Computer Science and Information Systems 18(1):93–114 https://doi.org/10.2298/CSIS200608039H

TrustRec: An Effective Approach to Exploit Implicit Trust and Distrust Relationships along with Explicit ones for Accurate Recommendations[⋆]

Masoud Reyhani Hamedani, Irfan Ali, Jiwon Hong, and Sang-Wook Kim[*]

Department of Computer and Software, Hanyang University,
Seoul, Korea, 04763
{masoud,irfan.ali,nowiz,wook}@hanyang.ac.kr

Abstract. Trust-aware recommendation approaches are widely used to mitigate the cold-start problem in recommender systems by utilizing trust networks. In this paper, we *point out* the problems of existing trust-aware recommendation approaches as follows: (P1) exploiting *sparse* explicit trust and distrust relationships; (P2) considering a *misleading* assumption that a user pair having a trust/distrust relationship *certainly* has a similar/dissimilar preference in practice; (P3) employing the *transitivity* of *distrust* relationships. Then, we propose *TrustRec*, a novel approach based on the matrix factorization that provides an effective *solution* to each of the aforementioned problems and incorporates all of them in a *single* matrix factorization model. Furthermore, TrustRec exploits *only* top-*k* most similar trustees and dissimilar distrustees of each user to improve both the computational cost and accuracy. The results of our extensive experiments demonstrate that TructRec outperforms existing approaches in terms of *both* effectiveness and efficiency.

Keywords: Recommender Systems, Collaborating Filtering, Trust Network, Matrix Factorization, Distrust Intransitivity

1. Introduction

Although collaborative filtering (CF) is a well-known technique in recommender systems [1, 2, 6], it performs *poorly* for users who have rated a *very small* number of items; this problem is called as a *cold-start user* problem. In the literature, several CF approaches have been proposed to solve this problem by exploiting the additional information such as social information [4–6, 8, 11, 16–20, 23–25], demographic information [21], crowd source information [14], uninteresting items [10], and location information [28] along with the ratings information. In particular, the *trust-aware* recommendation approaches [4, 5, 11, 16–19, 24, 25, 27] exploit a *trust network*, which is kind of social information.

In a trust network, users can establish two types of relationships: *trust* and *distrust*. Trust/distrust is a unidirectional relationship between two users, indicating that one *agrees/ disagrees* on the opinion of the other on some items where *explicit* trust/distrust relationships are established by users *themselves*, while the *implicit* ones are *inferred* based on

[⋆] The initial idea has been presented with some preliminary experimental results in ACM SAC 2017 [3]. The current paper is an extended version that contains some new ideas, formulations, and more extensive experimental results.
Corresponding author: Sang-Wook Kim (wook@hanyang.ac.kr)

some *evidence* such as the degree of similarity in users' preferences [16] or the transitivity of other trust/distrust relationships [12]. A user who trusts someone is called a *trustor* and a user who is being trusted by someone else is called a *trustee*; a user who distrusts someone is called a *distruster* and a user who is being distrusted by someone else is called a *distrustee*. Fig. 1 shows a sample trust network where thick and dotted thick arrows indicate explicit trust and distrust relationships, respectively (i.e., A is an explicit trustor of B while B is an explicit trustee of A). The thin arrow shows an *implicit* trust relationship between two users inferred by *transitivity*; F is an implicit trustor of B and B is an implicit trustee of F since F trusts A and A trusts B.

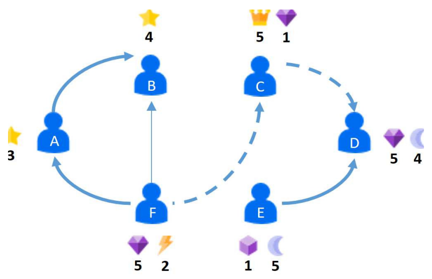

Fig. 1. A sample trust network

Some of the existing trust-aware recommendation approaches exploit *only* trust relationships [11,12,17,19] and some approaches exploit *both* trust and distrust relationships; the latter approaches achieve *better* accuracy than the former ones [4,5,16,18,24,25]. In this paper, we first point out the problems in the existing approaches as follows:

Problem 1: as shown in Table 1, which summarizes the statistics of three real-world datasets, *not* only ratings (i.e., R-density) but also explicit trust and distrust relationships (i.e., T/D-density) are *sparse*. Therefore, the approaches exploiting *only* explicit relationships may *fail* to infer the true preferences of users, in particular, in the case of cold-start users. To solve this problem, PushTrust [4], JNMF-SG [24], and Impl [16] infer implicit trust and distrust relationships and exploit them along with the explicit ones. Impl exploits implicit and explicit trust/distrust relationships *separately* in *two* different models, thus *unable* to solve the sparsity of explicit relationships. Although PushTrust and JNMF-SG exploit both implicit and explicit relationships in a same model, they still *suffer* from Problem 2, which will be explained below.

Problem 2: existing approaches assume that a user pair having an explicit trust/distrust relationship has a similar/dissimilar preference. However, they *neglect* the fact that some of the user pairs having explicit trust/distrust relationships may have dissimilar/similar preferences *in practice* as we show it in Section 3.2; likewise, the same situation may exist for implicit relationships. The similarity in preferences between users having explicit or implicit trust/distrust relationships need to be examined *before* being used in inferring users' preferences.

Problem 3: It has been shown that distrust relationships are intransitive, while trust relationships are transitive [7]. However, MF-TD [5], PushTrust [4], Impl [16], JNMF-SG [24], and RecSSN [25] consider distrust relationships as transitive ones, which may

Table 1. Statistics of some datasets

	Epinions	FilmTrust	Ciao
# Users	132,492	1,508	7,375
# Items	755,760	2,071	99,746
# Ratings	13,668,320	35,497	278,483
# R-density	0.015%	1.136%	0.037%
# Trusts	717,667	1,853	111,781
# Distrusts	123,705	0	0
# T/D-density	0.005%	0.081%	0.205%

cause *inaccurate inference* of a user's preferences. We clarify this problem with the following example. In Fig. 1, user *F* explicitly distrusts user *C* who explicitly distrusts user *D*. By transitivity of distrust relationships, user *F* should *implicitly* distrust user *D*, which means they have dissimilar preferences; however, as shown in Fig. 1, their ratings on the common item (i.e., diamond) are both 5 that indicates they have likely similar preferences.

In addition to these problems, to address the sparsity of explicit trust/distrust relationships, PushTrust [4] and JNMF-SG [24] exploit *all* implicit and explicit relationships of *each* user, which requires a *significantly high* computational cost (e.g., the complexity of PushTrust is $\mathcal{O}(n^2)$ where *n* denotes the number of users). Furthermore, as we show in Section 4, this problem also leads to inaccurate recommendations.

In this paper, we propose a novel trust-aware recommendation approach called *TrustRec*, which provides *effective solutions* to the three aforementioned problems as *inferring implicit relationships (IIR)*, *confirming trustees and distrustees (CTD)*, and *employing distrust's intransitivity (EDI)*, respectively. The IIR solution infers implicit trust/distrust relationships between a pair of users by computing the similarity between their rating, which increases the *density* of trust and distrust relationships a lot, thereby enabling us to infer users' preferences effectively. The CTD solution *confirms* the degree of similarity/dissimilarity of user pairs having explicit trust/distrust relationships in terms of their ratings. This allows us to infer the preferences of the target user by exploiting *only* her *real* similar trustees and dissimilar distrustees. The EDI solution considers the *intransitivity* of distrust relationships in the trust network. Also, TrustRec selects top-*k most* similar trustees and top-*k most* dissimilar distrustees of the target user from her *all* explicit as well as implicit trustees and distrustees to infer her preferences, which is expected to *not* only reduce the computational cost *but also* to improve the accuracy; we refer to it as the *selecting top-k relationships (STR)* solution. Furthermore, TrustRec utilizes a *novel* matrix factorization model that incorporates *all* the proposed solutions together into a *single* model. We evaluate our TrustRec and each of the four solutions by conducting extensive experiments with a real-world dataset. Our experimental results demonstrate that 1) *each* of our four solutions is *effective* in making accurate recommendations; 2) our TrustRec significantly outperforms existing approaches in terms of *both* effectiveness and efficiency. The contributions of this paper are summarized as follows:

- We point out the problems of existing trust-aware recommendation approaches.
- We propose an effective solution to each of the problems.
- We propose a novel matrix factorization model that incorporates *all* the aforementioned solutions together in a *single* model.

