Turk J Elec Eng & Comp Sci, Vol.18, No.1, 2010, © TÜBİTAK
doi:10.3906/elk-0808-21

P2P collaborative filtering with privacy

Cihan KALELİ, Hüseyin POLAT
Department of Computer Engineering, Anadolu University,
Eskisehir, 26470, TURKEY
e-mail: {ckaleli, polath}@anadolu.edu.tr

Abstract

With the evolution of the Internet and e-commerce, collaborative filtering (CF) and privacy-preserving collaborative filtering (PPCF) have become popular. The goal in CF is to generate predictions with decent accuracy, efficiently. The main issue in PPCF, however, is achieving such a goal while preserving users' privacy. Many implementations of CF and PPCF techniques proposed so far are centralized. In centralized systems, data is collected and stored by a central server for CF purposes. Centralized storage poses several hazards to users because the central server controls users' data.

In this work, we investigate how to produce naïve Bayesian classifier (NBC)-based recommendations while preserving users' privacy without using a central server. In a community of people, users might create a peer-to-peer (P2P) network. Through P2P network, users can communicate with each other and exchange data to produce predictions. We share the workload of prediction process and offer referrals efficiently using P2P network. We propose privacy-preserving schemes and analyze them in terms of accuracy, privacy, and efficiency. Our real data-based results show that our schemes offer accurate NBC-based predictions with privacy eliminating central server.

Key Words: *Privacy, P2P, collaborative filtering, naïve Bayesian classifier, accuracy.*

1. Introduction

With the advent of the Internet, e-commerce has become very popular. In order to increase their sales and profits, online vendors utilize various techniques like collaborative filtering (CF). To have competitive edge over competing companies, many online vendors employ CF techniques, which are widely used for filtering and recommendation purposes [1]. Online vendors incorporate recommendation systems that suggest products to customers on like-minded users' preferences about items they have bought before or showed interest.

The term "collaborative filtering" was first coined by the designers of Tapestry, a mail filtering software developed in the early nineties for the intranet at the Xerox Palo Alto Research Center [2]. There is an increasing commercial interest in CF technology with enhancing growth of e-commerce. With the number of users accessing the Internet growing and the number of items available online increasing, CF techniques are

becoming increasingly popular as part of online shopping sites. These sites leverage knowledge about the known preferences of users to recommend items of interest to other users [3].

CF systems first collect ratings for items from many users. Such ratings collected by a central server are stored on a user-item database. CF schemes then match together users who share the same interest or tastes. Such systems predict how well a user (active user a) will like an item that she did not buy before using a community of users' preferences about products [1, 3]. It is an assumption that similar users have similar preferences in CF [4]. Recommender systems predict a's preferences for items and provide predictions for single items (prediction) or a ranked list of items (top-N recommendation), which will most probably be liked by a. CF generally ignores the form and the content of the items and can therefore also be applied to non-textual items [4]. It can detect relationships between items that have no content similarities but are linked implicitly through the groups of users accessing them.

There are two main approaches of CF algorithms [5]: memory- and model-based CF techniques. In addition, there are hybrid CF approaches [3]. Memory-based algorithms utilize the entire user-item database to generate a prediction [6]. Although memory-based CF algorithms are widely used and they are successful; since they use entire user-item database, their computation time increases when number of users/items increases. Model-based CF algorithms provide recommendations by developing a model of user ratings. The model building process is performed by different machine learning algorithms such as Bayesian network, clustering, and rule-based approaches. Miyahara and Pazzani [7] propose to use naïve Bayesian classifier (NBC) to predict unknown class labels of target items. The Bayesian network model [5] formulates a probabilistic model for CF problem. The clustering model treats CF as a classification problem [8, 9, 10]; and works by clustering similar users in the same cluster. The rule-based approach [11] applies association rule discovery algorithms to find association between co-purchased items and then generates item recommendation based on the strength of the association between items.

