Trust-aware Privacy-Preserving Recommender System

Xiwei Wang*
Dept. of Computer Science
Northeastern Illinois University
Chicago, IL
xwang9@neiu.edu

Jun Zhang†
Dept. of Computer Science
University of Kentucky
Lexington, KY
jzhang@cs.uky.edu

Yin Wang‡
Dept. of Math and Computer Science
Lawrence Technological University
Southfield, MI
ywang12@ltu.edu

ABSTRACT

Recommender systems have achieved great success in providing product recommendations for online shopping. With recommender systems, customers can find their interested merchandise in a timely manner. It not only facilitates customers' purchases, but also promotes sales. While recommender systems can predict customers' preferences accurately, they suffer from privacy leakage in many aspects.

In this paper, we study an attack model in centralized recommender systems and present a privacy-preserving recommendation framework to neutralize such attack. In this model, an attacker holds some of the real customers' ratings and attempts to obtain their private preferences. The proposed framework is based on the weighted nonnegative matrix tri-factorization technique. It utilizes customers' trustworthiness to filter out unrelated ratings so that their privacy can be preserved. We performed experiments on two different datasets with respect to recommendation precision and privacy preservation level. The results demonstrate that our recommender system can distinguish between real customers and attackers to a great extent without compromising the prediction accuracy.

CCS Concepts

•Information systems → Personalization; Trust; Online shopping; •Security and privacy → Social aspects of security and privacy; Privacy protections; •Human-centered computing → Collaborative filtering; Social recommendation; Social tagging;

Keywords

collaborative filtering; nonnegative matrix factorization; privacy; recommender system; trust

1. INTRODUCTION

Recommender systems are widely deployed in online stores. With recommender systems, the merchants can provide personalized product recommendations to their customers. This not only promotes their sales but also facilitates customers' purchases. A recommender system is a program (or a set of programs) that utilizes algorithms to predict users'[1] purchase preferences by profiling their shopping patterns. Most popular recommender systems are based on collaborative filtering (CF) techniques, e.g., item/user correlation based CF's [19, 17, 22], singular value decomposition (SVD) based latent factor CF's [20], and nonnegative matrix factorization (NMF) based CF's [4, 29].

While CF based recommender systems can make reasonably accurate recommendations, the privacy issues were raised during this process [3, 18, 16]. As far as we know, almost all online shopping websites keep users' sensitive information on their central servers, such as users' purchase history as well as their ratings and comments on products. When writing reviews, some websites allow users to choose publishing or saving them as drafts. If the reviews are not sensitive, they can be simply published by their authors. However, if users just want to save those comments for personal purposes and do not hope to share with others, they can keep the drafts unpublished. By doing so, people can ensure that these comments are only visible to themselves. With that said, people's published reviews are exposed to the public so anyone can read them freely. Though these comments might not be considered as customers' privacy, malicious users are able to identify customers' private preferences on particular products by cheating recommender systems. In a typical attack model, the attackers create fake user profiles containing real customers' public comments and then obtain recommendations from the system [13]. Due to the connections (either implicit or explicit) between users' public preferences and private preferences, if a recommender sys-

*Xiwei Wang is an Assistant Professor in the Department of Computer Science, Northeastern Illinois University, Chicago, IL, 60625-4599 USA. e-mail: xwang9@neiu.edu

†Jun Zhang is a Professor in the Department of Computer Science, University of Kentucky, Lexington, KY, 40506-0633 USA. e-mail: jzhang@cs.uky.edu

‡Yin Wang is an Assistant Professor in the Department of Math and Computer Science, Lawrence Technological University, Southfield, MI, 48075 USA. e-mail: ywang12@ltu.edu.

[1]The terms "customer" and "user" are used interchangeably as they refer to the same thing in this context. Same interchangeability also applies to "product" and "item".

MOBIMEDIA 2016, June 18-19, Xi'an, People's Republic of China

DOI 10.4108/eai.18-6-2016.2264146

tem fails to distinguish real customers from malicious users, it would be highly possible that the private preferences of real customers can be exposed.

On some product review websites, such as Epinions.com and Ciao.com, in addition to writing reviews for items, users can also tag trust votes based on their opinions on others' reviews [8]. In other words, every user maintains a list of the users that he trusts. Now an assumption is added to the aforementioned attack model - attackers are not easily trusted by other people because they only duplicate real customers' reviews and ratings and create fake profiles in a relatively short time. We make this assumption because before a user tags his trusted users, he must have read many of their reviews in the past. If a newly created account has very similar or the same product reviews and ratings as people's trusted users, they might suspect that this account is a fake account so they would not trust it.

In this paper, we propose a privacy-preserving recommendation framework, named TrustRS, which preserves users' private preferences when providing accurate recommendations. The framework uses NMF based CF techniques that work on both users' rating information and their trustworthiness to provide personalized recommendations. We divide the procedure into two stages: unknown rating predictions and unrelated entries filtering. The raw ratings and trustworthiness information participate in the weighted nonnegative matrix tri-factorization [7, 24] to approximate the original rating matrix. User and item groups are then established according to the factor matrices. We use the group information to filter out unrelated entries so that a majority of real users' items of interest are concealed from attackers. We examined TrustRS on the Epinions [15] and Ciao datasets [23] in two aspects: (1) unknown rating prediction accuracy, and (2) user privacy preservation level. The results show that our framework can tell real customers from attackers to a great extent without compromising the prediction accuracy.

