On Top-k Recommendation using Social Networks

Xiwang Yang†, Harald Steck‡,*Yang Guo‡ and Yong Liu†
†Polytechnic Institute of NYU, Brooklyn, NY, USA 11201
‡Bell Labs, Alcatel-Lucent, New Jersey
xyang01@students.poly.edu, hsteck@gmail.com, Yang.Guo@alcatel-lucent.com
yongliu@poly.edu

ABSTRACT

Recommendation accuracy can be improved by incorporating trust relationships derived from social networks. Most recent work on social network based recommendation is focused on minimizing the root mean square error (RMSE). Social network based top-k recommendation, which recommends to a user a small number of items at a time, is not well studied. In this paper, we conduct a comprehensive study on improving the accuracy of top-k recommendation using social networks. We first show that the existing social-trust enhanced Matrix Factorization (MF) models can be tailored for top-k recommendation by including observed and missing ratings in their training objective functions. We also propose a Nearest Neighbor (NN) based top-k recommendation method that combines users' neighborhoods in the trust network with their neighborhoods in the latent feature space. Experimental results on two publicly available datasets show that social networks can significantly improve the top-k hit ratio, especially for cold start users. Surprisingly, we also found that the technical approach for combining feedback data (e.g. ratings) with social network information that works best for minimizing RMSE works poorly for maximizing the hit ratio, and vice versa.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications–Data Mining

General Terms

Algorithms, Design, Measurement, Experimentation

Keywords

Recommender System, Social Network, Matrix Factorization

1. INTRODUCTION

The idea of recommender systems (RS) is to automatically suggest items to each user that s/he may find appealing, e.g. see [1] for an overview. Traditional collaborative filtering approaches predict users' interests by mining user rating history data [3, 9, 16, 21–24]. In real life, people often resort to their friends for recommendations in real life. It is therefore tempting to improve RS by incorporating information on the trust relationships between users in social networks. In the literature, e.g. [4, 6–8, 11, 12, 15, 17–20], it was shown that using social network information in addition to feedback data (e.g. ratings) can significantly improve recommendation accuracy. While there is some improvement on the average recommendation accuracy over all users, the improvement is particularly significant for the so-called *cold users*, who have provided only very little feedback (e.g. very few ratings) [5, 7, 8, 15]. Most of the existing social network based recommender systems are optimized for the *root mean square error (RMSE)*, which has enjoyed perhaps the largest popularity among the various accuracy measures in the recommender literature. On the other hand, top-k recommendation, a small number k of items are recommended to a user at a time is pervasive in Real-World recommendation task.

In this paper, we provide a comprehensive study on improving the accuracy of top-k RS using social networks. Towards this goal, we first show that the existing social-trust-enhanced Matrix Factorization (MF) models [7,8,11,12] can be conveniently tailored for top-k recommendation by extending their training objective functions to include both the observed ratings and the missing ratings with the consideration that rating data are missing not at random(MNRA). For the Nearest Neighbor (NN) based top-k recommendation, we propose to combine users' neighborhoods in their trust networks with their neighborhoods in the *user latent feature space* derived from matrix factorization considering MNAR. To generate top-k recommendation from a combined neighborhood, instead of taking the weighted average over only the observed ratings, we propose to use voting-based algorithm as a simple approach to consider both observed and missing ratings.

To assess the performance of the proposed social-network-based top-k RSes, we then undertake a comprehensive comparison study using two publicly available real-world data sets: Epinions and Flixster. The major findings are:

1. Trust information derived from social networks significantly improves top-k hit ratio when incorporated properly both in MF and NN models.

2. Our proposed social network based NN models and RS trained with our modified objective function considerably outperform the only published top-k approach us-

*Now at Netflix Inc. Work was done while at Bell Labs.

RecSys'12, September 9–13, 2012, Dublin, Ireland.

ing social network information in recent literature [7], an NN approach.

3. Among the various ways of combing feedback data with social network information, the one that was found to be worst with respect to RMSE turns out to be the best concerning the top-k hit ratio. This illustrates that the technical details of minimizing RMSE can be very different from the ones that work to optimize top-k hit ratio.

This paper is organized as follows. Some related work is discussed in section 2. Section 3 outlines the top-k models using social network information. In particular, in Sections 3.1.1 to 3.1.3, we outline various matrix factorization (MF) models for combining feedback data and social trust information that were proposed in the literature. As they are typically geared towards RMSE, we modify their training objectives functions towards the top-k hit ratio. We extend the nearest neighbor (NN) models in Section 3.2. In Section 4, we present the comprehensive comparison study of the various approaches–with respect to the top-k hit ratio. The paper is concluded in Section 5.

2. RECOMMENDATION TASK

We consider the following *Real-World Recommendation Task*: for each user, the recommender system has to recommend a *small* number, say k, of items from among *all available* items. One may distinguish between two slightly different variants of this task: (a) all items are eligible for recommendation, including the items that have been rated[1] by the user in the past (this assumes that a user may consume an item possibly several times, e.g. listen to a song on the online radio); or (b) only those items, which have not been rated by the user in the past, are eligible for recommendation (this assumes that a user consumes an item at most once, e.g. purchase of a movie DVD).

A meaningful offline test ideally should provide a good approximation to the utility function optimized by the deployed system (e.g. user satisfaction, increase in sales). This immediately suggests the corresponding procedures for offline testing on available data:

- **all items:** for each user, all items are considered, whether rated or not by the user (in the training set or the test set).
- **all unrated items:** for each user, only those items are considered that have not been rated by the user in the training set. Note that this contains items with and without ratings in the test data.

