Learning to Rank with Trust and Distrust in Recommender Systems

Dimitrios Rafailidis
Department of Computer Science
University of Mons
Mons, Belgium
dimitrios.rafailidis@umons.ac.be

Fabio Crestani
Faculty of Informatics
Università della Svizzera italiana (USI)
Lugano, Switzerland
fabio.crestani@usi.ch

ABSTRACT

The sparsity of users' preferences can significantly degrade the quality of recommendations in the collaborative filtering strategy. To account for the fact that the selections of social friends and foes may improve the recommendation accuracy, we propose a learning to rank model that exploits users' trust and distrust relationships. Our learning to rank model focusses on the performance at the top of the list, with the recommended items that end-users will actually see. In our model, we try to push the relevant items of users and their friends at the top of the list, while ranking low those of their foes. Furthermore, we propose a weighting strategy to capture the correlations of users' preferences with friends' trust and foes' distrust degrees in two intermediate trust- and distrust-preference user latent spaces, respectively. Our experiments on the Epinions dataset show that the proposed learning to rank model significantly outperforms other state-of-the-art methods in the presence of sparsity in users' preferences and when a part of trust and distrust relationships is not available. Furthermore, we demonstrate the crucial role of our weighting strategy in our model, to balance well the influences of friends and foes on users' preferences.

CCS CONCEPTS

•Information systems →Collaborative and social computing systems and tools;

KEYWORDS

Learning to rank; social relationships; collaborative filtering

1 INTRODUCTION

The collaborative filtering strategy has been proved as an effective means for recommender systems, providing similar-minded users with similar recommendations [13]. However, in a real-world scenario the sparsity of users' preferences significantly limits the recommendation accuracy. To overcome the sparsity problem of users' preferences, several models exploit the selections of trust friends to generate trust-based recommendations [8, 18]. In trust-based recommendations, models consider that people tend to rely more on recommendations from friends they trust than on recommendations of anonymous people similar to them [12, 16]. The challenge in trust-based recommender systems is that we have to learn both about users' preferences and trust degrees, as friends do not have necessarily the same preferences [3, 8]. In addition, in online networks users may establish both trust and distrust relationships. For example Epinions[1], an e-commerce site for reviewing and rating products, allows users to evaluate other users based on the quality of their reviews, and form trust and distrust relations with them. In Slashdot[2] users post news and comments, and they can tag other users as friends or foes. The analyses in relevant studies point out that users might accept recommendations from their trusted users, but will certainly not rely on their distrusted foes [27, 29]. Thus, more recently a few attempts have been made to exploit both trust and distrust relationships in recommender systems, in studies such as those reported in [5, 6, 19, 26]. Following the collaborative filtering strategy, these models design different objective functions with trust and distrust relationships to handle the sparsity in users' preferences, assuming that the latent features of users and their friends should be as close as possible, while those of foes as far as possible. However, these methods do not capture well the correlations of users' preferences with friends' trust and foes' distrust degrees, as we will experimentally show in Section 5.

In addition, existing methods that exploit both trust and distrust relationships are mainly focused on the rating prediction problem, aiming to predict the missing ratings of users based on different loss functions. However, several studies point out that focussing on the top-N recommendation problem, that is predicting the products-items that users will see in a ranked recommendation list, reflects more on real-world scenarios [1, 33]. In this respect, several learning to rank models have been introduced to improve the personalized ranking performance directly in the top-N recommendation problem, such as the studies reported in [24, 25, 30]. Moreover, in [4], authors introduce different optimization algorithms to perform push at the top of the list, assuming that what it matters is the performance of the learning to rank models at the top of the list, the recommended items that users will actually see. While there are various attempts to incorporate trust relationships in learning to rank models, such as the studies reported in [3, 15, 20, 21, 33], these models ignore the selections of foes, which play an important role in boosting the recommendation accuracy [7, 28, 29].

RecSys'17, August 27-31, 2017, Como, Italy
© 2017 ACM. ISBN 978-1-4503-4652-8/17/08…$15.00
DOI: http://dx.doi.org/10.1145/3109859.3109879

[1]http://www.epinions.com/
[2]https://slashdot.org/

To overcome the limitations of existing methods we propose a learning to rank model with trust and distrust relationships, by balancing the influences of friends' and foes' selections on users' preferences. This paper's contributions are summarized as follows:

- We design a learning to rank model, trying to push the relevant items of users and their friends at the top of the recommendation list, while at the same time ranking low the foes' relevant items.
- We propose a weighting strategy to measure the influences of friends and foes. To better capture their trust and distrust degrees with users' preferences, we build an intermediate trust-preference user latent space with friends and their selections, and an intermediate distrust-preference space with foes and their selections.

Our experiments on the Epinions dataset show that the proposed approach outperforms competitive approaches in the presence of preferences' sparsity and when a part of trust and distrust relationships is missing. Also, we demonstrate the importance of our weighting strategy in our learning to rank model to compute the correlations of users' preferences with friends' trust and foes' distrust degrees.

The remainder of the paper is organized as follows. Section 2 reviews the related work, Section 3 defines our ranking problem with trust and distrust relationships and Section 4 details the proposed learning to rank model. Section 5 evaluates the performance of the proposed model against several state-of-the-art methods and, finally, Section 6 concludes the study.

2 RELATED WORK

2.1 Trust-based Recommendations

As trust relationships can leverage the recommendation accuracy, many different strategies have been introduced to exploit the selections of trust friends when producing recommendations. In [12], authors extend the probabilistic matrix factorization of [18] by weighting the user latent factors based on their trust relationships. In [16], a trust-based ensemble method is presented to combine matrix factorization with a trust-based neighbourhood model. In [11], Jamali and Ester combine TrustWalker [10] with a neighbourhood collaborative filtering strategy. In this study, authors run random walks on the trust network, formed by the trust relationships, and then they perform a probabilistic item selection strategy to generate recommendations. Guo et al. [8] extend SVD++ [13], to learn both the user preferences and the influence of her friends. This study considers both friends' ratings and trust degrees to produce recommendations. However, these methods treat the recommendation problem as a rating prediction problem in which different squared loss functions are used to minimize the prediction error. Nonetheless, methods that achieve low rating errors do not necessarily have high ranking performance in the recommendation lists [32]. This occurs because optimizing the predicted values of the ratings may not provide the best recommendation lists in many cases. For instance, if all the low-ranked ratings are predicted very accurately, but significant errors are made on the higher-ranked ratings, then these methods will not provide a high-quality personalized recommendation list to the end-user [1].