The contributions of this paper are threefold:

1. We proposed an attack model and the corresponding privacy measurement;
2. We applied trustworthiness to the weighted nonnegative matrix tri-factorization to improve prediction accuracy;
3. We grouped users and items according to the factor matrices of NMF and used them to filter out unrelated ratings for privacy preservation.

The remainder of this paper is organized as follows. Section 2 gives the related work. Section 3 defines the problem and related notations. Section 4 describes the main idea of the proposed approach. Section 5 presents the experiments and discusses the results. Some concluding remarks and future work are given in 6.

2. RELATED WORK

Privacy-preserving collaborative filtering (PPCF) was first proposed by Canny [3] and has been studied in both distributed systems [3, 28, 1, 27, 13, 21] and centralized systems [18, 14, 10, 5]. While most peer-to-peer (P2P) environments adopt distributed recommender systems, centralized systems are widely used by almost all of the most popular online vendors, e.g., eBay, Amazon, and Newegg.

Polat and Du [18] applied randomized perturbation techniques to the SVD based collaborative filtering to provide privacy-preserving recommendations. In their method, uniform or Gaussian noise is added to the users' real ratings and then the server predicts the unknown ratings by the perturbed data. Based on this model, Li et al. [14] used NMF instead of SVD as the fundamental collaborative filtering technique and they obtained better results than Polat's method. While both papers demonstrated that the proposed methods perturbed the data to a reasonable level and kept the prediction precision, it is not clear how much contribution can be made by the methods to the real world recommender systems. Different from Polat's and Li's work, Kaleli et al. [10] proposed a privacy-preserving naïve Bayesian classifier (PPNBC) CF approach. The approach employs randomized response techniques (RRT) [25] to protect users' privacy while producing referrals using a naïve Bayesian classifier (NBC). In their model, the RRT is applied to both the one-group scheme and the multi-group scheme for data distortion purposes. The distorted data is then fed to NBC for recommendations. Adding to [10], Bigle et al. [2] utilized pre-processing to improve the privacy-preserving recommendations. In their method, the masked data is pre-processed by identifying the most similar items to each item off-line; some of the unrated item entries are also filled to improve the density. Since the increasing number of features and groups would degrade the online performance of the multi-group PPNBC significantly, decreasing the amount of data involved in CF can be very beneficial. With the varying number of neighbors, the central server can decide how many items would be recommended to users. In our experiments, it was tested by the attack model for comparison purposes.

Recent works on using trustworthiness in collaborative filtering indicate that this information benefits prediction precision as well as privacy protection. Jamali et al. [9] proposed SocialMF, a recommender system that makes use of the matrix factorization technique with trust propagation in social networks. Similar to [9], Yang et al. [26] proposed TrustMF to handle data sparsity and cold start problems which happen commonly in collaborative filtering based recommender systems. In TrustMF, users are projected into low-dimensional latent feature spaces by the matrix factorization technique according to their relationships. By doing so, users' mutual influence on their own opinions is reflected in a more reasonable way. In [11], the authors presented a trust based recommendation scheme on vertically distributed data for privacy preservation. The scheme builds a trust web of users and then uses it to filter users' neighbors in order to protect the privacy. Our proposed framework incorporates trustworthiness into the weighted nonnegative matrix tri-factorizarion [24] to improve prediction accuracy and privacy protection.

In [13] and [21], the authors proposed group based privacy-preserving recommender systems in which recommendations are made at the group level as opposed to the individual level. Their experimental results show that user groups can be utilized as a natural protective mechanism for achieving promising privacy protection. In [13], items are grouped in terms of their ratings so users' public interests and private interests can be separated. The system then identifies and aggregates the preferences of group members. Recommendations are made by personalizing the group preferences locally to conceal uesrs' private interests. Motivated by this framework, we group items and users according to the factor matrices of NMF in our centralized system to distinguish real users from attackers.

3. PROBLEM DESCRIPTION

Assume the data owner has two matrices: a user-item rating matrix, denoted by $R \in \mathbb{R}^{m\times n}$, and a user-user trust matrix, denoted by $T \in \mathbb{R}^{m\times m}$, where there are m users and n items. An entry r_{ij} in R represents the rating left on item j by user i. The valid range of a rating's value varies from website to website. Some websites use the $1 \sim 5$ scale with 1 as the lowest rating (most disliked) and 5 as the highest rating (most favorated) while some others use the $-10 \sim 10$ scale with -10 as the lowest rating, 0 as the neutral rating, and 10 as the highest rating. The trust matrix T indicates trust relationships among users. Possible values of the entry t_{ik} in T are 1, 0, and -1. $t_{ik} = 1$ means that user i is trusted by user k, while $t_{ik} = -1$ means that user i is distrusted by user k. If the trust relationship is unknown, then $t_{ik} = 0$.

The rating matrix R is incomplete due to the fact that most users rate only a few items and most items are rated by a small number of users. The main task of recommender systems is to predict the unknown ratings in R, which are further used to make recommendations. The collaborative filtering based recommender systems predict ratings by using relevant users' preferences. In this scheme, if two users have very similar preferences, they are very likely to receive close or same recommendations. Malicious users can take this advantage to obtain real users' private interests, which are considered as sensitive information. Definition 1 gives the formal description of the attack model.

Definition 1. *An **Attack model** is a 4-tuple: Attack*(u, a) $= \{R_u, R_a, L_u, L_a\}$*, where* u *is a real user and* a *is an attacker;* R_u *and* R_a *are their rating vectors;* L_u *and* L_a *are their recommendation lists*[2].