Due to the data sparsity, the difference between these two variants is expected to be small, as confirmed by our experiments. We hence will only report results concerning the second variant in this paper.

The user's selection bias causes the observed feedback (e.g. ratings, purchases, clicks) in the data to be missing *not* at random (MNAR). This is an important issue in practice, but largely ignored in the literature (the few exceptions include [2, 13, 14, 25]). Selection bias may result from user's tendency to rate only the items they like or know. Compared to the (unknown) distribution over (a random subset of) all ratings, the distribution of observed ratings is skewed due to the selection bias. Recent work by Marlin et al. [13, 14] provided empirical evidence that the data typically used for training and testing recommender systems indeed exhibit a significant selection bias, i.e., the ratings are missing not at random: their histograms of the distribution of ratings in the Yahoo!LaunchCast data show that *low* ratings are much more likely to be missing from the observed data than *high* ratings (see Figure 2 in [13]).

RMSE *on the observed data* is agnostic to the selection bias, as the data in the training set and the test set are from the same skewed distribution. In contrast, in top-k recommendation, as the k recommended items have to be chosen from *all* items (that were not rated in the training set), the unknown distribution over all ratings influences the recommendation accuracy, and hence user satisfaction in practice. If the ratings in the available data had been missing at random (MAR), unbiased results could have been expected from the common test procedures using **observed ratings** only. It is shown in [25] that the top-k hit ratio has desirable properties when applied to all (unrated) items in MNAR test data. Note that approaches that perform well with respect to RMSE on the observed ratings may perform poorly with respect to the top-k hit-ratio on all items [2, 10, 25].

As to compute the top-k hit ratio or recall, for each user u, we rank the items i according to the predicted rating $\hat{R}_{i,u}$. An item is defined as *relevant* to a user in the test set if s/he finds it appealing or interesting (e.g., the assigned rating in the test data is above a certain threshold). For instance, in our experiments with Epinions data, the rating values range from 1,..., 5 stars, and we consider 5-star ratings as relevant (i.e. the user definitely liked these items), while other rating values and missing rating values are considered not relevant. Other choices led to similar results. Now the top-k hit ratio or recall can be defined as the fraction of relevant items in the test set that are in the top-k of the ranking list, denoted by $N(k, u)$, from among all relevant items, $N(u)$. For each user u, the top-k hit ratio is given by

$$\mathrm{H}(k, u) = \frac{N(k, u)}{N(u)}, \tag{1}$$

which can be aggregated over all users to obtain the average top-k hit ratio or recall for the test set. The recall is computed as follows:

$$recall = \frac{\sum_u N(k, u)}{\sum_u N(u)}, \tag{2}$$

Note that a higher top-k hit ratio or recall is better. In our experiments, the evaluation metric is recall. We noticed that recommender systems that perform well with respect to recall also perform similarly well regarding other ranking measures like precision or nDCG on all items, while the RMSE measure on observed ratings behaves very differently in comparison.

3. TOP-K RECOMMENDER SYSTEMS USING SOCIAL NETWORKS

Recommender systems using social network information were mainly developed to optimize RMSE on observed ratings, e.g. [6,8,11,12,15]. Various approaches are used. While neighborhood [4,7] and random walk [6] methods were used on the social network graph, matrix factorization methods were found to be the most accurate models [8, 11, 12].

For this reason, we start with matrix factorization (MF)

[1]For simplicity, we use 'ratings' as a synonym for feedback in this paper. As will become clear, the presented approach is applicable to both explicit and implicit feedback data.

approaches using social network information, and modify them as to optimize the top-k hit ratio (rather than the original RMSE). Each of the three models and its modification is outlined in the following sub-sections. Other than MF approaches, we also consider nearest neighbor (NN) approaches. In recent recommender literature, the only top-k approach using social network information is a NN method [7] to the best of our knowledge. In Section 3.2, we develop several variants of NN approaches by adopting user latent features derived from MF optimized for top-k hit ratios.

3.1 Top-k MF using Social Networks

In the following subsection, we briefly review the existing MF approaches in the literature that combine rating data with social network information [8, 11, 12]. As to optimize the top-k hit ratio (rather than RMSE), we modify their training function as to account for *all* items, rather than the observed ratings only, analogous to *AllRank* proposed in [25] for rating data.

The social network information is represented by a matrix $S \in \mathbb{R}^{u_0 \times u_0}$, where u_0 is the number of users. The directed and weighted social relationship of user u with user v (e.g. user u trusts/knows/follows v) is represented by a positive value $S_{u,v} \in (0, 1]$. An absent or unobserved social relationship is reflected by $S_{u,v} = s_m$, where typically $s_m = 0$.

