Information Sciences 374 (2016) 100–114

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Improving top-K recommendation with truster and trustee relationship in user trust network

Chanyoung Park, Donghyun Kim, Jinoh Oh, Hwanjo Yu*

Department of Computer Science and Engineering, POSTECH (Pohang University of Science and Technology), 77 Cheongam-Ro. Nam-Gu., Pohang, Gyeongbuk 37673, Republic of Korea

A R T I C L E I N F O

Article history:
Received 19 April 2016
Revised 28 August 2016
Accepted 9 September 2016
Available online 15 September 2016

Keywords:
Recommender system
Learning-to-Rank
Social network

A B S T R A C T

Due to the data sparsity problem, social network information is often additionally used to improve the performance of recommender systems. While most existing works exploit social information to reduce the *rating prediction error*, e.g., RMSE, a few had aimed to improve the *top-k ranking prediction accuracy*. This paper proposes a novel *top-k ranking* oriented recommendation method, TRecSo, which incorporates social information into recommendation by modeling two different roles of users as trusters and trustees while considering the structural information of the network. Empirical studies on real-world datasets demonstrate that TRecSo leads to a remarkable improvement compared with previous methods in top-k recommendation.

1. Introduction

Due to the ever increasing amount of information available to users on the Web, recommender systems have recently gained their popularity in industrial companies such as Amazon[1], Netflix[2], Facebook[3], to name a few. Indeed, it has been reported that Amazon are generating about 35% of their revenue through personalized recommendations provided by their own recommendation algorithms [21]. Moreover, recommender systems are widely used to help academic researchers in their decision making process when searching for relevant scientific articles [37,38,41]. Hence, improving the quality of recommender systems has been an attractive research problem for academic researchers.

In the beginning, a significant amount of research was devoted to accurately predicting the overall ratings to reduce the rating prediction error, e.g., RMSE, MAE [2,13]. Given a user-item rating matrix, their main goal was to provide the correct rating that a user will give to an item. However, rather than better predicted ratings, users are more interested in seeing a list of top-k items that aligns with their preferences. Moreover, it is well known that minimizing the rating prediction error does not always result in better top-k list of items [6,46]. Consequently, the Learning-To-Rank (LTR) [16] method, a supervised machine learning method that directly builds a ranking list from training data without an intermediate step of the rating prediction, has gained popularity to provide accurate results at top-k.

Despite the success of recommendation approaches, recommender systems suffer from an inherent limitation called the *data sparsity problem*, that is, the recommendation is hardly accurate due to lack of observations (i.e., ratings) because users

* Corresponding author.
E-mail addresses: pcy1302@postech.ac.kr (C. Park), kdh5377@postech.ac.kr (D. Kim), kurin@postech.ac.kr (J. Oh), hwanjoyu@postech.ac.kr (H. Yu).

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

http://dx.doi.org/10.1016/j.ins.2016.09.024
0020-0255/© 2016 Elsevier Inc. All rights reserved.

typically rate a small number of items. To tackle the data sparsity problem, researchers have tried to incorporate auxiliary information such as social network relationship among users [7–9,17,19], text reviews on items [15,23,41], time related information [1,14,43] and items' quality information [28,37,38]. Specifically, this paper focuses on incorporating the social network information of users in the top-k recommendation.

Recently, two top-k recommendation methods have been developed to incorporate the social network information based on the LTR approach. Specifically, Yao et al. [48] linearly combine a user's taste and her direct friends' tastes in optimizing the top-k recommendation. However, they do not utilize other important information hidden in the social network such as the structural information or truster-trustee relationship [45]. Zhao et al. [49] optimize the top-k recommendation from relative ordering that can be extracted from purchase history or browsing history, but they cannot handle numerical ratings directly. Note that numerical ratings usually contain much richer information on users preference than relative ordering.

This paper proposes a novel LTR-based top-k recommendation method, TRecSo, which leverages the social network information to optimize top-k recommendation. TRecSo is distinguished from previous methods in that it models two different roles of users as trusters and trustees while considering the structural information of the network. Precisely, we map users into two types of low dimensional spaces according to their roles, that is, *truster* space and *trustee* space under the following assumptions:

- When "*user A*" is given several choices of items, he turns to people he trusts to ask them their opinions about the items.
- Consequently, the behavior of "*user A*" will influence the people that trust "*user A*".

We summarize our contributions as follows:

- We develop a novel Learning-To-Rank (LTR) based top-k recommendation method that takes into account the social network information among users to alleviate the data sparsity problem.
- We map users into two types of low dimensional spaces according to their roles while considering the structural information of the trust network.
- Our experimental results on three real-world datasets indicate that our method considerably outperforms previous methods in top-k recommendation.

The remainder of this paper is organized as follows. Section 2 describes the related work; The problem is formally discussed in Section 3; In Section 4, we describe our proposed method, TRecSo, and demonstrate the learning algorithm; In Section 5, we describe the complexity of our proposed method followed by experimental results in Section 6. Finally in Section 7, we conclude the paper.

2. Related work

Triggered by the Netflix prize [2] that ended in 2009, the field of recommender system has seen rapid progress since then. In this section, we review the existing methods including traditional collaborative filtering methods, *social* network based recommender systems, top-k ranking oriented recommender systems and top-k ranking oriented *social* recommender systems, the most related work to ours.

2.1. Traditional collaborative filtering

Typically, traditional collaborative filtering (CF) methods can be divided into two major categories, memory-based CF and model-based CF. Memory-based CF uses similarity measures such as Pearson Correlation or Cosine Similarity between users or items, and recommend items based on the neighbors [31,50]. While gaining much popularity due to its easy implementation and interpretation, it does not scale well to large datasets and the performance decreases as ratings become sparse. Model-based CF, on the other hand, instead of simply computing similarities, learns a parametric model to fit the user-item ratings, which generally results in a better performance than memory-based CF [33]. Among various model-based methods [3,10,39], matrix factorization [13,26] is the most widely used method upon which our proposed method, TRecSo, is built.

2.2. Social recommender system

Due to the inherent nature of recommender system, where users only rate a small number of items, the user-item ratings data is typically very sparse, that is, the entries are mostly unknown. Recently, social network based recommender systems have gained spotlight due to the rapid growth of social network services including Facebook[4], Twitter[5]. In a way akin to the traditional CF, social recommender systems are also categorized [36] into memory-based method [11,22] and model-based method. Moreover, model-based methods are further divided into three different categories: Matrix co-factorization methods [17,34], ensemble methods [18,35] and regularization methods [12,19,47].

[4] http://www.facebook.com/.
[5] http://www.twitter.com/.

As the most representative memory-based social recommender approach, Jamali et al. [11] combine the memory-based CF based approach and the social network based approach under the random walk framework. They show that the prediction accuracy is improved by preferring raters with a shorter distance. Moreover, Ma et al. [17] were first to incorporate users' social network information into a matrix co-factorization based recommender system. They propose a probabilistic model that fuses the user-item rating data with the users' social network data by sharing a common user-specific latent factor and formulate the objective function as follows:

$$\min \sum_{i=1}\sum_{j=1}\left(r_{ij} - p_i^T q_j\right)^2 + \sum_{i=1}\sum_{k=1}\left(s_{ij} - p_i^T z_k\right)^2 \tag{1}$$

where $p_i \in \mathbb{R}^K$, $q_j \in \mathbb{R}^K$ and $z_k \in \mathbb{R}^K$ denote user-specific, item-specific latent vectors and factor-specific latent vectors, respectively; r_{ij} denotes the rating of user i on item j, and s_{jk} is equal to 1 if user j and user k are socially related, and 0 otherwise. In addition, Ma et al. [18] proposed an extended method that linearly combines a user's taste and her friends' tastes for modeling ratings as:

$$\hat{r}_{ij} = \alpha p_i^T q_j + (1-\alpha)\sum_{k\in T_i}\tilde{s}_{ik} p_k^T q_j \tag{2}$$

where $\tilde{s}_{ik}$ represents the normalized trust strength between a user i and user k; α balances between user's taste and his friends' tastes and T_i denotes the social friends of user i. Finally, Ma et al. [19] propose a regularization method under the assumption that a user's preference should be similar to that of his social friends', so they formulate the objective function by forcing the user latent feature vectors to be close to those of his friends' as follows:

$$\min \sum_{i}\sum_{j}\left(r_{ij} - p_i^T q_j\right)^2 + \alpha \sum_{i}\left(p_i - \sum_{k\in T_i} s_{ik} p_k\right)^2 \tag{3}$$

where α controls the importance of the social regularization.