To attack a recommender system, the attacker makes a fake profile R_a *according to the real user's preferences* R_u *and receives recommendations from the system. The closer* L_u *and* L_a *are, the more successful of an attack is achieved.*

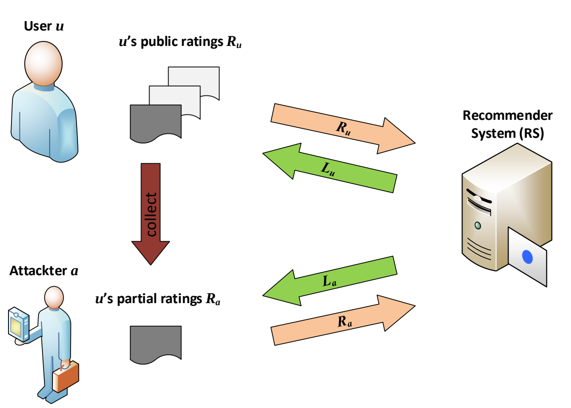

Figure 1: An attack model

As mentioned earlier, an attacker can easily collect real users' public preferences. However, it is not easy for attackers to be trusted by the same people who trust the attackees. Therefore, in our proposed attack model, we assume that attackers only possess attackees' partial public ratings but are not in other people's trust lists.

The purpose of our proposed recommendation framework is to distinguish real users from attackers by differing their recommendation lists without degrading the accuracy of the unknown rating predictions.

4. TRUST-AWARE PRIVACY-PRESERVING RECOMMENDATION FRAMEWORK

In this section, we will introduce the proposed recommendation framework —TrustRS in two phases: unknown rating predictions and unrelated entries filtering.

4.1 Unknown Rating Predictions

In [24], the authors used the weighted and constrained nonnegative matrix tri-factorization for privacy-preserving data updates. In TrustRS, we incorporate users' trustworthiness into a similar matrix factorization to approximate the original rating matrix.

Nonnegative matrix factorization (NMF) [12] is one of the most popular dimension reduction methods and has been applied in collaborative filtering to make product recommendations [29, 4]. A conventional NMF is defined as follows [12],

$$R_{m\times n} \approx U_{m\times k} \cdot V^T_{n\times k}. \quad (1)$$

The goal is to find a pair of orthogonal nonnegative matrices U and V that minimize the Frobenius norm $\|R-UV^T\|_F$. Thus the objective function for NMF is

$$min_{U\geq 0, V\geq 0} f(R, U, V) = \|R - UV^T\|^2_F. \quad (2)$$

It is worth noting that R must not contain missing values as NMF is not suitable for such matrices. If all missing values in R are replaced with zeros and NMF is performed on the imputed rating matrix, it will lead to very biased results. Moreover, the trustworthiness is expected to be incorporated into the matrix factorization, and the users as well as items are expected to be grouped in terms of the factor matrices. Therefore, additional constraints need to be applied to NMF. To do so, we propose the trust-incorporated NMF as illustrated in Eq. (3).

$$\begin{aligned} &min_{U\geq 0, S\geq 0, V\geq 0, B\geq 0} f(R, W_R, T, W_T, U, S, V, B) = \\ &\alpha \cdot \|W_R \circ (R - USV^T)\|^2_F + \beta \cdot \|W_T \circ (T - UB^T)\|^2_F \\ &+ \gamma \cdot (\|U\|^2_F + \|S\|^2_F + \|V\|^2_F + \|B\|^2_F), \end{aligned} \quad (3)$$

where $U \in \mathbb{R}^{m\times k}_+$, $S \in \mathbb{R}^{k\times l}_+$, $V \in \mathbb{R}^{n\times l}_+$, and $B \in \mathbb{R}^{m\times k}_+$. U, V, B are orthogonal matrices. $\circ$ denotes the element-wise multiplication.

In this equation, R is a rating matrix and T is a trust matrix. Since both R and T are incomplete, two weight matrices W_R and W_T are introduced to resolve the missing value issues [29]. They indicate the locations of the observed entries in R and T as

$$w_{Rij} = \begin{cases} 1 & if \quad r_{ij} \neq 0 \\ 0 & if \quad r_{ij} = 0 \end{cases} \quad (w_{Rij} \in W_R, r_{ij} \in R) \quad (4)$$

$$w_{Tij} = \begin{cases} 1 & if \quad t_{ij} \neq 0 \\ 0 & if \quad t_{ij} = 0 \end{cases} \quad (w_{Tij} \in W_T, t_{ij} \in T) \quad (5)$$

[2]The recommendation list in this framework contains all relevant items to the active user. They might receive either low or high ratings from this user.

In the first term of Eq. (3), an extra factor matrix S is utilized to absorb the different scales of R, U, and V for simultaneous row clustering and column clustering [7]. Ding et al. [6] showed that NMF and the K-Means algorithm are equivalent with respect to object clustering. In our proposed framework, we expect to group users and items according to the results of NMF without extra clustering overhead. Thus, by using S, the factor matrices U and V can be viewed as the cluster membership indicators for users and items, respectively.

