-specific latent feature vectors, respectively. We define the conditional distribution over the observed social network relationships as

$$p(C|U, Z, \sigma_C^2) = \prod_{i=1}^{m}\prod_{k=1}^{m} \mathcal{N}\left[\left(c_{ik}|g(U_i^T Z_k), \sigma_C^2\right)\right]^{I_{ik}^C}, \tag{1}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is the probability density function of the Gaussian distribution with mean μ and variance σ_C^2, and I_{ik}^C is the indicator function that is equal to 1 if user i trusts or knows user k and equal to 0 otherwise. The function $g(x)$ is the logistic function $g(x) = 1/(1 + \exp(-x))$, which makes it possible to bound the range of $U_i^T Z_k$ within the range [0, 1]. We also place zero-mean spherical Gaussian priors [Dueck and Frey 2004; Salakhutdinov and Mnih 2008b] on user and factor feature vectors.

$$\begin{aligned} p(U|\sigma_U^2) &= \prod_{i=1}^{m} \mathcal{N}(U_i|0, \sigma_U^2 \mathbf{I}) \\ p(Z|\sigma_Z^2) &= \prod_{k=1}^{m} \mathcal{N}(Z_k|0, \sigma_Z^2 \mathbf{I}) \end{aligned} \tag{2}$$

Hence, through a simple Bayesian inference, we have

$$\begin{aligned} &p(U, Z|C, \sigma_C^2, \sigma_U^2, \sigma_Z^2) \\ &\quad\propto\; p(C|U, Z, \sigma_C^2)p(U|\sigma_U^2)p(Z|\sigma_Z^2) \\ &\quad=\; \prod_{i=1}^{m}\prod_{k=1}^{m} \mathcal{N}\left[\left(c_{ik}|g(U_i^T Z_k), \sigma_C^2\right)\right]^{I_{ik}^C} \\ &\quad\times\; \prod_{i=1}^{m} \mathcal{N}(U_i|0, \sigma_U^2 \mathbf{I}) \times \prod_{k=1}^{m} \mathcal{N}(Z_k|0, \sigma_Z^2 \mathbf{I}). \end{aligned} \tag{3}$$

In online social networks, the value of c_{ik} is mostly explicitly stated by user i with respect to user k, which cannot accurately describe the relations between users since it contains noise and it ignores the graph structure information of the social network.

For instance, similar to the Web link adjacency graph in Zhou et al. [2005], in a trust-based social network, the confidence of trust value c_{ik} should be decreased if user i trusts a large number of users; however, the confidence of trust value c_{ik} should be increased if user k is trusted by lots of users. Hence, we employ the term c^*_{ik} which incorporates local authority and local hub values as a substitute for c_{ik} in Eq. (1),

$$p(C|U, Z, \sigma_C^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}\mathcal{N}\left[\left(c^*_{ik}|g(U_i^T Z_k), \sigma_C^2\right)\right]^{I^C_{ik}},$$

$$c^*_{ik} = \sqrt{\frac{d^-(v_k)}{d^+(v_i) + d^-(v_k)}} \times c_{ik}, \tag{4}$$

where $d^+(v_i)$ represents the outdegree of node v_i, while $d^-(v_k)$ indicates the indegree of node v_k.

3.1.3. User-Item Matrix Factorization. Now considering the user-item matrix, suppose we have m users, n movies, and rating values within the range [0, 1]. Actually, most recommender systems use integer rating values from 1 to R_{max} to represent the users' judgements on the items. In this article, without loss of generality, we map the ratings $1, \ldots, R_{max}$ to the interval [0, 1] using the function $f(x) = (x - 1)/(R_{max} - 1)$. Let R_{ij} represent the rating of user i for movie j, and $U \in R^{l\times m}$ and $V \in R^{l\times n}$ be latent user and movie feature matrices, with column vectors U_i and V_j representing user-specific and movie-specific latent feature vectors, respectively. We define the conditional distribution over the observed ratings as

$$p(R|U, V, \sigma_R^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}\mathcal{N}\left[\left(r_{ij}|g(U_i^T V_j), \sigma_R^2\right)\right]^{I^R_{ij}}, \tag{5}$$

where I^R_{ij} is the indicator function that is equal to 1 if user i rated movie j and equal to 0 otherwise. We also place zero-mean spherical Gaussian priors on user and movie feature vectors.

$$p(U|\sigma_U^2) = \prod_{i=1}^{m}\mathcal{N}(U_i|0, \sigma_U^2\mathbf{I})$$

$$p(V|\sigma_V^2) = \prod_{j=1}^{n}\mathcal{N}(V_j|0, \sigma_V^2\mathbf{I}) \tag{6}$$

Hence, similar to Eq. (3), through a Bayesian inference, we have

$$\begin{aligned}
p(U, V|R, \sigma_R^2, \sigma_U^2, \sigma_V^2)&\\
&\propto p(R|U, V, \sigma_R^2)p(U|\sigma_U^2)p(Z|\sigma_V^2)\\
&= \prod_{i=1}^{m}\prod_{j=1}^{n}\mathcal{N}\left[\left(r_{ij}|g(U_i^T V_j), \sigma_R^2\right)\right]^{I^R_{ij}}\\
&\times \prod_{i=1}^{m}\mathcal{N}(U_i|0, \sigma_U^2\mathbf{I}) \times \prod_{j=1}^{n}\mathcal{N}(V_j|0, \sigma_V^2\mathbf{I}).
\end{aligned} \tag{7}$$

3.1.4. Matrix Factorization for Social Trust Recommendation. As analyzed in Section 1, in order to reflect the phenomenon that a user's social connections will affect this user's

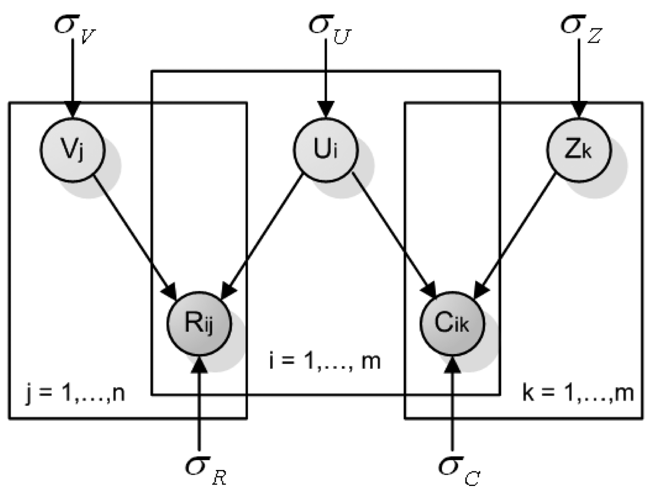


Fig. 2. Graphical model for social trust recommendation.

judgement of interest in items, we model the problem of social recommendation using the graphical model described in Figure 2, which fuses both the social network graph and the user-item rating matrix into a consistent and compact feature representation.