However, while the vast majority of commercial recommender systems provide recommended item lists to users because most users are only interested in seeing top-k items, the aforementioned social recommender systems mainly focus on minimizing the rating prediction error., e.g., RMSE, MAE. As a matter of fact, improving such traditional error criteria does not necessarily lead to improving top-k performance [6,24,46]. Therefore, in this paper we focus on finding a better top-k list of items rather than correctly predicting the ratings.

2.3. Top-k ranking oriented recommender system

As mentioned in the previous section, the eventual goal of a recommender system is to provide a top-k list of items as a recommendation. Consequently, several approaches have been proposed with the objective of top-k recommendation, which cast this problem as Learning-to-Rank (LTR) [16] where a personalized ranking list is directly generated from the training data without an intermediate step of the rating prediction. LTR based top-k recommendation approaches are categorized into pair-wise and list-wise models. Pair-wise models learn users' relative preferences of each item pair [27,29]. Although pair-wise models have shown substantial improvements in terms of top-k recommendation, they have issues with high computational complexity. Most recently, list-wise models have gained much attentions due to its scalability compared with pair-wise models [32,44]. Given a list of items as a training sample, list-wise models directly predict a ranking list of items for each user based on the distance between the ground truth ranking list and the predicted list. In this paper, we adopt the list-wise approach and demonstrate the superiority of our proposed model, TRecSo, over the relevant pair-wise and list-wise competitors.

2.4. Top-k ranking oriented social recommender system

Approaches that belong to this section are considered as direct baselines to our model, TRecSo. In this section, we introduce two state-of-the-art methods [48,49] and address their limitations.

Zhao et al. [49] propose a pair-wise LTR approach based on the assumption that users tend to show a higher preference to items that their friends prefer than items that they are not aware of at all. However, their approach cannot handle numerical ratings directly and they have high computational complexity due to the inherent nature of pair-wise LTR approach. Moreover, Yao et al. [48] adopt the rating model of [18], and linearly combine a user's taste and her friends' tastes for modeling ratings. Given the rating model as Eq. (2), they optimize the top-k recommendation lists by using a concept called *top-one probability*, which will be explained in Section 3 for later use in our model. However, they fail to utilize other important information hidden in social network such as the structural information and the truster-trustee relationship. In this paper, we demonstrate that our approach, TRecSo, outperforms these two state-of-the-art approaches over three real-world datasets including datasets used in [48] and [49].

3. Problem description

We first introduce the notations that we use throughout the paper. Let $\mathcal{U} = \{u_1, u_2, \dots, u_N\}$ be the set of users and $\mathcal{V} = \{v_1, v_2, \dots, v_M\}$ be the set of items, where N and M are number of users and items, respectively. The ratings given by users in $\mathcal{U}$ on items in $\mathcal{V}$ are represented by a rating matrix $\mathcal{R} = [r_{ij}]_{N\times M}$, where r_{ij} denotes the rating of u_i on v_j. Depending on applications, r_{ij} can be either a real number or a binary value. When users explicitly express their opinions on items, the rating value is a real number, often in range [1,5], and when $\mathcal{R}$ reflects users action, such as click or non-click and bookmark or non-bookmark, r_{ij} is a binary value. Although this paper focuses on the former case, it can be extended to the latter case as well. Without loss of generality, we convert the ratings of $1\dots5$ into the interval [0,1] by normalization. In addition, $\mathcal{S} = [s_{ik}]_{N\times N}$ is the user trust matrix, where $s_{ik} = 1$ denotes that u_i trusts u_k. It is worth noting that the matrix $\mathcal{S}$ is asymmetric, that is, $s_{ik} \neq s_{ki}$, which means that the fact that u_i trusts u_k does not imply that u_k trusts u_i. Epinion[6], Ciao[7] and Twitter[8] are the most representative applications that provide asymmetric relationship between users, and such asymmetric relationship is the generalized version of symmetric relationship. i.e, Friendship. Therefore, our method can be certainly applied to the case of symmetric relationship. With the aforementioned notations, we can formally specify our problem as follows:

Problem definition

Given: *The observed rating matrix $\mathcal{R}$ and the trust matrix $\mathcal{S}$*

Goal: *Recommend each user a ranking list of unobserved items considering their personal preferences.*

4. Method

In Section 4.1, we briefly explain the concept of *Plackett-Luce model* that is required for understanding our proposed model, TRecSo. Then, we explain how the rating and the trust relationship are modeled in Sections 4.2 and 4.3, respectively, and propose a unified model that combines the rating and the trust in Section 4.4, where we show how the unified model is learned.

4.1. Preliminary: Plackett-Luce model

Plackett-Luce model [20] computes the probability distribution over the permutations of items that each user has rated. The underlying assumption is that different permutations of items have different probability distributions, and a high permutation probability value implies that the permutation is more likely to be preferred by the user.

Probability of permutation: Given a set of M items $\mathcal{V}$, let $\pi = \{\pi_1, \pi_2, \dots, \pi_M\}$ be one of the possible permutations of $\mathcal{V}$. If its corresponding ratings are given by $\{r_{\pi_1}, r_{\pi_2}, \dots, r_{\pi_M}\}$, the probability of the permutation π is defined as follows:

$$P(\pi) = \prod_{i=1}^{M} \frac{\phi(r_{\pi_i})}{\sum_{k=i}^{M} \phi(r_{\pi_k})}$$

where r_{π_i} is the rating given to item π_i and $\phi(r) = e^r$. However, the time complexity of such computation is too high, since the number of different permutations is $M!$ for a set of M items. Therefore, to alleviate such problem, we employ the probability of only the top-ranked item among a given ranked list as follows:

$$P(\pi_i) = \frac{\phi(r_{\pi_i})}{\sum_{j=1}^{k} \phi(r_{\pi_j})} \tag{4}$$

Eq. (4) models the probability of an item scored r_{π_i} being ranked on the top-one position in the list and we call it *top-one probability*.

4.2. Modeling rating

Due to the asymmetry property of the trust matrix as mentioned in Section 3, we map each user into two different latent vectors, each of which represents truster or trustee. Such mappings are reasonable under the assumption that when a user is given several choices of items, he asks the people he trusts for their opinions about the items, and also the decision made by the user will influence the people that trust the user. Then, the rating of u_i on v_j is predicted as follows:

$$\hat{r}_{ij} = g\left(\mu + b_{u_i} + b_{v_j} + {q_j}^T\left(\alpha p_i + (1-\alpha)w_i + |I_i|^{-\frac{1}{2}} \sum_{t\in I_i} y_t + |T_i|^{-\frac{1}{2}} \sum_{v\in T_i} x_v\right)\right) \tag{5}$$

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

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

[8] http://www.twitter.com.

where $g(\cdot)$ is the logistic function that bounds the range of predicted ratings into [0,1]; μ is the average of all ratings; b_{u_i} and b_{v_j} represent the user and item biases for u_i and v_j, respectively. User biases and item biases should be taken into consideration because some users tend to give higher ratings than others, and some items (e.g., famous items) tend to receive higher ratings than others. Note that it is well known from the result of the Netflix Prize [2,13] that incorporating the concept of biases significantly boosts the performance of the matrix factorization based recommender system. q_j represents the latent vector of item j; p_i and w_i represent the user latent vectors as a truster and as a trustee role of u_i, respectively, which are also used to model the social information in Eq. (7). In other words, a user latent vector is modeled by a linear combination of truster specific latent vector and trustee specific latent vector, since a user plays a role of truster and trustee at the same time. α balances between the truster and trustee roles. (the higher the value of α, the more weight is given to the truster role of a user); I_i denotes the set of items rated by u_i; y_t is the latent vector of item t, which models implicit influence of items rated by u_i; T_i is the set of users that u_i trusts (e.g., whom u_i follows in social network); and x_v is the latent vector of whom u_i trusts, which models implicit influence of the users trusted by u_i. It is worth mentioning that although we do not explicitly model the trust propagation in our model, we include the implicit influence of social friends ($|T_i|^{-\frac{1}{2}} \sum_{v \in T_i} x_v$) of user i so that users with only a few ratings are rather modeled by their social friends. Note that this way of modeling the rating value is distinguished from existing methods [9,13] in that none of them integrated the concept of bias and two roles (Truster and Trustee) of users while at the same time considering the implicit influence of the users trusted by u_i.

