Product and User Dependent Social Network Models for Recommender Systems

Min Li, Zhiwei Jiang, Bin Luo, Jiubin Tang, Qing Gu*, and Daoxu Chen

Department of Computer Science, Nanjing University,
Nanjing, Jiangsu 210000, China
minli@software.nju.edu.cn, jzwpm@163.com, tjb@telecomjs.com,
{luobin,guq,cdx}@nju.edu.cn

Abstract. Social network based applications such as Facebook, Myspace and LinkedIn have become very popular among Internet users, and a major research problem is how to use the social network information to better infer users' preferences and make better recommender systems. A common trend is combining the user-item rating matrix and users' social network for recommendations. However, existing solutions add the social network information for a particular user without considering the different characteristics of the products to be recommended and the neighbors involved. This paper proposes a new approach that can adaptively utilize social network information based on the context characterized by products and users. This approach complements several existing social network based recommendation algorithms and thus can be integrated with existing solutions. Experimental results on Epinions data set demonstrate the added value of the proposed solution in two recommendation tasks: rating prediction and top-K recommendations.

1 Introduction

Recommender Systems have achieved great success and are becoming increasingly popular in real world applications. For example, online stores, such as Amazon and Netflix, provide customized recommendations for products or services based on a user's history. Many techniques have been proposed to make recommendations for the users, among which collaborative filtering is one of the most popular approaches. The task of collaborative filtering is to predict the utility of items to a particular user based on the user's history and other users' ratings.

With the increasing popularity of social network based applications such as Facebook, Myspace and LinkedIn, how to make recommendations with additional information from a user's social network has become an important research topic. In real life, we often turn to our friends for some recommendations. Besides, people with close relationship are likely to have similar tastes. Therefore, a user's social network may have two effects in the real world: help us infer users' preferences and influence users' behaviors. Hence, social network info might be an important element that recommender algorithms can take advantage of. Recently, several researchers have started to tackle this problem [11][10][9].For

* Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 13–24, 2013.

example, Jamali et al. proposed a a model-based approach utilizing matrix factorization techniques and incorporating trust propagation mechanisms [3]. Konstas et al. adopt a Random Walk framework and focus on investigating the role of additional relationships, such as friendships and social tags [6].

However, most of prior research only focused on a single-domain recommendation and thus the solutions are less likely to work well in open domain recommenders systems. There are three differences between the two kinds of recommender systems: 1) Data is sparser in the open-domain systems. Open-domain systems have much more items but less user feedback. That means the user-item rating matrix is sparser in open-domain systems. Thus traditional collaborative filtering cannot achieve as good performance as in the single-domain systems. 2) Data distribution varies according to the different domains. For example, in the Epinions data set, online stores get more reviews(average 10 ratings/item), yet books tend to get less reviews(average 2 ratings/item).[1] 3) The social network structure is more complicated than the single-domain system. Social network has been used to measure users' similarities and infer users' preferences in recommender systems. Most of prior research assumed that those people trusted by same user have the same influence for the user. However in the real world, people always selectively adopt others' opinions. Some persons are good at software, some persons are good at sports. People will consult different persons due to the products they want to purchase. Each person may involve in multiple social networks, we shouldn't consider them equally.

Recently, Hao et al.[8] introduced a framework combining social networks and collaborative filtering techniques for recommendation in an open domain data set *epinions.com*. However, similar to existing research on social network based recommender systems, their solution also combines the information using a static weight, without considering how to balance the weights between user-item ratings and social network information based on the context.

Motivated by early research on social network based recommender systems, this paper focuses on a problem that existing solutions have not addressed: how to differentiate the effect of social network info based on recommendation context. Without loss of generality, we focus on three variables that characterize the context: item category, the number of observed ratings for the user and characteristics of the neighbors. Our experiments are based on these three characteristics. We propose a solution to modify some existing social network based recommendation algorithms so that the context could be considered.

Based on experimental results, we found: 1) users' social networks influence users' behaviors and are useful for inferring users' preference; 2) how to balance the weights between user-item ratings and social network information is dependent on the recommendation context and neighbors involved. Our proposed approaches using adaptive weights can capture the recommendation context and thus outperform the approaches using a static weight; 3) utilizing social network information can help overcome the negative effect of rating variance, especially

[1] Based on the statistics of our crawled data.

in an open-domain recommender system; 4) weighted differentiation of each individual in a social network can better model the influence of the social network.