Based on Figure 2, we have

$$\begin{aligned} p(U, V, Z|C, R, \sigma_C^2, \sigma_R^2, \sigma_U^2, \sigma_V^2, \sigma_Z^2) \\ \propto\ & p(R|U, V, \sigma_R^2) p(C|U, Z, \sigma_C^2) \\ \times\ & p(U|\sigma_U^2) p(V|\sigma_V^2) p(Z|\sigma_Z^2). \end{aligned} \tag{8}$$

The log of the posterior distribution for the preceding equation is given by

$$\begin{aligned} \ln\ & p(U, V, Z|C, R, \sigma_C^2, \sigma_R^2, \sigma_U^2, \sigma_V^2, \sigma_Z^2) = \\ & -\frac{1}{2\sigma_R^2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij}^R (r_{ij} - g(U_i^T V_j))^2 \\ & -\frac{1}{2\sigma_C^2} \sum_{i=1}^{m} \sum_{k=1}^{m} I_{ik}^C (c_{ik}^* - g(U_i^T Z_k))^2 \\ & -\frac{1}{2\sigma_U^2} \sum_{i=1}^{m} U_i^T U_i - \frac{1}{2\sigma_V^2} \sum_{j=1}^{n} V_j^T V_j - \frac{1}{2\sigma_Z^2} \sum_{k=1}^{m} Z_k^T Z_k \\ & -\frac{1}{2} \left(\left(\sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij}^R \right) \ln \sigma_R^2 + \left(\sum_{i=1}^{m} \sum_{k=1}^{m} I_{ik}^C \right) \ln \sigma_C^2 \right) \\ & -\frac{1}{2} \left(ml \ln \sigma_U^2 + nl \ln \sigma_V^2 + ml \ln \sigma_Z^2 \right) + \mathcal{C}, \end{aligned} \tag{9}$$

where $\mathcal{C}$ is a constant that does not depend on the parameters. Maximizing the log-posterior over three latent features with hyperparameters (i.e., the observation noise variance and prior variances) kept fixed is equivalent to minimizing the following sum-of-squared-errors objective functions with quadratic regularization terms. We have

$$\begin{aligned}\mathcal{L}(R, C, U, V, Z) &= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}^{R}(r_{ij} - g(U_i^T V_j))^2 \\ &+ \frac{\lambda_C}{2}\sum_{i=1}^{m}\sum_{k=1}^{m} I_{ik}^{C}(c_{ik}^{*} - g(U_i^T Z_k))^2 \\ &+ \frac{\lambda_U}{2}\|U\|_F^2 + \frac{\lambda_V}{2}\|V\|_F^2 + \frac{\lambda_Z}{2}\|Z\|_F^2, \end{aligned} \tag{10}$$

where $\lambda_C = \sigma_R^2/\sigma_C^2$, $\lambda_U = \sigma_R^2/\sigma_U^2$, $\lambda_V = \sigma_R^2/\sigma_V^2$, $\lambda_Z = \sigma_R^2/\sigma_Z^2$, and $\|\cdot\|_F^2$ denotes the Frobenius norm. A local minimum of the objective function given by Eq. (10) can be found by performing gradient descent in U_i, V_j, and Z_k,

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial U_i} &= \sum_{j=1}^{n} I_{ij}^{R} g'(U_i^T V_j)(g(U_i^T V_j) - r_{ij})V_j \\ &+ \lambda_C \sum_{j=1}^{m} I_{ik}^{C} g'(U_i^T Z_k)(g(U_i^T Z_k) - c_{ik}^{*})Z_k + \lambda_U U_i, \\ \frac{\partial \mathcal{L}}{\partial V_j} &= \sum_{i=1}^{m} I_{ij}^{R} g'(U_i^T V_j)(g(U_i^T V_j) - r_{ij})U_i + \lambda_V V_j, \\ \frac{\partial \mathcal{L}}{\partial Z_k} &= \lambda_C \sum_{i=1}^{m} I_{ik}^{C} g'(U_i^T Z_k)(g(U_i^T Z_k) - c_{ik}^{*})U_i + \lambda_Z Z_k, \end{aligned} \tag{11}$$

where $g'(x)$ is the derivative of logistic function $g'(x) = \exp(x)/(1 + \exp(x))^2$. In order to reduce the model complexity, in all of the experiments we conduct in Section 4, we set $\lambda_U = \lambda_V = \lambda_Z$. The algorithm for learning U, V, and Z is straightforward: we first randomly initialize U, V, and Z, then iteratively update these three matrices based on their gradients until the value of the objective function converges.

3.1.5. Complexity Analysis. The main computation of gradient methods is evaluating the object function $\mathcal{L}$ and its gradients against variables. Because of the sparsity of matrices R and C, the computational complexity of evaluating the object function $\mathcal{L}$ is $O(\rho_R l + \rho_C l)$, where ρ_R and ρ_C are the numbers of nonzero entries in matrices R and C, respectively. The computational complexities for gradients $\frac{\partial \mathcal{L}}{\partial U}$, $\frac{\partial \mathcal{L}}{\partial V}$, and $\frac{\partial \mathcal{L}}{\partial Z}$ in Eq. (11) are $O(\rho_R l + \rho_C l)$, $O(\rho_R l)$, and $O(\rho_C l)$, respectively. Therefore, the total computational complexity in one iteration is $O(\rho_R l + \rho_C l)$, which indicates that the computational time of our method is linear with respect to the number of observations in the two sparse matrices. This complexity analysis shows that our proposed approach is very efficient and can scale up with respect to very large datasets.

3.2. Recommendation with Social Tags

In the previous section, we demonstrate how to recommend by incorporating users' social trust information. Actually, this general framework can also be easily extended to fuse the user-item rating matrix with social tags information. We can use a similar factor analysis approach by utilizing both users' rating information and tagging information at the same time in light of the facts that both users' rating information and users' tagging information can reflect their opinions about Web content. Specifically,

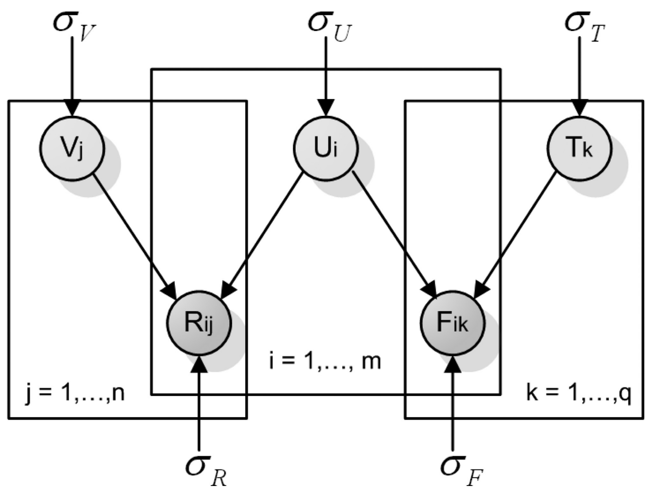


Fig. 3. Graphical model for recommendation with user tags.