While the first two terms focus on optimizing R and T, the third term of this equation is added to avoid overfitting. We use coefficients α, β, and γ to control the weight of each term. The corresponding update formulas are listed as follows:

$$U_{ij} = U_{ij} \cdot \frac{\{\alpha(W_R \circ R)VS^T + \beta(W_T \circ T)B\}_{ij}}{\{\alpha[W_R \circ (USV^T)]VS^T + \beta[W_T \circ (UB^T)]B + \gamma U\}_{ij}} \tag{6}$$

$$S_{ij} = S_{ij} \cdot \frac{\{\alpha U^T (W_R \circ R)V\}_{ij}}{\{\alpha U^T [W_R \circ (USV^T)]V + \gamma S\}_{ij}} \tag{7}$$

$$V_{ij} = V_{ij} \cdot \frac{\{\alpha (W_R \circ R)^T US\}_{ij}}{\{\alpha [W_R \circ (USV^T)]^T US + \gamma V\}_{ij}} \tag{8}$$

$$B_{ij} = B_{ij} \cdot \frac{\{\beta (W_T \circ T)^T U\}_{ij}}{\{\beta [W_T \circ (UB^T)]^T U + \gamma B\}_{ij}}. \tag{9}$$

As we mentioned above, rating matrices are usually very incomplete. In order to predict missing values, the trust-incorporated NMF initializes the factor matrices U, S, V, and B with random values. It then iteratively updates them according to the update formulas. The iterative procedure can be terminated either after a particular number of iterations or when the Frobenius norm $\|R - UV^T\|_F$ has only trivial differences between two consecutive iterations, i.e., the objective function's global minimum is achieved. Eventually, with the updated factor matrices, a complete rating matrix can be calculated by taking the product of the matrices, i.e., $\hat{R} = USV^T$, where all unknown values in R are predicted and present in $\hat{R}$.

With regard to the time complexity, assuming $k, l \ll \min(m, n)$, each iteration takes up to $O(mn(k+l) + (m^2 + n^2)k)$ calculations.

4.2 Unrelated Entries Filtering

In the previous section, we developed an NMF model that considers ratings as well as trustworthiness. The approximated rating matrix $\hat{R}$ has no missing values. To make personalized and privacy-preserving recommendations, TrustRS filters out unrelated ratings before it generates the recommendation lists.

To this purpose, users and items are grouped based on U and V in Eq. (3) so that only relevant items would be presented to the users. The following definitions define the user group and the interest group that would be used next.

Definition 2. *A* ***user group*** $g_{U_p} = \{T_{U_p}, D_{U_p}\}$*, where* T_{U_p} *is the set of users that belong to this group;* D_{U_p} *contains the membership degree of each user in* T_{U_p}*.*

Definition 3. *An* ***interest group*** $g_{I_q} = \{T_{I_q}, M_{I_q}, g'_{U_q}\}$*, where* T_{I_q} *is the set of items that belong to this group;* M_{I_q} *contains the average rating of each item in* T_{I_q}*;* g'_{U_q} *is the group of users (with their membership degrees) that have rated at least one item in* T_{I_q}*.*

Interest groups, denoted by $G_I = \{g_{I_1}, g_{I_2}, ..., g_{I_l}\}$, can be established in terms of matrix V according to the equivalence between NMF and the K-Means clustering algorithm [6]. For a given $V \in \mathbb{R}_+^{n \times l}$, items are clustered into l groups by Eq. (10).

$$c_i = \operatorname*{argmax}_{1 \le j \le l} V_{ij}, \tag{10}$$

where c_i is the index of the group that item i belongs to.

For each item group T_{I_q}, the average rating of every item is calculated and the related users are also identified. In this case, a single user may belong to multiple groups. This user grouping strategy is called the interest based grouping. The membership degree of user u to interest group g can be calculated by Eq. (11).

$$d_{ug} = (1 - dis_{ug}) \cdot wa_{ug} \cdot wb_{ug}, \tag{11}$$

where dis_{ug} is the normalized distance between user u's ratings and the corresponding average ratings in M_{I_q} of interest group g; wa_{ug} is the fraction of user u's ratings left on items in g; wb_{ug} is the fraction of items in g that are rated by user u. wa and wb are used to weight $(1 - dis_{ug})$ so the greater they are, the higher d_{ug} will be.

Algorithm 1 Rating filtering algorithm

Input:
Raw rating matrix: R;
Predicted rating matrix: $\hat{R} \in \mathbb{R}^{m \times n}$;
User groups: $G_T = \{g_{U_1}, g_{U_2}, ..., g_{U_k}\}$;
Interest groups: $G_I = \{g_{I_1}, g_{I_2}, ..., g_{I_l}\}$;
User index: u;
Membership degree threshold: m_u;

Output:
Recommendation list: L_u;

$max_{degree} = 0$, $g_{I_{closest}} = \varnothing$, $NL_I, NL_T = \varnothing$;
for $i = 1$ *to* l **do**
  if $d_{ui} \ge m_u$ and $d_{ui} > max_{degree}$ **then**
    $max_{degree} = d_{ui}$;
    $g_{I_{closest}} = g_{I_i}$;
  end if
end for
Add users in $g_{I_{closest}}$ with membership degree $\ge m_u$ to u's interest neighbor list NL_I;
Search in G_T and identify the user group g_{U_u} that u belongs to;
Add all users in g_{U_u} to u's trust neighbor list NL_T;
Take the intersection of NL_I and NL_T. The result is saved in NL_u;
Search in R to find items that are rated by users in NL_u. Save them in IL_u;
$L_u \leftarrow (j, \hat{R}_{uj})$, where $j \in IL_u$.

In addition to the interest based grouping, users are also clustered by matrix U like the way that items are clustered by matrix V. Note that $U \in \mathbb{R}_+^{m\times k}$ is updated by taking into account both the rating matrix R and the trust matrix T. Hence the groups obtained in this way, denoted by $G_T = \{g_{U1}, g_{U2}, ..., g_{Uk}\}$, reflect users' rating patterns as well as their trustworthiness. This grouping strategy is named the trust based grouping.