2 Social Recommendation Approaches

Our approach is to start with state-of-the-art social network recommendation algorithms, modify them so that product and neighborhood characteristics will be considered when we trade off the predicted user preferences (without considering social information) and user's neighbors' preferences.

Assume there are N items, M users in a recommender system. The rating of user i for item j is denoted by $r_{i,j}$. All the ratings from users to items are denoted by a user-rating matrix $R = \{r_{i,j}\}$. For some recommender systems, users are connected in a social network. For example, if user i selects user k as a trustable person or his/her friend, there is a directed connection from user i to user k. This network can be represented as a $M \times M$ matrix $S = \{s_{i,k}\}$, where $s_{i,k}$ denotes how well user i trust user k. In the simplest case, $s_{i,k} = 1$ means user i trusts user k, otherwise 0. The task is to recommend a list of items to a user, and good items are those that user is likely to purchase, rate high, or click.

2.1 Singular Value Decomposition

Singular Value Decomposition(SVD) is a widely used collaborative filtering algorithm. The central idea is factorizing the user-item rating matrix into low-rank approximation based on low-dimensional hidden representations of users and items, then utilizing them to predict the missing values in the rating matrix. Let $U \in \mathbb{R}^{D\times M}$ and $V \in \mathbb{R}^{D\times N}$ be latent user and item matrices, with column vectors $\mathbf{u}_i$ and $\mathbf{v}_j$ representing the latent/hidden vectors of user i and item j respectively. D is the dimension of latent vectors. There are various ways to find the latent representations of users and items. We can view it as a statistical modeling problem, where the observed ratings are generated as follows [12]

$$p(R|U, V, \sigma^2) = \prod_{r_{i,j}\in R} \mathcal{N}(r_{i,j}|\mathbf{u}_i^T\mathbf{v}_j, \sigma^2) \tag{1}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is a Gaussian distribution with mean μ and variance σ^2. The dot product of latent user and item vectors $\mathbf{u}_i^T\mathbf{v}_j$ is the expected mean of rating $r_{i,j}$. The latent vectors are assumed to be generated independently from Gaussian distributions of zero-mean: $p(U|\sigma_u^2) = \prod_{i=1}^{M} \mathcal{N}(\mathbf{u}_i|0, \sigma_u^2\mathbf{I})$ and $p(V|\sigma_v^2) = \prod_{j=1}^{N} \mathcal{N}(\mathbf{v}_j|0, \sigma_v^2\mathbf{I})$ where σ_u is the variance of the Gaussian distribution for users and σ_v is the variance of the Gaussian distribution for items. $\mathbf{I}$ is an identity matrix. Hence, the posterior distribution over the user and item latent vectors is given by

$$p(U, V|R, \sigma^2, \sigma_u^2, \sigma_v^2) \propto \prod_{r_{i,j}\in R} \mathcal{N}(r_{i,j}|\mathbf{u}_i^T\mathbf{v}_j, \sigma^2) \times \prod_{i=1}^{M} \mathcal{N}(\mathbf{u}_i|0, \sigma_u^2\mathbf{I}) \times \prod_{j=1}^{N} \mathcal{N}(\mathbf{v}_j|0, \sigma_v^2\mathbf{I})$$

We can find $\mathbf{u}_i$ and $\mathbf{v}_j$ by maximizing the above posterior likelihood. The rating for user i and item j, if not available, can be predicted as $r_{i,j} = \mathbf{u}_i^T\mathbf{v}_j$.

2.2 Factorization with Social Network

In trust-aware recommender systems, users express trust for other users. When user u trusts user k, they may have similar preference to some extent, or user k may affect user u's decisions. Social Trust Ensemble is a probabilistic framework that naturally fused users' tastes and their trusted friends' favors [8]. In this framework, the conditional distribution over the observed ratings is modeled as:

$$p(R,U,V|S,\sigma^2,\ \sigma_U^2,\sigma_V^2) \propto \prod_{i=1}^{M}\mathcal{N}(\mathbf{u}_i|0,\sigma_u^2\mathbf{I}) \times \prod_{j=1}^{N}\mathcal{N}(\mathbf{v}_j|0,\sigma_v^2\mathbf{I}) \tag{2}$$
$$\times \prod_{r_{i,j}\in R}[\mathcal{N}(r_{i,j}|(\alpha_{i,j}\mathbf{u}_i^T\mathbf{v}_j + (1-\alpha_{i,j})\sum_{k\in\tau(i)} s_{i,k}\mathbf{u}_k^T\mathbf{v}_j),\sigma^2)]$$