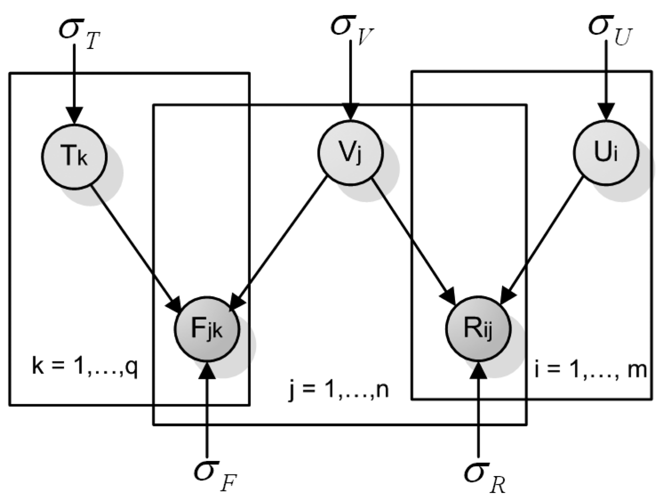


Fig. 4. Graphical model for recommendation with item tags.

on the one hand, we connect users' rating information with users' tagging information through the shared user latent feature space. The graphical model of this case is shown in Figure 3, where the matrix T represents the latent feature of each tag, and F_{ik} indicates how many times that user u_i used tag t_k. We can also have the similar object function as shown in Eq. (10) with the parameter λ_T^U controlling how many users' tag information should be used. On the other hand, we connect items' received rating information with items' received tagging information through the shared item latent feature space. The related graphical model is shown in Figure 4, where F_{jk} represents how many times that item v_j is tagged by tag t_k. In the objective function, we employ λ_T^V to control how many items' tag information should be incorporated.

The user latent feature space affects users' behaviors on both rating and tagging activities, while the item latent feature space determines both the received rating information and received tagging information.

4. EXPERIMENTAL ANALYSIS

In this section, we conduct several experiments to compare the recommendation quality of our social recommendation approach with other state-of-the-art collaborative filtering methods. We conduct the experiments on two different datasets: one is Epinions which is associated with a social trust network, another is Movielens which has tag information that is issued by different users.

Our experiments are intended to address the following questions.

(1) How does our approach compare with the published state-of-the-art collaborative filtering algorithms?
(2) How does the model parameter λ_C affect the accuracy of prediction?
(3) What is the performance comparison on users with different observed ratings?
(4) Can our algorithm achieve good performance even if users have no observed ratings?
(5) Is our algorithm efficient for large datasets?

4.1. Metrics

We use two metrics, the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE), to measure the prediction quality of our proposed approach in comparison with other collaborative filtering and trust-aware recommendation methods.

The metrics MAE is defined as

$$MAE = \frac{\sum_{i,j} |r_{i,j} - \widehat{r}_{i,j}|}{N}, \tag{12}$$

where $r_{i,j}$ denotes the rating user i gave to item j, $\widehat{r}_{i,j}$ denotes the rating user i gave to item j as predicted by a method, and N denotes the number of teted ratings. The metrics RMSE is defined as

$$RMSE = \sqrt{\frac{\sum_{i,j}(r_{i,j} - \widehat{r}_{i,j})^2}{N}}. \tag{13}$$

4.2. Compared Methods

In this section, in order to show the performance improvement of our recommendation algorithm with social contextual information (SoRec), we compare our algorithm with two baseline methods user mean and item mean, three state-of-the-art matrix factorization algorithms NMF [Lee and Seung 1999], SVD [Kurucz et al. 2007], and PMF [Salakhutdinov and Mnih 2008b], as well as a trust-aware recommendation algorithm trust [Massa and Avesani 2004].

4.3. Epinions Dataset

4.3.1. Description of the Epinions Dataset. A tremendous amount of data has been produced on the Internet every day over the past decade. Millions of people influence each other implicitly or explicitly through online social network services, such as Facebook[7]. As a result, there are many online opportunities to mine social networks for the purposes of social recommendations.

We choose Epinions as the data source for our experiments on social recommendation. Epinions.com is a well-known knowledge sharing and review site that was established in 1999. In order to add reviews, users (contributors) need to register for

[7]http://www.facebook.com

Table I. Statistics of User-Item Rating Matrix of Epinions

Statistics	User	Item
Max. Num. of Ratings	1,960	7,082
Avg. Num. of Ratings	12.21	7.56

Table II. Statistics of Social Trust Network of Epinions

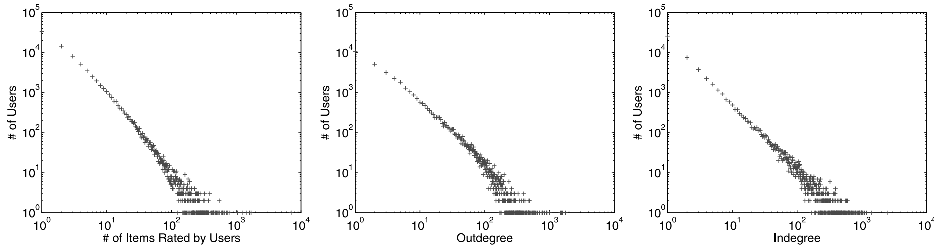
Statistics	Trust per User	Be Trusted per User
Max. Num.	1,763	2,443
Avg. Num.	9.91	9.91

free and they begin submitting their own personal opinions on topics such as products, companies, movies, or reviews issued by other users. Users can also assign products or reviews integer ratings from 1 to 5. These ratings and reviews will influence future customers when they are deciding whether a product is worth buying or a movie is worth watching. Every member of Epinions maintains a "trust" list which presents a network of trust relationships between users, and a "block (distrust)" list which presents a network of distrust relationships. This network is called the "web of trust", and is used by Epinions to reorder the product reviews such that a user first sees reviews by users that they trust. Epinions is thus an ideal source for experiments on social recommendation. Note that in this article, we only employ trust statements between users while ignoring the distrust statements, for the following two reasons: (1) The distrust list of each user is kept private in Epinions.com in order to protect the privacy of users, hence it is not available in our dataset. (2) As presented in Guha et al. [2004], the understanding of distrust is more complicated than trust, which indicates that the user trust latent feature space may not be the same as the user distrust latent feature space. The study of distrust-based social recommendation will be conducted as future work.