– We conduct extensive experiments with a real-world dataset, evaluating the effectiveness and efficiency of our TrustRec in comparison with existing approaches.

The rest of the paper is organized as follows. We discuss existing trust-aware recommendation approaches and point out their problems in Section 2. In Section 3, we present our solutions in detail and explain how to integrate them under a single matrix factorization model. In Section 4, we evaluate the effectiveness of our solutions in recommendation and also evaluate the effectiveness and efficiency of our TrustRec in comparison with existing approaches. In Section 5, we conclude our paper.

2. Background and Related Work

In this section, we explain CF by focusing on matrix factorization since our approach is built upon it. We also provide a brief overview of existing trust-aware recommendation approaches and point out their problems.

CF is a widely used technique in recommender systems because it is simple, effective, and efficient [2, 6, 22, 27]. Matrix factorization-based CF approaches have lately gained popularity since they scale *linearly* with the numbers of users and items [13]. Suppose $R \in \mathbb{R}^{n\times m}$ denotes a *user-item rating* matrix where each entry R_{ui} represents a rating given by user *u* on item *i*; *n* and *m* represent the total numbers of users and items, respectively; matrix factorization-based CF approaches try to obtain a latent *user* feature matrix *U* and a latent *item* feature matrix *V* such that $U \in \mathbb{R}^{n\times f}$ and $V \in \mathbb{R}^{f\times m}$ can effectively recuperate the rating matrix $R \cong \hat{R} = UV$ where *f* is the number of latent features. Matrix factorization tries to obtain *U* and *V* by minimizing the following objective function:

$$\mathcal{F}(U,V) = \sum_{u=1}^{n}\sum_{i=1}^{m}\left(r_{ui} - U_u^T V_i\right)^2 + \lambda_U \parallel U \parallel_F^2 + \lambda_V \parallel V \parallel_F^2 . \tag{1}$$

where $\|.\|_F^2$ denotes the Frobenius norm. We refer to the first part of Eq. 1 as the *factorization part*, which minimizes the difference between real users' ratings and their corresponding predicted ratings. We refer to the second and third parts as the *regularization part*; λ_U and λ_V are regularization parameters to avoid overfitting [2, 13].

SoRec [19], RSTE [17], SocialMF [12], and TrustMF [27] exploit *only* trust relationships. SoRec [19] is based on a probabilistic graphical model that factorizes a rating matrix and a trust matrix simultaneously by sharing a common user latent feature matrix. RSTE [17] is a linear combination of a traditional matrix factorization-based CF method and a trust-aware recommendation approach. However, both SoRec and RSTE exploit trust relationships in the factorization part of the matrix factorization model. In contrast, SocialMF [12] exploits trust relationships in the regularization part of the matrix factorization model and employs the transitivity of trust relationships. TrustMF [27] maps users into two low dimensional spaces, truster space and trustee space, by factorizing the trust relationship matrix and proposes a matrix factorization model that incorporates the truster model along with the trustee model to address the cold-start user problem.

Later, it has been observed that distrust relationships are also important as trust ones since exploiting them enables to make more accurate recommendations [4,5,16,18,24,25]. The earlier attempts include an approach that exploits trust and distrust relationships *separately* into *two* different models [18] where it is shown that the matrix factorization model

Table 2. Comparison of TrustRec with existing approaches

	MF-TD	PushTrust	JNMF-SG	RecSSN	TrustRec
Problem 1	✗	✓	✓	✗	✓
Problem 2	✗	✗	✗	✗	✓
Problem 3	✗	✗	✗	✗	✓

exploiting only trust relationships performs better than the one that exploiting only distrust relationships. The true effectiveness of exploiting distrust relationships is shown by MF-TD [5], PushTrust [4], JNMF-SG [24], and RecSSN [25] where they exploit both trust and distrust relationships together in a *single* model. Let us compare the latter approaches with our TrustRec in regarding to the three problems explained in Section 1 as follows; Table 2 summarizes the result where a '✗' mark denotes that the approach does not solve the problem and a '✓' mark denotes that the approach solves the problem.

(1) PushTrust and JNMF-SG solve Problem 1. PushTrust infers implicit trust relationships between a user and *all* those users who have no any explicit relationship with her. The explicit distrustees of a user are used to filter out dissimilar users from her explicit and implicit trustees based on their similarity score, which is computed by exploiting their latent feature vectors. JNMF-SG infers implicit trust and distrust relationships by clustering explicit ones using the ratio-cut spectral clustering technique [15] where users in the same cluster are regarded to have implicit trust relationships, while users in different clusters are regarded to have implicit distrust relationships. On the contrary, both MF-TD and RecSSN suffer from Problem 1 since they exploit *only* explicit relationships. (2) MF-TD, PushTrust, JNMF-SG, and RecSSN do not consider the degree of similarity/dissimilarity between users having trust/distrust relationships, thus *all* suffering from Problem 2. (3) MF-TD, JNMF-SG, and RecSSN exploit distrust relationships in the regularization part of a matrix factorization model, thereby employing the transitivity of distrust relationships; they *all* suffer from Problem 3. Although PushTrust exploits only trust relationships in the regularization part, the degree of similarity of users having trust relationships is not confirmed; some of those users may have dissimilar preferences. Therefore, PushTrust may *indirectly* exploit some distrust relationships in the regularization part, thereby suffering from Problem 3. As shown in Table 2, on contrary to the existing approaches, TrustRec addresses *all* the three aforementioned problems by providing an effective solution to each of them. In addition, both PushTrust and JNMF-SG suffer from high computational cost since they exploit all the explicit as well as implicit trust and distrust relationships for every user, which increases their computational cost significantly and also could introduce noise in inferring users' preferences.

3. Proposed Approach

In this section, we present our proposed approach, TrustRec, and each of the IIR, CTD, STR, and EDI solutions in detail. Finally, we present our matrix factorization model.

As shown in Fig. 2, TrustRec provides an effective solution to each of the problems described in Sections 1 and 2. TrustRec infers implicit relationships between users to solve Problem 1, considers the degree of similarity/dissimilarity of users having trust/distrust

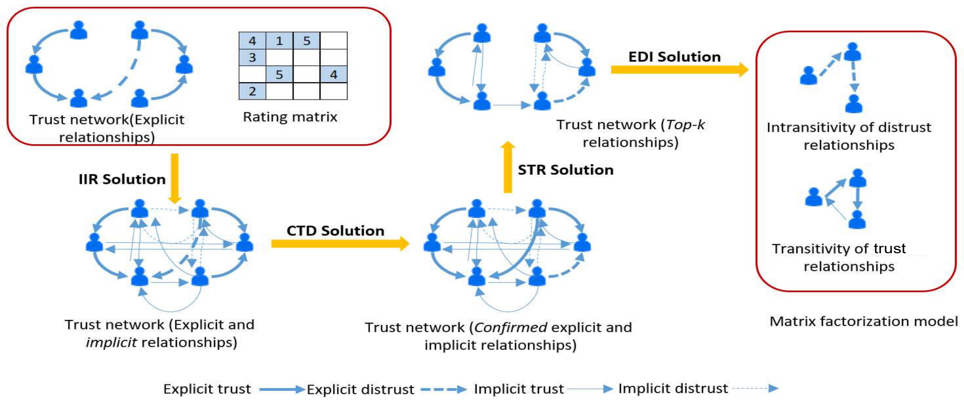
Trust network(Explicit relationships)
Rating matrix
IIR Solution
Trust network (Explicit and *implicit* relationships)
CTD Solution
Trust network (*Confirmed* explicit and implicit relationships)
STR Solution
Trust network (*Top-k* relationships)
EDI Solution
Intransitivity of distrust relationships
Transitivity of trust relationships
Matrix factorization model
Explicit trust
Explicit distrust
Implicit trust
Implicit distrust