For CF purposes, users' preferences about various products or items, also known as ratings or votes, can be collected explicitly or implicitly. Such ratings might be numeric showing how much a user likes or dislikes an item or binary showing whether a user likes an item or not. Although ratings are generally numeric; however, in some cases, CF systems collect binary ratings for their applications. Especially in market basket data analysis and document clustering, binary ratings are collected. When binary ratings are available and collecting ratings as binary is inevitable, to perform CF services efficiently, the most appropriate algorithms should be utilized. NBC is widely applied to offer CF services when binary data is utilized in CF process. NBC algorithm is one of the most successful classification algorithms and it is appropriate for producing recommendations from binary data.

Today's CF systems are very popular and widely used. However, they have some problems. The most important one is that they are a serious threat to individuals' privacy [12, 13]. The individuals share their data with data vendors, so there are several risks for individuals' privacy [14]. One of them is unsolicited marketing. Another risk is that users' profiles might be used in criminal case. Customer buying information and preferences are valuable assets, and they have been sold when some e-companies suffered bankruptcy. According to a survey conducted in 1999 [15], the privacy fundamentalists are concerned about any use of their data and are generally unwilling to provide their data to web sites. Pragmatists are also concerned about data use, but less than the fundamentalists. Users who are marginally concerned are generally willing to provide data to web sites under almost any condition, although they often express a mild general concern about privacy. Besides failing

protecting privacy, another challenge of a CF system is increasing computation time of recommendation process with increasing number of users/items. It is important to offer recommendations during an online interaction to many users. To be successful, CF systems should be able to provide predictions efficiently. Therefore, achieving higher performance is an important challenge for CF systems that should be achieved.

CF and privacy-preserving collaborative filtering (PPCF) schemes proposed so far are generally executed with a central server. Since users must send their ratings vectors to a central server for producing recommendations, the server controls all of the ratings. Therefore, there is a threat for users' privacy. The server performs all computations and provides many recommendations to loads of users during an online interaction. Using a central server for CF purposes is not desirable for privacy and performance reasons. It is still possible to produce CF services without using a central server. A group of people or peers can come together and construct a peer-to-peer (P2P) network for generating referrals. Centralized CF systems are used by online vendors and they are practical for e-commerce sites. However, for close communities or peers, like academicians, physicians, students, people in the same social class, and so on, generating recommendations using a P2P network is more desirable. In such communities, users can control their own data and peers. Using a P2P schema, we are able to break the control of a central server and prevent it from learning users' data. Also, if we apply P2P schema, we can divide the computation load among each peer; and each peer computes its own part. In this way, efficiency of CF algorithms might improve.

In this work, we investigate how to produce NBC-based CF services while preserving users' privacy without using a central server. We propose to use a P2P network among various peers to generate recommendations without violating their privacy. We study the privacy risks that might occur among such peers while providing predictions over a P2P network. To overcome such risks, privacy-preserving schemes are proposed. Due to privacy concerns, accuracy might be affected. To show how overall performance of our proposed schemes changes with privacy concerns, we conduct real data-based experiments. We analyze the schemes in terms of accuracy, privacy, and efficiency. Finally, we display our findings and provide suggestions.

2. Related works

Many of CF algorithms utilize a central server to offer referrals. However, for some applications, employing decentralized approaches is becoming popular. Implementing the CF algorithm in a decentralized way was initially proposed by Tveit [16], which presents a P2P architecture for producing recommendations for mobile customers of software agents. Although Olsson [17] proposes an improved mechanism, this new mechanism reduced the efficiency of the neighborhood formation phase. In [18], Sarwar et al. develop a different implementation of distributed CF for different domains of electronic commerce. Han et al. [19] propose a scalable P2P recommender system for CF. They call their distributed algorithm PipeCF. They utilize significance refinement and unanimous amplification to further improve scalability and accuracy. They implement their algorithm on a P2P structure through a distributed hash table method. Berkovsky et al. [20] propose a new CF approach to overcome sparseness and cold start problems by using multiple distributed information repositories. They utilize a multi-agent communication infrastructure. They employ CF to generate user-personalized recommendations over different data distribution policies.