The model assumes the ratings are generated from different Gaussian distributions. The mean of the Gaussian distribution that generates a rating $r_{i,j}$ is determined by the latent vectors of user $\mathbf{u}_i$ and item $\mathbf{v}_j$ as well as the users in user i's social network, which is denoted as $\tau(i)$. The contributions from the two parts are weighted by the parameter $\alpha_{i,j}$. In [8], $\alpha_{i,j}$ is fixed as the same value for different user i and item j. It ignores the recommendation context associated with ratings. We will discuss this issue and propose a new solution later.

2.3 Adaptive Weights Based on User and Product Characteristics

In formula (2), $\alpha_{i,j}$ and $s_{i,k}$ balance the information from users' own characteristics and their friends' favors. $\alpha_{i,j}$ controls how much the model should trust the user vs. the neighbors, and $s_{i,k}$ controls how much one should trust user k. A straightforward way is to define a fixed value for all the $\alpha_{i,j}$ [8]. For instance, $\alpha_{i,j} = 0.4$ for all $\langle i,j\rangle$ pairs means whatever the situation is, a user's own hidden representation contributes 40% and social network contributes 60%. However, in real life, how much to trust others depends on many factors. For example, if item k is a movie, user i may ask his/her friends or read reviews before watching it. If item k is a hard drive, the user i may have clear idea about his/her preferences (size, price range) and can judge the quality easily without consulting friends.

To make $\alpha_{i,j}$ context-sensitive, we propose to set the value of $\alpha_{i,j}$ based on the features of user i and item j using the following sigmoid function:

$$\alpha_{i,j} = sigmoid(\mathbf{w}^T\mathbf{f}_{i,j}) \tag{3}$$

where $\mathbf{w} \in \mathbb{R}^P$ and $\mathbf{f}_{i,j}$ is a P-dimensional feature vector about user i and item j. Each dimension of $\mathbf{f}_{i,j}$ corresponds to one feature, and each feature value could be binary or numeric. The features could include user characteristics (gender, location, etc.) and item characteristics(price, category, etc.). The features could also include interactions between users and items. For example, a binary value indicating whether user i is familiar with products in the same category/brand of item j, or the frequency of user i visiting the web pages mentioning product j. The sigmoid function is used to restrict the value of $\alpha_{i,j}$ between 0 and 1.

According to the formula (3), the recommendation algorithm decides how much to adopt the social networks' opinions based on the characteristics of users and items. The rating $r_{i,j}$ can be estimated as follows:

$$\hat{r}_{i,j} = sigmoid(\mathbf{w}^T \mathbf{f}_{i,j}) \mathbf{u}_i^T \mathbf{v}_j + (1 - sigmoid(\mathbf{w}^T \mathbf{f}_{i,j})) \sum_{k \in \tau(i)} s_{i,k} \mathbf{u}_k^T \mathbf{v}_j \quad (4)$$

We further assume $\mathbf{w}$ follows a Gaussian distribution $\mathcal{N}(0, \sigma_w^2 \mathbf{I})$. Thus the maximum likelihood estimation of the parameters can be learned by minimizing the following loss function (the negative log likelihood of the observation):

$$loss_{R,S,U,V,W} = \sum_{r_{i,j} \in R} \frac{1}{2} (\hat{r}_{i,j} - r_{i,j})^2 + \sum_{i=1}^{M} \frac{\lambda_u}{2} \parallel \mathbf{u}_i \parallel_2^2 + \sum_{j=1}^{N} \frac{\lambda_v}{2} \parallel \mathbf{v}_j \parallel_2^2 + \frac{\lambda_w}{2} \parallel \mathbf{w} \parallel_2^2$$

where $\lambda_u = \frac{\sigma^2}{\sigma_u^2}$, $\lambda_v = \frac{\sigma^2}{\sigma_v^2}$, $\lambda_w = \frac{\sigma^2}{\sigma_w^2}$.