Fig. 2. Overview of TrustRec

Table 3. Symbols and their meanings in TrustRec

Symbol	Meaning
$\mathbb{U} = \{u_1, ..., u_n\}$	Set of *n* users
$\mathbb{I} = \{i_1, ..., i_m\}$	Set of *m* items
$R \in \mathbb{R}^{n \times m}$	Sparse rating matrix
f	# of latent features in matrix factorization
$U \in \mathbb{R}^{n \times f}$	Latent features matrix for users
$V \in \mathbb{R}^{f \times m}$	Latent features matrix for items
$S \in \mathbb{R}^{n \times n}$	Similarity matrix for all user pairs
$T \in \{-1, +1\}^{n \times n}$	Explicit relationships matrix
$G \in \{-1, +1\}^{n \times n}$	Inferred implicit relationships matrix
X_u	Row *u* in matrix *X*
$\overline{X}_u$	Average of all values in X_u
$X_{u,v}$	Value at row *u* and column *v* in matrix *X*
$X\,(u)$	Set of column indexes for non-null values in X_u
$X_+(u)$	Set of trustees of user *u* in matrix *X*
$X_-(u)$	Set distrustees of user *u* in matrix *X*
$S_+(u,k)$	Set of top-*k* trustees of user *u*
$S_-(u,k)$	Set of top-*k* distrustees of user *u*

relationships to solve Problem 2, and employs a novel matrix factorization model incorporating distrust relationships into the factorization part which makes it exploit the intransitivity of distrust relationships to solve Problem 3. Furthermore, TrustRec selects only the top-*k* relationships of a user regardless of the relationship type (i.e., explicit or implicit) to improve both efficiency and effectiveness. Given a sparse rating matrix *R* and a trust network *T*, *the problem that our TrustRec tries to solve is to infer accurately the missing values in R*. Table 3 lists all the notations and their meanings used in this paper.

3.1. Inferring Implicit Relationships (IIR)

One of possible solutions to the sparsity problem of explicit relationships is to exploit additional information such as implicit relationships, which is performed by PushTrust, Impl, and JNMF-SG; however, all these approaches suffer from Problem 2 as discussed before. We also solve the sparsity problem by applying the same solution; however, to avoid Problem 2, we infer implicit relationships as follows. We compute the similarity scores between *all* possible users based on their ratings; if the similarity score between a pair of users is *higher/lower* than a given threshold and an explicit trust/distrust relationship does *not* exist between them, we add an implicit trust/distrust relationship between them. We note that Impl also infers implicit relationships based on the similarity score of the users' ratings. However, it selects *topmost* similar/dissimilar users to a user as her implicit trustees/distrustees. Thus, it is likely that some users having negative/positive similarity scores (i.e., having dissimilar/similar preferences) with the user could be selected as her implicit trustees/distrustees; Impl may be *unable* to accurately infer the implicit trustees/distrustees of a user. On the contrary, our IIR solution infers implicit trustees/distrustees of a user *only if* the similarity score between them is greater/smaller than a given threshold, thereby leading to the inference of implicit trustees/distrustees of a user more effectively.

We utilize the Pearson correlation coefficient (PCC) as a similarity measure to compute the similarity between users u and v based on their ratings as follows; we utilized PCC since it is the most commonly used similarity measure in recommender systems [2, 26].

$$P(u,v) = \frac{\sum_{i \in R(u) \cap R(v)} \left(R_{u,i} - \overline{R_u}\right) \cdot \left(R_{v,i} - \overline{R_v}\right)}{\sqrt{\sum_{i \in R(u) \cap R(v)} \left(R_{u,i} - \overline{R_u}\right)^2} \cdot \sqrt{\sum_{i \in R(u) \cap R(v)} \left(R_{v,i} - \overline{R_v}\right)^2}}. \tag{2}$$

By following [2] and [9], to make the similarity score more reliable, we compute the similarity between two users *only if* they have at least *h* co-rated items; if the number of co-rated items is less than *h*, their implicit relationship is *not* inferred. Let $G \in \{-1, +1\}^{n \times n}$ be a matrix that represents the inferred *implicit* relationships between users where $G_{u,v} = 1$ and $G_{u,v} = -1$ denote implicit trust and distrust relationships between users *u* and *v*, respectively. We infer the implicit relationships between *u* and *v* as follows:

$$G_{u,v} = \begin{cases} 1, & P(u,v) > \sigma_T \text{ and } T_{u,v} = \textit{null}. \\ -1, & P(u,v) < \sigma_D \text{ and } T_{u,v} = \textit{null}. \end{cases} \tag{3}$$

where σ_T/σ_D denotes a threshold to ensure that the similarity/dissimilarity between *u* and *v* is sufficient to have an implicit trust/distrust relationship and $T_{u,v} = null$ indicates that the explicit relationship does *not* exist between *u* and *v*.

3.2. Confirming Trustees and Distrustees (CTD)

In the Epinions dataset, we analyzed the similarity scores between all user pairs having explicit relationships, presented in Table 4. Surprisingly, $45,208$ (i.e., 6.3%) out of $717,667$ user pairs having an explicit trust relationship have *negative* similarity scores, which indicates that these users actually have *dissimilar* preferences. Also, $19,498$ (i.e.,

15.76%) out of $123,705$ user pairs having an explicit distrust relationship have *positive* similarity scores, which indicates that these users actually may have *similar* preferences. These results show that, even if users *explicitly* trust/distrust each other, they may have dissimilar/similar preferences in practice. Therefore, it seems *necessary* to *confirm* the degree of similarity/dissimilarity of users having explicit trust/distrust relationships.

Table 4. Analysis on explicit relationships in the Epinions dataset

# Trust relationships	# user-pairs with similarity scores	# user-pairs with negative similarity scores
717,66	523,983	45,208
# Distrust relationships	# user-pairs with similarity scores	# user-pairs with positive similarity scores
123,705	64,175	19,498

In our TrustRec, the similarity/dissimilarity of users having implicit trust/distrust relationships are confirmed *by default* since implicit relationships are inferred based on the similarity score between users' ratings. Therefore, we *only* need to confirm the degree of similarity/dissimilarity of users having explicit trust/distrust relationships by utilizing the similarity scores, which are *already* computed in the IIR solution: for a user pair having an explicit relationship, if their similarity score is greater than σ_T, we consider it as a *trust* relationship; if their similarity score is less than σ_D, we consider it as a *distrust* relationship. To incorporate this solution in our approach, we modify matrix *T* as follows:

$$T_{u,v} = \begin{cases} 1, & P(u,v) > \sigma_T \text{ and } T_{u,v} \neq \textit{null}. \\ -1, & P(u,v) < \sigma_D \text{ and } T_{u,v} \neq \textit{null}. \end{cases} \quad (4)$$

where $T_{u,v} = 1$ and $T_{u,v} = -1$ show explicit trust and distrust relationships between users *u* and *v,* respectively.

3.3. Selecting Top-*k* Relationships (STR)

PushTrust [4] and JNMF-SG [24] exploit *all* implicit and explicit relationships of every user, which increases the *computation cost* significantly and also may introduce *noise* in inferring users' preferences [11]. To solve this problem, we select *only* the top-*k* trust and distrust relationships of each user *regardless* of their relationship type (i.e., implicit or explicit). The top-*k* trust and distrust relationships of a target user are those relationships with *k* trustees and *k* distrustees who have the highest and lowest similarity scores with her, respectively. However, selecting top-*k* relationships is *challenging* since if two users involved in an explicit relationship have *less* than *h* co-rated items, their similarity score *cannot* be computed as explained in Section 3.1. Ignoring those user pairs may make the sparsity problem of explicit relationships more serious. Hereafter, we refer to a pair of users who have an explicit relationship with *less* than *h* co-rated items as an *unconfirmed user pair* and refer to the user in that pair as an *unconfirmed* trustor, distruster, trustee, or distrsutee depending on her role in the relationship. Also, we refer to a pair of users who have an explicit relationship with *more* than *h* co-rated items as a *confirmed user pair* and refer to the user in that pair as a *confirmed* trustor, distruster, trustee, or distrsutee. The similarity score between users in a confirmed user pair is computed easily by Eq. (2). We estimate the similarity score for unconfirmed user pairs as follows.