With the advent of e-commerce and increasing concerns about privacy, PPCF is becoming increasingly popular. Polat and Du [21, 22] propose various centralized PPCF schemes, where users first disguise their ratings and send them to a central server. Since data is perturbed, the server is not able to learn true data.

In addition to centralized PPCF schemes, various decentralized PPCF approaches have been proposed in the literature. In [23], the authors study how to produce NBC-based CF services while preserving users' privacy using randomized response techniques (RRT). They utilize single- and multi-group schemes to disguise users' binary ratings. Canny [12, 24] proposes two PPCF schemes in which users control all of their own private data; a community of users can compute a public "aggregate" of their data without disclosing individual users' data. His schemes are based on distributed computation of a certain aggregate of users' data. He uses homomorphic encryption to allow sums of encrypted vectors to be computed and decrypted without exposing individual data. Berkovsky et al. [25] investigate how a decentralized approach to users' profiles storage might mitigate some of the privacy concerns of CF. They try to resolve privacy hazards by storing the users' profiles only on the client side. In their scheme, users' profiles are not revealed. Only similarity values are transferred over the network. Lathia et al. [26] propose a new measure of similarity, which can be calculated without breaking users' privacy over a distributed environment. Their new method works by estimating the number of concordant, discordant, and tied pairs of ratings between two users.

Kaleli and Polat [23] study centralized privacy-preserving NBC-based CF. In their scheme; the central server controls all of the users' data. Users' ratings are masked by dividing them into multi groups. With increasing number of groups, online computation cost increases. We investigate how to offer NBC-based CF services while preserving users' privacy without using a central server. In this way, we want to break the control of the server and divide the online computation work among various users or peers. After constructing a P2P network, peers are able to communicate and exchange different information to produce recommendations. In order to achieve privacy, we also employ RRT. However, unlike schemes in [23], where data is disguised in the same way by each user, in our proposed schemes, data can be differently masked.

3. P2P NBC-based collaborative filtering with privacy

Miyahara and Pazzani [7] apply NBC to CF, where they define two classes, like and dislike. Since customers vote items as like (1) or dislike (0), the sparse user ratings matrix includes Boolean values indicating whether the user rated items as 1 or 0. a's ratings for items are class labels of the training examples. In the user ratings matrix, other users correspond to features and the matrix entries correspond to feature values. The naïve assumption states that features are independent given the class label. Therefore, the probability of an item belonging to $class_j$, where j is like or dislike, given its n feature values, can be written as

$$p(class_j|f_{1,,}f_2,..., f_n) \propto |p(class_j) \prod_{i=1}^{n} p(f_i|class_j), \tag{3.1}$$

where both $p(class_j)$ and $p(f_i \mid class_j)$ can be estimated from training data and f_i corresponds the feature value of target item (q) for user i. To assign a target item q to a class, the probability of each class is computed, and the example is assigned to the class with the highest probability. Only known features and the data that both users commonly rated are used for predictions.

NBC is one of the most successful learning algorithms. Although it is applied to CF productively, however, with increasing number of users/items, its performance degrades significantly. Kaleli and Polat [23] show how to provide NBC-based predictions while preserving users' privacy. However, in their scheme, a central server collects disguised users' ratings. It controls all of the data and conducts all computations online to offer

recommendations. Since data is valuable asset, the server can use them for malicious purposes. Users mask their data in the same way in order to have consistently masked data. They propose to use one- and multi-group schemes. With increasing number of groups, however, their schemes' performances decrease considerably. We want to produce decentralized NBC-based CF services while preserving users' privacy. In our scheme, each user or peer controls her own data. We break the control of central server. Moreover, recommendation computation workload is distributed among the peers. Therefore, performance improves noticeably. Unlike the schemes proposed by [23], data is masked differently and our schemes are still able to provide predictions from such inconsistently perturbed data.