The solution can be found using conjugate gradient algorithm. The gradient of $\mathbf{u}_i$, $\mathbf{v}_j$ and $\mathbf{w}$ can be calculated as below:

$$\frac{\partial loss}{\partial \mathbf{u}_i} = \sum_{r_{i,j} \in R} \alpha_{i,j} (\hat{r}_{i,j} - r_{i,j}) \mathbf{v}_j + \sum_{t \in \varphi(i)} \sum_{r_{t,j} \in R} (1 - \alpha_{i,j}) (\hat{r}_{t,j} - r_{t,j}) s_{t,i} \mathbf{v}_j + \lambda_u \mathbf{u}_i$$

$$\frac{\partial loss}{\partial \mathbf{v}_j} = \sum_{r_{i,j} \in R} (\hat{r}_{i,j} - r_{i,j}) (\alpha_{i,j} \mathbf{u}_i + (1 - \alpha_{i,j}) \sum_{k \in \tau(i)} s_{i,k} \mathbf{u}_k) + \lambda_v \mathbf{v}_j$$

$$\frac{\partial loss}{\partial \mathbf{w}} = \sum_{r_{i,j} \in R} (\hat{r}_{i,j} - r_{i,j}) (\mathbf{u}_i^T \mathbf{v}_j - \sum_{k \in \tau(i)} s_{i,k} \mathbf{u}_k^T \mathbf{v}_j) \alpha'_{i,j} \mathbf{f}_{i,j} + \lambda_w \mathbf{w}$$

where $\alpha'_{i,j} = exp(\mathbf{w}^T \mathbf{f}_{i,j}) / (1 + exp(\mathbf{w}^T \mathbf{f}_{i,j}))^2$ is the derivative of the sigmoid function. $\varphi(i)$ is the set of all the users who trust user i.

2.4 Adaptive Weights Based on Individual Neighbors

$s_{i,k}$ captures how a particular neighbor k affects the prediction. According to the definition of $s_{i,k}$ and $\tau(i)$, we have the following three approaches:

Social Trust Model. A straightforward way is adopting a commonly used social network definition of recommender systems, which is so-called *social trust network*. In this scenario, $\tau(i)$ is the social network explicitly expressed by user i. For example, in *Epinions.com*, each user can express his/her Web of Trust by marking some other users as "trustable". Then the set $\tau(i)$ contains all the users who are selected by user i. There are several possible reasons that user i add user k into his/her trust list. First, they might know each other in the real life. Second, user i has read the reviews and ratings provided by user k, and found them valuable or consistent with his/her own tastes. In both cases, social trust network has much potential to be utilized for better inference of users' preferences. It is worth mentioning that the trust value is binary in most

recommender systems. This means we do not know how much user i trusts each individual in the trust list. If we simply treat all trusted users on user i's list equally, the definition of $s_{i,k}$ is $s_{i,k} = \frac{1}{|\tau(i)|}$ where $|\tau(i)|$ is the number of trusted users by user i in the set $\tau(i)$.

Social Influence Model. The social trust model mentioned above utilizes a user's social network to infer the user's preference. Now we further discuss how we model the social network influencing users' behaviors. Consider a scenario in the real world, where user i knows nothing about the movie "Avatar" initially. He found more and more people around him have watched the movie, are talking about it and rate it highly. Then there is a high probability that user i will be influenced by people around and go to the theater for "Avatar", even if he usually does not watch Action Sci-Fi movies or movies in general.

To model the influence from one's social network, we restrict $s_{i,k}$ as follows: $s_{i,k} = 1$ if user k purchased or rated item j; otherwise $s_{i,k} = 0$. While predicting $r_{i,j}$, the social influence network being considered contains all the users who are trusted by user i and also purchased/rated the target item j.

Neighborhood Model with Implicit Social Network . The above models treat different individual's opinion in the social network equally. However, people adopt others' opinions differently. For close friends that people know well, they trust them highly. In this case, we probably want to use a high value for $s_{i,k}$. For people they are not familiar with, one may cautiously take the advice. In this case, we may want a low value for $s_{i,k}$. Even for the same person, people will trust him/her in varying degrees in different recommendation contexts. Besides, social network information is not always available for a recommender system. Based on above two considerations, we propose to utilize user's neighborhoods, which can be found using standard collaborative filtering algorithms, as implicit social network. In this model, $\tau(i)$ is the top-N nearest neighbors of user i. To calculate the similarity between users, several similarity measures have been proposed before. Without loss of generality, we use cosine similarity in the space of items. We use $\mathbf{U}_i$ and $\mathbf{U}_k$ to indicate the ith and kth row of the rating matrix. Then similarity between user i and user k is defined as $sim_{i,k} = \frac{\mathbf{U}_i \mathbf{U}_k}{\|\mathbf{U}_i\| \cdot \|\mathbf{U}_k\|}$. According to the similarities between users, we select top N nearest neighbors for each user i as the implicit social network $\tau(i)$. $s_{i,k}$ is defined based on the similarity $sim_{i,k}$ with a normalizing factor so that $\sum_{k \in \tau(i)} s_{i,k} = 1$:

$$s_{i,k} = \frac{sim_{i,k}}{\sum_{t \in \tau(i)} sim_{i,t}} \tag{5}$$

3 Experimental Methodology

We collect evaluation data set from *Epinions.com* which is a consumers review website. Users can review items and provide integer ratings from 1 to 5. Epinions also provides the user profiles and item descriptions, such as item category. As a

trust-aware system, users can explicitly express the trust statements in Epinions. Each user maintains a "trust" list which includes some trustable users.

Researchers have used Epinions data set for various research on recommender systems, however, none of the existing data sets contain all the information we need. The data set used in this paper is a new collection we collected by crawling *Epinions.com* on Oct 2009. We first crawled the ratings and trust statements of the top reviewers and then move to the users who trust top reviewers or who are trusted by top reviewers. We crawled users' ratings and trust statements following users' social networks. As a result, we collected a data set that contains 56,859 users, 271,365 items, and 1,154,812 ratings. There are totally 603,686 trust statements. Most of the items are assigned into one category by *epinions.com*. 10,994(19.3%) users only rate one item. 26,712 users(47%) rated no more than 5 items. We use two sets of binary features to represent recommendation context. The first is item categories assigned by *Epinions.com*. The second is the group id that characterizes the number of items the user rated. We classify users into 7 groups (1:"1", 2:"2-5", 3:"6-10", 4:"11-20", 5:"21-40", 6:"41-80", 7:">80").

We carry out experiments on two recommendation tasks:

Rating Prediction Given a user i and an item j, the task is to predict the rating of user i on item j. For this task, we randomly select 80% rating data for training, 10% for testing, and 10% for cross validation (hold out data set). The prediction accuracy is measured by Root Mean Square Error(RMSE).

Top-K Recommendation In real life, a user wants the system to suggest a list of top K items that the user has not yet rated/purchased/seen before.

We design the experiments to answer the following questions: 1) How does the setting of the factor $\alpha_{i,j}$ affect the performance? 2) How does the selection of a user's social network $\tau(i)$ affect the performance? 3) Does weighting each neighbor' opinion differently improve the performance?

To answer question 1), we compare two different settings of $\alpha_{i,j}$. One is to define a fixed value for all the $\alpha_{i,j}$ [8]. The other is to assign adaptive weights based on characteristics of users and items (Section 2.3). To differentiate the two settings, we use "*A*" for the approaches with $\alpha_{i,j}$ that is adaptive for different users and items, and "*F*" for the approaches with a fixed value for all $\alpha_{i,k}$. To answer question 2), we compare the three models in Section 2.4 to utilize social network information. The models are denoted by "Trust", "Influence" and "Neighborhood" respectively. To answer question 3), we compare two settings of $s_{i,k}$ when using neighborhood as an implicit social network: using the similarities measure as formula (5) vs. assigning equal weights to all the neighbors.

The algorithms compared in our experiments are summarized as follows:

- *SVD*: Baseline approach as described in Section 2.1.
- *F-Trust*: Social trust network with a fixed α value (Section 2.4).
- *A-Trust*: Social trust network with adaptive α values (Section 2.3).
- *F-Influence*: Social influence network with a fixed α value (Section 2.4).
- *A-Influence*: Social influence network with adaptive α values.
- *F-Neighborhood*: This approach uses neighborhood as implicit social network (Section 2.4) and a fixed α value.

- *A-Neighborhood*: This approach uses neighborhood as implicit social network and adaptive α values.
- *F-Neighborhood-E*: A variation of *F-Neighborhood* that sets $s_{i,k} = 1/|\tau(i)|$.
- *A-Neighborhood-E*: A variation of *A-Neighborhood* that sets $s_{i,k} = 1/|\tau(i)|$.

All the approaches are based on the parameter setting $\lambda_u = \lambda_v = \lambda_w = 0.2$. For the *Neighborhood* based approaches, we use the top 10 nearest neighbors. Based on validation set, we found the fixed α values ($\alpha = 0.3$ for *F-Trust*, $\alpha = 1.0$ for *F-Influence*, and $\alpha = 0.2$ for *F-Neighborhood*.)