By adopting the *top-one probability* in Eq. (4), we are now able to formulate the objective function aiming at minimizing the uncertainty between the training list and the predicted list of ratings by using cross-entropy measure as follows, which can be interpreted as list-wise ranking prediction:

$$\begin{aligned}\mathcal{L}(b_U, b_V, p, w, q, y, x) = &-\sum_{i=1}^{N} \sum_{j \in I_i} P_{l_i}(r_{ij}) \log P_{l_i}(\hat{r}_{ij}) + \frac{\lambda_p}{2}||p_i||_F^2 + \frac{\lambda_w}{2}||w_i||_F^2 \\ &+ \frac{\lambda_v}{2}||q_j||_F^2 + \frac{\lambda_c}{2}\left(\sum_{t \in I_i}||y_t||_F^2 + \sum_{v \in T_i}||x_v||_F^2\right) \\ &+ \frac{\lambda_b}{2}\left(||b_{u_i}||_F^2 + ||b_{v_j}||_F^2\right) \end{aligned} \tag{6}$$

where $P_{l_i}(r_{ij})$ models the probability of an item scored r_{ij} being ranked on the top-one position in u_i's ranked list l_i; λ_p, λ_w, λ_v, λ_c and λ_b are regularization parameter for truster vector, trustee vector, item vector, implicit vectors and bias vectors, respectively. Note that the last five terms are regularization terms to avoid model over-fitting.

4.3. Modeling trust

Intuitively, the unknown relationship $\hat{s}_{ik}$ between u_i and u_k can be estimated as follows:

$$\hat{s}_{ik} = g(b_{p_i} + b_{w_k} + w_k^T p_i) \tag{7}$$

where b_{p_i} and b_{w_k} represent the truster bias and trustee bias, respectively. By sharing the term p_i and w_k in Eq. (5) and Eq. (7), and by *simultaneously learning both latent models*, we are able to properly model the different roles of users as trusters and trustees.

To reflect the structural information of the network, s_{ik} is adjusted based on the degree of nodes such that it gives lower weights to those who *trust* many users (large out-degrees) and gives higher weights to those who *are trusted* by many users (large in-degrees):

$$s_{ik}^* = \sqrt{\frac{Indegree(v_k)}{Outdegree(v_i) + Indegree(v_k)}} \times s_{ik} \tag{8}$$

where v_i and v_k are nodes for u_i and u_k in the network, respectively [17]; $Indegree(v)$ and $Outdegree(v)$ are links that come into the node v and links that go out from the node v. It is worth noting that the model performance has been in fact improved by this adjustment and it is demonstrated in Section 6.5.

Likewise, the objective function for minimizing the cross-entropy between the probability distribution of training list and the predicted list of trust values can be formulated as follows:

$$\begin{aligned}\mathcal{L}(b_p, b_w, p, w) = &-\sum_{i=1}^{N} \sum_{k \in T_i} P_{l_i}(s_{ik}^*) \log P_{l_i}(\hat{s}_{ik}) + \frac{\lambda_b}{2}\left(||b_{p_i}||_F^2 + ||b_{w_i}||_F^2\right) \\ &+ \frac{\lambda_p}{2}||p_i||_F^2 + \frac{\lambda_w}{2}||w_i||_F^2 \end{aligned} \tag{9}$$

where $P_{l_i}(s_{ik})$ models the probability of trust strength s_{ik} being ranked on the top-one position in u_is ranked list l_i. Note that the last three terms are regularization terms to avoid the model over-fitting.

4.4. Unified model

So far, we have described how the ratings and the trust relations are modeled in TRecSo. In this section, we demonstrate the unified model that integrates the two models, and we show how the parameters are learned.

In order to jointly model the ratings and the social relationships among users, we formulate a unified model by combining Eq. (6) and Eq. (9), which is given as follows:

$$\begin{aligned}\mathcal{L} = &-\sum_{i=1}^{N}\sum_{j\in I_i} P_{l_i}(r_{ij})\log P_{l_i}(\hat{r}_{ij}) - \lambda_t\sum_{i=1}^{N}\sum_{k\in T_i} P_{l_i}(s^*_{ik})\log P_{l_i}(\hat{s}_{ik})\\ &+\frac{\lambda_p}{2}||p_i||_F^2+\frac{\lambda_w}{2}||w_i||_F^2+\frac{\lambda_v}{2}||q_j||_F^2+\frac{\lambda_c}{2}\left(\sum_{t\in I_i}||y_t||_F^2+\sum_{v\in T_i}||x_v||_F^2\right)\\ &+\frac{\lambda_b}{2}(|||b_{u_i}||_F^2+||b_{v_j}||_F^2+|b_{p_i}||_F^2+||b_{w_i}||_F^2)\end{aligned} \tag{10}$$

where λ_t controls the importance of trust regularization, which means that the higher the λ_t, the more impact a user's trusters have on the user's preference. Note that in addition to the regularization terms in Eq. (10), we adopt weighted-λ-regularization [51] to further avoid model over-fitting. Moreover, in order to reduce the model complexity, we set $\lambda_p = \lambda_w = \lambda_v = \lambda_c = \lambda$.

Optimization procedure: Having formulated the non-convex objective function as shown Eq. (10), we compute the gradient of each latent vector, i.e., $p_i, w_k, q_j, b_{u_i}, b_{p_i}, b_{w_k}, b_{q_j}, y_t, x_v$, and learn them by stochastic gradient descent [25] from which we obtain the local minimum solution. The gradients are given as follows:

$$\frac{\partial\mathcal{L}}{\partial p_i} = \alpha\sum_{k\in I_i} e_{ij}\cdot q_j + \lambda_t\sum_{v\in T_i} e_{iv}\cdot w_v + (\lambda+\lambda_t)\cdot p_i \tag{11}$$

$$\frac{\partial\mathcal{L}}{\partial w_k} = (1-\alpha)\sum_{j\in I_k} e_{ik}\cdot q_j + \lambda_t\sum_{i\in T_k} e_{ik}\cdot p_i + (\lambda+\lambda_t)\cdot w_k \tag{12}$$

$$\frac{\partial\mathcal{L}}{\partial q_j} = \alpha\sum_{i\in U_j} e_{ij}\left(\alpha p_i + (1-\alpha)w_i + |I_i|^{-\frac{1}{2}}\sum_{t\in I_i} y_t + |T_i|^{-\frac{1}{2}}\sum_{v\in T_i} x_v\right)q_j + \lambda q_j \tag{13}$$

$$\frac{\partial\mathcal{L}}{\partial b_{u_i}} = \sum_{j\in I_i} e_{ij} + \lambda b_{u_i} \tag{14}$$

$$\frac{\partial\mathcal{L}}{\partial b_{p_i}} = \lambda_t\sum_{v\in T_i} e_{iv} + \lambda_t b_{p_i} \tag{15}$$

$$\frac{\partial\mathcal{L}}{\partial b_{w_k}} = \lambda_t\sum_{i\in T_k} e_{ik} + \lambda_t b_{w_k} \tag{16}$$

$$\frac{\partial\mathcal{L}}{\partial b_{q_j}} = \sum_{i\in U_j} e_{ij} + \lambda b_{q_j} \tag{17}$$

$$^{\forall}t\in I_i,\ \frac{\partial\mathcal{L}}{\partial y_t} = \sum_{j\in I_i} e_{ij}|I_i|^{-\frac{1}{2}}q_j + \lambda y_t \tag{18}$$

$$^{\forall}v\in T_i,\ \frac{\partial\mathcal{L}}{\partial x_v} = \sum_{j\in I_i} e_{ij}|T_i|^{-\frac{1}{2}}q_j + \lambda x_v \tag{19}$$

$$\text{where } e_{ij} = \left(\frac{\exp(g(\hat{r}_{ij}))}{\sum_{k\in I_i}\exp(g(\hat{r}_{ik}))} - \frac{\exp(r_{ij})}{\sum_{k\in I_i}\exp(r_{ik})}\right)g\prime(\hat{r}_{ij})$$

$$e_{ik} = \left(\frac{\exp(g(\hat{s}_{ik}))}{\sum_{k\in I_i}\exp(g(\hat{s}_{ik}))} - \frac{\exp(s^*_{ik})}{\sum_{k\in I_i}\exp(s^*_{ik})}\right)g\prime(\hat{s}_{ik}).$$

Note that $g\prime(x) = \exp(x)/(1+\exp(x))^2$ is the derivative of logistic function $g(x)$. The detailed steps of TRecSo are shown in Algorithm 1 and the notations are explained in Table 1. Note that the algorithm terminates when the difference in loss between two iterations is less than 0.000001 or when reaching the predetermined number of iterations.