Our proposed schemes should preserve users' privacy, provide accurate recommendations efficiently. However, privacy, accuracy, and performance are conflicting goals. Although it is a challenge to define privacy succinctly, we can define it, as follows: All peers or users including active users should not be able to learn the true values of other peers' ratings. Moreover, since learning rated items might be more damaging, they should not learn rated items. To achieve privacy, we utilize RRT. Due to privacy concerns and RRT, it is expected that accuracy might worsen. However, recommendations provided with privacy concerns should still be accurate. Therefore, predictions estimated from masked data while preserving privacy should be as close as possible to true ratings. In other words, accuracy losses due to privacy concerns should be small and make it possible to offer acceptable recommendations. It is desirable to offer many referrals to loads of users during an online interaction. Therefore, our schemes' online computation times should be small and make it possible to generate referrals to various peers. Compared to centralized privacy-preserving NBC-based CF schemes, our decentralized schemes are expected to achieve higher performance because online computations are split among different peers.

RRT was first introduced by Warner [27] as a technique to estimate the percentage of people in a population that has attribute A. The interviewer asks each respondent two related questions, the answers to which are opposite to each other. Respondents are able to choose one of the questions to answer using a randomizing device. With probability θ, respondents choose the first question while with probability 1- θ; they select the second question to answer. Although the interviewer learns the responses, she does not know which question was answered.

RRT can be used to mask users' binary ratings. Ratings vectors usually contain binary ratings for rated items and blank cells for unrated items. An example of a ratings vector for user i is V_i = (1 1 0 – 0 1 – 0 0 1 1), where the bar "–" denotes an item not rated. To mask V_i using RRT, i uniformly randomly chooses a random number (r_i) over the range [0, 1]. If it is less than or equal to θ, then user i sends V_i. Otherwise, the user sends the false data. In other words, she sends the exact opposite of the ratings, which is called $\underline{V}_i$, where $\underline{V}_i$ = (0 0 1 – 1 0 – 1 1 0 0). With probability θ, true data is sent while false data is sent with probability 1-θ. Since random numbers are known by users only, the data collector does not know whether the received data is true or not. Users might decide to put their ratings into multiple groups; and mask ratings in each group independently.

In addition to centralized CF schemes, decentralized methods are receiving attention lately. With increasing popularity of the Internet, users are able to construct small networks to exchange and share various information over the Web. Decentralized approaches have advantageous like workload sharing, easy information exchange, eliminating control of central servers, controlling their own data, and so on. Therefore, we propose to use P2P network to generate NBC-based referrals. In a close community, users can construct a P2P network. The members of the network might academicians in a university, practitioners in a hospital, students in a

school, workers in a company, etc. Anyone's friends in her messenger list might be a good example for a P2P network. Peers in the network help each other to produce recommendations. Users in the P2P network might have ratings vectors for different categories like movies, books, music CDs, etc. Using such ratings, they are able to involve predictions processes for their peers. To produce a recommendation without privacy concerns over a P2P network, the peers follow the following steps:

1. When a or peer who wants a prediction for a target item (q), she sends a request including q to other peers (train users) in the network.

2. Those peers who rated $q$and want to involve CF process send their response back to a.

3. After determining train users or peers, a sends her ratings vectors to them. She then computes $p(class_j)$ values for both classes.

4. Each peer then computes $p(f_i \mid class_j)$ values for each class based on a's and their ratings and q. They then send the calculated values to a.

5. After receiving all values from all peers, a computes $p(class_j \mid f_{1,}, f_2, ..., f_n)$ probabilities for both classes using Eq. (3.1). She finally assigns her target item to the class with the highest probability.