The dataset used in our experiments is collected by crawling the Epinions.com site on January 2009. It consists of 51,670 users who have rated a total of 83,509 different items. The total number of ratings is 631,064. The density of the user-item rating matrix is less than 0.015%. We can observe that the user-item rating matrix of Epinions is very sparse, since the densities for the two most famous collaborative filtering datasets Movielens (6,040 users, 3,900 movies and 1,000,209 ratings) and Eachmovie (74,424 users, 1,648 movies and 2,811,983 ratings) are 4.25% and 2.29%, respectively. Moreover, an important factor in choosing the Epinions dataset is that user social trust network information is not included in the Movielens and Eachmovie datasets. The statistics of the Epinions user-item rating matrix is summarized in Table I. As to the user social trust network, the total number of issued trust statements is 511,799. The statistics of this data source is summarized in Table II.

We also observe a number of power-law distributions in our dataset, including items per user distribution, and social trust network outdegree and indegree distributions. The distributions are shown in Figure 5.

4.3.2. Comparison. We use different amounts of training data (90%, 80%, 70%, 60%) to test all the algorithms. Training data 90%, for example, means we randomly select 90% of the ratings from Epinions dataset as the training data to predict the remaining 10% of ratings. The random selection was carried out 5 times independently. The experimental results are shown in Table III. The parameter settings of our approach are $\lambda_C = 20$, $\lambda_U = \lambda_V = \lambda_Z = 0.001$, and in all the experiments conducted in the following



(a) (b) (c)

Fig. 5. Power-law distributions of the Epinions dataset. (a) Items per user distribution; (b) trust graph outdegree distribution; (c) trust graph indegree distribution.

Table III. MAE Comparison with Other Approaches on Epinions Dataset

Methods		90% Training	80% Training	70% Training	60% Training
User Mean		0.9294	0.9319	0.9353	0.9384
Item Mean		0.8936	0.9115	0.9316	0.9528
Trust		0.9005	0.9044	0.9082	0.9153
5D	NMF	0.8938	0.8975	0.9229	0.9430
	SVD	0.8739	0.8946	0.9214	0.9421
	PMF	0.8678	0.8946	0.9127	0.9350
	SoRec	**0.8442**	**0.8638**	**0.8751**	**0.8948**
10D	NMF	0.8712	0.8951	0.9211	0.9408
	SVD	0.8702	0.8921	0.9189	0.9382
	PMF	0.8651	0.8886	0.9092	0.9328
	SoRec	**0.8404**	**0.8580**	**0.8722**	**0.8921**

(A smaller MAE value means a better performance).

Table IV. RMSE Comparison with Other Approaches on Epinions Dataset

Methods		90% Training	80% Training	70% Training	60% Training
User Mean		1.1927	1.1968	1.2014	1.2082
Item Mean		1.1678	1.1973	1.2276	1.2505
Trust		1.1697	1.1761	1.1797	1.1894
5D	NMF	1.1649	1.1861	1.2090	1.2311
	SVD	1.1635	1.1845	1.2067	1.2298
	PMF	1.1583	1.1798	1.2008	1.2271
	SoRec	**1.1333**	**1.1530**	**1.1690**	**1.1892**
10D	NMF	1.1621	1.1832	1.2073	1.2294
	SVD	1.1600	1.1812	1.2011	1.2268
	PMF	1.1544	1.1760	1.1968	1.2230
	SoRec	**1.1293**	**1.1492**	**1.1660**	**1.1852**

(A smaller RMSE value means a better performance).

sections, we set all of the parameters λ_U, λ_V, and λ_Z equal to 0.001. From Table III and Table IV, we can observe that our approach outperforms the other methods. The improvements are significant, which shows the promising future of our recommendation approach.

4.3.3. Impact of Parameter λ_C. The main advantage of our recommendation approach is that it incorporates the social trust network information, which helps predict users' preferences. In our model, parameter λ_C balances the information from the user-item rating matrix and the user social trust network. If $\lambda_C = 0$, we only mine the user-item rating matrix for matrix factorization, and if $\lambda_C = \infty$, we only extract information from the social network to predict users' preferences. In other cases, we fuse information from the user-item rating matrix and the user social network for probabilistic matrix factorization and, furthermore, to predict ratings for active users.

Figure 6 shows the impacts of λ_C on MAE and RMSE. We observe that the value of λ_C impacts the recommendation results significantly, which demonstrates that fusing the user-item rating matrix with the user social trust network greatly improves the recommendation accuracy. As λ_C increases, the prediction accuracy also increases at first, but when λ_C surpasses a certain threshold, the prediction accuracy decreases with further increase of the value of λ_C. This phenomenon confirms the intuition that fusing the user-item rating matrix and the user social trust network can generate better performance than only purely using each of these two resources separately. From Figure 6, we observe that for this Epinions dataset, our social recommendation method achieves the best performance when λ_C is around 20, while smaller values like $\lambda_C = 0.1$ or larger values $\lambda_C = 100$ can potentially degrade the model performance.

4.3.4. Performance on Different Users. One main task we target in this article is to provide accurate recommendations when users only supply a few ratings, or even have no rating records. Although previous work has noticed this critical problem, few approaches perform well when few user ratings are given. Hence, in order to compare our approach with the other methods thoroughly, we first group all the users based on the number of observed ratings in the training data, and then evaluate prediction accuracies of different user groups. The experimental results are shown in Figure 7. Users are grouped into 6 classes: "$1-10$", "$11-20$", "$21-40$", "$41-80$", "$81-160$", and "> 160", denoting how many ratings users have rated.

Figure 7(a) summarizes the distributions of testing data according to groups in the training data (90% as training data). For example, there are a total of 3,360 user-item pairs to be predicted in the testing dataset in which the related users in the training dataset have rating numbers from 1 to 10. In Figure 7(b) and Figure 7(c), we observe that our SoRec algorithm consistently outperforms other methods even when users only rated very few ratings.

4.3.5. Efficiency Analysis. The complexity analysis in Section 3.1.5 states that the computational complexity of our approach is linear with respect to the number of ratings, which proves that our approach is scalable to very large datasets. Actually, our approach is very efficient even when using a very simple gradient descent method. In the experiments using 90% of the data as training data, each iteration only needs less than 2 seconds. Also, as shown in Figure 8, when using 90% of the data as training data, our method needs less than 300 iterations to converge, which only needs approximately 10 minutes. When using 60% of the data as training data, we only need less than 5 minutes to train the model. All the experiments are conducted on a normal personal computer containing an Intel Pentium D CPU (3.0 GHz, Dual Core) and 1 gigabyte memory.