4 Experimental Results

4.1 Results on Rating Prediction

Table 1. Performance comparison

Model	Dimensionality		
	D=5	D=10	D=20
SVD	1.0747	1.0683	1.0812
F-Trust	1.0516	1.0434	1.0528
A-Trust	1.0481	1.0387	1.0416
F-Influence	1.0740	1.0673	1.0820
A-Influence	1.0682	1.0618	1.0664
F-Neighborhood	1.0262	1.0212	1.0270
A-Neighborhood	1.0238	1.0142	1.0022

Table 2. Performance on the subsets

Model	Influence Subset		Trust Subset	
	RMSE	α	RMSE	α
SVD	1.0467	-	1.0773	-
F-Trust	1.0009	0.3	1.0382	0.3
A-Trust	1.0002	-	1.0347	-
F-Influence	1.0447	0.9	1.0779	1.0
A-Influence	1.0373	-	1.0747	-
F-Neighborhood	1.0115	0.3	1.0323	0.2
A-Neighborhood	1.0023	-	1.0251	-

Table 1 summarizes the results on the whole test data. We conduct experiments on three latent vector dimensions: 5, 10, and 20. There are several things worth mentioning. First, it shows social network information is valuable. Social network based approaches outperformed baseline SVD. Second, it shows *A-Neighborhood* performs better than other methods. The improvement of using neighbors over SVD is not surprising. Because factorization captures global structure of the rating matrix, while neighborhood captures local regularization of the data space. Combining these complementary information has the same effect as the Netflix competition winner's solution, which combines nearest neighbors with factorization models [7]. However, it is interesting to see that neighborhood models perform better than social influence and social trust models, since neighborhood models do not use any user identified social network info. Third, it shows the performance of every approach improves when we vary α based on recommendation context (users and items). The improvements are different when using different social network information. One possible reason is the sparsity of social networks. In Epinions data set, almost every user has neighborhood, while only 59.5% of ratings in the test data have social trust information and only 18.2% have social influence information. Therefore, the overall performance may be dominated by the rating pairs without explicit social information. That also answers why *F-Influence* performs best with $\alpha = 1.0$.

Performance of Different Social Networks. To focus on the effect of different social networks, we created two test data subsets. One subset (*Trust Subset*) consists of ratings with social trust info. Both *Trust* and *Neighborhood* based

approaches can be used to predict all the test cases, while *Influence* based approaches can not be used on part of this subset. Thus the second subset (*Influence Subset*) is much smaller and consists of the ratings with all three kinds of social info available. This data set contains 6618 users and 21,599 ratings. Table 2 shows the results on the two smaller test data sets.[2] We observe that *Trust* based approaches are comparable with *Neighborhood* based approaches, although *Neighborhood* based approaches are clearly better than the others on the whole test data (Table 1). On the influence subset, trust based approaches outperforms *Neighborhood* based approaches. The results suggest that a recommender system may want to use a hybrid neighborhood-trust network model.

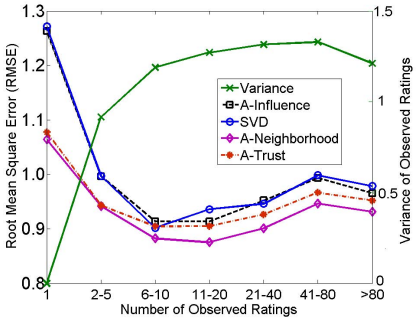

Fig. 1. Comparison on different number of observed ratings based on the whole test data

Table 3. Performance on two settings of $s_{i,k}$ when using neighborhood as implicit social network

identical $s_{i,k}$	
Model	RMSE
F-Neighborhood-E	1.0257
A-Neighborhood-E	1.0200

weighted $s_{i,k}$	
Model	RMSE
F-Neighborhood	1.0212
A-Neighborhood	1.0163

Performance on Different Users. We analyze how the size of training data per user affects the performance of different algorithms. We group all the users into 7 classes based on the number of observed ratings in the training data. Figure 1 shows the macro RMSE on different user groups. The horizontal axis describes how many training ratings are available for a user in that class. It shows that *A-Trust* and *A-Neighborhood* almost consistently outperform *SVD* and *A-Influence*, especially when users have less than 6 ratings. It's surprising that RMSE increases when the number of observed ratings is more than 10. To understand this, we look at the rating variance for each user group. We find that variance and RMSE have the similar trend, both of them tend to increase after observing more than 10 ratings (Figure 1). When user has fewer ratings, those ratings usually are about one or two aspects and thus the variance is small; when the user provide more ratings, those ratings consist of user's multiple interests. When we use all ratings to predict a rating in a specific aspect, products that are irrelevant to the target item may hurt the performance. Therefore, the initial decrease of RMSE is because the increase of observed ratings makes the model know more about users while the influence of rating variance confuses the model