3.1.1 Social Recommendation (SoRec) Model

Social Recommendation (SoRec) was introduced in [12]. In this model, the social network matrix S (see beginning of Section 3.1) may be slightly modified as follows [12]:

$$S^*_{u,v} = S_{u,v}\sqrt{\frac{d_v^-}{d_u^+ + d_v^-}},$$

where d_u^+ is the out-degree of user u in the social network (i.e. the number of users whom u follows/trusts), and d_v^- is the in-degree of user v in the network (ie the number of users who follow/trust user v). The predicted ratings are obtained from the model as follows:

$$\hat{R} = r_m + QP^\top, \tag{3}$$

with matrices $P \in \mathbb{R}^{i_0 \times j_0}$ and $Q \in \mathbb{R}^{u_0 \times j_0}$, where $j_0 \ll i_0, u_0$ is the rank; and $r_m \in \mathbb{R}$ is a (global) offset. Besides the rating data, also the social network information is used for training this model. The social relationships are predicted as follows:

$$\hat{S}^* = s_m + QZ^\top, \tag{4}$$

where $Z \in \mathbb{R}^{u_0 \times j_0}$ is a third matrix in this model, besides P and Q. Note that the matrix Q is shared among the two equations (3) and (4). Due to this constraint, one can expect Q (i.e. the user profiles Q_u for each user u) to reflect information from both the ratings and the social network as to achieve accurate predictions for both. Note that the matrix Z is not needed for predicting rating values, and hence may be discarded after the matrices P and Q have been learned. Both (3) and (4) are combined as follows in the training objective function. Analogous to [25], we modify the training function as to account for *all* items (instead of RMSE on the *observed* ratings) for improved top-k hit-rate on the test data:

$$\begin{aligned}&\sum_{\text{all } u}\sum_{\text{all } i} W_{u,i}\left(R^{\text{o\&i}}_{u,i} - \hat{R}_{u,i}\right)^2 + \sum_{\text{all } u}\sum_{\text{all } v} W^{(S)}_{u,v}\left(S^*_{u,v} - \hat{S}^*_{u,v}\right)^2 \\ +\quad & \lambda\left(||P||_F^2 + ||Q||_F^2 + ||Z||_F^2\right),\end{aligned} \tag{5}$$

where $||\cdot||_F$ denotes the Frobenius norm of the matrices, and λ is the usual regularization parameter. $R^{\text{o\&i}}_{u,i}$ equals the actual rating value in the training data if observed for user u and item i; otherwise the value $R^{\text{o\&i}}_{u,i} = r_m$ is imputed. The training weights are [25]

$$W_{u,i} = \begin{cases} 1 & \text{if } R^{\text{obs}}_{u,i} \text{ observed} \\ w_m & \text{otherwise} \end{cases}. \tag{6}$$

The term concerning the social network (in the second line) is analogous to the first term concerning the ratings. In particular, the absent or unobserved social links are treated analogous to the missing ratings in AllRank [25], i.e. we impute the value s_m with weight $w_m^{(S)}$. Like $W_{u,i}$ in (6), $W^{(S)}_{u,v}$ is defined as follows:

$$W^{(S)}_{u,v} = \gamma \cdot \begin{cases} 1 & \text{if } S^*_{u,v} \text{ observed} \\ w_m^{(S)} & \text{otherwise} \end{cases}, \tag{7}$$

where $\gamma \geq 0$ determines the weight of the social network information compared to the rating data. Obviously, $\gamma = 0$ corresponds to the extreme case where the social network is ignored when learning the matrices P and Q. As γ increases, the influence of the social network increases. The effect is that the user profiles Q_u and Q_v of two users u and v become more similar to each other if they are friends. While only positive social relationships are considered in the original model [12], we note that this model allows also for *negative* values of $S_{u,v}$, representing e.g. distrust among users. This objective function can be optimized using the popular (stochastic) gradient descent method.

3.1.2 Social Trust Ensemble (STE) Model

Recommendation with Social Trust Ensemble (STE) was introduced in [11]. The predicted ratings are obtained from a model comprising the matrices $P \in \mathbb{R}^{i_0 \times j_0}$ and $Q^{u_0 \times j_0}$:

$$\hat{R}_{u,i} = r_m + \alpha Q_u P_i^\top + (1-\alpha)\sum_v S_{u,v} Q_v P_i^\top, \tag{8}$$

where we omitted the logistic function, as we found its effect rather negligible in our experiments. The reason is that only the ranking/order of the predicted rating values is important for the top-k hit ratio, while it is irrelevant if the predicted rating values are confined to valid rating values (e.g. the interval $[1, 5]$ stars). The trade-off between the feedback data (ratings) and the social network information is determined by $\alpha \in [0, 1]$. Obviously, the social network information is ignored for $\alpha = 1$, while $\alpha = 0$ assigns the highest possible weight to the social network information. Intermediate values of α result in a weighted combination of the information from both sources. (8) is equivalent to the matrix notation

$$\hat{R} = r_m + S_\alpha QP^\top, \tag{9}$$

where $S_\alpha = \alpha I + (1-\alpha)S$, and I is the identity matrix. Analogous to the previous section, we modify the training function to be geared towards the top-k hit ratio as follows:

$$\sum_{\text{all } u}\sum_{\text{all } i} W_{u,i}\cdot\left(R^{\text{o\&i}}_{u,i} - \hat{R}_{u,i}\right)^2 + \lambda\left(||P||_F^2 + ||Q||_F^2\right), \tag{10}$$

where $||\cdot||_F$ denotes the Frobenius norm. $W_{u,i}$ and $R^{\text{o\&i}}_{u,i}$ are defined as in the previous section. Again, this training objective function can be optimized efficiently using stocahstic gradient descent.

3.1.3 Social MF Model

The SocialMF model was proposed in [8], and was found to outperform SoRec and STE with respect to RMSE. Each of the rows of the social network matrix S has to be normalized to 1, resulting in the new matrix S^* with $S^*_{u,v} \propto S_{u,v}$, and $\sum_v S^*_{u,v} = 1$ for each user u.