2.2 Learning to Rank

Learning to rank methods have been widely studied in recommender systems. The goal is to define a personalized ranking function, and then learn loss functions to improve the ranking performance in the top-N recommendation problem [1]. Representative learning to rank models in recommender systems are CofiRank [30] and CLiMF [25], which use loss functions based on Normalized Discounted Cumulative Gain and Reciprocal Rank, respectively. The Bayesian personalized ranking framework of [24] makes a prediction for every pair of items concerning their relative ordering in the recommendation list. In [4], authors introduce various learning to rank models, which follow different strategies when learning the personalized ranking functions, to account for the fact that users focus on recommendations at the top of the list. However, all the aforementioned learning to rank models do not exploit trust nor distrust relationships in their learning process. Instead, various learning to rank models focus on the ranking performance when generating recommendations with trust relationships. Grimberghe et al. [15] combine Multi-Relational matrix factorization with the Bayesian personalized ranking framework of [24] to model users' feedback both on items and on trust relationships. Zhao et al. [33] present a trust-based Bayesian personalized ranking model that incorporates trust relationships into a pair-wise ranking model, assuming that users tend to assign higher ranks to items that their friends prefer. In [20], a learning to rank model is presented, considering how well the relevant items of users and their friends have been ranked at the top of the list. In [21], authors extend the model of [20] by combing different learning to rank strategies into a joint model to leverage the recommendation accuracy with trust relationships. Chaney et al. [3] infer each user's preferences and the influence of her friends by introducing a Bayesian model that performs Social Poisson factorization. However, all the above learning to rank models ignore users' distrust relationships when generating recommendations.

2.3 Recommendation with Trust and Distrust

Ma et al. [17] present a social regularization method that shares a common user-item matrix, factorized by ratings and social relationships. This work introduces a trust-based, as well as a distrust-based model to exploit trust and distrust relationships in each model separately. The goal of the trust-based model is to minimize the distances of latent features between trust users, while the distrust-based model tries to maximize the latent features' distances between distrust users. Recently, a few attempts have been made to exploit both trust and distrust relationships at the same time in recommender systems. Forsati et al. [6] use trust and distrust relationships in a matrix factorization framework using a hinge loss function. This method assumes that the trust and distrust relationships between users are considered as similarity/dissimilarity in their preferences. Then, the latent features are computed in a manner such that the latent features of foes who are distrusted by a certain user have a guaranteed minimum dissimilarity gap from the worst dissimilarity of friends who are trusted by this same user. In [5], a recommendation strategy is introduced to rank the latent features of users, based on the users' trust and distrust relationships. This method also considers the neutral relationships (users who have no relation to a certain user), aiming to rank the neutral users' latent features after the friends' latent features and before those of foes. In [19] a signed graph is constructed, considering positive and negative weights for the trust and distrust relationships, respectively. Then, a spectral clustering approach is used to generate clusters in the signed graph. The clusters are extracted on condition that users with positive connections should lie close to each other, while users with negative ones should lie far to each other. Following a joint non-negative matrix factorization framework the final recommendations are generated, by co-factorizing the user-item and user-cluster associations. Tang et al. [26] consider a signed graph with trust and distrust relationships and capture local and global information from the signed graph. Local information reveals the correlations among users and her friends/foes, and the global information reveals the reputation of the user in the whole social network, as users tend to trust users with high global reputation. Then, they exploit both local and global information in a matrix factorization technique to compute the recommendations. Nonetheless, none of these studies aim to improve the personalized ranking performance in their learning strategies.

3 PROBLEM DEFINITION

Let $\mathcal{N}$ and $\mathcal{I}$ be the sets of users and items, with $n = |\mathcal{N}|$ being the number of users and $m = |\mathcal{I}|$ the number of items. Our learning to rank model inputs the user-item matrix $\mathbf{R} \in \mathbb{R}^{n \times m}$ with user preferences, expressed by any type of user-item interactions such as ratings, number of views and clicks, and so on. Following the collaborative filtering strategy of matrix factorization, we assume that the recommendations are in the factorized matrix $\hat{\mathbf{R}} \in \mathbb{R}^{n \times m}$ of the user-item interaction matrix $\mathbf{R}$. In addition, we consider two adjacency matrices $\mathbf{A}^+ \in \mathbb{R}^{n \times n}$ and $\mathbf{A}^- \in \mathbb{R}^{n \times n}$, corresponding to the weights of the trust and distrust relationships. Based on the adjacency matrices $\mathbf{A}^+$ and $\mathbf{A}^-$, for each user i we calculate the neighbourhoods N_i^+ and N_i^- with her friends and foes, respectively. With these settings our problem is formally defined as follows:

Definition 1 (Problem). *Given (i) the friends in the adjacency matrix $\mathbf{A}^+$, (ii) the foes in the adjacency matrix $\mathbf{A}^-$, and (iii) the users' preferences in the user-item interaction matrix $\mathbf{R}$, the goal of the proposed learning to rank model is to compute the factorized matrix $\hat{\mathbf{R}}$, by pushing the relevant items of users and their friends at the top of the recommendation list, and push down the relevant items of foes.*

4 PROPOSED APPROACH

4.1 Learning to Rank Model

Let $\mathcal{X}_i$ and $\mathcal{Y}_i$ be the sets of relevant and irrelevant items of user i. Aiming at improving the ranking performance at the top of the recommendation list, the reverse height of a relevant item x_i is defined as follows [4]:

Definition 2 (Reverse Height). *The reverse height $RH_i(x_i)$ of a relevant item $x_i \in X_i$ in the recommendation list of user i is the number of irrelevant items $y_i \in Y_i$ ranked above x_i.*

Let r_i be the personalized ranking function of user i. According to Definition 1, the reverse height is computed as follows:

$$RH_i(x_i) = \sum_{y_i \in \mathcal{Y}_i} \mathbf{1}\Big[r_i(x_i) \leq r_i(y_i)\Big] \tag{1}$$

where $\mathbf{1}$ is the indicator function. As the indicator is a not convex function and we have to measure the influences of friends and foes, we take the following surrogate:

$$\begin{aligned} g_i(x_i, y_i) &= r_i(x_i) - r_i(y_i) \\ &+ \frac{1}{|\mathcal{N}_i^+|} \sum_{j \in \mathcal{N}_i^+} \Bigg[\mathbf{W}_{ij}^+ \frac{1}{|\mathcal{X}_j|} \sum_{x_j \in \mathcal{X}_j} \Big(r_j(x_j) - r_j(y_i)\Big)\Bigg] \\ &- \frac{1}{|\mathcal{N}_i^-|} \sum_{v \in \mathcal{N}_i^-} \Bigg[\mathbf{W}_{iv}^- \frac{1}{|\mathcal{X}_v|} \sum_{x_v \in \mathcal{X}_v} \Big(r_v(x_v) - r_v(y_i)\Big)\Bigg] \end{aligned} \tag{2}$$

where r_j and r_v are the personalized ranking functions of a friend j and a foe v, respectively, and $\mathcal{X}_j$ and $\mathcal{X}_v$ are the sets of items marked as relevant by j and v, accordingly. The term in the middle line of Eq. (2) measures the influence of friends in the surrogate g_i, that is how many relevant items x_j of friend j are ranked above the user's i irrelevant item y_i. Accordingly, the term in the last line of Eq. (2) penalizes the influence of foes based on their relevant items x_v that are ranked above y_i. This means that we have to push the relevant items $x_i \in \mathcal{X}_i$ of user i and $x_j \in \mathcal{X}_j$ of friend j above the irrelevant items $y_i \in \mathcal{Y}_i$ of user i. Also, we have to push down the relevant items $x_v \in \mathcal{X}_v$ of foe v below the irrelevant items y_i. In doing so, the relevant items x_i and x_j will be pushed at the top of the personalized recommendation list of user i and the relevant items x_v will be ranked low.