With privacy concerns, a should not be able to learn the true ratings and the rated items of train users; and the train users should not learn a's ratings and rated items. Note that it is important to hide rated items, as well because it might be more damaging to reveal which items are rated. To increase privacy level, a perturbs her private data differently for each peer. To mask her data, a follows the following steps before sending her data to the train peers:

1. a first finds number of rated (m_r) and unrated $(m - m_r)$ items, where m denotes the number of items.

2. She decides a random integer (α_a) between 0 and 100. She then uniformly randomly chooses another random integer (δ_a) over the range [0, α_a].

3. She uniformly randomly selects δ_a percent of her unrated items' cells (w) and then fills half of them with 1s and the remaining cells with 0s. With increasing α_a values, a adds more randomness to her ratings vector. That might make accuracy worse while increasing privacy level. Therefore, she is able to decide α_a value in such a way to achieve a required balance between accuracy and privacy.

4. a uniformly randomly chooses $n_p \quad x \quad M_t$ number of θ (θ_{aiM}) values over the range [0, 0.5] because complementary θ_{ai} values produce the same results in terms of the randomness, where n_p shows the number of peers sending response to a to involve prediction process, $i = 1, 2, \ldots, n_p$, and M_t shows the total number of groups.

5. To determine number of groups for each peer, a uniformly randomly selects n_p number of M (M_{ai}) values over the range [2, γ], where M is a positive integer and γ should be a small number because as stated in [24], when M is bigger than 5, performance becomes worse.

6. After determining M_{ai} values for each peer, a divides her ratings into M_{ai} groups. She then uniformly randomly selects M_{ai} random numbers (r_{aiM}) for each peer over the range [0, 1]. And then she compares r_{aiM} random values with corresponding θ_{ai} values. If r_{aiM} is bigger than θ_{ai}, then a reverses her ratings in the corresponding group; otherwise, she keeps the same ratings values, as explained previously.

7. And finally, a sends her masked data to each peer together with the corresponding M_{ai} values. Remember that a masks her data differently for each peer and M_{ai} values are needed to compute probability values.

The peers cannot learn true ratings and rated items of a, because they do not know randomly selected blank cells and random numbers. Once they received perturbed ratings, q and M_{ai} values from a, each peer estimates $p(f_i \mid class_j)$ values as follows. Since the train peers receive M_{ai} values, they know each possible situations of a's perturbed vector. For example, if M_{ai} is 3, a's disguised ratings vector can have eight possible situations, such as TTT, TTF, TFT, etc., where T and F represent true and false data, respectively. For each possible case, the peers estimate $p(f_i \mid class_j)$ values for both classes. For M_{ai} is 3, the peer i estimates eight $p(f_i \mid class_j)$ values for both classes by considering eight possibilities. After each peer computes such values for both classes by considering each possible case, they then send them to a. Since a knows how she disguised her data for each peer, she considers those values that represent true ratings for both classes and then disregards the remaining values. Moreover, a can easily compute $p(class_j)$ values. And finally, a calculates $p(class_j \mid f_{1,}, f_2, ..., f_n)$ values; and assigns q to the class with the highest probability.

Due to data disguising by grouping ratings into multiple groups using RRT, there is no accuracy losses, because the peers consider all possible cases and a uses the values for true ratings vector. However, since a fills some of her blank cells for unrated items with v_{ad}, accuracy might become worse because v_{ad} might not represent true votes for unrated items.

3.1. Privacy attacks

In our proposed scheme, explained previously, although data is masked, the peers including the active peer might pose various privacy risks. Such risks might come from the train peers because they receive data from active peers and they might try to learn the true ratings and the rated items. The active peer might be able derive data from the results, which she receives from various train peers. In multiple scenarios, active peers can derive useful information about the train peers' data from $p(f_i \mid class_j)$ values. We group the privacy risks into two groups, and explain them as follows:

3.1.1. Train peers' privacy attacks

After receiving a disguised ratings vector from a, any train peer tries to learn the true ratings and the rated items. Since θ_{ai} and r_{ai} values, number of blank cells, and the blank cells are known by a, only, each peer cannot learn the true ratings and the rated items by herself. Although each peer cannot do anything to a's data, however, they might decide to collaborate to derive a's data. With increasing number of peers involving the collaboration, the chances they might learn a's data increase.