The predicted ratings are obtained from the model, comprising the matrices $P \in \mathbb{R}^{i_0 \times j_0}$ and $Q^{u_0 \times j_0}$, as follows:

$$\hat{R} = r_m + QP^\top, \tag{11}$$

where we again omitted the logistic function, as we found its effect rather negligible in our experiments. Like before, we modify the training function in [8] as to better optimize the top-k hit ratio (instead of RMSE):

$$\begin{aligned} &\sum_{\text{all } u} \sum_{\text{all } i} W_{u,i} \cdot \left(R^{\text{o\&i}}_{u,i} - \hat{R}_{u,i} \right)^2 \\ + \quad & \beta \sum_{\text{all } u} \left((Q_u - \sum_v S^*_{u,v} Q_v)(Q_u - \sum_v S^*_{u,v} Q_v)^\top \right) \\ + \quad & \lambda \left(||P||_F^2 + ||Q||_F^2 \right) \end{aligned} \tag{12}$$

The tradeoff between the feedback data (ratings) and the social network information is determined by $\beta \geq 0$. Obviously, the social network information is ignored for $\beta = 0$, while increasing values of β shift the tradeoff more and more towards the social network information. The term in the second line constitutes a constraint that a user profile Q_u should be similar to the (weighted) average of his/her friends' profiles Q_v (measured in terms of the square error). We optimize also this modified training function by means of stochastic gradient descent.

3.2 Nearest Neighbor Methods

In a NN method, top-k recommendations are generated not from all items, but only from items "liked" by a subset of users who are "nearest" (under certain distance metric) to the target user. The neighborhood of a user can be calculated using collaborative filtering, or it can be just a set of directly or indirectly connected friends in a social network. This makes it convenient to incorporate social trust into NN based top-k recommendation.

Basically, NN based RS is a different approach from MF based RS. In Real-World systems, there are lots of user's feedbacks every day, e.g., as it is reported that there are billions of the like buttons served daily in facebook. NN based RS enjoys a unique advantage in that it can incrementally integrate user's new feedback into recommendation. Because nearest neighbors of a user is comparably stable within a short period, so a user's new feedbacks influence the recommendation to its neighbors in real time. While, in MF based approach, in order to integrate user's new feedbacks, it requires new matrix factorization which is not so efficient when deployed in real systems.

To the best of our knowledge, [7] is the only work that incorporates social network into NN based top-k recommender system. Two neighborhood based approaches are studied in [7] and their performance are comparable. We select one model, termed as *Trust-cf*, as the baseline for comparison.

In Trust-cf, Breadth First Search (BFS), starting from a source user u, is performed to obtain a set of trusted neighbors, namely trusted neighborhood. Meanwhile, collaborative filtering (CF) neighborhood consists of users who are close to the source user u in terms of Pearson Correlation coefficient. The items rated highly by users in either neighborhoods are considered to be candidates for top-k recommendation. Trust-cf calculates the predicted rating for a candidate item as the weighted average of all observed ratings in the two neighborhoods. The weight for a user in the trusted neighborhood is set to $1/d_v$, where d_v is the depth of user v from user u in the trust network. The weight for a user in the CF neighborhood is the Pearson Correlation coefficient between this user and the source user. If an item has predicted ratings from both neighborhoods, two predicted ratings are combined using weighted average with weights proportional to the neighborhood size for this item. Finally, Trust-cf sorts all the candidate items by their predicted ratings and recommends the top-k to the source user.

We propose a set of social network based NN approaches to achieve high top-k hit ratio by considering both social trust and MNAR. We always denote by k_1 the number of nearest users identified by the Collaborative Filtering (CF) approach, and by k_2 the number of trusted users identified by the social network based approach.

- **CF-ULF approach**. *CF-ULF* uses MF (i.e., *AllRank* [25]) to obtain the user latent features. The users are then clustered in the user latent feature space using the Pearson correlation coefficient. The k_1 users nearest to the source user u are identified. The relevant items of these nearest users are voted to form the top-k recommended items. The voting for the candidate items are computed as follows:

$$Vote_{u,i} = \sum_{v \in N_u} \sum_i sim(u,v) \delta_{i \in I_v}, \tag{13}$$

where δ is the Kronecker delta; I_v denotes the set of *relevant* items of user v; and N_u is the set of k_1 nearest neighbors of user u (as determined by the Pearson correlation). $Vote_{u,i}$ is the voting concerning item i for user u; the k_1 nearest neighbors of user u are weighted according to their similarity $sim(u,v)$ with user u, measured in terms of the Pearson correlation coefficient between user u and v (in user latent feature space).

- **PureTrust approach.** *PureTrust* approach employs the breadth-first search (BFS) in the social network to find k_2 trusted users to the source user u. The voting scheme is similar to the scheme employed in *CF-ULF*.

$$Vote_{u,i} = \sum_{v \in N_u^{(t)}} \sum_i w_t(u,v) \delta_{i \in I_v}, \tag{14}$$

where $N_u^{(t)}$ is the set of trusted users of u, and $w_t(u,v)$ is the voting weight from user v. We set $w_t(u,v) = 1/d_v$, where d_v is the depth of user v in the BFS tree rooted at user u.