The weighting matrices $\mathbf{W}^+ \in \mathbb{R}^{n \times n}$ and $\mathbf{W}^- \in \mathbb{R}^{n \times n}$ in Eq. (2) are formally defined as follows:

Definition 3 (Trust Weighting Matrix). *Given a friend $j \in \mathcal{N}_i^+$ of user i and their preferences in the user-item interaction matrix $\mathbf{R}$, each element $\mathbf{W}_{ij}^+$ measures the correlation between j and i, by considering their preferences and trust degree.*

Definition 4 (Distrust Weighting Matrix). *Given a foe $v \in \mathcal{N}_i^-$ of user i and their preferences in $\mathbf{R}$, each element $\mathbf{W}_{iv}^-$ expresses the correlation between v and i, based on their preferences and distrust degree.*

Our weighting strategy to calculate matrices $\mathbf{W}^+$ and $\mathbf{W}^-$ is presented in the subsequent Section 4.2.

In our ranking model, we have to learn the ranking functions r_i, r_j and r_v in a collaborative manner. As previously stated in Section 3, following the collaborative filtering strategy in our model we consider the recommendations in the factorized matrix $\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^\top$, with $\mathbf{U} \in \mathbb{R}^{n \times k}$ and $\mathbf{V} \in \mathbb{R}^{m \times k}$ being the user and item factor matrices and k the number of latent factors. By taking the respective k-dimensional latent vectors of matrices $\mathbf{U}$ and $\mathbf{V}$, Eq. (2) becomes:

$$\begin{aligned} g_i(x_i, y_i) &= \mathbf{u}_i{}^\top(\mathbf{v}_{x_i} - \mathbf{v}_{y_i}) \\ &+ \frac{1}{|\mathcal{N}_i^+|} \sum_{j \in \mathcal{N}_i^+} \Bigg[\mathbf{W}_{ij}^+ \frac{1}{|\mathcal{X}_j|} \sum_{x_j \in \mathcal{X}_j} \Big(\mathbf{u}_j^\top(\mathbf{v}_{x_j} - \mathbf{v}_{y_i})\Big)\Bigg] \\ &- \frac{1}{|\mathcal{N}_i^-|} \sum_{v \in \mathcal{N}_i^-} \Bigg[\mathbf{W}_{iv}^- \frac{1}{|\mathcal{X}_v|} \sum_{x_v \in \mathcal{X}_v} \Big(\mathbf{u}_v^\top(\mathbf{v}_{x_v} - \mathbf{v}_{y_i})\Big)\Bigg] \end{aligned} \tag{3}$$

As the surrogate in Eq. (3) is not convex, we reformulate Eq. (2) as a minimization problem with respect to the latent matrices **U** and **V** using the convex logistic function:

$$\min_{\mathbf{U},\mathbf{V}} RH_i(x_i) = \sum_{y_j \in Y_j} \log\left(1 + \exp(-g_i(\mathbf{x}_i, \mathbf{y}_i))\right) \tag{4}$$

In our model we solve the minimization problem in Eq. (4) via gradient descent, using the following update rules at the t-h iteration:

$$\mathbf{u}_i^{t+1} \leftarrow \mathbf{u}_i^t - \eta \frac{\partial RH}{\partial_{\mathbf{u}_i}}, \quad i = 1 \ldots n \tag{5}$$

$$\mathbf{v}_j^{t+1} \leftarrow \mathbf{v}_j^t - \eta \frac{\partial RH}{\partial_{\mathbf{v}_j}}, \quad j = 1 \ldots m \tag{6}$$

where η is the learning parameter and $\partial RH/\partial_{\mathbf{u}_i}$ and $\partial RH/\partial_{\mathbf{v}_j}$ are the respective gradients. When optimizing Eq. (4) the influence of friends' and foes' selections on the user i's preferences might become negative in the surrogate g_i of Eq. (3), that is the case of $P_i < 0$, with the term P_i being equal to:

$\forall i \in \mathcal{N}$

$$\begin{aligned} P_i = & \frac{1}{|\mathcal{N}_i^+|} \sum_{j \in \mathcal{N}_i^+} \left[\mathbf{W}_{ij}^+ \frac{1}{|\mathcal{X}_j|} \sum_{x_j \in \mathcal{X}_j} \left(\mathbf{u}_j^\top (\mathbf{v}_{x_j} - \mathbf{v}_{y_i}) \right) \right] \\ & - \frac{1}{|\mathcal{N}_i^-|} \sum_{v \in \mathcal{N}_i^-} \left[\mathbf{W}_{iv}^- \frac{1}{|\mathcal{X}_v|} \sum_{x_v \in \mathcal{X}_v} \left(\mathbf{u}_v^\top (\mathbf{v}_{x_v} - \mathbf{v}_{y_i}) \right) \right] \end{aligned} \tag{7}$$

Hence, during the optimization of Eq. (4) with the update rules of Eqs. (5)-(6), we define a boolean vector $\mathbf{h}^t \in \{0,1\}^n$ for each iteration t in the gradient descent algorithm, by replacing the surrogate g_i of Eq. (3) with $g_i(x_i, y_i) = {\mathbf{u}_i}^\top (\mathbf{v}_{x_i} - \mathbf{v}_{y_i}) + \mathbf{h}_i^t P_i$. The i-th entry $\mathbf{h}_i^t$ is calculated as follows: if $P_i < 0$ then $\mathbf{h}_i^t = 0$, and 1 otherwise. Having calculated the factor matrices **U** and **V** by minimizing Eq. (4), we compute the factorized matrix $\hat{\mathbf{R}}$ as the product of $\mathbf{U}\mathbf{V}^\top$. Then, for each user i we order the items in the i-th row of $\hat{\mathbf{R}}$ in a descending order, and select the top-N items to generate the final recommendations.

So the challenge in our learning to rank model is how to weigh the influences of friends and foes when optimizing Eq. (4), capturing users' preferences with trust and distrust degrees in the weighting matrices $\mathbf{W}^+$ and $\mathbf{W}^-$.

4.2 Weighting Strategy

Let $\mathbf{U}^+ \in \mathbb{R}^{n \times k^+}$ and $\mathbf{U}^- \in \mathbb{R}^{n \times k^-}$ be the user factor matrices of $\mathbf{A}^+$ and $\mathbf{A}^-$, with k^+ being the number of latent factors of $\mathbf{U}^+$, and k^- the number of latent factors of $\mathbf{U}^-$. We also consider the user factor matrix $\mathbf{U} \in \mathbb{R}^{n \times k}$ of the user-item interaction matrix **R**. The three user factor matrices **U**, $\mathbf{U}^+$ and $\mathbf{U}^-$ generate three respective latent spaces, that is a $n \times k$ - dimensional latent space with users' preferences of **U**, a $n \times k^+$ - dimensional space with trust correlations of $\mathbf{U}^+$ and a $n \times k^-$ - dimensional space with distrust correlations of $\mathbf{U}^-$.