From Figure 8, we also observe that when using a small value of λ_C, such as $\lambda_C = 0.1$ or $\lambda_C = 1$, after 50 or 100 iterations, the model begins to overfit, while a larger λ_C, such as $\lambda_C = 20$, does not have the overfitting problem. These experiments clearly demonstrate that in this Epinion dataset, using little social network information can cause

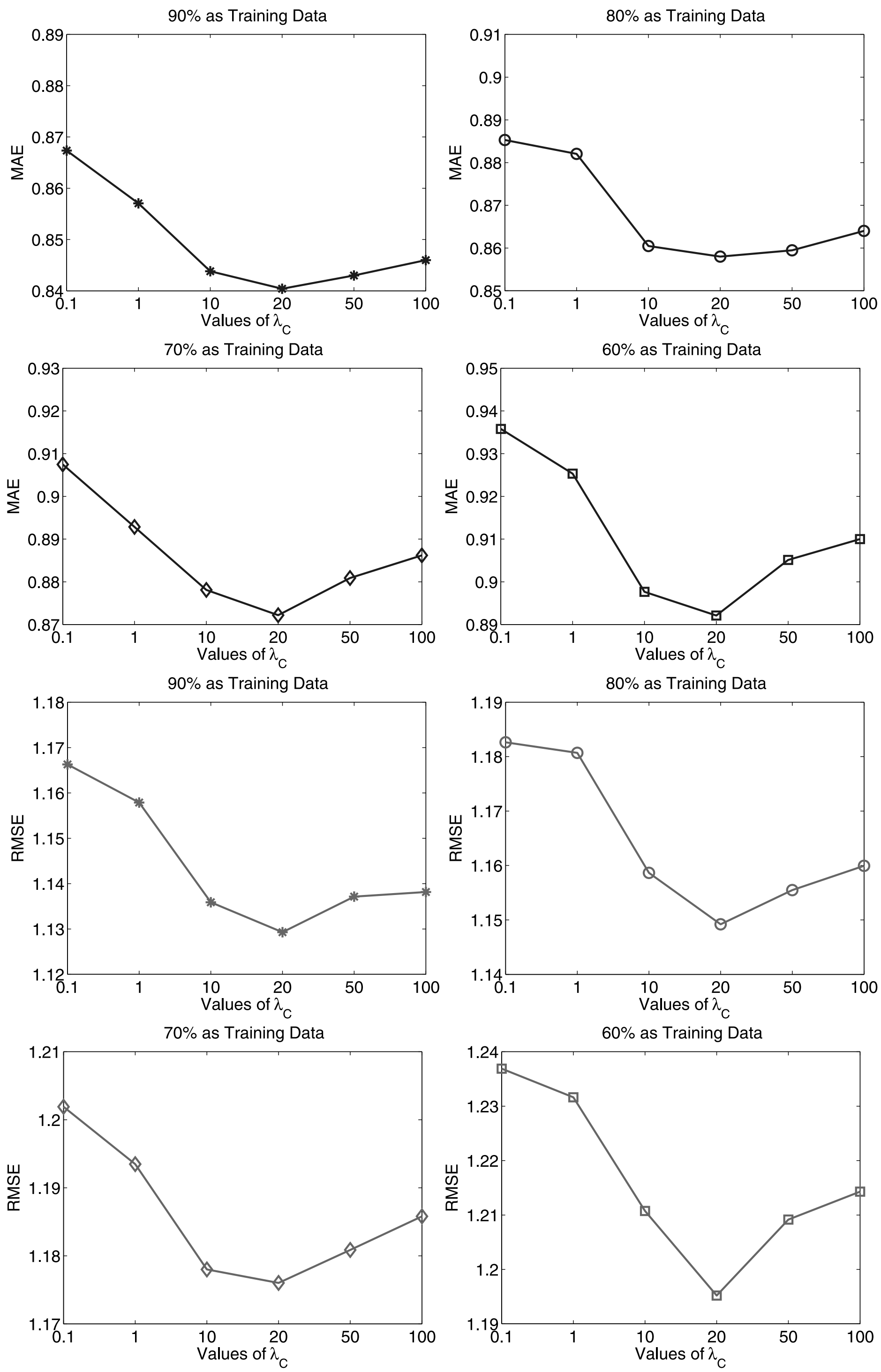
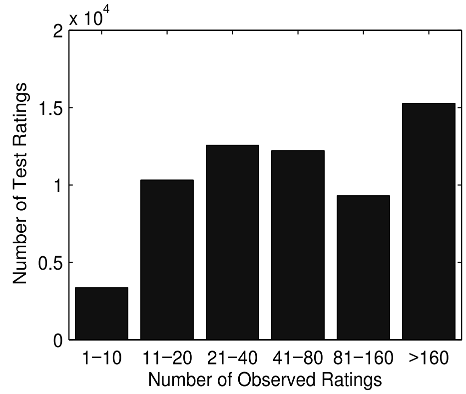
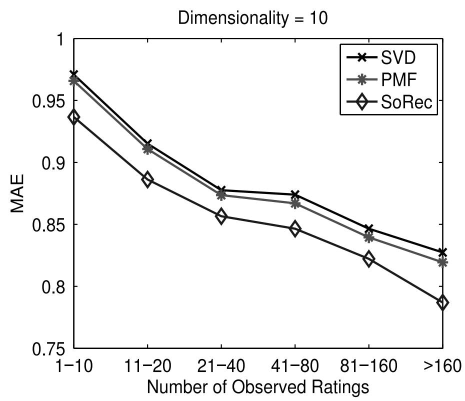


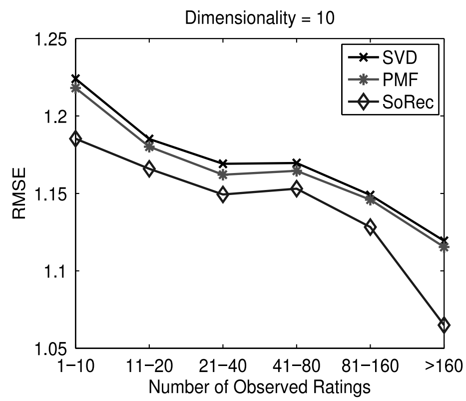
Fig. 6. Impact of parameter λ_C (dimensionality = 10).



(a) distribution of testing data (90% as training data)



(b) MAE comparison on different user rating scales (90% as training data)



(c) RMSE comparison on different user rating scales (90% as training data)

Fig. 7. Performance comparison on different users.