Algorithm 1 illustrates the procedures to recommend items to a user u. The general idea of this algorithm is to recommend user u the items that are rated by u's neighbors. The corresponding ratings are predicted by the NMF model proposed in the previous section.

5. EXPERIMENTAL STUDY

5.1 Data Description

In the experiments, we examine TrustRS on the Epinions[15] and Ciao[23] datasets. They both provide rating values and trustworthiness information. Table 1 collects the statistics of the datasets.

Table 1: Statistics of the data

Dataset	#users	#items	#ratings	#trusts
Ciao	2,056	1,458	53,312	36,432
Epinions	7,260	2,440	172,497	49,248

The raw Epinions dataset has 22,166 users and 296,227 items with 912,441 ratings and 355,217 trust values. Due to the memory limit, 7,260 users (who have rated at least 10 items) and 2,440 items with 172,497 ratings and 49,248 trust values were selected for testing. The ratings were divided into two subsets: a training set and a test set. To build the test set, we chose users with more than 100 ratings and 20 trust values. For each of these users, 20 ratings were randomly selected and added into the test set. The rest of the ratings were added to the training set. Consequently, there were 171,577 ratings in the training set and 920 ratings in the test set.

Likewise, only partial ratings were selected from the Ciao dataset for the experiments. In this case, there were 2,056 users and 1,458 items with 53,312 ratings and 36,432 trust values. The test set contained 2,960 ratings from users who had at least 60 ratings and 20 trust values. The remaining 50,352 ratings formed the training set.

It is worth mentioning that both datasets provide the helpfulness of the ratings. For example, a user left 5 stars on an item and another user thought the helpfulness of this rating was 7 out of 10. This additional "rating of a rating" makes users' feedback more reliable. Therefore, in our experiments, we pre-processed all the ratings by considering their helpfulness. In this example, the user's rating becomes $original\ \ rating \times helpfulness\ \ degree = 5 \times (7/10) = 3.5$ as opposed to 5 accordingly.

For each dataset, users who have ratings in the test set are called test users. Some of the test users were chosen and 30% of their ratings were used to create the attackers[3]. For instance, if a test user has 100 ratings in the training set and 20 ratings in the test set, 30 of his training ratings are used to create an attacker profile. The test ratings of this attacker are identical to the real user's, meaning that the attacker has 20 ratings in the test set as well. Moreover, attackers are not trusted by anyone, so they do not have trust values.

[3]In the experiments, the active users include both test users and attackers.

5.2 Evaluation Strategy

The experiments tested TrustRS in two aspects: unknown rating predictions and user privacy preservation. We used the mean absolute error (MAE) to measure the prediction accuracy:

$$MAE = \frac{1}{|TestSet|} \sum_{r_{ij} \in TestSet} |r_{ij} - p_{ij}| \qquad (12)$$

where $|TestSet|$ is the size of the test set; r_{ij} is an observed rating in the test set; p_{ij} is an entry in $\hat{R}$, the approximated rating matrix.

To obtain the privacy preservation level, we calculated the recall rates derived from Eq. (13) for both real users and the corresponding attackers. We then measured how much the recall rates were reduced. More specifically, higher recall rates for real users and lower recall rates for attackers are expected because fewer items that are related to real users would be recommended to attackers. Therefore a higher privacy preservation level is achieved.

$$Recall_u = \frac{|L_u \cap T_u|}{|T_u|}, \qquad (13)$$

where L_u is the set of items that are recommended to user u, T_u is the set of items that appear in the test set and are rated by user u.

We compared our recommender system against two existing algorithms that were discussed in Section 2: the NMF based CF with random perturbation [14] (referred to as RandNMF) and the naïve Baysian classifier based PPCF with pre-processing [2] (referred to as PRNBC).

Since RandNMF focuses on perturbing the rating matrix instead of refining the recommendation list, no ratings are filtered out, and the recall rates are always 1. Thus we only compared it with TrustRS in unknown rating predictions. PRNBC, in contrast, has a filtering step so both prediction accuracy and privacy preservation level were investigated on this algorithm.

5.3 Results and Discussion

In this section, we study the experimental results with respect to privacy preservation level and the prediction accuracy.

5.3.1 *Privacy Preservation*

As discussed in Section 2, PRNBC applies a pre-processing step to the naïve Baysian classifier based PPCF to filter out less important neighbors. By doing so, the online performance of PPCF can be improved. A side effect is that the number of recommended items can be controlled – greater number of neighbors result in more recommended items. This facilitates the comparisons because different neighbor sizes can be probed for PRNBC to obtain the real users' recall rates that are very close to the recall rates produced by TrustRS. Then how much the recall rate is reduced from real users to attackers on both methods can be measured.

Note that the authors in [2] did not study the MAE of PRNBC but converted the integer ratings to binaries. Whereas in our experiments, we expected to examine both MAE's and recall rates so this conversion was not carried out. However, more classes (e.g., five classes in a $1 \sim 5$ rating system, and two classes in a binary rating system) require significantly more computation time. To compensate for it, the one-group scheme was used as opposed to the multi-group scheme.

Table 2: Parameter setup

Dataset	α	β	γ	k	l	m_u
Ciao	0.1	0.3	0.6	20	13	0.0005
Epinions	0.1	0.7	0.2	20	20	0.0017

Table 2 lists the parameter setup for TrustRS. α, β, and γ are the coefficients that control the weight of each term in Eq. (3). k and l are utilized as the dimensions of the factor matrices as well as the number of user/item clusters. m_u is the membership degree threshold required by Algorithm 1.