4.2.1 *Trust-based Weighting Matrix.*

As pointed out in several studies, the preferences of trust friends do not necessarily match [3, 8]. Hence, we compute the weighting matrix $\mathbf{W}^+$, to capture users' preferences with their trust correlations in an intermediate *trust-preference user factor* space, formed by a matrix $\mathbf{M}^+ \in \mathbb{R}^{k \times k^+}$ which is defined as follows:

Definition 5 (Trust-Preference User Factor Matrix). *Given the k and k^+ latent dimensions of the n users in* **U** *and* $\mathbf{U}^+$*, the intermediate trust-preference user factor matrix* $\mathbf{M}^+$ *captures the correlations of users' preferences with friends' trust degrees in an intermediate user latent space, subject to:*

$$\mathbf{W}^+ = \mathbf{U} \times \mathbf{M}^+ \times {\mathbf{U}^+}^\top \tag{8}$$

To calculate the user similarities in the latent spaces, we consider the k-dimensional latent vectors u_i in **U** and the k^+-dimensional vectors u_i^+ in $\mathbf{U}^+$. We define the respective similarity functions s, s^+ and sm^+ in the user latent spaces of **U**, $\mathbf{U}^+$ and $\mathbf{M}^+$ as follows:

$\forall i \in \mathcal{N}$ and $j \in \mathcal{N}_i^+$

$$s(\mathbf{u}_i, \mathbf{u}_j) = \frac{1}{1 + \exp\left(-{\mathbf{u}_i}^\top \mathbf{u}_j\right)} \tag{9}$$

$$s^+(\mathbf{u}_i^+, \mathbf{u}_j^+) = \frac{1}{1 + \exp\left(-{\mathbf{u}_i^+}^\top \mathbf{u}_j^+\right)} \tag{10}$$

$$sm^+(\mathbf{u}_i, \mathbf{u}_j^+) = \frac{1}{1 + \exp\left(-{\mathbf{u}_i}^\top \mathbf{M}^+ \mathbf{u}_j^+\right)} \tag{11}$$

where the product $\mathbf{u}_i^\top \mathbf{M}^+ \mathbf{u}_j^+$ reflects on the embedded distance of $\mathbf{u}_i$ and $\mathbf{u}_j^+$ in the intermediate latent space of $\mathbf{M}^+$. Since users i and j are friends, having (probably) similar preferences, we have to ***maximize*** their similarities in the three latent spaces **U**, $\mathbf{U}^+$ and $\mathbf{M}^+$. We transform the problem of calculating the intermediate trust-preference user factor matrix into a minimization problem with respect to the user factor matrices **U**, $\mathbf{U}^+$ and $\mathbf{M}^+$. To achieve this we take the negative natural logarithmic function of the respective similarities. For example, the similarities in the latent space of **U** are recomputed as follows: $-\sum_{i,j} \log\left(s(\mathbf{u}_i, \mathbf{u}_j)\right)$. Summarizing, we formulate our minimization problem as the following objective function with respect to the user factor matrices **U**, $\mathbf{U}^+$ and $\mathbf{M}^+$:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{U}^+,\mathbf{M}^+} L^+(\mathbf{U}, \mathbf{U}^+, \mathbf{M}^+) = & - \sum_{i \in \mathcal{N}, j \in \mathcal{N}_i^+} \log\left(s(\mathbf{u}_i, \mathbf{u}_j)\right) \\ & - \sum_{i \in \mathcal{N}, j \in \mathcal{N}_i^+} \log\left(s^+(\mathbf{u}_i^+, \mathbf{u}_j^+)\right) - \sum_{i \in \mathcal{N}, j \in \mathcal{N}_i^+} \log\left(sm^+(\mathbf{u}_i, \mathbf{u}_j^+)\right) \\ & + \lambda^+ \sum_{c=1}^{k} ||\mathbf{u}_c||_2 + \gamma^+ \sum_{c=1}^{k^+} ||\mathbf{u}_c^+||_2 + \delta^+ ||\mathbf{M}^+||_1 \end{aligned} \tag{12}$$

In the last line, the first two terms are the ℓ_2 norm regularization terms to avoid model overfitting. The third term is the ℓ_1 norm regularization to force the intermediate trust-preference user factor matrix to be sparse, as both the trust relationships in $\mathbf{A}^+$ and the users' preferences in **R** are sparse. Parameters λ^+, γ^+ and δ^+ control the influences of the respective regularization terms. To optimize the minimization problem in Eq. (12), we use a gradient-based alternating optimization algorithm, that is, we optimize each matrix/variable via gradient descent by fixing the other two [9]. Hence, to compute the three matrices/variables in alternating optimization we calculate the gradients of the objective function in Eq. (12) with respect to matrices **U**, $\mathbf{U}^+$ and $\mathbf{M}^+$ as follows:

$$\frac{\partial L^+(\mathbf{U},\mathbf{U}^+,\mathbf{M}^+)}{\partial_{\mathbf{u}_i}} = -\sum_{i\in\mathcal{N}, j\in\mathcal{N}_i^+} \frac{\exp(-\mathbf{u}_i{}^\top \mathbf{u}_j)}{1+\exp(-\mathbf{u}_i{}^\top \mathbf{u}_j)}\mathbf{u}_j - \sum_{i\in\mathcal{N}, j\in\mathcal{N}_i^+} \frac{\exp(-\mathbf{u}_i{}^\top \mathbf{M}^+\mathbf{u}_j^+)}{1+\exp(-\mathbf{u}_i{}^\top \mathbf{M}^+\mathbf{u}_j^+)}\mathbf{M}^+\mathbf{u}_j^+ + 2\lambda^+\sum_{c=1}^{k}(\mathbf{u}_i)^c \tag{13}$$

$$\frac{\partial L^+(\mathbf{U},\mathbf{U}^+,\mathbf{M}^+)}{\partial_{\mathbf{u}_i^+}} = -\sum_{i\in\mathcal{N}, j\in\mathcal{N}_i^+} \frac{\exp(-\mathbf{u}_i^{+\top} \mathbf{u}_j^+)}{1+\exp(-\mathbf{u}_i{}^\top \mathbf{u}_j^+)}\mathbf{u}_j^+ - \sum_{i\in\mathcal{N}, j\in\mathcal{N}_i^+} \frac{\exp(-\mathbf{u}_i{}^\top \mathbf{M}^+\mathbf{u}_j^+)}{1+\exp(-\mathbf{u}_i{}^\top \mathbf{M}^+\mathbf{u}_j^+)}\mathbf{u}_i{}^\top\mathbf{M}^+ + 2\gamma^+\sum_{c=1}^{k^+}(\mathbf{u}_i^+)^c \tag{14}$$

$$\frac{\partial L^+(\mathbf{U},\mathbf{U}^+,\mathbf{M}^+)}{\partial_{\mathbf{M}^+}} = -\sum_{i\in\mathcal{N}, j\in\mathcal{N}_i^+} \frac{\exp(-\mathbf{u}_i{}^\top \mathbf{M}^+\mathbf{u}_j^+)}{1+\exp(-\mathbf{u}_i{}^\top \mathbf{M}^+\mathbf{u}_j^+)}\mathbf{u}_i\mathbf{u}_j^{+\top} + \delta^+\sum_{c=1}^{k}\sum_{d=1}^{k^+} sign(\mathbf{M}^+_{cd}) \tag{15}$$