We propose the following solution against this privacy risk: a should select θ_{ai} values over the range [0, 1] in such way that the mean value of θ for those groups including the same items should be 0.5. In this way, when the train peers collaborate to learn the true ratings of the same items fall into the same group, the

probability of the received ratings to be true will be 0.5. Since the ratings are binary, this value does not help them at all. The peers cannot increase the chance to learn the rated items by collaboration, because a fills the same unrated items' cells for all peers. Prediction generations continue as any peer asks referrals. The active peer asks recommendations for different items time-to-time. In each process, she sends her masked data to each train peer. However, this might violate her privacy. Therefore, the active peer starts with the masked ratings vector. In the subsequent each recommendation process, she removes the oldest ratings and adds new ratings. Note that customers frequently buy items like movies, CDs, books, and so on over the Internet. Their ratings vectors include more and more ratings as they purchase new items. As shown by [28], using the recent ratings leads better quality recommendations. Therefore, the peers might decide to remove the oldest ratings and utilize the newest ratings to obtain high quality predictions. Since the ratings vectors employed in recommendation processes are changed in each prediction generation process, this prevents the peers from deriving data about each others' ratings through multiple prediction processes.

3.1.2. Active peer's privacy attacks

The attacks posed by a can be briefly explained as follows:

1. a can learn which peers rated the q which did not because when she sends a request to her peers on the P2P network for producing predictions, only those peers who rated the q send responses to her to involve in the process. By asking predictions for various items in multiple scenarios, a can learn which peers rated which items.

To get meaningful and trustworthy predictions, it is not possible to completely resolve this attack, because those peers who rated q should be involve in prediction process. However, we propose the following solution based on randomness to partially overcome this problem: When a request is sent by a, each peer on the network uniformly randomly selects a random value λ_i and a random number r_i over the range [0, 1]. If r_i is less than λ_i, then the peer i sends a response to a to participate in prediction process. On the average, half of the peers on the network join in referral computation. If they did not rate the q, they use their default vote (v_{id}) to fill its entry. Since a does not know λ_i and r_i values, she cannot learn which peers rated or not rated q.

2. In multiple scenarios, a might decide to use fake ratings and by changing the values of a single cell in her ratings vector each time to derive data about other peers' data. Since a gets $p(f_i \mid class_j)$ values, which are computed on other peers' and a's data, by changing the value of a single cell each time, a is able to derive other peers' data.

We propose the following scheme to overcome this attack: In each recommendation process, each peer first decides a random integer α_i between 0 and 100. They then uniformly randomly choose another random integer δ_i over the range [0, α_a]. They finally uniformly randomly select δ_i percent of their unrated items' cells and fill half of them with 1s and half of them with 0s like active peers. Since a does not know α_i and δ_i values and each time the peers utilize different values, she cannot derive information about the train peers' data in multiple scenarios.

3. Unlike previous attack, a might be able to derive data from $p(f_i \mid class_j)$ values computed for both classes and all possible cases without using fake ratings and without making any malicious change in her ratings

vector. She can compare the results of multiple recommendation processes to learn the true ratings and the rated items of other peers. However, she cannot derive useful information through this attack, because the train peers fill some of their randomly selected unrated items' cells with fake votes, as explained in the previous attack.