Let *u* and *v* be an unconfirmed user pair where *u* is an unconfirmed trustor and *v* is an unconfirmed trustee. We regard that the preference of *v* is similar to *u* as much as those of *u*'s confirmed trustees; in other words, the degree of similarity between a user and her confirmed trustees are exploited to estimate the similarity score between the user and her unconfirmed trustee. We consider *two* possible situations as follows: first, *u* has some confirmed trustees (i.e., $S(u) \bigcap T_+(u) \neq \emptyset$); second, *u* has *no* confirmed trustees and her explicit trustees are *all* unconfirmed ones (i.e.,$S(u) \bigcap T_+(u) = \emptyset$). Note that, according to Table 3, $S(u) \bigcap T_+(u)$ denotes the confirmed trustees of *u*. In the first situation, we estimate the similarity score between *u* and *v*, $S(u, v)$, as follows:

$$S(u,v) = \frac{\sum_{z \in S(u) \bigcap T_+(u)} P(u,z)}{|S(u) \bigcap T_+(u)|}. \tag{5}$$

where $P(u, z)$ is the similarity score between users *u* and *z* computed by Eq. (2); we regard the *average* of similarity scores between *u* and all her confirmed trustees as the estimated similarity score between *u* and *v*.

In the second situation, since the number of confirmed trustees of u is equal to zero, we estimate the similarity between *u* and *v* as follows:

$$S(u,v) = \frac{\sum_{y \in \mathbb{U}} \sum_{z \in S(y) \bigcap T_+(y)} P(y,z)}{\sum_{y \in \mathbb{U}} |S(y) \bigcap T_+(y)|}. \tag{6}$$

where we regard the *average* of the similarity scores between two users in *all* the existing confirmed user pairs having explicit trust relationship as the estimated similarity score between *u* and *v*.

Let *u* and *w* be an unconfirmed user pair where *u* is an unconfirmed distruster and *w* is an unconfirmed distrustee. We regard that the preference of *w* is dissimilar to *u* as much as those of *u*'s confirmed distrustees; in other words, the degree of dissimilarity between a user and her confirmed distrustees are exploited to estimate the similarity score between the user and her unconfirmed distrustees. We consider *two* possible situations as follows: first, *u* has some confirmed distrustees (i.e., $S(u) \bigcap T_-(u) \neq \emptyset$); second, *u* has *no* confirmed distrustees and her explicit distrustees are *all* unconfirmed ones (i.e., $S(u) \bigcap T_-(u) = \emptyset$). Note that, according to Table 3, $S(u) \bigcap T_-(u)$ denotes the confirmed distrustees of user *u*. In the first situation, $S(u, w)$ is estimated as follows:

$$S(u,w) = \frac{\sum_{z \in S(u) \bigcap T_-(u)} P(u,z)}{|S(u) \bigcap T_-(u)|}. \tag{7}$$

where we regard the *average* of similarity scores between *u* and *all* her confirmed distrustees as the estimated similarity score between *u* and *w*.

In the second situation (i.e., $S(u) \bigcap T_-(u) = \emptyset$), we estimate the similarity between *u* and *w* as follows:

$$S(u,w) = \frac{\sum_{y \in \mathbb{U}} \sum_{z \in S(y) \bigcap T_-(y)} P(y,z)}{\sum_{y \in \mathbb{U}} |S(y) \bigcap T_-(y)|}. \tag{8}$$

where we regard the *average* of the similarity scores between two users in *all* the existing confirmed user pairs having explicit distrust relationship as the estimated similarity score between *u* and *w*.

Once a similarity score is assigned to every user pair having a relationship, we select the top-*k* trust/distrust relationships of a user according to the similarity scores between the user and her trustees/distrustees sorted in *descending/ascending* order; we select *equal numbers* of trust and distrust relationships of a user by following previous work [4,5,24,25] where it has been shown that exploiting equal numbers of trust and distrust relationships provides the best accuracy. As shown in Section 4, applying the STR solution reduces the computational cost. In addition, it improves the accuracy since it somehow *refines* the trustees/distrustees to a target user and only exploits those trustees/distrustees who are highly similar/dissimilar to her to infer her preferences.

3.4. Employing Distrust's Intransitivity (EDI)

As discussed earlier, employing the transitivity of distrust relationships may cause to treat implicit trustees of a user as her implicit distrustees in a wrong way. As a result, a user's preferences may be inferred incorrectly, thereby leading to inaccurate recommendations. Moreover, it has been shown that trust is transitive while distrust is intransitive [7]. Therefore, it should be a natural choice to employ the transitivity of trust and intransitivity of distrust in recommendation. To employ the intransitivity of distrust relationships, we exploit them in the factorization part of Eq. (1) as follows:

$$\min_{U,V}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(R_{i,j}-U_i^TV_j-\lambda_D\left(\frac{\sum_{v\in S_-(u,k)}U_v}{|S_-(u,k)|}\right)V_j\right)^2+\lambda_U\|U_i\|_F^2+\lambda_V\|V_j\|_F^2\,. \quad (9)$$

where λ_D as a parameter controls the importance of distrustees' latent feature vectors.

By exploiting distrust relationships in the factorization part, latent feature vectors of distrustees are utilized *only* to predict the target user's original ratings, then those predicted ratings are used to infer users' latent feature vectors. Since the distrustees of a target user can *directly* affect only her original ratings rather than her latent feature vector, our proposed approach *employs* the intransitivity of distrust relationships successfully.

3.5. Unified Matrix Factorization Model

To incorporate the four aforementioned solutions together, we propose a novel matrix factorization model as follows:

$$\begin{aligned}\min_{U,V}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(R_{i,j}-U_i^TV_j-\lambda_D\left(\frac{\sum_{v\in S_-(u,k)}U_v}{|S_-(u,k)|}\right)V_j\right)^2+\\ \lambda_U\|U_i\|_F^2+\lambda_V\|V_j\|_F^2+\lambda_T\|U_i-\left(\frac{\sum_{v\in S_+(u,k)}U_v}{|S_+(u,k)|}\right)\|_F^2\,. \end{aligned} \quad (10)$$

where λ_T denotes a parameter for controlling the importance of trustees' latent feature vectors. Following SocialMF [12], we add a term into the regularization part of Eq. (9), which minimizes the difference between the latent feature vector of a user and the average of her trustees' latent feature vectors. As a result, the latent feature vector of a user is learned by referring to her trustees, and the latent feature vectors of her trustees are learned by referring to their trustees recursively, thereby making our approach to employ the transitivity of trust relationships.