As the ℓ_1 norm regularization term in Eq. (12) is not differentiable at zero, we use the *sign* function in the last term of Eq. (15), with $sign(\mathbf{M}^+_{ij}) = 1$ if $\mathbf{M}^+_{ij} > 0$, $sign(\mathbf{M}^+_{ij}) = -1$ if $\mathbf{M}^+_{ij} < 0$, and 0 otherwise. With the gradients in Eqs.(13)-(15) we optimize the objective function in Eq. (12) via alternating optimization, to compute the trust-preference user factor matrix $\mathbf{M}^+$, and the user factor matrices $\mathbf{U}$ and $\mathbf{U}^+$. Then, based on Eq. (8), we compute the weighting matrix $\mathbf{W}^+$ to capture users' preferences with friends' trust degrees.

4.2.2 Distrust-based Weighting Matrix. To compute the weighting matrix $\mathbf{W}^-$, we calculate the correlations of users' preferences with foes' distrust degrees. We define an intermediate *distrust-preference user factor* matrix as follows:

Definition 6 (Distrust-Preference User Factor Matrix). *Given the k and k^- latent dimensions of the n users in $\mathbf{U}$ and $\mathbf{U}^-$, the intermediate distrust-preference user factor matrix $\mathbf{M}^- \in \mathbb{R}^{k\times k^-}$ contains the correlations of users' preferences with foes' distrust degrees in an intermediate user latent space, subject to:*

$$\mathbf{W}^- = \mathbf{U}\times\mathbf{M}^-\times\mathbf{U}^{-\top} \tag{16}$$

Accordingly, we have the similarity function s in the latent space of $\mathbf{U}$, and the respective similarity functions s^- and sm^- in $\mathbf{U}^-$ and $\mathbf{M}^-$ as follows:

$\forall i \in \mathcal{N}$ and $v \in \mathcal{N}_i^-$

$$s(\mathbf{u}_i,\mathbf{u}_v) = \frac{1}{1+\exp(-\mathbf{u}_i{}^\top\mathbf{u}_v)} \tag{17}$$

$$s^-(\mathbf{u}_i^-,\mathbf{u}_v^-) = \frac{1}{1+\exp(-\mathbf{u}_i^{-\top}\mathbf{u}_v^-)} \tag{18}$$

$$sm^-(\mathbf{u}_i,\mathbf{u}_v^-) = \frac{1}{1+\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)} \tag{19}$$

The similarity of the foes' latent vectors have to be ***minimized*** in the user preference latent space in $\mathbf{U}$ and in the intermediate space in $\mathbf{M}^-$, as foes do not have similar preferences [28, 29]. Instead, as users i and v are foes we have to ***maximize*** their similarities in the distrust space of $\mathbf{U}^-$, to express their distrust degree. First we transform the maximization problem of foes' similarity in $\mathbf{U}^-$ as a minimization problem, using the negative natural logarithmic function $-\sum_{i,v}\log(s(\mathbf{u}_i,\mathbf{u}_v))$. Hence, we formulate the following objective function as a minimization problem to compute $\mathbf{U}$, $\mathbf{U}^-$ and $\mathbf{M}^-$:

$$\min_{\mathbf{U},\mathbf{U}^-,\mathbf{M}^-} L^-(\mathbf{U},\mathbf{U}^-,\mathbf{M}^-) = \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} s(\mathbf{u}_i,\mathbf{u}_v) - \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-}\log(s^-(\mathbf{u}_i^-,\mathbf{u}_v^-)) + \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} sm^-(\mathbf{u}_i,\mathbf{u}_v^-) + \lambda^-\sum_{c=1}^{k}||\mathbf{u}_c||_2 + \gamma^-\sum_{c=1}^{k^-}||\mathbf{u}_c^-||_2 + \delta^-||\mathbf{M}^-||_1 \tag{20}$$

The respective gradients of the objective function L^- are equal to:

$$\frac{\partial L^-(\mathbf{U},\mathbf{U}^-,\mathbf{M}^-)}{\partial_{\mathbf{u}_i}} = \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} \frac{\exp(-\mathbf{u}_i{}^\top\mathbf{u}_v)}{[1+\exp(-\mathbf{u}_i{}^\top\mathbf{u}_v)]^2}\mathbf{u}_v + \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} \frac{\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)}{[1+\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)]^2}\mathbf{M}^-\mathbf{u}_v^- + 2\lambda^-\sum_{c=1}^{k}(\mathbf{u}_i)^c \tag{21}$$

$$\frac{\partial L^-(\mathbf{U},\mathbf{U}^-,\mathbf{M}^-)}{\partial_{\mathbf{u}_i^-}} = -\sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} \frac{\exp(-\mathbf{u}_i^{-\top}\mathbf{u}_v^-)}{1+\exp(-\mathbf{u}_i^{-\top}\mathbf{u}_v^-)}\mathbf{u}_v^- + \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} \frac{\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)}{[1+\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)]^2}\mathbf{u}_i{}^\top\mathbf{M}^- + 2\gamma^-\sum_{c=1}^{k^-}(\mathbf{u}_i^-)^c \tag{22}$$

$$\frac{\partial L^-(\mathbf{U},\mathbf{U}^-,\mathbf{M}^-)}{\partial_{\mathbf{M}^-}} = \sum_{i\in\mathcal{N}, v\in\mathcal{N}_i^-} \frac{\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)}{[1+\exp(-\mathbf{u}_i{}^\top\mathbf{M}^-\mathbf{u}_v^-)]^2}\mathbf{u}_i\mathbf{u}_v^{-\top} + \delta^-\sum_{c=1}^{k}\sum_{d=1}^{k^-} sign(\mathbf{M}^-_{cd}) \tag{23}$$

Similarly, with the gradients in Eq.(21)-(23) we optimize Eq. (20) and calculate the three matrices/variables $\mathbf{U}$, $\mathbf{U}^-$ and $\mathbf{M}^-$. Having computed the three matrices we calculate the weighting matrix $\mathbf{W}^-$ in Eq. (16) with users' preferences and foes' distrust degrees.