- **Trust-CF-ULF approach.** *Trust-CF-ULF approach* is the combination of user latent feature space based collaborative filtering (CF-ULF) approach and social network based approach. We firstly find k_1 closest neighbors from the CF neighborhood, then find k_2 closet neighbors from the trust neighborhood which are not in the k_1 set. Then users in the combined neighborhood vote for their relevant items. $w(u,v)$ is defined as:

$$Vote_{u,i} = \sum_{v \in N_u^{(c)}} \sum_i w(u,v) \delta_{i \in I_v}, \tag{15}$$

where, $N_u^{(c)}$ is the combined neighborhood.

$$w(u,v) = \begin{cases} sim(u,v) & \text{if } v \in N_u \\ w_t(u,v) & \text{if } v \in N_u^{(t)} \end{cases} \tag{16}$$

In the basic version of *Trust-CF-ULF*, we set $k_1 = k_2$.

• **Trust-CF-ULF-best approach.** *Trust-CF-ULF-best* improves upon *Trust-CF-ULF* by dynamically tuning the value of k_1 and k_2 so as to obtain the best recall results.

The main differences between our proposed NN methods and Trust-cf are: 1) Our CF neighbors are derived from user latent features obtained from MF (i.e., *AllRank* [25]), which is expected to have higher top-k hit ratio than the Pearson correlation coefficient based only on observed ratings; 2) We use voting, instead of the weighted averaging of the observed ratings, to construct top-k recommendations. Voting can be treated as the simplest approach to consider all items with and without ratings. As will be shown in Section 4.2.4, our NN models outperform the existing social network based NN models.

4. EXPERIMENTS

In this section, we perform experiments for the proposed top-k RSes on Epinions[2] and Flixster[3] datasets. We focus on the *top-k hit ratio* or *recall* for testing recommendation accuracy (as motivated in Section 2). Concerning the three MF models, SoRec, STE and SocialMF (see Section 3.1), we used rank $j_0 = 10$, like in [8, 11, 12]. We find that Trust information significantly improves top-k hit ratio when incorporated properly both in MF and NN models. We also find that our proposed NN based RS and MF models trained with our modified objective function considerably outperform the existing top-k approach using social network information in recent literature [7]. Moreover, among the three models for combining rating data with social network information, the model with the worst performance concerning RMSE surprisingly turns out to achieve the best top-k hit ratio. This illustrates that approaches that work well for the vastly popular RMSE are not necessarily useful for optimizing the more realistic top-k hit ratio or recall.

4.1 Dealing with High Computation Cost

Training on all items admittedly increases the computation complexity, which is another key performance metric, other than accuracy, in designing recommender systems. Good recommendation algorithms not only need to provide accurate results, but also need to be scalable to large problems. To work with two large real-world datasets, we conducted our experiments on a Linux server with four E5640 Intel Xeon CPUs. Each CPU has four cores, and each core has 12.3 MB cache. The shared memory size is 12 GB. We implemented multi-thread C++ programs to parallelize large-scale matrix operations encountered in model training and parameter optimization. The running times for different models ranges from seconds to hours. For the STE model, we could not afford the computation cost to get the exact optimal solution, and we resorted to approximation methods. We found that the stochastic gradient descend and gradient descend methods easily got stuck in local minima when training with missing ratings, while ALS performed better in many cases.

4.2 Experiments on Epinions Dataset

4.2.1 Dataset

Epinions is a consumer opinion site where users review various items, such as cars, movies, books, software, etc., and assign ratings to the items. The ratings are in the range of 1(min) to 5(max). Users also assign trust values (i.e. a value of 1) to other users whose reviews and/or ratings they find valuable. No trust value indicates that a user either does not know the other, or distrusts him. We used the Epinions dataset[4] published by the authors of [26].

The Epinions data set consists of 71,002 users with a total number of 104,356 rated items. The total number of reviews is 571,235, and the total number of pairwise, directed trust relationships is 508,960. In our experiments, the data set is divided into two sub-sets: the training set and the test set. For users with less than five ratings, one randomly selected rating is put into the test set. For users with five ratings or more, 10% of the randomly selected ratings are moved to the test set. We define cold user as user who had rated fewer than 5 items. We further split the test set randomly into two disjoint sets of equal size. The first test set is used for cross-validation during training as to determine the tuning parameters in our objective function. The second test set is used as a truly held-out data set for final evaluation of the trained model. We report the result of testing the second test set. We consider 5-star ratings as relevant[5] to a user, i.e. the user definitely likes these items, and report the recall test results mostly for the *top 500* items (as defined in Section 2). The reason we set $k = 500$ is as follows. In the Epinions data set, there is a much larger number of items than users, which is different from many other data sets, e.g. the Netflix dataset. [6] Thus, using a small value for k will produce generally poor results for all the compared methods. Actually, we have performed experiments for $k = 5$. The recall of modified SoRec model on all users and cold users were 2.06% and 2.45% respectively, and the recall of modified *No Trust* on all users and cold users were 1.31% and 0.93% respectively. Nonetheless, we show the results of recall as the value of k changes in Figure 3.

4.2.2 Recall for MF Models

We found the following tuning parameters of the training objective functions for the MF models to result in the highest recall: $\lambda = 0.4$, $r_m = -1$, $w_m = 0.0002$ for all models; the optimal tradeoff between the rating data and the social network information is determined by the parameters β for SocialMF, α for STE, and γ (with $w_m^{(S)} = 0.00003, s_m = 0$) for SoRec. The results are shown in Figure 1. As expected, it is important to find the right tradeoff between the social network information and the rating data.