Moreover, as explained before, our model also employs the intransitivity of distrust relationships. We can find a local minimum of Eq. (10) by performing *gradient descent* on U_u and V_i for all users *u* and all items *i* as follows:

$$\begin{aligned}
\frac{\partial \mathcal{F}}{\partial U_u} =& 2\sum_{u=1}^{n}\left(R_{u,i}-\left(U_u-\lambda_D\left(\frac{\sum_{v\in S_-(u,k)}U_v}{|S_-(u,k)|}\right)\right)^T V_i\right)^2 \\
&\cdot\left(1-\lambda_D\left(\frac{\sum_{v\in S_-(u,k)}U_v}{|S_-(u,k)|}\right)^T V_i\right) \\
&+2\lambda_U U_u+2\lambda_T\left(U_u-\frac{\sum_{v\in S_+(u,k)}U_v}{|S_+(u,k)|}\right) \\
&-\lambda_T\sum_{\{v|u\in S_+(v,k)\}}\left(U_v-\frac{\sum_{w\in S_+(v,k)}U_w}{|S_+(w,k)|}\right) \\
\frac{\partial \mathcal{F}}{\partial V_i} =& 2\sum_{i=1}^{m}\left(R_{u,i}-\left(U_u-\lambda_D\left(\frac{\sum_{v\in S_-(u,k)}U_v}{|S_-(u,k)|}\right)\right)^T V_i\right)^2 \\
&\cdot\left(U_u-\lambda_D\left(\frac{\sum_{v\in S_-(u,k)}U_v}{|S_-(u,k)|}\right)\right)+2\lambda_V V_i\,.
\end{aligned} \tag{11}$$

Our TrustRec infers implicit trust relationships between users *not* only by applying the IIR solution but *also* by employing the transitivity of trust; for simplicity, we call it as an *ETT* solution. However, IIR and ETT solutions solve the problems of *different* sets of cold-start users as summarized in Table 5. In the case of those users having almost a zero number of ratings but a few explicit trust relationships, the ETT solution solves their sparsity problem since it infers implicit trust relationships by employing the transitivity of explicit ones. In the case of those users who have rated a small number of items but do not have explicit relationships, the IIR solution solves their sparsity problem by inferring implicit trust and distrust relationships based on the similarity score between users' ratings. If a particular set of users have both ratings and explicit trust relationships, clearly they would take advantage of both solutions. The worst case could happen when users have neither ratings nor explicit trust relationships, which is out of the scope of this paper[1].

[1] One possible solution to this problem is to exploit content information associated with users such as demographic information [21] and location information [28].

Table 5. Applicability of TrustRec to various sets of cold-start users

# of explicit trust relationships	# of ratings	
	Low	Zero
Low	IIR & ETT	ETT
Zero	IIR	×

4. Evaluation

In this section, we evaluate the effectiveness and efficiency of our TrustRec by performing extensive experiments with a real-world dataset. The objective of our experimental study is to answer the following key questions:

Q_1: What are the best *values* of TrustRec's parameters to get the highest accuracy?
Q_2: Are all solutions (i.e., IIR, CTD, STR, and EDI) in TrustRec really effective to achieve better accuracy?
Q_3: How much accurate is TrustRec for *all users* in comparison with existing approaches?
Q_4: How much accurate is TrustRec for *cold-start users* in comparison with existing approaches?
Q_5: How much is the *training time* of TrustRec in comparison with those of existing approaches?

4.1. Experimental Setup

In our experiments, we employed Epinions since it is a real-world dataset, publicly available, and widely used to evaluate recommender systems in the literature as in references [4], [17], [16], [25], [27], and [12]. Epinions contains users' ratings on items and explicit trust/distrust relationships between users. We used 80% of total ratings as a training set and other 20% as a testing set. All required codes were implemented in Java and all the experiments were conducted on an Intel machine equipped with four Core i7-2600K CPUs, 24GB RAM, and a 64-bit Windows 10 operating system. In order to evaluate the effectiveness, we utilized the mean average error (MAE) [2] and the root mean squared error (RMSE) [2] as the two well-known accuracy metrics in the literature, which are defined as follows:

$$MAE = \frac{\sum_{(u,i)\in E} |\hat{R}_{u,i} - R_{u,i}|}{|E|}\,. \tag{12}$$

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in E} \left(\hat{R}_{u,i} - R_{u,i}\right)^2}{|E|}}\,. \tag{13}$$

where *E* denotes the set of ratings in the testing set and $\hat{R}_{u,i}$ does the predicted rating for user *u* on item *i*.

We compared the effectiveness and efficiency of TrustRec with those of the following approaches:

– *MF*: it is based on matrix factorization and exploits *only ratings* [13].

- *SocialMF*: it exploits ratings and explicit trust relationships [12].
- *RSTE*: it exploits ratings and explicit trust relationships [17]. The difference between SocialMF and RSTE is that SocialMF employs the transitivity of trust relationships while RSTE does *not*.
- *TrustMF*: it exploits ratings and explicit trust relationships [27]. However, it not only factorizes the rating matrix but also factorizes the trust matrix and incorporates both of them in the matrix factorization model.
- *Impl*: it exploits ratings and both implicit trust and distrust relationships [16].
- *RecSSN*: it exploits ratings and both explicit trust and distrust relationships [25].
- *PushTrust*: it exploits ratings along with *both* explicit and implicit trust/distrust relationships [4].

4.2. Experimental Results

In this section, we answer questions Q_1 to Q_5, one by one.

***Q_1*: Parameter Tuning** There are five parameters in Eq. (10) as *f*, λ_D, λ_T, λ_U, and λ_V where *f* denotes the number of latent features, λ_D and λ_T control the importance of latent feature vectors of distrustees and trustees of a target user, respectively; λ_U and λ_V are the regularization parameters. Also, *h*, σ_T, σ_D, and *k* are other important parameters used in our proposed solutions; the similarity score between two users is computed if they have at least *h* co-rated items; in the IIR solution, σ_T and σ_D are utilized to infer implicit relationships; in the CTD solution, σ_T and σ_D are utilized to confirm the degree of similarity/dissimilarity of users having explicit trust/distrust relationships, respectively; also, in the STR solution, top-*k* trustees/distrustees are selected for each user. Most existing approaches set the values of both λ_U and λ_V as 0.001 to reduce the complexity [4, 5, 12, 16–19, 24, 25]; we follow the same practice. On the contrary, the best values of λ_T and λ_D in existing approaches are quite different and are heavily *dependent* on matrix factorization models; thus we decided to find their best values in our TrustRec. We set the value of *h* as 5 by following [2, 9]. Also, heuristically, we set the values of σ_T and σ_D as 0.1 and -0.1, respectively.

For the rest of parameters, *f*, *k*, λ_T, and λ_D, we employed a two-step approach to determine their best values since finding the best combination among all possible values for these parameters is computationally too expensive. In the first step, while we assigned an *identical* value to λ_T and λ_D (i.e., $\lambda_T = \lambda_D$), we tried to find the best values of *f*, *k*, and λ_T as follows: we set the value of *f* as 5 and 10; for each value of *f*, we set the value of λ_T from 0.1 to 1.0 in step of 0.1; we set the value of *k* as 5, 10, 25, 50, and number of *all* available implicit and explicit relationships. Finally, we measured RMSE and MAE of TrustRec when it was equipped with each of the aforementioned settings. Table 6 shows the results of this parameter where the boldface numbers represent the best accuracy; the best accuracy is observed when *f*, *k*, and λ_T are set as 10, 5, and 0.7, respectively.

In the second step, we tried to find the best *individual* values of λ_T and λ_D as follows: we set $f = 10$ and $k = 5$ based on the result of our parameter tuning in the first step and changed the values of λ_T and λ_D from 0.1 to 1.0 in step of 0.1; then, we measured RMSE and MAE of TrustRec when it was equipped with each of the aforementioned settings. Tables 7 shows the results where the boldface numbers show the best accuracy in the

Table 6. Results of parameter tuning (first step)