[2] In the rest of this paper, all the experimental results are using 10-dimensional latent vector setting, where 10 is found by the validation data set.

and hurts the performance. It suggests that user ratings on one category may hurt the prediction of user ratings in another category.

Impact of Parameter $s_{i,k}$. In our approaches, the parameter $s_{i,k}$ indicates how much user i would trust user k. Table 3 shows the results of two settings of $s_{i,k}$ when using neighborhood as an implicit social network. It is clear that weighting others' opinions based on the similarity (*-Neighborhood) can achieve better performance than treating all opinions equally (*-Neighborhood-E).

4.2 Further Analysis about Social Influence

We did some further analysis by looking at the weights learned (the component values of $\mathbf{w}$)[3] for different contexts for social recommendation model, and the goal is to answer the following questions: 1) How does the number of observed ratings affect the weight of social network? 2) How does the size of a user's trust list affect the weight of social network? 3) Is a user more likely to be influenced when his uncertainty about the product is high? (We assume a user may be more uncertain about the product quality if the product quality tends to be subjective, such as for books/movies, instead of objective, such as for PC/memory.)

Figure 2(a) shows that, as the number of training ratings increases, the weights learned by *A-Neighborhood* become smaller, while the weights learned by *A-Trust* increase. The weight is a tradeoff between uncertainty about neighbors' ratings vs. uncertainty about the user's own ratings. In *A-Neighborhood*, the neighbors found are unreliable when the user has fewer ratings, therefore, *A-Neighborhood* does not weight neighbors' opinions high in these cases [1]. In *A-Trust*, the user's own prediction is more reliable when the number of ratings is high, thus *A-Trust* does not weight neighbors' opinions high.

To answer question 2), we introduce a new feature, the size of social trust network, for *A-Trust*. Figure 2(b) shows the weights learned by *A-Trust* increases with the size of social trust network. That means the model considers larger social trust networks less reliable than smaller ones. One possible reason is that, a large social network is more likely be selected arbitrarily by a user, while a small social network tends to be selected more seriously and hence more reliable.

Figure 2(c) shows the learned weights for different categories. It seems that categories more related to personal experiences tend to have higher weights. Instead, the categories whose ratings are more subjective tend to have lower weights, probably because a user is more uncertain about these products and is likely to be influenced by people they trust.

4.3 Results on Top-K Recommendations

A more realistic task for a recommender system is to recommend K items that users may like. In this section, we simulate the real scenario and investigate the effect of our approaches on the task of top-K recommendations.

[3] According to formula (4), a larger weight value means that less emphasis is placed on social network information.

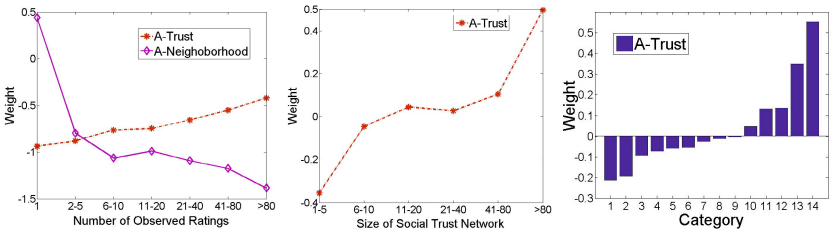

Fig. 2. Learned weights for different features. In Figure(c), item categories form 1 to 14 are: Books, Music, Kids & Family, Hotel & Travel, Software, Sports & Outdoors, Pets, Electronics, Games, Wellness & Beauty, Movies, Education, Online Stores & Services, Personal Finance.