The recall test results for the optimal tuning parameters are summarized for all these MF models in Table 1. While all three models show an improvement in recall compared to *No Trust* case, it is particularly large for SoRec. The SoRec model with our modified training objective function outperforms *No Trust* by 23.1% in terms of overall recall and 101.8% in terms of cold-user recall. It shows that the social network is very helpful in terms of top-k recommendation, especially for recommendations for cold start users. Moreover, recall is even slightly higher for cold users than it is for all users. This may be explained by the fact that cold users have a slight tendency to rate popular items (i.e. items with a large number of ratings), which can naturally

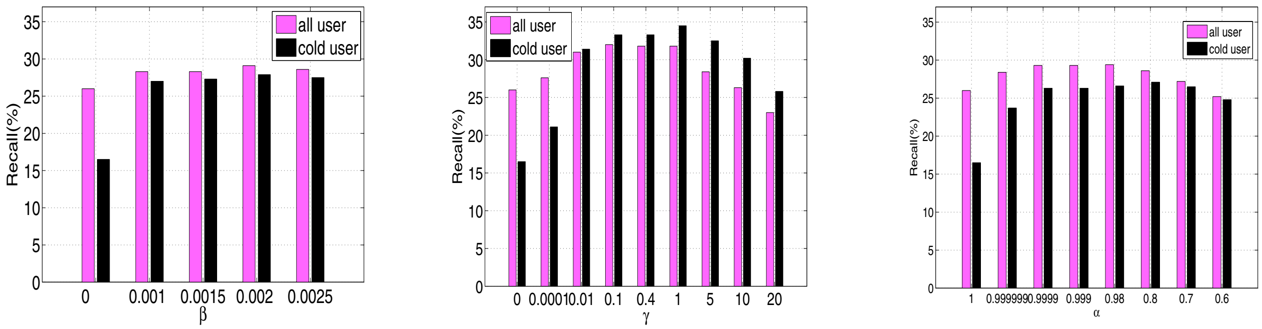
[2]http://www.epinions.com/

[3]http://www.flixster.com/

[4]http://alchemy.cs.washington.edu/data/epinions/

[5]Considering both 4 and 5 star ratings as relevant, experiments showed similar differences among the various approaches.

[6]http://en.wikipedia.org/wiki/Netflix_Prize



(a) SocialMF model (b) SoRec model (c) STE model

Figure 1: Top-500 Recall for social-network-based matrix-factorization models on Epinions data.

test users	MF models			
	No Trust	SocialMF	STE	SoRec
original training (on observed ratings)				
all	1.9%	3.5%	2.7%	2.6%
cold	1.5%	1.0%	2.8%	2.9%
modified training (on all ratings)				
all	26.0%	29.1%	29.4%	32.0%
cold	16.5%	27.9%	26.6%	33.3%

Table 1: Epinions data: recall (top 500) in percent for three MF models trained with original and modified training objective. 'No Trust' is the baseline MF model that only uses rating data.

be recommended more accurately. In the Epinions data, the average item rated by a cold user has received 102 ratings, while the average item rated by all users has received only 93 ratings. Table 1 shows that the SoRec model with our modified training objective function achieves the best recall. This may be unexpected, as the SoRec model was found to achieve a worse RMSE than STE in [11], and STE was found to have a worse RMSE than SocialMF in [8]. This result illustrates that the best way of combining rating data with social network information concerning the popular RMSE measure is not necessarily the best way to maximize recall.

4.2.3 RMSE for MF Models

Apart from optimizing for recall, we also determined the optimal tuning parameters as to minimize RMSE, and found $\lambda = 0.1$, $r_m = 4$, $w_m = 0$, $j_0 = 10$, which resulted in the following RMSE values:

- $RMSE = 1.174$, if only the rating data is used,
- $RMSE = 1.095$, for SocialMF (with $\beta = 20$),
- $RMSE = 1.157$, for STE (with $\alpha = 0.5$),
- $RMSE = 1.117$, for SoRec (with $\gamma = 50$ and $w_M^{(S)} = 0$).

These results are consistent with RMSE results in the literature [8, 11, 12]. It verifies that social network information is useful for improving RMSE.

4.2.4 Recall for NN Models

As a further comparison, Figures 2 shows the recall test results we obtained for various nearest neighbor models, which are outlined in Section 3.2. To the best of our knowledge, this includes *Trust-cf*, the only top-k approach using social network information [7]. In the Trust-cf model, k_1 is set to be 5 which leads to the best recall in user-based CF. The top-500 recommendation result on the Epinions dataset of Trust-cf is shown in Figure 2(c).

Among the NN approaches, the one in Figure 2(c) achieves a considerably worse hit ratio than any of the NN approaches in Figure 2(a) and 2(b). This poor performance of the *Trust-cf* approach is due to the following reason: *Trust-cf* predicts the rating value of a user in terms of the average rating values of the user's friends–which is obviously based on the *observed ratings only*. In contrast, the various NN approaches in Figure 2(a) and 2(b) use voting–which is the simplest possible way of accounting for *all items*, i.e. by counting 0 for an absent rating and counting 1 for an observed relevant rating (with weights defines in Section 3.2). As the rating value is ignored, this is the simplest possible approach to account for all items during training. Though recall of NN based RS is not as good as MF based RS. As we mentioned in Section 3.2, NN based approach has the advantage of integrating new feedbacks incrementally while MF based approach not able to.