The parameters were determined by probing different combinations. We first fixed k, l, and m_u to test α, β, and γ in $\{0.1, 0.2, ..., 0.9\}$ under the constraint $\alpha + \beta + \gamma = 1$. Then we probed k and l in $\{1, 2, ..., 20\}$ with everything else unchanged. m_u was obtained last. We selected the parameters by taking into account both the MAE's and the reductions of the recall rates.

In this table, β is greater than α on both datasets. It means that trust matrices played a more important role than rating matrices. Two k's and one l are 20, which is the largest number we probed so it is very likely that better results might exist if giving larger numbers. Nonetheless, the more clusters that are produced, the longer computation time is required.

Table 3: Privacy protection level

Dataset	Method	RecallU	RecallA	%Reduced
Ciao	PRNBC	0.9295	0.9228	0.007%
	TrustRS	0.9297	0.4595	50.581%
Epinions	PRNBC	0.8886	0.8815	0.008%
	TrustRS	0.8870	0.5935	33.088%

The privacy protection levels, measured by the recall rate reduction percentages, are illustrated in Table 3. "RecallU" represents the average recall rate for real users, and "RecallA" represents that of attackers. It can be seen that TrustRS greatly reduced the recall rates from real users to attackers. For example, the Ciao dataset has 2,960 ratings in its test set. TrustRS recommended $2,960 \times 0.9297 \approx 2,751$ items to real users but only $2,960 \times 0.4595 \approx 1,360$ items to attackers. It protected half of the real users' relevant items from being exposed to attackers. PRNBC, on the contrary, merely reduced the recall rates. The reduction percentages of TrustRS on Epinions dataset are not as high as those on Ciao, but it still outperformed PRNBC to a great extent.

We recorded the recall rates with varying membership degree threshold m_u as well. Since a user might belong to multiple interest groups, this value affects the results in two aspects: (1) it determines which groups are considered as the active users' interest groups; (2) it controls the number of similar users that would be recognized as the active users' interest neighbors. A smaller m_u causes higher recall rates and vice versa. With that said, m_u should not be extremely small because the filtering step will fail to distinguish real users from attackers thus no privacy can be preserved. Figure 2 plots the recall rates on two datasets, in which $m_u \in \{0.0001, 0.0002, ..., 0.01\}$. With the increasing

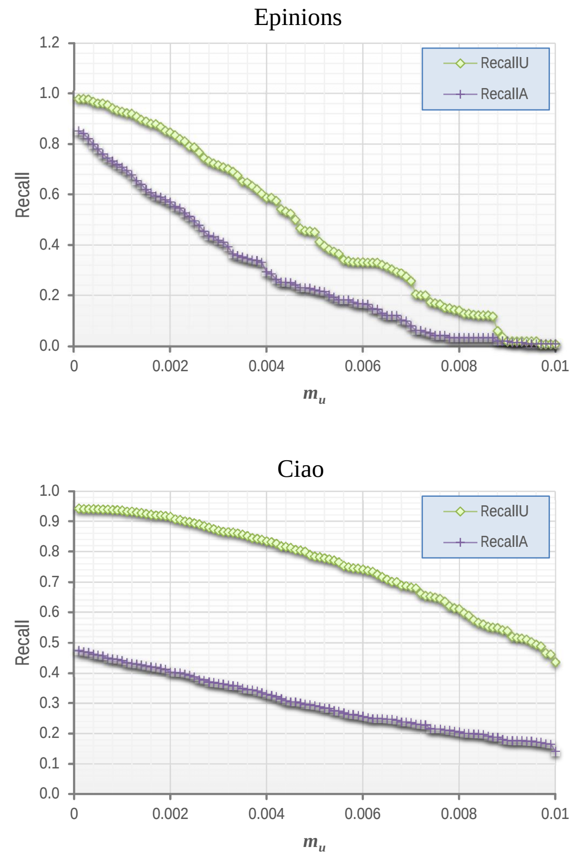

Figure 2: Recall rates with varying m_u in TrustRS

m_u, both recall rates decreased but they tended to be closer when m_u was very small or very large.

It is depicted in Algorithm 1 that TrustRS filters the unrelated items based on user groups G_T and interest groups G_I. While m_u directly affects the selection of a user's interest neighbors, there is no extra constraint on the formation of his trust neighbors. That is to say, all users in the same group as an active user are saved in his trust neighbor list NL_T. In Eq. (3), U is optimized for both R and T. Therefore, the user groups established from U are highly related to customers' rating patterns and trustworthiness, which are consequential to identifying malicious users. To investigate the impact of NL_T on the final recall rates, each active user's trust neighbor list was sorted according to the membership degrees of the neighbors in the list, and the top n neighbors were kept for filtering.

Figure 3 charts the recall rates with multiple n's. The curves in this figure indicate that when just a few trust neighbors were used, TrustRS was not able to tell real users from attackers. With more neighbors' participation, the recall rates increasingly differed but kept stable after some points. The results imply the significance of trustworthiness.

As a reference, we also studied the recall rates for PRNBC with different number of neighbors in Figure 4. When more neighbors were retained, the recall rates increased subsequently. Nevertheless, the algorithm failed to tell real users from attackers as their recall rates were almost identical.