Algorithm 1 TRecSo algorithm

```
Input: R: User-Item rating matrix, S: User-User trust relation matrix
Output: Learned model parameters $\mathbf{p}, \mathbf{w}, \mathbf{q}, \mathbf{b_u}, \mathbf{b_p}, \mathbf{b_w}, \mathbf{b_q}, \mathbf{y}, \mathbf{x}$
1:  repeat
2:    for $i = 1$ to $N$ do
3:      for $j = 1$ to $M$ do
4:        Calculate $\frac{\partial\mathcal{L}}{\partial p_i}, \frac{\partial\mathcal{L}}{\partial w_i}, \frac{\partial\mathcal{L}}{\partial q_j}, \frac{\partial\mathcal{L}}{\partial b_{u_i}}, \frac{\partial\mathcal{L}}{\partial b_{q_j}}, \frac{\partial\mathcal{L}}{\partial y_t}, \frac{\partial\mathcal{L}}{\partial x_v}$ for all $t \in I_i$ and $v \in T_i$ using Eq. (11), (12), (13), (14), (17), (18), (19)
5:        Update $p_i \leftarrow p_i - \gamma \frac{\partial\mathcal{L}}{\partial p_i}$
6:        Update $w_i \leftarrow w_i - \gamma \frac{\partial\mathcal{L}}{\partial w_i}$
7:        Update $q_j \leftarrow q_j - \gamma \frac{\partial\mathcal{L}}{\partial q_j}$
8:        Update $b_{u_i} \leftarrow b_{u_i} - \gamma \frac{\partial\mathcal{L}}{\partial b_{u_i}}$
9:        Update $b_{q_j} \leftarrow b_{q_j} - \gamma \frac{\partial v}{\partial b_{q_j}}$
10:       Update $y_t \leftarrow y_t - \gamma \frac{\partial\mathcal{L}}{\partial y_t}$
11:       Update $x_v \leftarrow x_v - \gamma \frac{\partial\mathcal{L}}{\partial x_v}$
12:     end for
13:   end for
14:   for $i = 1$ to $N$ do
15:     for $k = 1$ to $N$ do
16:       Calculate $\frac{\partial\mathcal{L}}{\partial p_i}, \frac{\partial\mathcal{L}}{\partial w_k}, \frac{\partial\mathcal{L}}{\partial b_{p_i}}, \frac{\partial\mathcal{L}}{\partial b_{w_k}}$ using Eq. (11), (12), (15), (16)
17:       Update $p_i \leftarrow p_i - \gamma \frac{\partial\mathcal{L}}{\partial p_i}$
18:       Update $w_k \leftarrow w_k - \gamma \frac{\partial\mathcal{L}L}{\partial w_k}$
19:       Update $b_{p_i} \leftarrow b_{p_i} - \gamma \frac{\partial\mathcal{L}}{\partial q_j}$
20:       Update $b_{w_k} \leftarrow b_{w_k} - \gamma \frac{\partial\mathcal{L}}{\partial b_{u_i}}$
21:     end for
22:   end for
23: until Convergence
```

Table 1
Notation.

Symbol	Description
N, M, K	number of users, items and latent dimensions
$\mathbf{p} \in \mathbb{R}^{N\times K}$, $\mathbf{w} \in \mathbb{R}^{N\times K}$	truster, trustee specific latent matrix
$\mathbf{q} \in \mathbb{R}^{M\times K}$	item latent matrix
$\mathbf{b_u} \in \mathbb{R}^{N}$, $\mathbf{b_q} \in \mathbb{R}^{M}$	user-bias, item-bias latent vector
$\mathbf{b_p} \in \mathbb{R}^{N}$, $\mathbf{b_w} \in \mathbb{R}^{N}$	truster-bias, trustee-bias latent vector
$\mathbf{y} \in \mathbb{R}^{M\times K}$	item latent matrix that represents implicit influence
$\mathbf{x} \in \mathbb{R}^{N\times K}$	user latent matrix that represents implicit influence
γ	learning rate

5. Complexity analysis

The computational time of learning the proposed model TRecSo is dominated by the computation of the loss function $\mathcal{L}$ in Eq. (10) and its gradients with respect to feature vectors given in Eqs. (11)–(19). Let $|\mathcal{R}|$ and $|\mathcal{S}|$ be the number of observed ratings and trust relations, respectively. Then, the complexity of evaluating $\mathcal{L}$ is $\mathcal{O}(K|\mathcal{R}|c + d|S|)$, where K is the size of latent dimensionality, $c = max(a, b)$ and $d = max(b, K)$. Note that a and b are the average ratings an item has received and the average ratings an has user rated. Since the matrices $\mathcal{R}$ and $\mathcal{S}$ are extremely sparse as shown in Table 2, the values of $|\mathcal{R}|$ and $|S|$ are substantially smaller. Furthermore, the computational complexity of computing gradients $\frac{\partial L}{\partial p_i}, \frac{\partial L}{\partial w_i}, \frac{\partial L}{\partial q_j}, \frac{\partial L}{\partial b_{u_i}}, \frac{\partial L}{\partial b_{q_j}}, \frac{\partial L}{\partial y_t}, \frac{\partial L}{\partial x_v}$ for all $t \in I_i$ and $v \in T_i$, that is, Eqs. (11)–(19), are $\mathcal{O}(K|\mathcal{R}| + K|S|), \mathcal{O}(K|\mathcal{R}| + K|S|), \mathcal{O}(a|\mathcal{R}| + K|\mathcal{R}|a + K|\mathcal{R}|b + K|\mathcal{R}|), \mathcal{O}(|\mathcal{R}|a + |\mathcal{R}|), \mathcal{O}(|\mathcal{R}|a + |\mathcal{R}|), \mathcal{O}(|\mathcal{R}|a + K|\mathcal{R}|a), \mathcal{O}(|\mathcal{R}|a + K|\mathcal{R}|b)$. Consequently, the overall complexity per iteration is $\mathcal{O}(K|\mathcal{R}|c + d|S|)$. Since $c \ll |\mathcal{R}|$ or $|S|$, the overall computational complexity is linear with respect to the number of observed ratings and trust relations. Therefore, unlike the pair-wise LTR methods as mentioned in Section 2.3, our proposed model can be scaled to large datasets.

Table 2
Data statistics.

	Rating				Trust		
	User	Item	Rating	Density (%)	User	Links	Density (%)
FilmTrust	1,508	2,071	35,497	1.1366	1,642	1,853	0.0687
Ciao	7,375	99,746	278,483	0.0379	7,375	111,781	0.2055
Epinion	40,163	139,738	664,824	0.0118	49,289	487,183	0.0201

6. Experiments

In this section, we carry out experiments to compare the quality of the top-k recommendation of our method, TRecSo, with several state-of-the-art methods on three real-world datasets. Our experiments are designed to verify the following questions:

1. How does TRecSo perform compared with other related competitors?
2. Does considering the social network structure as in Eq. (8) enhance the performance of TRecSo?
3. How does the trade-off parameters of TRecSo affect the quality of the top-k recommendation?
4. How does the dimensionality of user and item latent space affect the performance of TRecSo?

6.1. Datasets

We use three public real-world datasets, each of which contains both user-item ratings and trust relationships among users. The trust relations between users are asymmetric in all three datasets. Note that the ratings in Epinions[9] and Ciao[10] are integers in range [1,5], whereas those in Filmtrust[11] are real values in range [0.5, 4] with step 0.5. The detailed data statistics are described in Table 2.