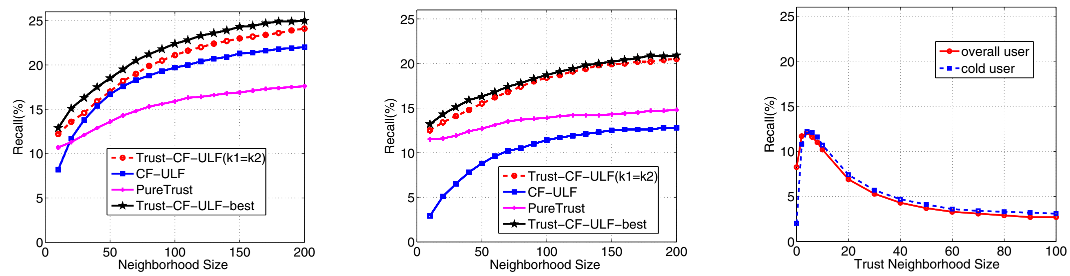
4.3 Experiments on Flixster Dataset

Flixster is a social network site where users add other users to their friend lists to form a social network. Flixster data has about one million users, who rate movies and share reviews. The ratings in Flixster have ten discrete values from 0.5 to 5, with step size of 0.5. Flixster is different from Epinions in that social relations in Flixster are bi-directional. The Flixster data [7] used here is from [8]. The Flixster social network has 26.7 million connections. The trace consists of 8.2 million movie ratings on 49,000 movies and 1 million users. The number of users who made at least one rating is 150,000. Despite the different properties of the Epinions and Flixster data sets, the results on the Flixster data confirm our results on the Epinions data. The result on Flixster dataset is very similar to Epinions dataset. Due to space limit, we cannot present all results here. The complete results on Flixster dataset is available in our technical report [8].

We split the data into a training set and a disjoint test set. For users with less than 10 ratings, we randomly choose one rating and put it into test set. For users with 10 or more than 10 ratings, we randomly chose 5% as put them into the test set. We further split the test set randomly into two disjoint

[7] http://www.sfu.ca/ sja25/datasets/

[8] `http://eeweb.poly.edu/faculty/yongliu/docs/topk_tr.pdf`



(a) Nearest Neighbor: all users (b) Nearest Neighbor:cold users (c) Trust-cf: $k_1 = 5$, varies k_2

Figure 2: Top-500 Recall by Nearest Neighbor based Models on Epinions dataset

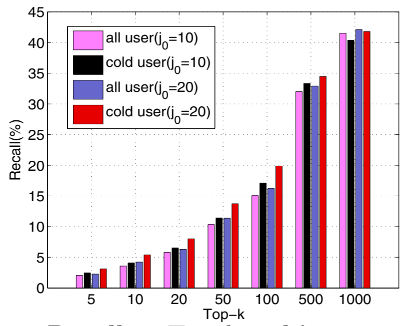


Figure 3: Recall vs Top-k and impact of Dimensionality on Epinions data.

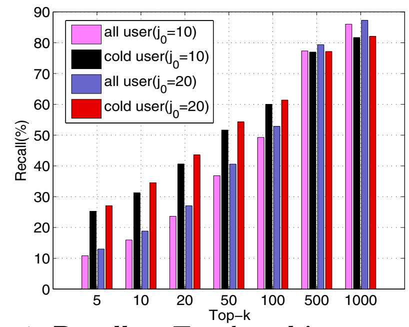


Figure 4: Recall vs Top-k and impact of Dimensionality on Flixster data.

sets of equal size and report results of testing the second test set akin to Epinions case. We defined rating values of 4 or larger as relevant for computing recall on the test set. We report recall for the top 100 items recommendation. The top-k hit ratio of different k value is presented in Figure 4.

The optimal values of tuning parameters are: $\lambda = 0.1$, $r_m = 1.0$ and $w_m = 0.2$. The optimal β is 0.1 in modified SocialMF model, and the optimal α is 0.99 in STE model. The optimal γ is 0.05 (with $w_m^{(S)} = 0.2$,$s_m = 0$) in modified SoRec model. For these optimal training parameters, the recall test results are summarized for all the MF models in Table 2.

test users	MF models			
	No Trust	SocialMF	STE	SoRec
original training (on observed ratings)				
all	4.4%	4.7%	5.3%	8.2%
cold	6.3%	6.6%	7.2%	15.4%
modified training (on all ratings)				
all	44.3%	45.2%	47.1%	49.1%
cold	30.8%	38.3%	47.6%	59.2%

Table 2: Flixster data: recall (top 100) in percent for three MF models trained with original and modified training objective. 'No Trust' is the baseline MF model that only uses rating data.

As before, SoRec with modified training objective function achieves the largest recall. We can see from Table 2 that SoRec model with our modified training objective function outperforms *No Trust* by 10.8% in terms of overall recall and 92.2% in terms of cold user recall. It again shows that social network is very helpful in terms of top-k recommendation especially for recommendation of cold start users. Note that the improvement for the cold users over all the users is particularly pronounced for the Flixtser data, as cold users have a rather strong tendency to rate popular items. In the Flixster data, the average item rated by a cold user has received 9,601 ratings, while the average item rated by all users has received only 5,367 ratings.