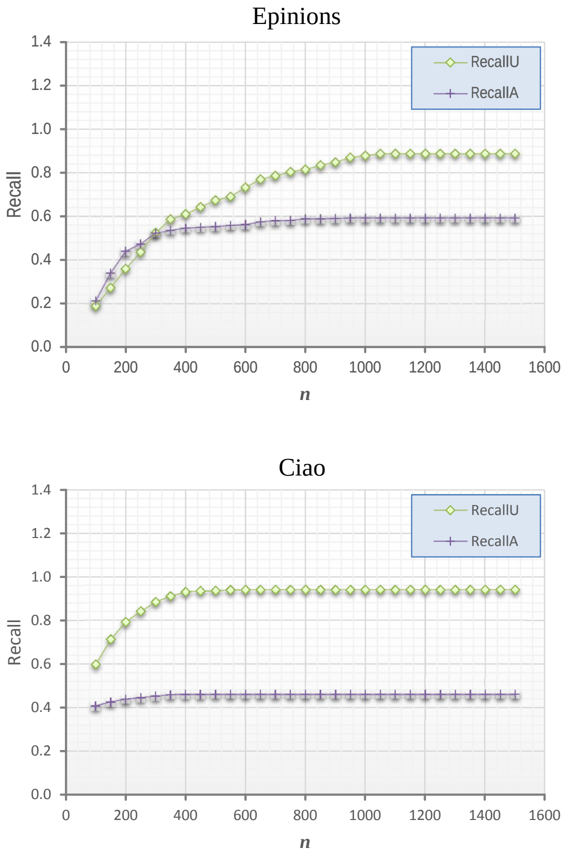

Figure 3: Recall rates with varying $\#neighbors$ in TrustRS

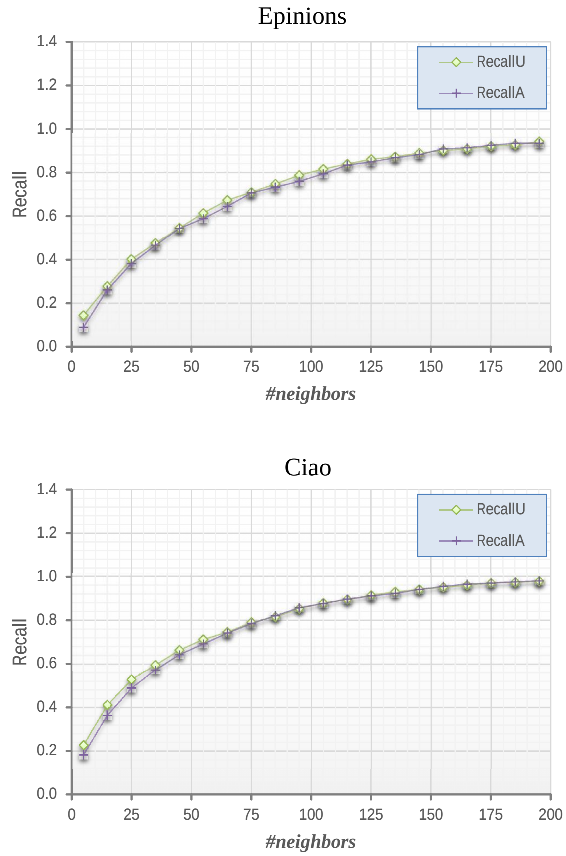

Figure 4: Recall rates with varying $\#neighbors$ in PRNBC

5.3.2 Prediction Accuracy

The prediction accuracy was judged in terms of MAE on all the ratings in the test set. Thus the second step of TrustRS, which is unrelated entries filtering, was ignored. We used the same parameter setup listed in Table 2.

In RandNMF, we calculated the z-scores for each user and replaced the values in the original rating matrix R with z-scores. The missing entries were imputed by the users' mean ratings. We added the noise drawn from the normal distribution to the imputed matrix for perturbation purposes.

The MAE's of all three methods are shown in Table 4. For RandNMF, we probed the rank of the underlying matrix factorization in $\{1, 2, ..., 20\}$ and took the lowest MAE's. As one can see, TrustRS produced the lowest prediction errors on both datasets. The MAE's of RandNMF were not significantly worse than TrustRS. We consider this result as normal owing to the fact that TrustRS is based on NMF. However, it uses weight matrices to avoid biased missing value imputation and incorporates trustworthiness to improve prediction accuracy. PRNBC, the naïve Bayesian classifier based recommendation algorithm, performed worst and the prediction errors were remarkably higher than others. We attribute it to the randomized response techniques (RRT) used in the one-group scheme [10]. Because in this scheme, each user's ratings are placed in the same group. According to the RRT, they either remain unchanged or are completely reversed (e.g., changing 1 to 5, 2 to 4). It perturbs the ratings to a certain degree and protects the privacy, but the prediction errors would increase as well.

Table 4: Prediction accuracy

Dataset	Method	MAE	
Ciao	RandNMF	0.4162	$rank = 1$
	PRNBC	1.7851	-
	TrustRS	0.2725	-
Epinions	RandNMF	0.5480	$rank = 1$
	PRNBC	1.0202	-
	TrustRS	0.4084	-

As a summary, our proposed recommendation framework TrustRS outperformed two existing privacy-preserving recommendation algorithms with regard to unknown rating prediction accuracy and privacy preservation level. The results demonstrate that the trustworthiness information makes a substantial contribution to correctly filtering out unrelated ratings. The membership degree threshold needs to be carefully determined due to its direct impact on the recall rates for both real users and attackers.

6. CONCLUSION AND FUTURE WORK

In this paper, we studied an attack model that aims at finding customers' potentially interested items by cheating recommender systems. We proposed a weighted nonnegative matrix tri-factorization based privacy-preserving recommendation framework, named TrustRS, to neutralize this type of attack. TrustRS utilizes customers' trustworthiness to filter out unrelated ratings so that their privacy can be preserved. Experiments conducted on two popular product review datasets show that TrustRS can preserve users' privacy to a great extent without compromising the prediction accuracy.