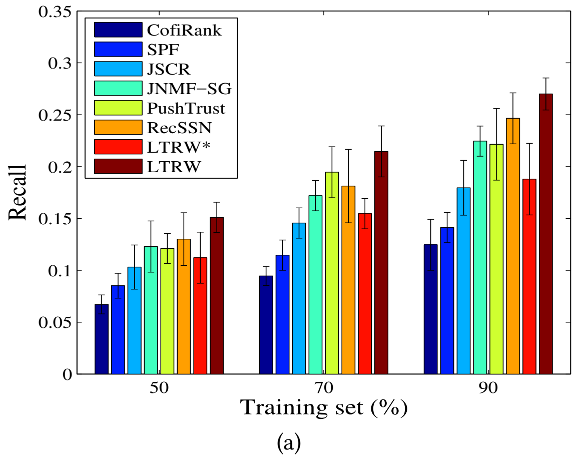
5 EXPERIMENTS

5.1 Evaluation Setup

We evaluate our experiments on a real-world dataset from Epinions[3] [7], also used in related studies [5, 6, 19]. This dataset contains n=119,867 users, m=676,436 product-items and 12,328,927 ratings, with 452,123 trust and 92,417 distrust relationships. We split the dataset into the training and test sets. Unless stated otherwise, in our experiments we consider the 70% of the ratings as training set, and exploit all trust and distrust relationships. To determine the parameters of each examined model, the training set is further split into two subsets: the cross-validation training set and the cross-validation test set.

To remove user bias from our results, we consider an item as relevant if a user has rated it above her average ratings, and irrelevant otherwise. For users with less than five ratings, we consider

[3] http://www.trustlet.org/epinions.html



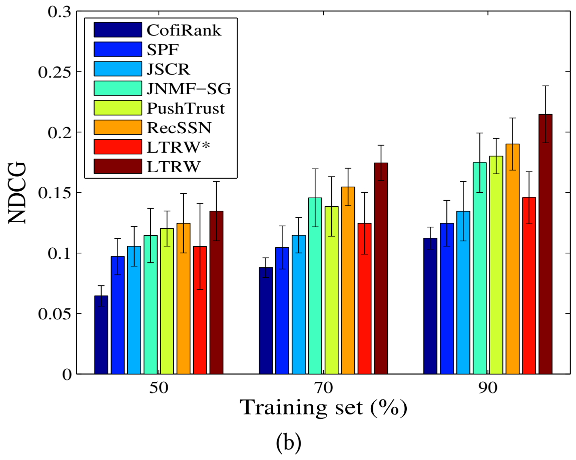


Figure 1: Effect on (a) recall and (b) NDCG, when varying the size of the training set.

the most highly rated item as relevant. To evaluate the top-N recommendation performance of the examined models we report the ranking-based metrics recall and Normalized Discounted Cumulative Gain (NDCG) [4, 21, 33]. Recall is the ratio of the relevant items in the top-N ranked list over all the relevant items for each user. NDCG measures the ranking of the relevant items in the top-N list. For each user the Discounted Cumulative Gain (DCG) is defined as: $DCG@N = \sum_{j=1}^{N} \frac{2^{rel_j}-1}{\log_2 j+1}$, where rel_j represents the relevance score of item j, that is binary in our case, i.e., relevant or irrelevant. NDCG is the ratio of DCG/iDCG, where iDCG is the ideal DCG value given the ratings in the test set. We repeated our experiment five times, and we report mean values and standard deviations of recall and NDCG over the five runs.

5.2 Compared Methods

We compare our proposed approach with the following baseline methods:

- **CofiRank** [30]: a baseline learning to rank model that ignores users' trust and distrust relationships.
- **SPF** [3]: a trust-based probabilistic model that performs Social Poisson factorization, to match user latent preferences for items with the latent influences of her friends. This model considers only trust relationships.
- **JSCR** [21]: a trust-based learning to rank model that focuses on the top of the list, accounting how well the selections of trust friends have been ranked. This model exploits only trust relationships as well.

We also evaluate the performance of the following trust and distrust-based models:

- **JNMF-SG** [19]: a method that co-factorizes user-item and user-cluster associations, by partitioning users into clusters with a spectral clustering approach based on users' trust and distrust relationships.
- **PushTrust** [5]: a recommendation strategy to compute the latent features of users, based on users' trust and distrust relationships. PushTrust also considers the neutral relationships, aiming to rank the neutral users' latent features after the friends' latent features and before those of the foes.
- **RecSSN** [26]: a recommendation method in social signed networks, that considers trust and distrust relationships when generating recommendations. RecSSN captures both local and global information from the signed graph and then exploits both types of information in a matrix factorization technique with users' preferences.
- **LTRW***: a variant of our approach that avoids the weighting strategy of Section 4.2. In LTRW* we set $\mathbf{W}^+ \leftarrow \mathbf{A}^+$ and $\mathbf{W}^- \leftarrow \mathbf{A}^-$. This variant is used to show the importance of our weighting strategy in our learning to rank model when it is missing.
- **LTRW**: the proposed learning to rank model with our weighting strategy to balance the influences of friends and foes on users' preferences when generating recommendations.

The parameters of the examined methods have been determined via cross-validation (Section 5.1) and in our experiments we report the best results. The parameter analysis of building the trust and distrust-preference user latent spaces in our weighting strategy is further studied in Section 5.6.

5.3 Performance Evaluation

We evaluate the performance of the examined methods by varying the number of ratings in the training set, using all trust and distrust relationships. In this set of experiments we set the length of the recommendation list to top-N=20. Fig. 1(a)-(b) show the effect on recall and NDCG, respectively. We can make the following observations:

- CofiRank has limited performance as it does not exploit users' trust or distrust relationships, thus being negatively affected by the sparsity in users' preferences when reducing the training set sizes. Compared to the baseline method of CofiRank, the trust-based methods SPF and JSCR improve recall and NDCG, as both methods use the selections of trust friends, as well as they capture the users' preferences and the friends' influence in their learning strategies.

- Clearly, the competitive methods of JNMF-SG, PushTrust and RecSSN that generate recommendations with trust and distrust relationships can significantly boost the recommendation accuracy. This indicates that both the selections of friends and foes are important in users' personalized recommendations.
- Evaluated against the proposed LRTW model, the LTRW* variant significantly reduces the recall and NDCG metrics. This occurs because LTRW* does not apply the weighting strategy of Section 4.2, hence it does not learn the influences of users' preferences with friends' trust and foes' distrust degrees. Consequently, the selections of social friends and foes dominate the user's personalized preferences.
- Using the paired t-test ($p < 0.05$), we found out that the proposed LTRW model is superior over all methods, achieving an average relative improvement of 14.67% and 11.25% in terms of recall and NDCG, respectively, compared to the second best method (PushTrust or RecSSN). This happens because LTRW does not only perform the weighting strategy to balance the influences of friends and foes, but also focusses on the ranking performance at the top of the recommendation list. On the contrary, the competitive methods of PushTrust and RecSSN do not account for the ranking performance in their learning strategies.

Fig. 2 shows the effect on recall, with changes in the length of the top-N recommendation list. In this set of experiments we train our models with 70% of the ratings using all social relationships. Again, LTRW beats its competitors in all settings. Evaluated against the second best method, LTRW achieves a relative improvement of 13.48% on average.

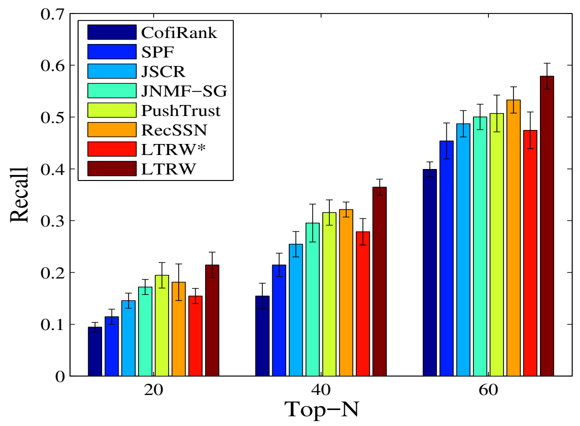


Figure 2: Effect on recall when varying the top-N rec.