6.2. Experimental protocol

We adopt an experiment protocol called *Weak generalization* that has been widely used for evaluating the performance of top-k recommender system [42,48]. Weak generalization is evaluated by predicting the rank of unrated items for users known at training time. For our experiments, we randomly select n=10, 20, 50 observed ratings for each user and used them for training, and the model performance is evaluated on the remaining observed ratings. In order to evaluate the models on at least 10 items per user, we remove users with less than 20, 30, 60 rated items, respectively.

Parameter Setting: For all the comparing methods, we set the parameters with optimal values based on the cross-validation. As for our proposed method, TRecSo, we set $\alpha = 0.4, \lambda = 0.1, \lambda_t = 0.8$ and $\gamma = 0.01$ for Filmtrust and Ciao datasets, and $\alpha = 0.4, \lambda = 0.5, \lambda_t = 0.5$ and $\gamma = 0.001$ for Epinion dataset, where γ is the learning rate. As mentioned in Section 4.4, we set $\lambda_p = \lambda_w = \lambda_v = \lambda_c = \lambda$ in order to reduce the model complexity. Note that for all the experiments, we set the size of dimensionality to 5, and the results reported in this section are the mean values over 5 runs on 5 different datasets.

6.3. Evaluation metric

The standard evaluation metrics for the recommender systems with the task of predicting the correct ratings (not the list of top-k items) are Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). Both metrics measure the distance between the true ratings and the predicted ratings. However, since our paper aims at improving the top-k recommendation quality, we use Normalized Discounted Cumulative Gain (NDCG), which is one of the most commonly used metric in the field of information retrieval. Generally, when Web users use search engines, rather than examining all the pages returned by the search engine, they look at only a few pages on the top of the returned list. Indeed, NDCG pays more attention to the top of a ranked list, and gives a high score to the list that contains more relevant items near the top position of the ranked list. In other words, NDCG takes the actual rating values into consideration, and gives a high score to the list with high ratings near the top position. For example, regarding NDCG@1, a list with an item that received a 5-star rating on the top-1 position gets a higher score than a list with an item that received a 4-star rating on the top-1 position. The NDCG value at kth position with respect to u_i is defined as follows:

$$NDCG_i@k = Z \sum_{j=1}^{k} \frac{2^{r_{ij}} - 1}{\log(1 + j)}$$

[9] http://www.trustlet.org/downloaded_epinions.html.
[10] http://www.jiliang.xyz/datasetcode/ciao.zip.
[11] http://www.librec.net/datasets/filmtrust.zip.

where Z is a normalization constant to make the NDCG of the optimal ranking equal to 1. Finally, by calculating the value of $NDCG_i@k$ of each u_i in $\mathcal{U}$ and taking the mean value of all the users, we obtain $NDCG@k$ as follows:

$$NDCG@k = \frac{1}{|\mathcal{U}|}\sum_{i=1}^{N} NDCG_i@k \tag{20}$$

where $|\mathcal{U}|$ is the number of users in $\mathcal{U}$. Moreover, while metrics such as Precision@k (the ratio of relevant items in the predicted top-k ranked list) and Recall@k (the ratio of the relevant items over all the relevant items for each user in the predicted top-k ranked list) are not appropriate for our case because these metrics are better suited for datasets with implicit feedback where only binary relevance information is given, we show the results for Precision@k and Recall@k to verify the superiority of our method in various evaluation metrics for ranking.

6.4. Competitors

Since our method is a social network based learning-to-rank (LTR) recommendation method whose goal is to optimize the top-k recommendations, we compare with the following state-of-the-art recommendation methods. Our competitors are divided into three different categories: Traditional CF method, ratings-only-based LTR method and social network-based LTR method.

1. Traditional CF method
   - ItemKNN: A traditional recommendation method based on similarity of items.
2. Ratings-only-based LTR methods
   - WRMF [26]: A weighted matrix factorization algorithm with implicit feedback data (One-class collaborative filtering).
   - BPRMF [29]: An item recommendation algorithm based on pair-wise Learning-to-Rank strategy combined with matrix factorization.
   - ListRank [32]: A list-wise Learning-to-Rank method combined with matrix factorization.
3. Social network-based LTR methods
   - SBPR [49]: An extended version of BPRMF by including social network information (Pair-wise Learning-to-Rank).
   - SoRank [48]: A social network based list-wise Learning-to-Rank algorithm that linearly combines a users taste and her direct friends tastes in optimizing the top-k recommendation.
   - TRecSo: Our proposed method.

Our competitors are selected for the following reasons. First, by comparing the performance of traditional CF method (memory-based CF) with WRMF (model-based CF), we show that the model-based CF outperforms the memory-based CF as mentioned in Section 2.1. Second, we include ratings-only-based LTR methods to verify the benefit of incorporating the social network information in the top-k recommendation task. Finally, most recent social network-based LTR methods are considered as our direct competitors in order to justify the benefit of our proposed method.

Note that we did not consider methods such as [9,17,19,22,45] because their focus was minimizing the rating prediction error (RMSE and MAE) rather than optimizing top-k recommendation. Our method is implemented on top of LibRec[12], a Java library for recommender system, where most of our competitors are implemented.

6.5. Performance analysis

Comparison with competitors : We report the results of all the tested users in Fig. 1. In order to evaluate our method in Precision and Recall, we consider the 4–5 star ratings as relevant to a user and 1–3 as irrelevant. As shown in the figure, our proposed method (TRecSo) generally outperforms the competitors in all three datasets. Note that the performance benefit of TRecSo is especially significant in NDCG for the following two reasons.

1. The information loss is inevitable during the conversion of the explicit feedback datasets into implicit feedback datasets. Therefore, the performance benefit of TRecSo is not as significant in Precision and Recall.
2. NDCG is originally designed for evaluating the performance of methods based on the explicit feedback datasets.

Moreover, from the comparison with LRMF [32] we observe that incorporating the social information significantly improves the recommendation on top-k items. In addition, we conclude from the comparison with SoRank [48] that modeling the two different roles of users as trusters and trustee and incorporating the implicit influence of the users trusted by each user instead of linear combination of users and their friends boosts the performance of top-k recommendation.

Consideration of social network structure: In order to answer the second question that we mentioned in the beginning of Section 5, we conduct experiments to see whether the consideration of the social network structure of users improves the top-k recommendation performance on three datasets. As shown in Fig. 2, we observe that adjusting s_{ik} by using Eq. (8) indeed improves the NDCG values. The results clearly indicate that giving lower weights to users who trust many users and giving higher weights to those who are trusted by many users reflects the social phenomena of the real-world.

[12] http://www.librec.net/.