4.4 Impact of Dimensionality and Top-k

Figure 3 and 4 depict recall vs. the top-k number with dimensionality $j_0 = 10$ and $j_0 = 20$ for Epinions data and Flixster data respectively. We can see from Figure 3 and 4 that dimensionality $j_0 = 20$ performs better than $j_0 = 10$. This is because larger dimensionality captures more latent features of users and items and hence improves recall. It should be noted that top-k hit ratio of Flixster data is much better than Epinions data. Counting that the number of items in Epinions dataset is about twice of Flixster dataset, still, we find that recall of Flixster is more than twice of Epinions for top-5 to top-500 recommendations. This is probably because of the fact that Epinions data is a multi-category data which contains items from many categories (cars, movies, books, software, etc.), while items in Flixster are all movies which makes the recommendation easier in general. Furthermore, users in Flixster dataset averagely have more social connections and item ratings compared to Epinions dataset.

5. CONCLUSIONS

Social recommendation is prevalent in the real-world, but top-k recommendation using online social networks has been insufficiently studied in the recommendation literature. In this paper, we presented a comprehensive study on improving the accuracy of top-k recommendation using trust information derived from social networks. We showed that the existing social-network based recommender-systems can be conveniently tailored for top-k recommendations by modifying their training objective functions to account for both observed ratings and missing ratings. Through experiments on two large-scale data sets, we made three majors findings: 1)Trust information significantly improves the top-k hit-ratio when incorporated properly both in MF and NN models; 2)Our proposed social network based NN models and RS trained with our modified objective function considerably outperform the only published top-k approach using social network information in recent literature [7], an NN approach; 3) surprisingly, we found that among the various ways of combing feedback data with social network information, the one that was found to be worst with respect to RMSE turns out to be the best concerning the top-k hit ratio, and vice versa. This illustrates that the technical details of minimizing RMSE can be very different from the ones that work to optimize the (more realistic) top-k hit ratio. Our work demonstrated that top-k recommendations pose unique challenges, and social trust information, when incorporated properly, can significantly improve the hit ratio of top-k recommendations.

6. ACKNOWLEDGMENTS

The authors would like to thank Debasis Mitra and Shiv Panwar for initiating this collaboration. Harald Steck would also like to thank Tin Kam Ho for her encouragement and support of this work. Xiwang Yang and Yong Liu were partially supported by USA NSF grant CNS-1018032 to conduct this work.

7. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17:734–49, 2005.

[2] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-N recommendation tasks. In *ACM Conference on Recommender Systems*, pages 39–46, 2010.

[3] S. Funk. Netflix update: Try this at home, 2006. `http://sifter.org/ simon/journal/20061211.html`.

[4] J. A. Golbeck. Computing and applying trust in web-based social networks, 2005.

[5] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *IEEE International Conference on Data Mining (ICDM'08).*

[6] M. Jamali and M. Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2009.

[7] M. Jamali and M. Ester. Using a trust network to improve top-n recommendation. In *ACM Conference on Recommender Systems (RecSys)*, 2009.

[8] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *ACM Conference on Recommender Systems (RecSys)*, 2010.

[9] R. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 11:2057–78, 2010.

[10] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 426–34, 2008.

[11] H. Ma, I. King, and M. R. Lyu. Learning to recommend with social trust ensemble. In *ACM conference on Research and development in information retrieval (SIGIR)*, 2009.

[12] H. Ma, H. Yang, M. R. Lyu, and I. King. Sorec: Social recommendation using probabilistic matrix factorization. In *International Conference on Information and Knowledge Management (CIKM'08).*

[13] B. Marlin and R. Zemel. Collaborative prediction and ranking with non-random missing data. In *ACM Conference on Recommender Systems (RecSys)*, 2009.

[14] B. Marlin, R. Zemel, S. Roweis, and M. Slaney. Collaborative filtering and the missing at random assumption. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

[15] P. Massa and P. Avesani. Trust-aware recommender systems. In *ACM Conference on Recommender Systems (RecSys)*, 2007.

[16] R. Pan, Y. Zhou, B. Cao, N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *IEEE International Conference on Data Mining (ICDM)*, 2008.

[17] X. Yang, Y. Guo and Y. Liu. "Bayesian-inference based recommendation in online social networks". *The 30th Annual IEEE International Conference on Computer Communications (INFOCOM)*, 2011.

[18] X. Yang, H. Steck and Y. Liu. Circle-based Recommendation in Online Social Networks. *Proc. of 2012 ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'12).*

[19] Q. Yuan, S. Zhao, L. Chen, Y. Liu , S. Ding, X. Zhang and W. Zheng. Augmenting collaborative recommender by fusing explicit social relationships. Recommender Systems & the Social Web workshop, RecSys, 2009.

[20] V. Vasuki, N. Natarajan, Z. Lu and I. Dhillon. Affiliation Recommendation using Auxiliary Networks. In *ACM Conference on Recommender Systems (RecSys)*, 2010.

[21] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *KDDCup*, 2007.

[22] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *International Conference on Machine Learning (ICML)*, 2007.

[23] R. Salakhutdinov and N. Srebro. Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. In *Advances in Neural Information Processing Systems 24 (NIPS)*, 2010.

[24] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *International Conference on Machine Learning (ICML)*, pages 720–7, 2003.

[25] H. Steck. Training and testing of recommender systems on data missing not at random. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 713–22, 2010.

[26] M. Richardson and P. Domingos. Mining Knowledge-Sharing Sites for Viral Marketing, In ACM Conference on Knowledge Discovery and Data mining, 2002.