5.4 Effect of Missing Social Relationships

In the next set of experiments, we evaluated the performance of the examined models when a part of social relationships is missing. This experiment is performed to measure the performance of models when not all social relationships are available. We ensure that the same percentage of trust and distrust relationships is removed when downsizing the number of relationships. In this set of experiments CofiRank is used for reference as its performance is not affected when varying the percentage of relationships. Fig. 3 presents the effect on recall with top-N=20 with the models trained on 70% of the ratings. We can observe that all social-based models that exploit users' relationships, either trust-based or trust and distrust-based ones, are negatively affected when less relationships are used, having a significant drop on recall. In particular, when exploiting the 75% and 50% of all available relationships the average drops on recall of all social-based methods are 20.26% and 30.94%, respectively. These results indicate that the numbers of trust and distrust relationships play a crucial role in trust-based and trust and distrust-based models when computing recommendations. Evaluated against the second best method our LTRW model maintains the quality of recommendations high, achieving an average improvement of 13.14% in this set of experiments.

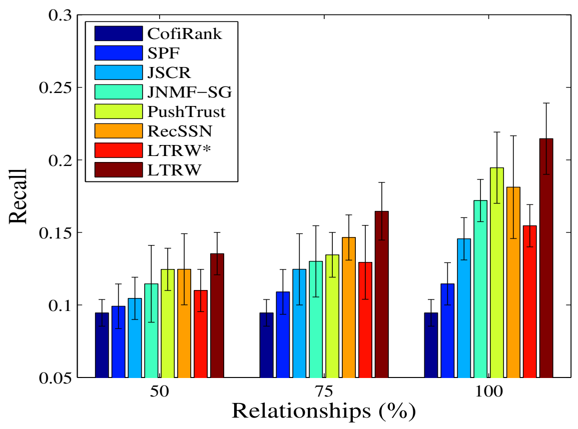


Figure 3: Effect on recall when varying the percentage of social (trust and distrust) relationships.

5.5 Discussion

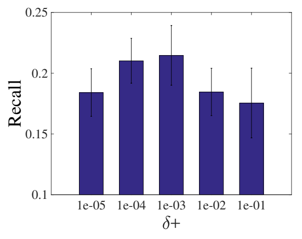
Exploiting both trust and distrust relationships can clearly leverage the recommendation accuracy. As shown in our experiments all trust and distrust-based models that use the selections of friends and foes outperform the baseline model of CofiRank, and the trust-based models of SPF and JSCR. However, the performances of trust and distrust-based models vary by following different strategies when exploiting users' trust and distrust relationships. JNMF-SG first forms a signed graph based on trust and distrust relationships, and then tries to capture the correlations of friends and foes in the clusters of the graph, on condition that friends and foes should not be clustered together. The final recommendations are produced by co-factorizing the cluster-based associations and the users' preferences. Instead, PushTrust considers the latent features of neutral users, that is users that have no relation to a certain user, and then rank the neutral users' latent features below those of the friends' and above the foes' latent features. However, PushTrust does not focus on the ranking performance at the top of the list in a way that ignores the rankings of friends' and foes' selected items in the list, thus having lower recommendation accuracy than the proposed LTRW model. RecSSN calculates the correlations of users and friends/foes in the signed graph based on friends'/foes' circles in the graph. Also, RecSSN exploits the selections of users that have high reputation in the graph, assuming that users trust the selections of users with high reputation. Nonetheless, RecSSN does not focus on the ranking performance when producing recommendation, as well as does not capture well the correlations of users' preferences with trust degrees in friends' circles and distrust degrees in foes' circles, which explains the limited performance of RecSSN, compared to LTRW.

The proposed LTRW model beats all the competitive trust and distrust-based models in all sets of experiments. LTRW significantly boosts the recommendation accuracy by exploiting two key factors. First it focusses at the ranking performance at the top of the list, by taking into account that users' and friends' relevant items should be ranked high in the list, while those of the foes should be ranked low. In addition, the weighting strategy has a significant contribution to our model, to better capture users' preferences with friends' trust and foes' distrust degrees in the intermediate trust and distrust-preference user latent spaces. Our experiments demonstrate the crucial role of our weighting strategy in the proposed LTRW model, as its LTRW* variant has poor performance with an average relative drop of 27.74% in all experiments.

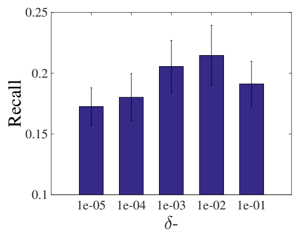
5.6 Analysis of Trust/Distrust-Preference User Latent Spaces

In this section we study the influence of trust and distrust-preference user latent spaces in the LRTW model, corresponding to matrices $\mathbf{M}^+ \in \mathbb{R}^{k \times k^+}$ and $\mathbf{M}^- \in \mathbb{R}^{k \times k^-}$ of Section 4.2. First, we fix the parameters of the ℓ_2 norm regularization terms $\lambda^+ = \gamma^+ = 1e-04$ in Eq. (12) and $\lambda^- = \gamma^- = 1e-04$ in Eq. (20), as we observed that higher and lower values result in model overfitting and underfitting, respectively. Fig. 4(a)-(b) report the effect on recall, when varying parameters δ^+ and δ^-, having impact on the respective ℓ_1 norm regularization terms of $\mathbf{M}^+$ in Eq. (12) and on $\mathbf{M}^-$ in Eq. (20). LTRW achieves the best performance when we set $\delta^+ = 1e-03$, where higher values of δ^+ force $\mathbf{M}^+$ to be more sparse when optimizing Eq. (12). Similarly, we select $\delta^- = 1e-02$ when computing $\mathbf{M}^-$. An interesting observation is that we fix $\delta^+ < \delta^-$, mainly because the intermediate space $\mathbf{M}^+$ with users' preferences and friends' trust degrees is less sparse than the space of $\mathbf{M}^-$ with user' preferences and foes' distrust degrees, as there are less foes than friends (Section 5.1). This complies with several studies reporting that users tend to establish less distrust relationships than trust ones [26–28].

Fig. 5 presents the effect on recall with changes in the user dimensions of the $k \times k^+$ trust-preference space of $\mathbf{M}^+$ and $k \times k^-$ distrust preference space of $\mathbf{M}^-$. In this set of experiments we vary the user latent dimensions from 10 to 100 by a step of 10. As shown in Fig. 5 the best performance is achieved when $k = 60$, $k^+ = 50$ and



(a)



(b)

Figure 4: Effect on recall with changes in (a) δ^+ and (b) δ^-.