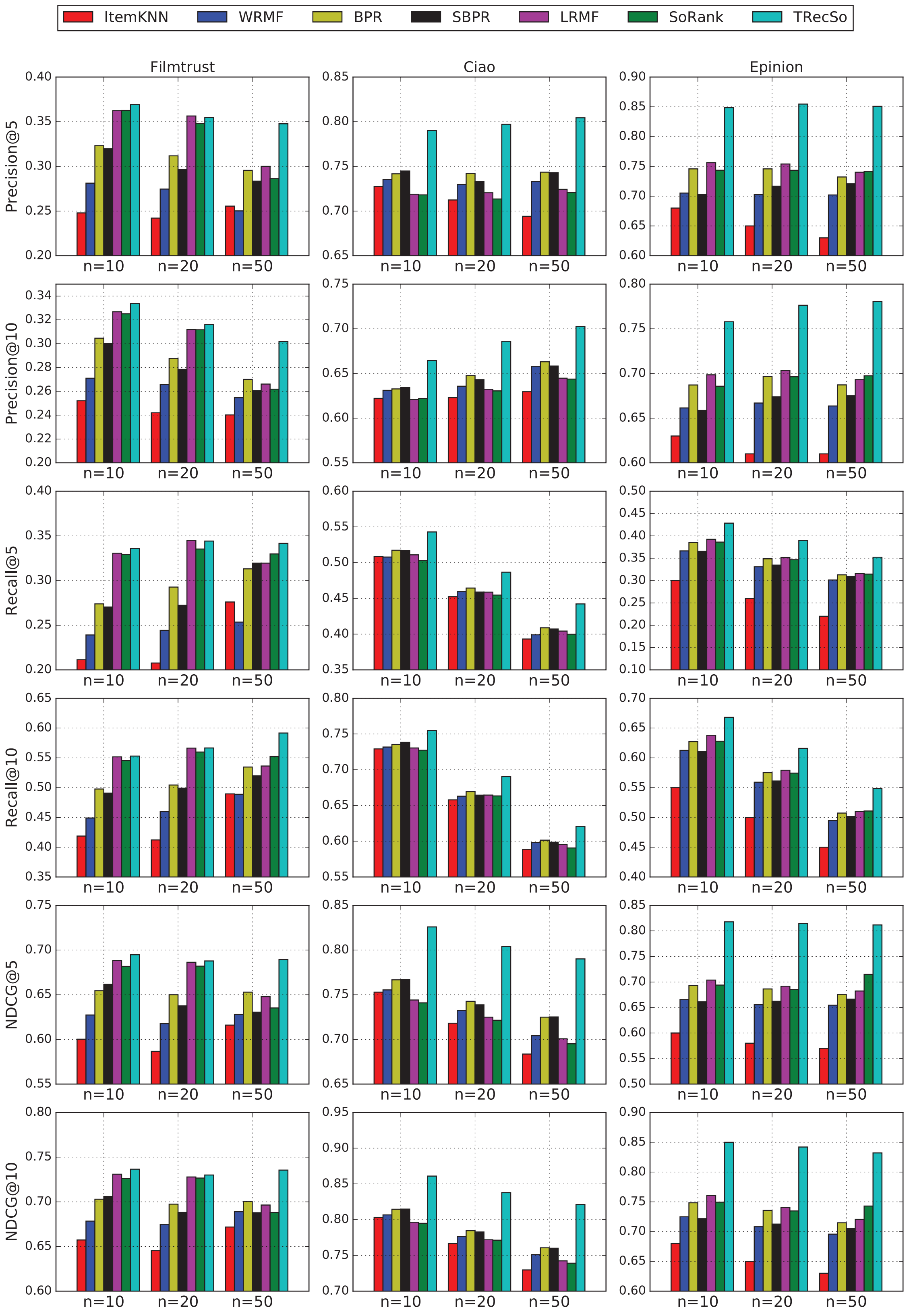


Fig. 1. Performance comparison on three datasets.

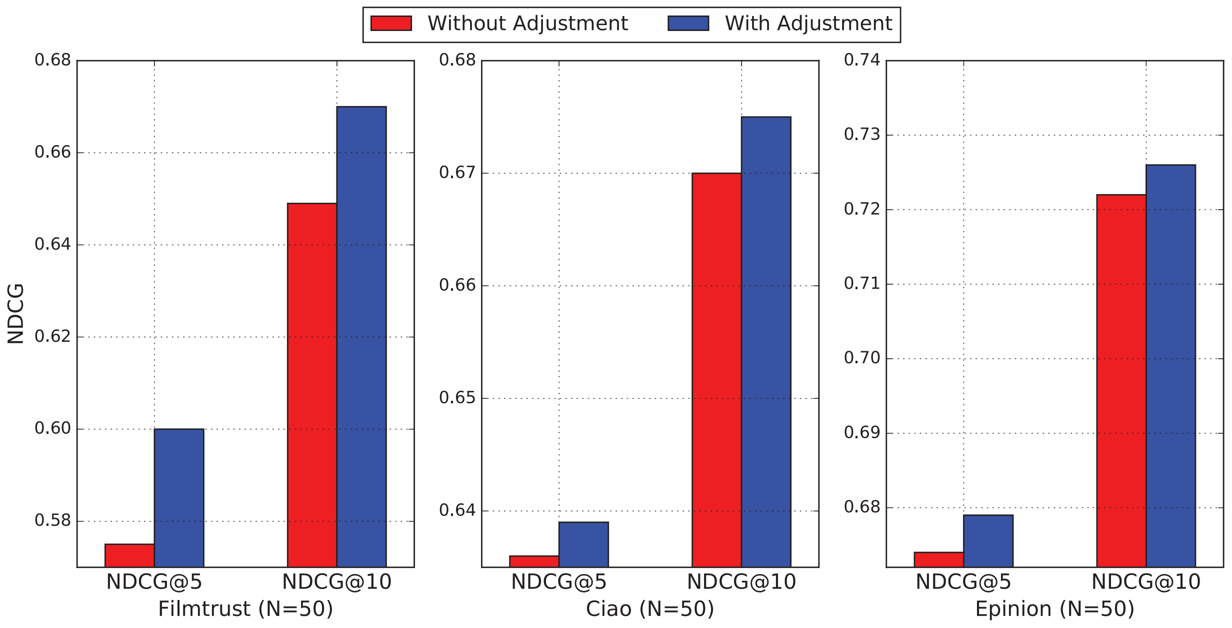


Fig. 2. Impact of considering the structure of the social network.

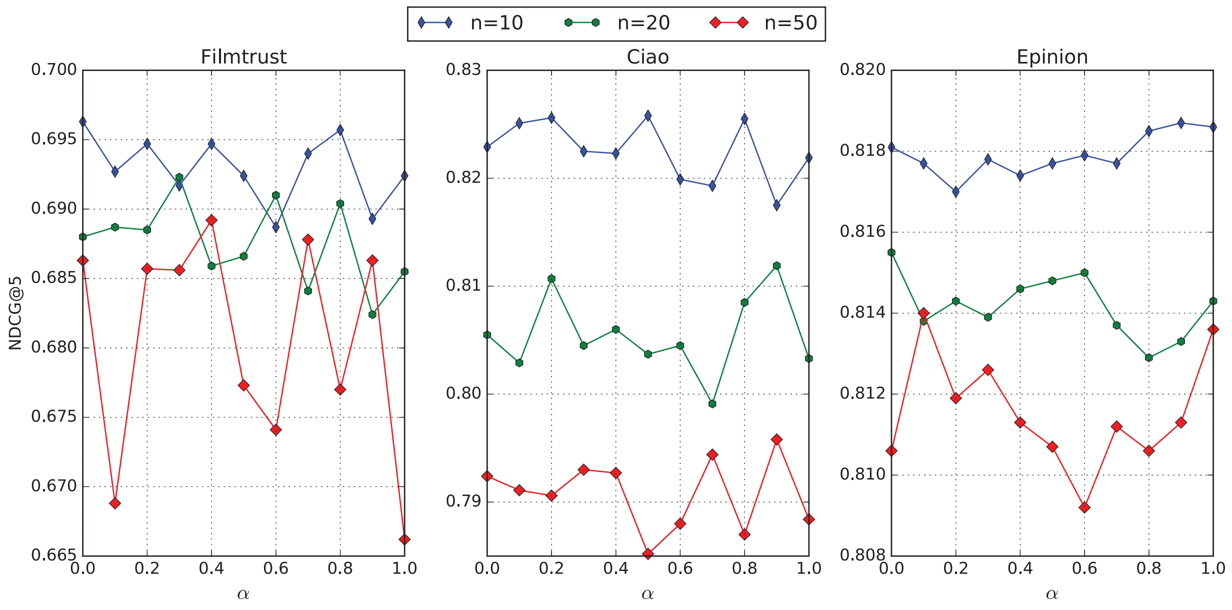


Fig. 3. Impact of Parameter α on three datasets.

6.6. Impact of parameters α and λ_t

In our proposed model, TRecSo, there are two trade-off parameters that should be tuned, i.e., α and λ_t. In order to determine the best performing parameters on each dataset, we adjust one parameter while fixing the other. We first investigate the impact of α while fixing λ_t to 0.8. As explained in Section 4, α is the parameter for balancing the relative importance of influence of truster and trustee. Fig. 3 illustrates the result of varying the value of α when the size of dimensionality is set to 5. The results show that the proper value of α exists, and it helps improve the recommendation quality.