	$F = 5$									
	$k = 5$		$k = 10$		$k = 25$		$k = 50$		all	
λ_D=λ_T	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
0.1	0.878	1.090	1.071	1.268	1.111	1.305	0.582	0.782	1.270	1.469
0.2	0.607	0.806	1.121	1.311	0.956	1.162	0.582	0.784	1.099	1.203
0.3	1.046	1.247	1.039	1.242	0.610	0.818	0.640	0.858	1.153	1.371
0.4	0.605	0.803	0.640	0.844	0.676	0.893	0.733	0.963	0.944	1.130
0.5	0.621	0.821	0.657	0.861	0.705	0.920	0.750	0.964	0.863	1.078
0.6	0.613	0.811	0.642	0.843	0.686	0.888	0.732	0.927	0.745	0.941
0.7	0.584	0.782	0.609	0.810	0.650	0.846	0.697	0.878	0.718	0.930
0.8	0.560	0.757	0.568	0.773	0.607	0.809	0.645	0.828	0.708	0.894
0.9	**0.542**	**0.745**	0.543	0.747	0.556	0.766	0.573	0.781	0.655	0.845
1.0	0.544	0.765	0.541	0.763	0.559	0.807	0.548	0.775	0.668	0.887
	$F = 10$									
	$k = 5$		$k = 5$		$k = 25$		$k = 50$		all	
0.1	0.820	1.049	0.596	0.820	0.582	0.800	0.567	0.782	0.858	1.177
0.2	0.612	0.840	0.816	1.026	0.581	0.799	0.570	0.788	0.850	1.103
0.3	0.733	0.965	1.026	1.234	0.610	0.843	0.632	0.880	0.941	1.201
0.4	0.601	0.830	0.608	0.844	0.677	0.940	0.728	1.007	0.733	1.016
0.5	0.573	0.800	0.619	0.850	0.671	0.919	0.721	0.969	0.729	1.000
0.6	0.639	0.838	0.578	0.800	0.614	0.838	0.650	0.866	0.661	0.957
0.7	**0.522**	**0.743**	0.537	0.772	0.549	0.783	0.565	0.799	0.677	0.962
0.8	0.530	0.771	0.546	0.797	0.560	0.820	0.574	0.847	0.686	0.971
0.9	0.569	0.830	0.573	0.837	0.601	0.872	0.647	0.931	0.656	0.950
1.0	0.625	0.898	0.631	0.905	0.610	0.876	0.620	0.891	0.643	0.928

columns and the italic boldface numbers represent the best accuracy in the whole table. As observed, when the values of λ_T and λ_D are *identical* or *very close* to each other, TrustRec provides *high* accuracy; the best accuracy is observed when $\lambda_T = \lambda_D = 0.7$. More specifically, we can assign an identical value to λ_D and λ_T in the range 0.6 to 0.8 or even two values with 0.1 difference in the same range. Table 8 summarizes the final result of our parameter tuning.

Q_2: Effectiveness of Proposed Solutions To answer Q_2, we evaluate TrustRec by removing each of the proposed solutions *one* at a time as follows. To show the effectiveness of the STR solution, we exploit *all* implicit and explicit trustees/distrustees of users; we refer to this version of TrustRec as *TrustRec-S*. To show the effectiveness of the CTD solution, we employ the *top-k* trustees and distrustees for each user without confirming their degree of similarity and dissimilarity with her; we refer to this version as *TrustRec-C*. To show the effectiveness of the IIR solution, we employ *top-k* trustees and distrustees of users that are obtained *only* from their explicit relationships; we refer to it as *TrustRec-I*. To show the effectiveness of the EDI solution, we exploit distrust relationships in the regularization part of the matrix factorization model; we refer to it as *TrustRec-E*.

Table 9 shows the effectiveness of our original TrustRec and its aforementioned versions. The original TrustRec universally outperforms TrustRec-S since the latter one exploits *all* implicit and explicit trustees/ distrustees of a user, some of which may adversely affect the inference of her preferences when their preferences are not that similar/dissimilar to hers. This result also *coincides* with the observation found in Tables 6

Table 7. Results of parameter tuning (second step)

RMSE										
	λ_D									
λ_T	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.1	1.049	**0.784**	0.792	0.799	0.806	0.814	0.821	0.826	0.840	0.854
0.2	**0.782**	0.840	**0.783**	0.972	0.797	0.798	0.806	0.821	0.836	0.831
0.3	0.792	0.792	0.965	**0.783**	0.790	0.791	0.798	0.814	0.841	0.852
0.4	0.801	0.800	0.830	0.799	**0.782**	0.806	0.791	0.806	0.844	**0.809**
0.5	0.808	0.808	0.807	0.807	0.801	**0.782**	0.782	0.798	0.831	0.827
0.6	0.816	0.816	0.815	0.815	0.804	0.838	0.781	0.791	0.896	0.854
0.7	1.113	1.062	0.822	0.821	0.822	0.821	***0.743***	0.781	0.830	0.921
0.8	0.845	0.825	0.925	0.827	0.827	0.827	0.804	**0.771**	0.849	0.985
0.9	0.829	0.936	0.827	0.834	0.830	0.829	0.826	0.847	**0.830**	1.08
1.0	0.850	1.077	1.165	0.828	0.827	0.829	0.827	1.122	1.183	0.898
MAE										
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.1	0.820	**0.565**	0.571	0.574	0.578	0.580	0.598	0.602	0.611	0.629
0.2	**0.565**	0.612	**0.564**	0.571	0.580	0.570	0.590	0.597	0.600	0.616
0.3	0.571	0.571	0.732	**0.564**	0.680	0.563	0.580	0.589	0.596	0.630
0.4	0.580	0.580	0.705	0.601	0.583	0.563	0.570	0.580	0.596	**0.603**
0.5	0.588	0.588	0.588	0.589	**0.573**	**0.559**	0.563	0.571	0.589	0.637
0.6	0.594	0.594	0.595	0.595	0.578	0.639	0.550	0.563	0.641	0.712
0.7	0.870	0.813	0.597	0.600	0.603	0.574	***0.522***	0.563	0.580	0.805
0.8	0.618	0.596	0.690	0.598	0.605	0.607	0.563	**0.530**	0.601	0.893
0.9	0.604	0.702	0.599	0.609	0.600	0.605	0.607	0.601	**0.569**	0.895
1.0	0.616	0.846	0.940	0.651	0.700	0.763	0.800	0.890	0.908	0.625

Table 8. Final results of parameter tuning

	f	λ_D	λ_T	λ_U	λ_V	h	σ_T	σ_D	k
Value	10	0.7	0.7	0.001	0.001	5	0.1	-0.1	5

where as the value of k *increases*, the accuracy of TrustRec *decreases*; the *lowest* accuracy is observed when *all* the implicit and explicit trustees/distrustees are exploited. Also, the STR solution contributes to obtain higher *efficiency* since exploiting only top-k trustees/distrustees requires much less training time than exploiting all of them. TrustRec outperforms TrustRec-C since the latter exploits trustees/distrustees of each user *without* confirming the similarity scores between them and her; if the trustees/distrustees of a user have actually dissimilar/similar preferences with hers, it negatively affects the inference of her preferences. TrustRec shows better accuracy than TrustRec-I because the latter does *not* exploit implicit relationships. TrustRec outperforms TrustRec-E since the latter one employs the transitivity of distrust relationships that causes the implicit trustees of a user to be considered incorrectly as her implicit distrustees, thereby adversely affecting the inference of the user's true preferences. Also, TrustRec-I performs better than TrustRec-E even though both versions exploit only explicit relationships; the reason is that TrustRec-I employs the intransitivity of distrust relationships while TrustRec-E does not.

In summary, *each* of our solutions employed in TrustRec is *effective* in recommendation and contributes to achieve higher accuracy; the *best* accuracy is obtained when *all* the solutions are employed *together* (i.e., the original TrustRec).

Table 9. Effectiveness of TrustRec with/without each of the proposed solutions

	TrustRec	TrustRec-S	TrustRec-C	TrustRec-I	TrustRec-E
RMSE	0.743	0.889	0.779	0.819	0.852
MAE	0.522	0.702	0.555	0.590	0.612

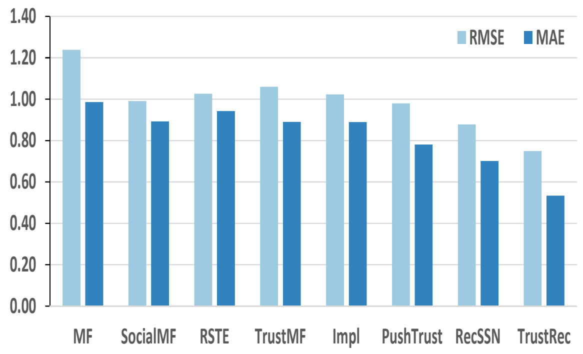

Fig. 3. Accuracy comparison for *all* users

Q_3: Accuracy for All Users To answer Q_3, we compared the accuracy of TrustRec with those of existing approaches explained in Section 4.1; Fig. 3 demonstrates the results. MF shows the *worse* accuracy since, on the contrary to trust-aware recommendation approaches, it exploits *only* ratings; this result clearly shows the power of employing trust networks in recommendation. The approaches exploiting both of trust and distrust relationships (i.e., Impl, RecSSN, PushTrust, and TrustRec) outperform those ones exploiting only trust relationships (i.e., SocialMF, RSTE, and TrustMF); this result shows that distrust relationships are also important as trust ones and exploiting them leads to make more accurate recommendations as already observed in previous work [4] [16] [25]. Among Impl, RecSSN, PushTrust, and TrustRec, Impl performs worst since it exploits *only* implicit relationships between users, which are obtained based on the similarity in their ratings. As already discussed (i.e., in Sections 1 and 2), most users usually give ratings on a very small number of items where the implicit trust and distrust relationships may not be inferred well for them. In addition, Impl employs the transitivity of distrust in recommendation; these two problems would cause its low accuracy.