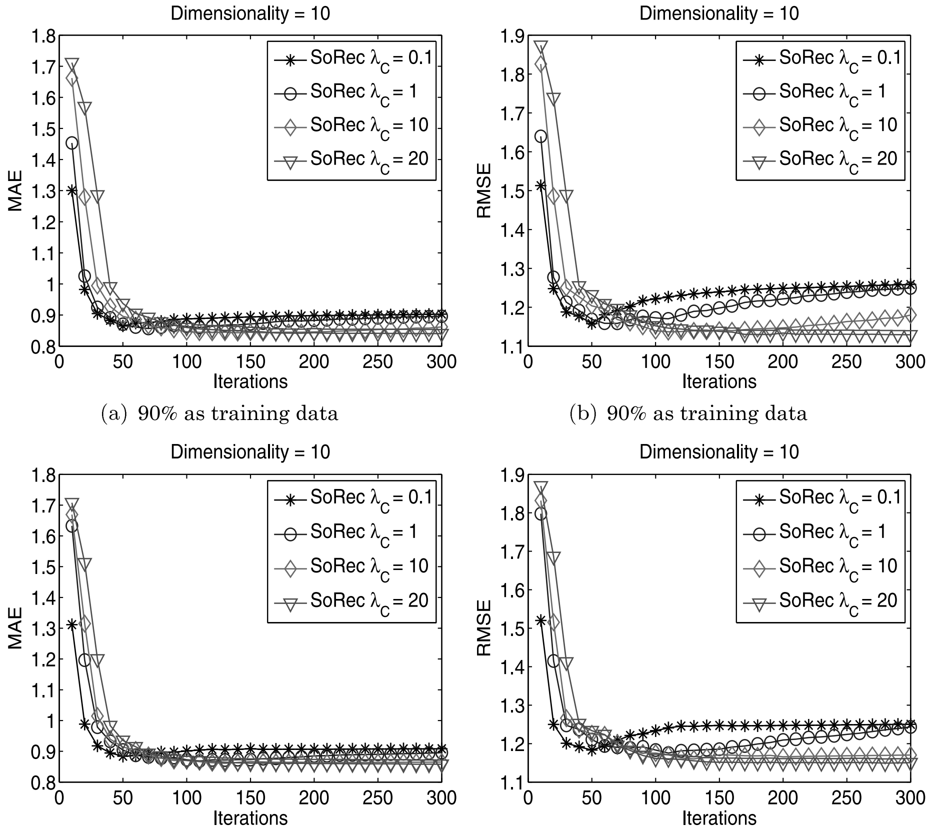
overfiting problem, and that the predictive accuracy can be improved by incorporating more social network information.

4.4. MovieLens Dataset

4.4.1. Description of the MovieLens Dataset. MovieLens is a famous recommender system. The dataset we employ in this article is the 10M/100K dataset. This dataset contains 10,000,054 ratings and 95,580 tags added to 10,681 movies by 71,567 users of the online movie recommender service MovieLens.

4.4.2. Comparison. In the comparison, we employ different amounts of training data, including 80%, 50%, 30%, 10%. 80% training data means we randomly select 80% of the ratings from the MovieLens 10M/100K dataset as the training data, and leave the remaining 20% as prediction performance testing. The procedure is carried out 5 times independently, and we report the average values in this article.

As introduced in Section 3.2, we can incorporate social tag information in two ways: (1) the first method is to treat the tags as the favors of users (we call this



(a) 90% as training data (b) 90% as training data

(c) 80% as training data (d) 80% as training data

Fig. 8. Efficiency analysis.

method SoRecUser, and it is related to the graphical model shown in Figure 3 with the parameter λ_T^U); (2) the second method is to interpret the tags as the properties of items (we call this method SoRecItem, and it is associated with the graphical model shown in Figure 4 with the parameter λ_T^V).

In the comparison, we set $\lambda_T^U = 0.1$ and $\lambda_T^V = 10$. The MAE results and RMSE results are reported in Table V and Table VI, respectively. From the results, we can see that our SoRecUser and SoRecItem approaches consistently outperform the baseline methods and the state-of-the-art recommendation algorithms, especially when there is a small amount of training data, which is equivalent to data sparsity in reality. In addition, it is necessary to notice that in the MovieLens 10M/100K dataset, all the selected users have rated at least 20 movies, but in reality, according to the famous power-law distribution phenomenon, in almost all kinds of Web activities, most users only rated very few items. Thus, we can see the improvement of our method is significant, and this again shows the promising future of our approach.

As to the parameters λ_T^U and λ_T^V, basically, they share similar trends with Figure 6, hence we do not show the detailed results here.

4.4.3. Performance on Items with Different Number of Tags. One major contribution of this article is incorporating social tagging information with traditional rating information

Table V. MAE Comparison with Other Approaches on MovieLens Dataset

Methods		80% Training	50% Training	30% Training	10% Training
User Mean		0.7686	0.7710	0.7742	0.8234
Item Mean		0.7379	0.7389	0.7399	0.7484
10D	NMF	0.6328	0.6556	0.6911	0.7428
	SVD	0.6169	0.6376	0.6821	0.7315
	PMF	0.6162	0.6354	0.6648	0.7189
	SoRecUser	0.6156	0.6347	0.6613	0.7115
	SoRecItem	**0.6155**	**0.6334**	**0.6526**	**0.6963**
20D	NMF	0.6319	0.6526	0.6721	0.7419
	SVD	0.6167	0.6355	0.6570	0.7264
	PMF	0.6156	0.6350	0.6569	0.7128
	SoRecUser	0.6147	0.6338	0.6547	0.7084
	SoRecItem	**0.6142**	**0.6303**	**0.6487**	**0.6951**

(A smaller MAE value means a better performance).

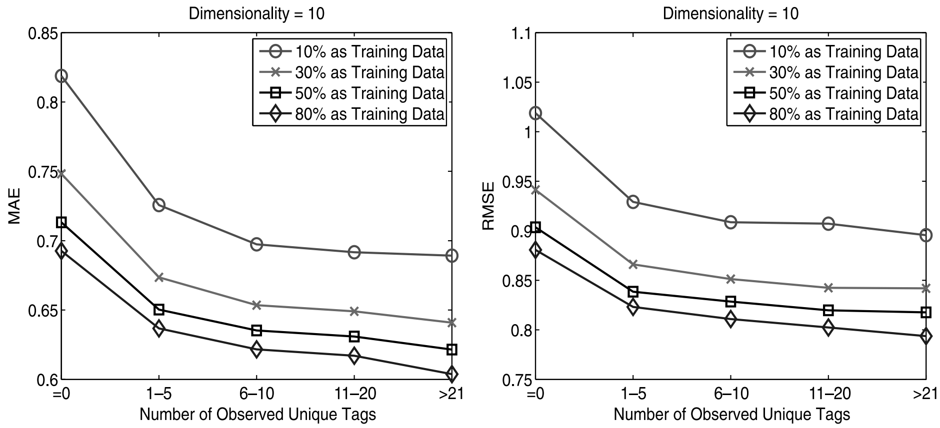
Table VI. RMSE Comparison with Other Approaches on MovieLens Dataset

Methods		80% Training	50% Training	30% Training	10% Training
User Mean		0.9779	0.9816	0.9869	1.1587
Item Mean		0.944	0.9463	0.9505	0.9851
10D	NMF	0.8320	0.8521	0.8942	0.9798
	SVD	0.8087	0.8330	0.8815	0.9703
	PMF	0.8078	0.8326	0.8647	0.9336
	SoRecUser	0.8077	0.8304	0.8596	0.9206
	SoRecItem	**0.8058**	**0.8274**	**0.8487**	**0.9028**
20D	NMF	0.8309	0.8501	0.8768	0.9759
	SVD	0.8054	0.8301	0.8575	0.9638
	PMF	0.8025	0.8275	0.8553	0.9258
	SoRecUser	0.8022	0.8250	0.8511	0.9187
	SoRecItem	**0.8018**	**0.8208**	**0.8436**	**0.9015**

(A smaller RMSE value means a better performance).