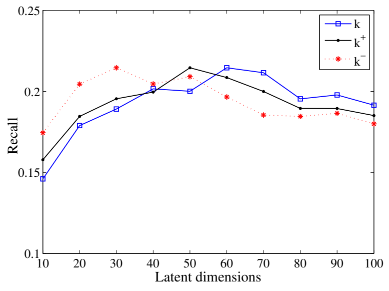


Figure 5: Effect on recall with changes in the user latent dimensions of trust/distrust-preference spaces.

$k^- = 30$, as the model becomes unstable for higher latent dimensions. The model has limited performance when selecting lower latent dimensions, as it does not capture well users' preferences with friends' trust and foes' distrust degrees in the intermediate spaces of $\mathbf{M}^+$ and $\mathbf{M}^-$, respectively.

6 CONCLUSIONS

We presented LTRW, a learning to rank model with trust and distrust relationships, aiming to improve the performance at the top of the list. We introduced a weighting strategy to balance the influences of friends' and foes' selections in our model. In particular we build two intermediate trust/distrust-preference user latent spaces to capture the correlations of users' preferences with friends' trust and foes' distrust degrees, accordingly. Our experiments demonstrate the superiority of the proposed LTRW model over several baselines, achieving a relative improvement of 13.13%, evaluated against the second best model.

Users in recommender systems tend to explore new items, instead of interacting with items they have previously liked in the past, making users' preferences change over time [14, 31]. In addition, while users interact with each other, they establish new relationships and adapt existing ones [2, 27]. This means that users' trust and distrust relationships evolve too. As future work we plan to investigate the performance of our learning to rank model with evolving trust and distrust relationships, to capture users' preference dynamics in recommender systems [22, 23].

ACKNOWLEDGMENTS

Dimitrios Rafailidis was supported by the COMPLEXYS and INFORTECH Research Institutes of University of Mons.

REFERENCES

[1] Charu C. Aggarwal. 2016. *Recommender Systems - The Textbook.* Springer.

[2] Yi Cen, Rentao Gu, and Yuefeng Ji. 2013. Sign Inference for Dynamic Signed Networks via Dictionary Learning. *J. Applied Mathematics* 2013 (2013), 708581:1–708581:10.

[3] Allison June-Barlow Chaney, David M. Blei, and Tina Eliassi-Rad. 2015. A Probabilistic Model for Using Social Networks in Personalized Item Recommendation. In *RecSys.* 43–50.

[4] Konstantina Christakopoulou and Arindam Banerjee. 2015. Collaborative Ranking with a Push at the Top. In *WWW.* 205–215.

[5] Rana Forsati, Iman Barjasteh, Farzan Masrour, Abdol-Hossein Esfahanian, and Hayder Radha. 2015. PushTrust: An Efficient Recommendation Algorithm by Leveraging Trust and Distrust Relations. In *RecSys.* 51–58.

[6] Rana Forsati, Mehrdad Mahdavi, Mehrnoush Shamsfard, and Mohamed Sarwat. 2014. Matrix Factorization with Explicit Trust and Distrust Side Information for Improved Social Recommendation. *ACM Trans. Inf. Syst.* 32, 4 (2014), 17:1–17:38.

[7] Ramanathan V. Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. 2004. Propagation of trust and distrust. In *WWW.* 403–412.

[8] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. In *AAAI.* 123–129.

[9] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. 2013. Low-rank Matrix Completion Using Alternating Minimization. In *STOC.* 665–674.

[10] Mohsen Jamali and Martin Ester. 2009. *TrustWalker*: a random walk model for combining trust-based and item-based recommendation. In *KDD.* 397–406.

[11] Mohsen Jamali and Martin Ester. 2009. Using a trust network to improve top-N recommendation. In *RecSys.* 181–188.

[12] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *RecSys.* 135–142.

[13] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD.* 426–434.

[14] Yehuda Koren. 2010. Collaborative filtering with temporal dynamics. *Commun. ACM* 53, 4 (2010), 89–97.

[15] Artus Krohn-Grimberghe, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2012. Multi-relational Matrix Factorization Using Bayesian Personalized Ranking for Social Network Data. In *WSDM.* 173–182.

[16] Hao Ma, Irwin King, and Michael R. Lyu. 2009. Learning to recommend with social trust ensemble. In *SIGIR.* 203–210.

[17] Hao Ma, Michael R. Lyu, and Irwin King. 2009. Learning to recommend with trust and distrust relationships. In *RecSys.* 189–196.

[18] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. 2008. SoRec: social recommendation using probabilistic matrix factorization. In *CIKM.* 931–940.

[19] Dimitrios Rafailidis. 2016. Modeling trust and distrust information in recommender systems via joint matrix factorization with signed graphs. In *SAC.* 1060–1065.

[20] Dimitrios Rafailidis and Fabio Crestani. 2016. Collaborative Ranking with Social Relationships for Top-N Recommendations. In *SIGIR.* 785–788.

[21] Dimitrios Rafailidis and Fabio Crestani. 2016. Joint Collaborative Ranking with Social Relationships in Top-N Recommendation. In *CIKM.* 1393–1402.

[22] Dimitrios Rafailidis and Alexandros Nanopoulos. 2014. Modeling the dynamics of user preferences in coupled tensor factorization. In *RecSys.* 321–324.

[23] Dimitrios Rafailidis and Alexandros Nanopoulos. 2016. Modeling Users Preference Dynamics and Side Information in Recommender Systems. *IEEE Trans. Systems, Man, and Cybernetics: Systems* 46, 6 (2016), 782–792.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI.* 452–461.

[25] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. 2012. CLiMF: Learning to Maximize Reciprocal Rank with Collaborative Less-is-more Filtering. In *RecSys.* 139–146.

[26] Jiliang Tang, Charu C. Aggarwal, and Huan Liu. 2016. Recommendations in Signed Social Networks. In *WWW.* 31–40.

[27] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. 2016. A Survey of Signed Network Mining in Social Media. *ACM Comput. Surv.* 49, 3 (2016), 42:1–42:37.

[28] Jiliang Tang, Xia Hu, Yi Chang, and Huan Liu. 2014. Predictability of Distrust with Interaction Data. In *CIKM.* 181–190.

[29] Patricia Victor, Chris Cornelis, Martine De Cock, and Ankur Teredesai. 2011. Trust- and Distrust-Based Recommendations for Controversial Reviews. *IEEE Intelligent Systems* 26, 1 (2011), 48–55.

[30] Markus Weimer, Alexandros Karatzoglou, Quoc V. Le, and Alexander J. Smola. 2007. COFI RANK - Maximum Margin Matrix Factorization for Collaborative Ranking. In *NIPS.* 1593–1600.

[31] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff G. Schneider, and Jaime G. Carbonell. 2010. Temporal Collaborative Filtering with Bayesian Probabilistic Tensor Factorization. In *SDM.* 211–222.

[32] Xiwang Yang, Harald Steck, Yang Guo, and Yong Liu. 2012. On Top-k Recommendation Using Social Networks. In *RecSys.* 67–74.

[33] Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging Social Connections to Improve Personalized Ranking for Collaborative Filtering. In *CIKM.* 261–270.