PushTrust exploits implicit as well as explicit relationships, while RecSSN exploits only explicit relationships; however, PushTrust performs *worse* than RecSSN. The reason is that PushTrust exploits implicit trustees for a user without conforming their degree of similarity with her where some of them may have *dissimilar* preferences with the user (i.e., suffering from Problem 2). On the contrary, although RecSSN exploits only explicit relationships, it exploits a number of explicit trustees/distrustees of a user who may have actual dissimilar/similar preferences with her; therefore, RecSSN performs better than PushTrust. RecSSN shows *lower* accuracy than our TrustRec since RecSSN employs the transitivity of distrust relationships and thus may consider implicit trustees of a user as her implicit distrustees, failing to infer her true preferences. Moreover, it cannot make

Table 10. Accuracy improvement(%) obtained by TrustRec over other methods with all users

	MF	SocialMF	RSTE	TrustMF	Impl	PushTrust	RecSSN
RMSE	39.4	24.3	26.9	29.2	26.6	23.4	15.6
MAE	45.8	40.2	43.3	40.0	39.9	31.5	23.8

accurate recommendations for cold-start users since it only considers the explicit relationships.

In summary, TrustRec *significantly* outperforms *all* existing approaches and shows higher accuracy in terms of both RMSE and MAE due to the following reasons: (1) the availability of a sufficient number of implicit trust and distrust relationships due to the IIR solution; (2) the confirmation of similarity/dissimilarity degree among users having trust/distrust relationships due to the CTD solution; (3) the employment of the transitivity of trust relationships and the intransitivity of distrust ones due to the EDI solution. Furthermore, by applying the STR solution, TrustRec somehow *refines* the similar/dissimilar trustees/distrustees to a target user and only exploits the trustees who are highly similar and the distrustees who are highly dissimilar to her. In other words, when we consider all the trustees/distrustees to a target user, some of them may adversely affect the inference of her preferences since their preferences are not that similar/dissimilar to hers. Table 10 represents the *percentage* of *improvement* in accuracy obtained by TrustRec over other approaches; the average improvement in terms of RMSE and MAE over existing approaches are 26.4% and 37.7%, respectively.

Q_4: Accuracy for Cold-start Users To answer Q_4, we compared the accuracy of TrustRec with those of other approaches in the case of considering *only* cold-start users; by following [4], we chose cold-start users as those having rated at most 20 item. Fig. 4 shows the results; as we expected, by comparing Figs 3 and 4, it is observed that the accuracy of *all* approaches decrease when only cold-start users are considered. However, our findings here *coincide* with the ones observed when all users are considered (i.e., shown in Fig. 3); even when considering only cold-start users, all the approaches outperform MF since it exploits only ratings. The approaches exploiting both of trust and distrust relationships (i.e., Impl, RecSSN, PushTrust, and TrustRec) outperform those ones exploiting only trust relationships (i.e., SocialMF, RSTE, and TrustMF), which means distrust relationships are important as trust ones even when only cold-start users are considered. Among Impl, RecSSN, PushTrust, and TrustRec, again Impl performs worst since it exploits only implicit relationships and considers distrust relationships to be transitive. With the same reasons explained on Fig. 3, although PushTrust exploits both implicit and explicit relationships, its accuracy is less that RecSSN where only explicit relationships are exploited. RecSSN shows lower accuracy than TrustRec since it employs the transitivity of distrust relationships and also it basically cannot make accurate recommendations for cold-start users.

When considering only the cold-start users, our TrustRec again significantly outperforms all existing approaches since, on the contrary to other approaches, it employs the four effective solutions IIR, CTD, EDI, and STR in recommendation process. Table 11 represents the percentage of improvement in accuracy obtained by TrustRec over other approaches; the average improvement in terms of RMSE and MAE over existing approaches are 28.3% and 28.0%, respectively.

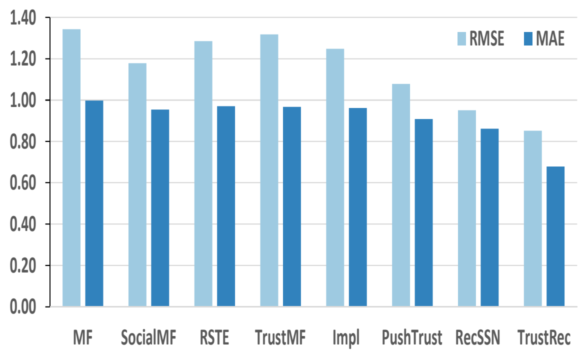

Fig. 4. Accuracy comparison for *cold-start* users

Table 11. Accuracy improvement(%) obtained by TrustRec over other methods with cold-start users

	MF	SocialMF	RSTE	TrustMF	Impl	PushTrust	RecSSN
RMSE	36.5	27.7	33.7	35.3	31.7	21.0	12.5
MAE	31.9	28.8	30.0	29.7	29.3	25.2	21.1

Q_5: Efficiency To answer Q_5, first, we measured the *training time* (i.e., efficiency) of TrustRec *with* different values of parameter *k*, which is shown in Table 12. Then, we compared its efficiency with those of existing approaches, which is shown in Fig. 5. We define the training time as the total amount of time spent on learning user and item latent feature matrices. We do not consider the time spent on predicting ratings on users' unrated items since once the user and item latent feature matrices are obtained, *all* the approaches will require the same time to make predictions. As observed in Table 12, the training time of TrustRec increases smoothly as the value of *k* increases, which means TrustRec is *scalable*. The reason is that it exploits relatively a small number of trust and distrust relationships, thanks to the STR solution.

Table 12. Training time of TrustRec with different *k*

	$k = 5$	$k = 10$	$k = 25$	$k = 50$	$k = all$
Time (hour)	0.5	1	2	5	10

In Fig. 5, we set the value of k for TrustRec as 5 since TrustRec shows its best accuracy (i.e., Table 8) and efficiency (i.e., Table 12) when $k = 5$. Although RSTE exploits only explicit trust relationships, its training time is more than RecSSN that exploits both explicit trust and distrust relationships. This is because RSTE exploits trust relationships in the factorization part of the model, thereby using a user u's trustees $|R_u|$ (i.e., the number of ratings given by *u* in a training set) times in a *single* iteration. On the contrary, RecSSN exploits trust and distrust relationships in the regularization part of the model; as a result, it exploits trustees and distrustees of a user *only once* in each iteration. TrustRec *significantly* outperforms *all* other approaches in terms of efficiency due to the STR solution where only top-k similar trustees and dissimilar distrustees of each user are exploited.