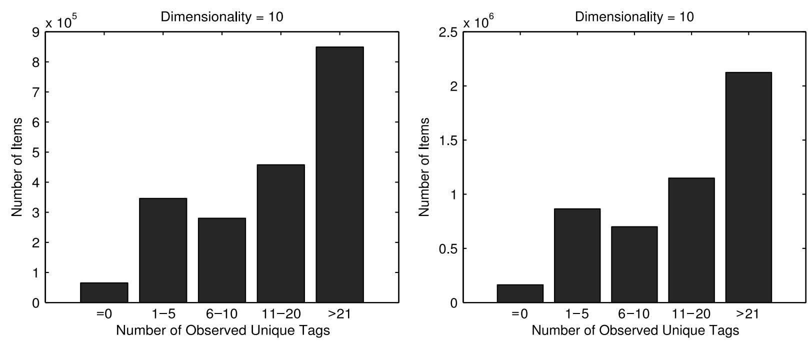
to improve prediction quality. In order to further investigate how the number of tags attached to one item affects the prediction accuracies, we first group all the items based on the number of unique tags they have been annotated with, then evaluate the prediction accuracies on different groups. We divide the items into 5 groups based on the number of unique tags that have been annotated: "= 0", "1-5", "6-10", "11-20", and "$\geq$21".

Experimental results are presented in Figure 9. This figure shows the prediction accuracies (measured with MAE and RMSE) of groups of items annotated with different numbers of unique tags, and the results of different amounts of training data are all presented. We only report the results on dimensionality = 10. From Figure 9, we can see that incorporating tags information can improve prediction quality significantly. In addition, as the number of annotated unique tags increases, the prediction quality first improves drastically, then gradually stabilizes after the number of tags surpasses some threshold value (around 20 in this dataset). This phenomenon is reasonable,

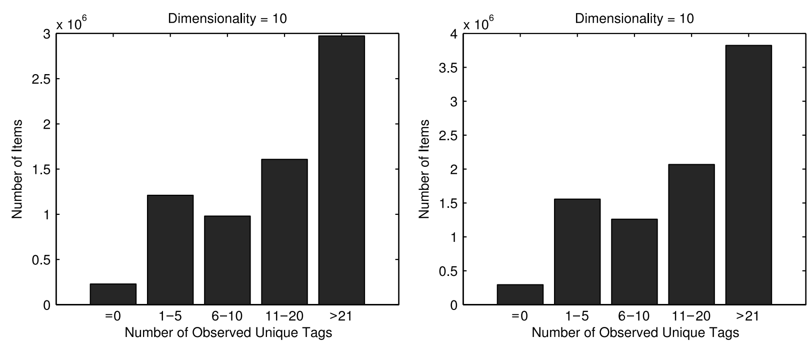


(a) MAE (b) RMSE

Fig. 9. Performance comparison on items with different no. of tags.



(a) 80% as training data (b) 50% as training data



(c) 30% as training data (d) 10% as training data

Fig. 10. Tag distributions of testing data on different amount of training data.

because with more tags' information, the concept of an item can be represented more accurately, but too many tags result in redundancy in representing the concepts of the items. Figure 10 shows the tag distributions of testing data on different amount of training data.

5. CONCLUSIONS AND FUTURE WORK

In this article, in order to alleviate the data sparsity problem in traditional recommender systems, we present a novel, efficient, and general recommendation framework fusing a user-item rating matrix with social contextual information using probabilistic matrix factorization. The experimental results show that our approach outperforms the other state-of-the-art collaborative filtering algorithms, and the complexity analysis indicates it is scalable to very large datasets. Moreover, the data fusion method using probabilistic matrix factorization we introduce in this article is not only applicable to recommendation with social contextual information, but also extensible to other popular research topics, such as social search.

For future work, we employ the inner product of two vectors to fit the observed data in this article; this approach assumes that the observed data is a linear combination of several latent factors. Although we use the logistic function to constrain the inner product, a more natural and accurate improvement over this assumption is to use a kernel representation for the two low-dimensional vectors, such as a Gaussian kernel or a polynomial kernel, which map the relations of the two vectors into a nonlinear space, and thus lead to an increase in the model's performance.

Moreover, we only employ interuser trust information in this article, but in many online social networks, distrust information is also stated by many users. Because a user trust feature space may not be consistent with the corresponding user distrust feature space, we cannot simply incorporate the distrust information into our model. In the future, we need to investigate the following two problems: whether distrust information is useful to increase the prediction quality, and how to incorporate this distrust information to obtain better-quality results.

Furthermore, when fusing the social trust network information, we ignore the information diffusion or propagation between users. A more accurate approach is to consider the diffusion process between users. Hence, we need to replace the social network matrix factorization with the social network diffusion processes. This consideration will help alleviate the data sparsity problem and will potentially increase the prediction accuracy.

Lastly, we either associate tags with users or associate tags with items. Actually, we can design a more general framework to incorporate tags with users and items simultaneously. This consideration will provide more information than either of the proposed methods, hence can further improve the recommendation quality.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers and editor for their helpful comments.

REFERENCES

BANERJEE, S. AND RAMANATHAN, K. 2008. Collaborative filtering on skewed datasets. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. 1135–1136.

BEDI, P., KAUR, H., AND MARWAHA, S. 2007. Trust based recommender system for semantic web. In *Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI'07)*. 2677–2682.

BELL, R., KOREN, Y., AND VOLINSKY, C. 2007. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proceedings of the International SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'07)*. 95–104.

BREESE, J. S., HECKERMAN, D., AND KADIE, C. 1998a. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*. Morgan Kaufmann, San Francisco, CA, 43–52.

BREESE, J. S., HECKERMAN, D., AND KADIE, C. 1998b. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI'98)*.

CANNY, J. 2002. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02)*. 238–245.