The real world recommender systems process a large amount of user data in every second. People expect to receive instant product recommendations immediately after they log into their accounts. It requires the recommender systems to be highly scalable. In the future, we plan to study recommendation algorithms on big datasets and adapt our proposed privacy-preserving recommendation framework to large-scale scenarios. We will also explore other factors, e.g., temporal information, that can be considered for malicious user detections.

7. REFERENCES

[1] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci. Enhancing Privacy and Preserving Accuracy of a Distributed Collaborative Filtering. In *Proceedings of the 2007 ACM Conference on Recommender Systems*, pages 9–16, 2007.

[2] A. Bilge and H. Polat. Improving Privacy-Preserving NBC-based Recommendations by Preprocessing. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 143–147, 2010.

[3] J. Canny. Collaborative Filtering with Privacy. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 45–57, 2002.

[4] G. Chen, F. Wang, and C. Zhang. Collaborative Filtering Using Orthogonal Nonnegative Matrix Tri-factorization. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops*, pages 303–308, 2007.

[5] R. Chen, M. Xie, and L. V. Lakshmanan. Thwarting Passive Privacy Attacks in Collaborative Filtering. *Database Systems for Advanced Applications*, 8422:218–233, 2014.

[6] C. Ding, X. He, and H. D. Simon. On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 606–610, 2005.

[7] C. Ding, T. Li, W. Peng, and H. Park. Orthogonal Nonnegative Matrix Tri-Factorizations for Clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 126–135, 2006.

[8] J. Huang, F. Nie, H. Huang, Y. Lei, and C. Ding. Social Trust Prediction Using Rank-k Matrix Recovery. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 2647–2653, 2013.

[9] M. Jamali and M. Ester. A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks. In *Proceedings of the 4th ACM conference on Recommender Systems*, pages 135–142, 2010.

[10] C. Kaleli and H. Polat. Providing Private Recommendations Using Naïve Bayesian Classifier. *Advances in Intelligent Web Mastering*, 43:168–173, 2007.

[11] C. Kaleli and H. Polat. Privacy-Preserving Trust-based Recommendations on Vertically Distributed Data. In *Proceedings of the 5th IEEE International Conference on Semantic Computing*, pages 376 – 379, 2011.

[12] D. D. Lee and H. S. Seung. Algorithms for Non-negative Matrix Factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[13] D. Li, Q. Lv, H. Xia, L. Shang, T. Lu, and N. Gu. Pistis: A Privacy-Preserving Content Recommender System for Online Social Communities. In *Proceedings of the 2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 79–86, 2011.

[14] T. Li, C. Gao, and J. Du. A NMF-Based Privacy-Preserving Recommendation Algorithm. In *Proceedings of the 1st International Conference on Information Science and Engineering*, pages 754–757, 2009.

[15] P. Massa and P. Avesani. Trust-aware Bootstrapping of Recommender Systems. In *ECAI 2006 Workshop on Recommender Systems*, pages 29–33, 2006.

[16] F. McSherry and I. Mironov. Differentially Private Recommender Systems: Building Privacy into the Netflix Prize Contenders. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 627–636, 2009.

[17] K. Miyahara and M. J. Pazzani. Collaborative Filtering with the Simple Bayesian Classifier. In *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, pages 679–689, 2000.

[18] H. Polat and W. Du. SVD-based Collaborative Filtering with Privacy. In *Proceedings of the 2005 ACM Symposium on Applied Computing*, pages 791–795, 2005.

[19] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 175–186, 1994.

[20] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of Dimensionality Reduction in Recommender Systems – A Case Study. In *Proceedings of ACM WebKDD Workshop*, 2000.

[21] S. Shang, Y. Huiy, P. Huiz, P. Cuff, and S. Kulkarni. Privacy Preserving Recommendation System Based on Groups. *http://arxiv.org/abs/1305.0540*, 2013.

[22] X. Su and T. M. Khoshgoftaar. Collaborative Filtering for Multi-class Data Using Belief Nets Algorithms. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, pages 497–504, 2006.

[23] J. Tang, H. Gao, and H. Liu. mTrust: Discerning Multi-faceted Trust in a Connected World. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*, pages 93–102, 2012.

[24] X. Wang, J. Zhang, P. Lin, N. Thapa, Y. Wang, and J. Wang. Incorporating Auxiliary Information in Collaborative Filtering Data Update with Privacy Preservation. *International Journal of Advanced Computer Science and Applications*, 5(4):224–235, May 2014.

[25] S. L. Warner. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *Journal of the American Statistical Association*, 60(309):63–69, March 1965.

[26] B. Yang, Y. Lei, D. Liu, and J. Liu. Social Collaborative Filtering by Trust. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages

2747–2753, 2013.

[27] J. Zhan, C.-L. Hsieh, I.-C. Wang, T. sheng Hsu, C.-J. Liau, and D.-W. Wang. Privacy-Preserving Collaborative Recommender Systems. *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)*, 40(4):472–476, 2010.

[28] S. Zhang, J. Ford, and F. Makedon. A Privacy-Preserving Collaborative Filtering Scheme with Two-way Communication. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, pages 316–323, 2006.

[29] S. Zhang, W. Wang, J. Ford, and F. Makedon. Learning from Incomplete Ratings Using Non-negative Matrix Factorization. In *Proceedings of the 6th SIAM International Conference on Data Mining*, pages 548–552, 2006.