After discovering the proper values of α, we then further explore the second trade-off parameter λ_t, which controls the importance of trust regularization. Fig. 4 shows the performance of TRecSo while changing the values of λ_t and fixing $\alpha = 0.5$. It is clear from the results that incorporating the trust network, that is when $\lambda_t > 0$, as auxiliary information to

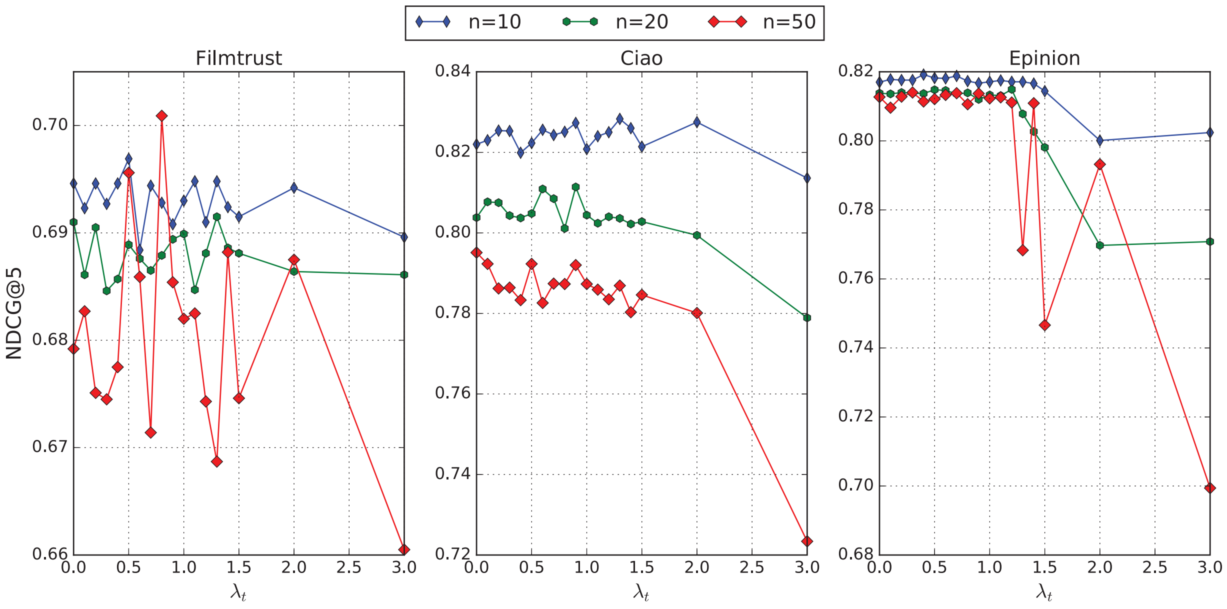


Fig. 4. Impact of Parameter λ_t on three datasets.

the ratings improves the performance of top-k recommendation. However, we observe that the legitimate balance between the influence from the ratings and the influence from the trust network should be empirically discovered, and the proper value differs among datasets.

6.7. Dimensionality analysis

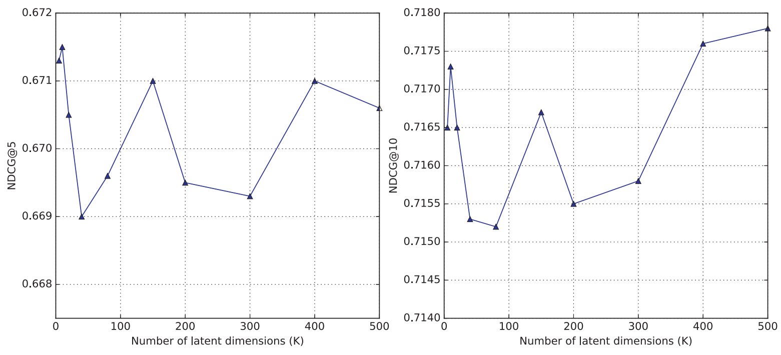
In this section, we address the fourth question that we introduced in Section 6, that is, to analyze the effect of the number of latent dimensions of user and item latent vectors on the performance of TRecSo. Generally, it is known from literatures that the performance of the recommendation improves as the number of latent dimensions increases [4,30]. Fig. 5 shows the performance with respect to the number of latent dimensions of our proposed model.

Interestingly enough, while experiments on Epinion dataset (Fig. 5c) show the trend of performance improvement as the number of latent dimensions increases, we could not discover any particular trends from the experiments on Filmtrust and Ciao datasets (Fig. 5a and 5b). Although precisely interpreting the meaning of each latent dimension is infeasible, we assume that it represents the profile of users' interest and items' features. For datasets with relatively small number of users and items, such as Filmtrust and Ciao, a large number of latent dimensionality would surpass the inherent number of profiles of users and items. However, for datasets like Epinion, which is composed of a large number of users and items, the performance of recommendation improves as the number of latent dimensionality increases. Nevertheless, if the number of dimensions is too large, the complexity will significantly increase. Therefore, we need to find a proper number of latent dimensions in order to balance the trade-off between the performance and the complexity.

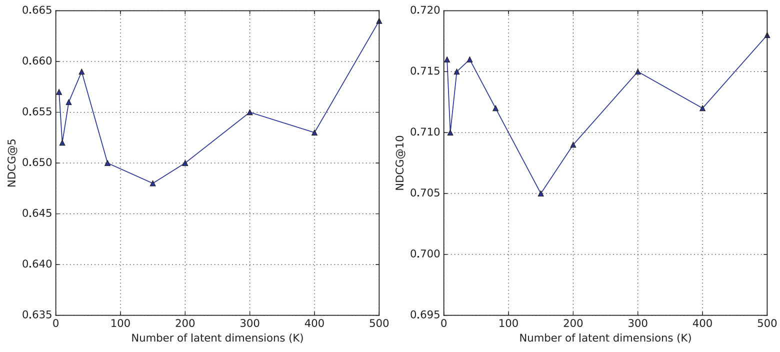
7. Conclusion and future work

This paper proposes TRecSo, a novel LTR based recommendation method that optimizes the top-k ranking prediction accuracy by additionally considering the social network information. Specifically, TRecSo integrates the social network information into the Learning-To-Rank (LTR) based objective function for recommendation. Thanks to the flexibility (can be generalized to symmetric social relationship) and the low complexity (compared with pair-wise LTR approaches) of our model, our proposed method can be easily integrated into a real-world applications where user-item interaction history and user social network information are given. Comprehensive experimental results show that TRecSo significantly outperforms the state-of-the-art algorithms in the top-k ranking accuracy of recommendation.

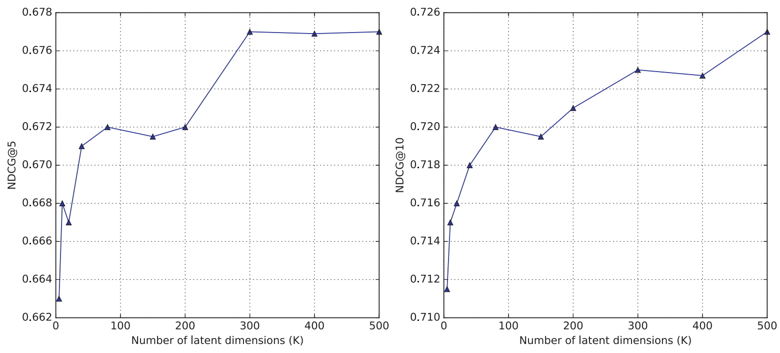
Recall that our work is based on the concept of *top-one probability* instead of top-k probability because the loss in time complexity outweighs the gain in performance. As a future work, we plan to extend our model to *top-k probability* that considers top-k items in a list rather than *top-one probability* to see whether the performance improvement can be achieved without compromising much computational complexity. Second, we also plan to investigate on other variant models for computing probability of permutations [5,40] rather than the *top-one probability*.



(a) Filmtrust dataset (n=20)



(b) Ciao dataset (n=20)



(c) Epinion dataset (n=20)

Fig. 5. Impact of latent dimensionality on three datasets.

Acknowledgment

This work was supported by the Industrial Core Technology Development Program (10049079 , Development of Mining core technology exploiting personal big data) funded by the Ministry of Trade, Industry and Energy (MOTIE, Korea), ICT R&D program of MSIP/IITP [14-824-09-014, Basic Software Research in Human-level Lifelong Machine Learning (Machine Learning Center)]

References

[1] G. Adomavicius, A. Tuzhilin, Context-aware recommender systems, in: Recommender Systems Handbook, Springer, 2011, pp. 217–253.
[2] J. Bennett, S. Lanning, The netflix prize, in: Proceedings of KDD Cup and Workshop, 2007, 2007, p. 35.