CAO, B., SUN, J.-T., WU, J., YANG, Q., AND CHEN, Z. 2008. Learning bidirectional similarity for collaborative filtering. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases Part I (ECMLPKDD'08)*. 178–194.

DAS, A. S., DATAR, M., GARG, A., AND RAJARAM, S. 2007. Google news personalization: Scalable online collaborative filtering. In *Proceedings of the 16th International Conference on World Wide Web (WWW'07)*. 271–280.

DESHPANDE, M. AND KARYPIS, G. 2004. Item-Based top-n recommendation. *ACM Trans. Inf. Syst. 22,* 1, 143–177.

DUECK, D. AND FREY, B. 2004. Probabilistic sparse matrix factorization. Tech. rep. PSI TR 2004-023. Department of Computer Science, University of Toronto.

GAO, B., LIU, T.-Y., ZHENG, X., CHENG, Q.-S., AND MA, W.-Y. 2005. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'05)*. 41–50.

GUHA, R., KUMAR, R., RAGHAVAN, P., AND TOMKINS, A. 2004. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web (WWW'04)*. 403–412.

HERLOCKER, J. L., KONSTAN, J. A., BORCHERS, A., AND RIEDL, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*. 230–237.

HERLOCKER, J. L., KONSTAN, J. A., TERVEEN, L. G., AND RIEDL, J. T. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. 22,* 1, 5–53.

HEYMANN, P., KOUTRIKA, G., AND GARCIA-MOLINA, H. 2008. Can social bookmarking improve web search? In *Proceedings of the International Conference on Web Search and Web Data Mining (WSDM'08)*. 195–206.

HOFMANN, T. 2003. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval (SIGIR'03)*. 259–266.

HOFMANN, T. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst. 22,* 1, 89–115.

HUANG, Z., CHEN, H., AND ZENG, D. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst. 22,* 1, 116–142.

JIN, R., CHAI, J. Y., AND SI, L. 2004. An automatic weighting scheme for collaborative filtering. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'04)*. 337–344.

KOHRS, A. AND MERIALDO, B. 1999. Clustering for collaborative filtering applications. In *Proceedings of the International Conference on Computational Intelligence for Modeling Control and Automation (CIMCA)*.

KOREN, J., ZHANG, Y., AND LIU, X. 2008. Personalized interactive faceted search. In *Proceeding of the 17th International Conference on World Wide Web (WWW'08)*. 477–486.

KOREN, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the International SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'08)*. 426–434.

KURUCZ, M., BENCZUR, A. A., AND CSALOGANY, K. 2007. Methods for large scale svd with missing values. In *Proceedings of the KDD Cup and Workshop*.

LEE, D. D. AND SEUNG, H. S. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature 401,* 6755, 788–791.

LI, X., GUO, L., AND ZHAO, Y. E. 2008. Tag-based social interest discovery. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. 675–684.

LINDEN, G., SMITH, B., AND YORK, J. 2003. Amazon.com recommendations: Item-to-Item collaborative filtering. *IEEE Internet Comput.*, 76–80.

LONG, B., ZHANG, Z. M., WÚ, X., AND YU, P. S. 2006. Spectral clustering for multi-type relational data. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*. 585–592.

MA, H., KING, I., AND LYU, M. R. 2007. Effective missing data prediction for collaborative filtering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*. 39–46.

MA, H., YANG, H., LYU, M. R., AND KING, I. 2008. Sorec: Social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM'08)*. 931–940.

MARLIN, B. 2004. Modeling user rating profiles for collaborative filtering. In *Advances in Neural Information Processing Systems*, Vol.16, S. Thrun, L. Saul, and B. Schölkopf Eds., MIT Press, Cambridge, MA.

MASSA, P. AND AVESANI, P. 2004. Trust-Aware collaborative filtering for recommender systems. In *Proceedings of CoopIS/DOA/ODBASE Conference*. 492–508.

RAMAGE, D., HEYMANN, P., MANNING, C. D., AND GARCIA-MOLINA, H. 2009. Clustering the tagged web. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining (WSDM'09)*. 54–63.

RENNIE, J. D. M. AND SREBRO, N. 2005. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine Learning (ICML'05)*.

RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM, P., AND RIEDL, J. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM Conference on Computer Supported Cooperative Work*.

SALAKHUTDINOV, R. AND MNIH, A. 2008a. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning (ICML'08)*.

SALAKHUTDINOV, R. AND MNIH, A. 2008b. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, Vol. 20.

SALAKHUTDINOV, R., MNIH, A., AND HINTON, G. 2007. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*. 791–798.

SARWAR, B., KARYPIS, G., KONSTAN, J., AND REIDL, J. 2001. Item-Based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW'01)*. 285–295.

SCHENKEL, R., CRECELIUS, T., KACIMI, M., MICHEL, S., NEUMANN, T., PARREIRA, J. X., AND WEIKUM, G. 2008. Efficient top-k querying over social-tagging networks. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. 523–530.

SEN, S., LAM, S. K., RASHID, A. M., COSLEY, D., FRANKOWSKI, D., OSTERHOUSE, J., HARPER, F. M., AND RIEDL, J. 2006. Tagging, communities, vocabulary, evolution. In *Proceedings of the 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW'06)*. 181–190.

SI, L. AND JIN, R. 2003. Flexible mixture model for collaborative filtering. In *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*.

SINGLA, P. AND RICHARDSON, M. 2008. Yes, there is a correlation - From social networks to personal behavior on the web. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. 655–664.

SINHA, R. R. AND SWEARINGEN, K. 2001. Comparing recommendations made by online systems and friends. In *Proceedings of the DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*.

SONG, Y., ZHUANG, Z., LI, H., ZHAO, Q., LI, J., LEE, W.-C., AND GILES, C. L. 2008. Real-Time automatic tag recommendation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. 515–522.

SREBRO, N. AND JAAKKOLA, T. 2003. Weighted low-rank approximations. In *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*. 720–727.

ZHANG, Y. AND KOREN, J. 2007. Efficient bayesian hierarchical user modeling for recommendation system. In *Proceedings of the Annual ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*. 47–54.

ZHOU, D., SCHOLKOPF, B., AND HOFMANN, T. 2005. Semi-Supervised learning on directed graphs. In *Advances in Neural Information Processing Systems*, Vol. 17.

ZHOU, D., ZHU, S., YU, K., SONG, X., TSENG, B. L., ZHA, H., AND GILES, C. L. 2008. Learning multiple graphs for document recommendations. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. 141–150.

ZHU, S., YU, K., AND GONG, Y. 2008. Predictive matrix-variate t models. In *Advances in Neural Information Processing Systems*, Vol. 20, J. Platt and S. Roweis Eds., MIT Press, Cambridge, MA, 1721–1728.

Received June 2009; revised June 2010; accepted August 2010