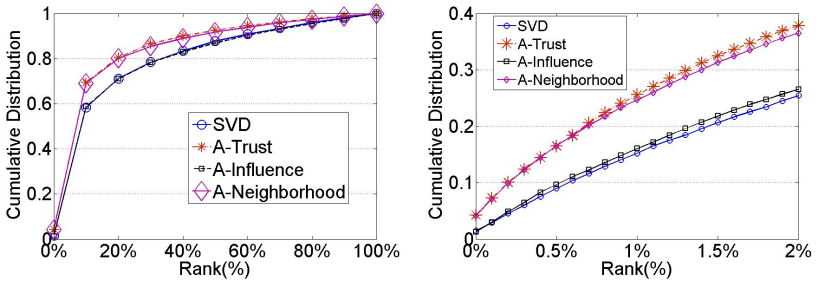

Fig. 3. Performance on Top-K Recommendation Task. The right plot concentrates on the top 2% ranking region.

Previous works on this task tend to adopt classic IR measures such as P@N and Recall [4][2]. However, without complete relevance judgements for each individual user, standard IR evaluation is almost infeasible. Thus we use a variation of the evaluation method in [7]. We randomly sample 10% from the rating data set $\langle i, j, r_{i,j} \rangle$. Then for each user in the sampled data set, we randomly choose one user-item pair with a 5-star rating. This gives 15,025 user-item testing pairs. To simulate the scenario that we only want to recommend the 5-star items to users, we treat 5-star pairs as relevant. The Epinions data is an open-domain data set with multiple categories. Intuitively, a book and a song are hard to compare. We assume that a user wants to purchase one specific kind of item, such as a book, and the system needs to rank items in this category. Therefore, for each testing pair $\langle i, j \rangle$, we randomly sample 1000 additional items which user i has not rated from the same category as item j. For example, if user i purchased a book, we randomly select 1000 additional books as candidates to be ranked.

Figure 3 compares four methods: *SVD*, *A-Trust*, *A-Influence*, *A-Neighborhood*. In real systems, only top K items might be recommended. Therefore, we focus on the top 2% ranking area (top 20 ranked items out of 1000) (Figure 3(b)). First, it shows all the social network based approaches outperform *SVD*. That means we can benefit from utilizing social network information in top-K recommendation

task. Second, it shows *A-Influence* is not as good as *A-Trust* and *A-Neighborhood* due to the sparsity of social network.

5 Conclusions and Future Work

We investigated three ways to combine social network and matrix factorization for recommender systems. All three methods work better than the baseline method. This means social network is useful for recommendation. The three methods have different properties. When social trust information is applicable, Social Trust Model always works better than SVD, especially when a user has few ratings. Social Influence Model is not always applicable. When it is applicable, making recommendations using the influence from people a user trust can also improve the performance for two tasks. When the social network information is not available, we can find implicit N Nearest Neighbors and use the Neighborhood model to combine neighbors' predictions with the SVD prediction. This is a first step to adaptively weight the info from neighbors. Future work includes adapting the influence for other social network based recommendation methods.

Acknowledgement. This work is supported by the National High-Tech Research and Development Plan of China (863) under Grant 2006AA01Z177, the National Natural Science Foundation of China (NSFC) under Grant No. 60873027, 61021062, and the National 973 Program of China under Grant No. 2009CB320705.

References

1. Berkovsky, S., Kuflik, T., Ricci, F.: Distributed collaborative filtering with domain specialization. In: RecSys, pp. 33–40 (2007)
2. Gunawardana, A., Meek, C.: A unified approach to building hybrid recommender systems. In: RecSys, pp. 117–124 (2009)
3. Jamali, M., Ester, M.: A transitivity aware matrix factorization model for recommendation in social networks. In: IJCAI, pp. 2644–2649 (2011)
4. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: CIKM, pp. 247–254 (2001)
5. Kautz, H.A., Selman, B., Shah, M.A.: Referral web: Combining social networks and collaborative filtering. Commun. ACM 40(3), 63–65 (1997)
6. Konstas, I., Stathopoulos, V., Jose, J.M.: On social networks and collaborative recommendation. In: SIGIR, pp. 195–202 (2009)
7. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD, pp. 426–434 (2008)
8. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: SIGIR, pp. 203–210 (2009)
9. Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I.: Recommender systems with social regularization. In: WSDM, pp. 287–296 (2011)
10. Massa, P., Avesani, P.: Trust-aware recommender systems. In: RecSys, pp. 17–24 (2007)
11. McDonald, D.W.: Recommending collaboration with social networks: a comparative evaluation. In: CHI, pp. 593–600 (2003)
12. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: NIPS (2007)