[3] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1998, pp. 43–52.
[4] B. Cao, N.N. Liu, Q. Yang, Transfer learning for collective link prediction in multiple heterogenous domains, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 159–166.
[5] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, H. Li, Learning to rank: from pairwise approach to listwise approach, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 129–136.
[6] P. Cremonesi, Y. Koren, R. Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: Proceedings of the Fourth ACM Conference on Recommender Systems, ACM, 2010, pp. 39–46.
[7] G. Guo, J. Zhang, N. Yorke-Smith, A novel bayesian similarity measure for recommender systems, IJCAI, 2013.
[8] G. Guo, J. Zhang, D. Thalmann, N. Yorke-Smith, Etaf: an extended trust antecedents framework for trust prediction, ASONAM, 2014.
[9] G. Guo, J. Zhang, N. Yorke-Smith, Trustsvd: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings., in: AAAI, 2015, pp. 123–129.
[10] T. Hofmann, Latent semantic models for collaborative filtering, ACM Trans. Inf. Syst. 22 (1) (2004) 89–115.
[11] M. Jamali, M. Ester, Trustwalker: a random walk model for combining trust-based and item-based recommendation, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 397–406.
[12] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, in: Proceedings of the Fourth ACM Conference on Recommender Systems, ACM, 2010, pp. 135–142.
[13] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2008, pp. 426–434.
[14] Y. Koren, Collaborative filtering with temporal dynamics, Commun. ACM 53 (4) (2010) 89–97.
[15] G. Ling, M.R. Lyu, I. King, Ratings meet reviews, a combined approach to recommend, in: Proceedings of the 8th ACM Conference on Recommender Systems, ACM, 2014, pp. 105–112.
[16] T.-Y. Liu, Learning to rank for information retrieval, Found. Trends Inf. Retr. 3 (3) (2009) 225–331.
[17] H. Ma, H. Yang, M.R. Lyu, I. King, Sorec: social recommendation using probabilistic matrix factorization, in: Proceedings of the 17th ACM Conference on Information and Knowledge Management, ACM, 2008, pp. 931–940.
[18] H. Ma, I. King, M.R. Lyu, Learning to recommend with social trust ensemble, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2009, pp. 203–210.
[19] H. Ma, D. Zhou, C. Liu, M.R. Lyu, I. King, Recommender systems with social regularization, in: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, ACM, 2011, pp. 287–296.
[20] J.I. Marden, Analyzing and Modeling Rank Data, CRC Press, 1996.
[21] M. Marshall, Aggregate Knowledge raises 5M from Kleiner, on a roll, 2006. URL http://venturebeat.com/2006/12/10/aggregate-knowledge-raises-5m-from-kleiner-on-a-roll/.
[22] P. Massa, P. Avesani, Trust-aware recommender systems, in: Proceedings of the 2007 ACM Conference on Recommender Systems, ACM, 2007, pp. 17–24.
[23] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: Proceedings of the 7th ACM Conference on Recommender Systems, ACM, 2013, pp. 165–172.
[24] S.M. McNee, J. Riedl, J.A. Konstan, Being accurate is not enough: how accuracy metrics have hurt recommender systems, in: CHI'06 Extended Abstracts on Human Factors in Computing Systems, ACM, 2006, pp. 1097–1101.
[25] J. Oh, W.-S. Han, H. Yu, X. Jiang, Fast and robust parallel sgd matrix factorization, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 865–874.
[26] R. Pan, Y. Zhou, B. Cao, N.N. Liu, R. Lukose, M. Scholz, Q. Yang, One-class collaborative filtering, in: Proceedings of the Eighth IEEE International Conference on Data Mining, 2008. ICDM'08., IEEE, 2008, pp. 502–511.
[27] W. Pan, L. Chen, Gbpr: group preference based bayesian personalized ranking for one-class collaborative filtering., in: IJCAI, 13, 2013, pp. 2691–2697.
[28] C. Porcel, E. Herrera-Viedma, Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries, Knowl. Based Syst. 23 (1) (2010) 32–39.
[29] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty–Fifth Conference on Uncertainty in Artificial Intelligence, AUAI Press, 2009, pp. 452–461.
[30] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Application of dimensionality reduction in recommender system-a case study, Technical Report, DTIC Document, 2000.
[31] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, ACM, 2001, pp. 285–295.
[32] Y. Shi, M. Larson, A. Hanjalic, List-wise learning to rank with matrix factorization for collaborative filtering, in: Proceedings of the Fourth ACM Conference on Recommender Systems, ACM, 2010, pp. 269–272.
[33] X. Su, T.M. Khoshgoftaar, A survey of collaborative filtering techniques, Adv. Artif. Intell. 2009 (2009) 4.
[34] J. Tang, X. Hu, H. Gao, H. Liu, Exploiting Local and Global Social Context for Recommendation, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI '13, AAAI Press, Beijing, China, 2013, pp. 2712–2718.
[35] J. Tang, H. Gao, H. Liu, mtrust: discerning multi-faceted trust in a connected world, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, ACM, 2012, pp. 93–102.
[36] J. Tang, X. Hu, H. Liu, Social recommendation: a review, Soc. Netw. Anal. Min. 3 (4) (2013) 1113–1133.
[37] A. Tejeda-Lorente, J. Bernabé-Moreno, C. Porcel, E. Herrera-Viedma, Integrating quality criteria in a fuzzy linguistic recommender system for digital libraries, Proc. Comput. Sci. 31 (2014a) 1036–1043.
[38] A. Tejeda-Lorente, C. Porcel, E. Peis, R. Sanz, E. Herrera-Viedma, A quality based recommender system to disseminate information in a university digital library, Inf. Sci. 261 (2014b) 52–69.
[39] L.H. Ungar, D.P. Foster, Clustering methods for collaborative filtering, in: AAAI Workshop on Recommendation Systems, 1, 1998, pp. 114–129.
[40] M.N. Volkovs, R.S. Zemel, Boltzrank: learning to maximize expected ranking gain, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 1089–1096.
[41] C. Wang, D.M. Blei, Collaborative topic modeling for recommending scientific articles, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2011, pp. 448–456.
[42] M. Weimer, A. Karatzoglou, Q.V. Le, A.J. Smola, COFI RANK - Maximum Margin Matrix Factorization for Collaborative Ranking, in: J.C. Platt, D. Koller, Y. Singer, S. Roweis (Eds.), Advances in Neural Information Processing Systems 20, MIT Press, Cambridge, MA, 2008, pp. 1593–1600.
[43] L. Xiong, X. Chen, T.-K. Huang, J.G. Schneider, J.G. Carbonell, Temporal collaborative filtering with bayesian probabilistic tensor factorization., in: SDM, 10, SIAM, 2010, pp. 211–222.
[44] J. Xu, H. Li, Adarank: a boosting algorithm for information retrieval, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2007, pp. 391–398.
[45] B. Yang, Y. Lei, D. Liu, J. Liu, Social collaborative filtering by trust, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, AAAI Press, 2013, pp. 2747–2753.
[46] X. Yang, H. Steck, Y. Guo, Y. Liu, On top-k recommendation using social networks, in: Proceedings of the Sixth ACM Conference on Recommender Systems, ACM, 2012, pp. 67–74.
[47] X. Yang, H. Steck, Y. Liu, Circle-based recommendation in online social networks, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2012, pp. 1267–1275.

[48] W. Yao, J. He, G. Huang, Y. Zhang, Sorank: incorporating social information into learning to rank models for recommendation, in: Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion, International World Wide Web Conferences Steering Committee, 2014, pp. 409–410.
[49] T. Zhao, J. McAuley, I. King, Leveraging social connections to improve personalized ranking for collaborative filtering, in: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, ACM, 2014, pp. 261–270.
[50] Z.-D. Zhao, M.-S. Shang, User-based collaborative-filtering recommendation algorithms on hadoop, in: Proceedings of the Third International Conference onKnowledge Discovery and Data Mining, 2010. WKDD'10., IEEE, 2010, pp. 478–481.
[51] Y. Zhou, D. Wilkinson, R. Schreiber, R. Pan, Large-scale parallel collaborative filtering for the netflix prize, in: Algorithmic Aspects in Information and Management, Springer, 2008, pp. 337–348.