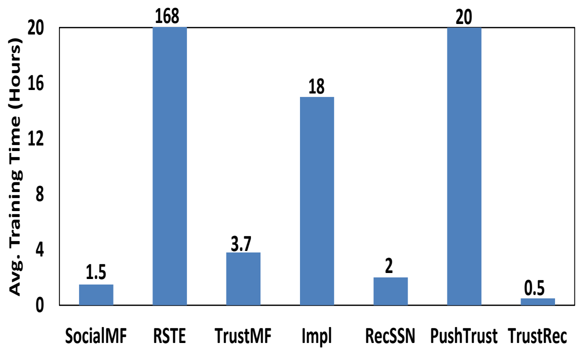

Fig. 5. Comparison of training time

5. Conclusions

In this paper, we analyzed some real-world datasets and observed that explicit trust/distrust relationships are very sparse and some users despite having trust/distrust relationships could have dissimilar/similar preferences in real life. Also, we pointed out that existing trust-aware recommendation approaches require a high computational cost and employing the transitivity of distrust relationships misleads us to considering implicit trustees of a user as her implicit distrustees. We proposed TrustRec that provides an effective solution to each of the aforementioned problems, incorporates all of them together in a single matrix factorization model, and effectively exploits both implicit and explicit relationships. TrustRec exploits more trust/distrust relationships between users by inferring implicit trust/distrust relationships, confirms the degree of similarity/dissimilarity of users having trust/distrust relationships, exploits only top-k most similar/dissimilar trustees/distrustees of each user, and employs the transitivity of trust relationships and the intransitivity of distrust ones. The results of our extensive experiments with a real-world data showed that: 1) each of the proposed solutions is really effective and contributes to achieve better accuracy; 2) TrustRec outperforms all other existing approaches in terms of both accuracy and efficiency when considering all users as well as cold-start users only.

Acknowledgments. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2019R1G1A1007598 and No. NRF-2020R1A2B5B03001960), the research fund of Hanyang University (NO. HY-201800000003218), and the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-01373, Artificial Intelligence Graduate School Program, Hanyang University).

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. on Knowledge and Data Engineering 17(6), 734–749 (2005)
2. Aggarwal, C.C.: Recommender Systems: The Textbook. Springer, 1st edn. (2016)
3. Ali, I., Hong, J., Kim, S.: Exploiting implicit and explicit signed trust relationships for effective recommendations. In: ACM SAC. pp. 804–810 (2017)
4. Forsati, R., Barjasteh, I., Masrour, F., Esfahanian, A., Radha, H.: Pushtrust: An efficient recommendation algorithm by leveraging trust and distrust relations. In: ACM RecSys. pp. 51–58 (2015)
5. Forsati, R., Mahdavi, M., Shamsfard, M., Sarwat, M.: Matrix factorization with explicit trust and distrust side information for improved social recommendation. ACM Trans. on Information Systems 32(4), 17:1–17:38 (2014)
6. Gohari, F., Aliee, F.S., Haghighi, H.: A new confidence-based recommendation approach: Combining trust and certainty. Information Sciences 422, 21–50 (2018)
7. Guha, R., Kumar, R., Raghavan, P., Tomkins, A.: Propagation of trust and distrust. In: WWW. pp. 403–412 (2004)
8. Guo, G., Zhang, J., Smith, N.: A novel recommendation model regularized with user trust and item ratings. IEEE Trans. on Knowledge and Data Engineering 28(7), 1607–1620 (July 2016)
9. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: ACM SIGIR. pp. 230–237 (1999)
10. Hwang, W., Parc, J., Kim, S., Lee, J., Lee, D.: Told you i didn't like it! exploiting uninteresting items for effective collaborative filtering. In: IEEE ICDE. pp. 349–360 (May 2016)
11. Jamali, M., Ester, M.: Trustwalker: A random walk model for combining trust-based and item-based recommendation. In: ACM SIGKDD. pp. 397–406 (2009)
12. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: ACM RecSys. pp. 135–142 (2010)
13. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42(8), 30–37 (2009)
14. Lee, J., Jang, M., Lee, D., Hwang, W., J.Hong, Kim, S.: Alleviating the sparsity in collaborative filtering using crowdsourcing. In: CrowdRec, ACM RecSys. pp. 1–6 (2013)
15. Luxburg, U.V.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (Dec 2007)
16. Ma, H.: An experimental study on implicit social recommendation. In: ACM SIGIR. pp. 73–82 (2013)
17. Ma, H., King, I., Lyu, M.: Learning to recommend with social trust ensemble. In: ACM SIGIR. pp. 203–210 (2009)
18. Ma, H., Lyu, M., King, I.: Learning to recommend with trust and distrust relationships. In: ACM RecSys. pp. 189–196 (2009)
19. Ma, H., Yang, H., Lyu, M., King, I.: Sorec: Social recommendation using probabilistic matrix factorization. In: ACM CIKM. pp. 931–940 (2008)

20. Park, C., Kim, D., Oh, J., Yu, H.: Improving top-k recommendation with truster and trustee relationship in user trust network. Information Sciences 374, 100–114 (2016)
21. Park, S., Chu, W.: Pairwise preference regression for cold-start recommendation. In: ACM RecSys. pp. 21–28 (2009)
22. Pham, M., Cao, Y., Klamma, R., Jarke, M.: A clustering approach for collaborative filtering recommendation using social network analysis. Journal of Universal Computer Science 17(4), 583–604 (feb 2011)
23. Punam, B., Pooja, V.: Empowering recommender systems using trust and argumentation. Information Sciences 279, 569–586 (2014)
24. Rafailidis, D.: Modeling trust and distrust information in recommender systems via joint matrix factorization with signed graphs. In: ACM SAC. pp. 1060–1065 (2016)
25. Tang, J., Aggarwal, C., Liu, H.: Recommendations in signed social networks. In: WWW. pp. 31–40 (2016)
26. Wang, Y., Deng, J., Gao, J., Zhang, P.: A hybrid user similarity model for collaborative filtering. Information Sciences 418-419, 102–118 (2017)
27. Yang, B., Lei, Y., Liu, J., Li, W.: Social collaborative filtering by trust. IEEE Trans. on Pattern Analysis and Machine Intelligence 39(8), 1633–1647 (Aug 2017)
28. Yin, H., Cui, B., Chen, L., Hu, Z., Zhang, C.: Modeling location-based user rating profiles for personalized recommendation. ACM Trans. Knowledge Discovery from Data 9(3), 1–41 (2015)

Masoud Reyhani Hamedani received the B.S. degree in computer science from Shahid Bahonar University, Kerman, Iran, in 2004, and the M.S. degree in software engineering from Payame Nour University, Tehran, Iran, in 2009, and the PhD degree in computer science from Hanyang University, Seoul, Korea in 2016. He worked as a postdoc researcher in Dankook University, Yongin, Korea until February 2018. In March 2018, he joint Hanyang University and currently is working as an assistant research professor in the Computational Social Science Research Center, Department of Computer and Software. His current research interests include data science, feature representation learning, similarity computation in social network, and deep learning.

Irfan Ali received B.S. and M.S. degrees in computer science from Mehran University of Engineering and Technology Hyderabad, Pakistan in 2012, and the PhD degree in computer science from Hanyang University, Seoul, Korea in 2018.

Jiwon Hong received his BS in computer science from Hanyang University, Seoul, Korea, in 2009. He is currently completing the PhD degree in computer and software at Hanyang University. His research interests include data mining, database, social network analysis, and recommender system.

Sang-Wook Kim received the B.S. degree in computer engineering from Seoul National University, in 1989, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), in 1991 and 1994, respectively. In 2003, he joined Hanyang University, Seoul, Korea, where he currently is a professor at the Department of Computer Science and Engineering and the director of the Brain-Korea21-Plus research program. He is also leading a National Research Lab (NRL) Project funded by the National Research Foundation since 2015. From 2009 to 2010, he visited the Computer Science Department, Carnegie Mellon University, as a visiting professor. From 1999 to 2000, he worked with the IBM T. J. Watson Research Center, USA, as a postdoc. He also visited the Computer Science Department at Stanford University as a visiting researcher in 1991. He is an author of more than 200 papers in refereed international journals and international conference proceedings. His research interests include databases, data mining, multimedia information retrieval, social network analysis, recommendation, and web data analysis. He is a member of the ACM and the IEEE.

Received: June 08, 2020; Accepted: October 10, 2020.