3.2. Privacy and overhead costs analysis

We can analyze our schemes in terms of privacy, as follows: The train peers do not figure out the rated items due to randomly filled unrated items' cells with bogus ratings. Note that a masks the same vector differently and sends it to each peer. The peers might guess the randomly filled cells. The probability of guessing α_a is 1 out of 100 because it is determined over the range [0, 100]. After guessing it, the probability of guessing the correct δ_a is 1 out of α_a. After that they can figure out the number of filled cells (w) with the help of the blank cells in the perturbed vector when it is not totally filled. Since the peers know that half of the filled cells ($w/2$) filled with 1s and the remaining $w/2$ cells filled with 0s, the probabilities of guessing which cells filled with 1s and 0s are 1 out of $C_{w_1}^{m_1}$ and $C_{w_0}^{m_0}$, where m_1 and m_0 represent the numbers of 1s and 0s, respectively; w_1 and w_0 show the numbers of randomly filled cells with 1s and 0s, respectively; and C_y^z shows the number of ways of selecting y outcomes from z possibilities. Therefore, the probability of figuring out the filled unrated items' cells is 1 out of $[100\ x\ \alpha_a \quad xC_{w_1}^{m_1} x \quad C_{w_0}^{m_0}]$. After that they can easily learn the rated items.

Each peer is not able to figure out a's true ratings by herself. They might decide to collaborate to increase the chance of guessing the true ratings. However, due to the underlying data disguising and selection of θ_{ai} values explained previously, they are not able to do that. Since the peers do not know the θ_{aiM} values and the random numbers (r_{aiM}), they cannot learn whether the received data is true or false in the same group. Even if they figure out the ratings in one group, they cannot learn the votes in other groups, because data is perturbed independently in different groups. Thus, with increasing number of groups, privacy enhances.

Remember that a receives $p(f_i \mid class_j)$ values rather than the peers' ratings vectors, the peers fill some of their randomly selected blank cells with fake ratings, they fill different cells in each process, and they estimate $p(f_i \mid class_j)$ values on filled vectors. Therefore, a is not able to learn the true ratings and the rated items through $p(f_i \mid class_j)$ values.

Finally, a might try to learn who rated the target item q or not. However, since the peers on the network decide to join the recommendation process or not based on the comparisons on λ_i and r_i values and they are known by the peers only, a cannot figure out who really rated the q or not.

Note that privacy, accuracy, and performance are conflicting goals. Due to our schemes, extra costs, such as storage, communication, and computation costs are expected. In order to reach higher performance, additional costs due to privacy concerns should be small. Unlike offline costs, online costs are critical for overall performance. Therefore, overhead online costs due to our privacy-preserving schemes should still make it possible to offer predictions efficiently.

Our schemes introduce additional storage costs, however, they are negligible. The peers on the network should save their default votes. Extra storage costs due to privacy concerns for each peer are order of 1 because each peer saves her default rating in a 1×1 matrix. Therefore, extra storage costs for all peers to save their default ratings are negligible. However, a should additionally save her filled ratings vector in a $1 \times m$ vector. Moreover, she should save M_{ai}, θ_{ai}, and r_{aiM} values. Therefore, on average, additional storage costs for a

are on the order of $n_p \times \gamma$. Since γ is a small constant, due our privacy-preserving schemes, extra storage costs are small.

Due to privacy concerns, the proposed schemes do not cause any extra communication. However, the amount of information exchanged between a and train peers increases due to our scheme. a additionally sends her filled ratings and M_{ai} values to other peers. The train peers sends $p(f_i \mid class_j)$ values for all possible cases rather than single values for both classes to a. Therefore, on average, overhead communication costs in terms of the amount of data are order of $2^{\gamma/2}$. Note again that γ is a small constant and a should be able to determine its value in such a way to achieve required levels of privacy and efficiency.

Supplementary computation costs due to data disguising are negligible. However, the train peers compute $p(f_i \mid class_j)$ values, on average, $2^{\gamma/2}$ times for both classes rather than a single value. Therefore, additional computation costs are order of $2^{\gamma/2}$. Remember that that γ is a small constant and the overhead costs are split among all train peers in our P2P network-based schemes. In the schemes proposed by [24], with increasing M values, computation costs increase exponentially, because the central server conducts all computations. However, in our schemes, all train peers join the computation and the costs are split among them. They perform such computations in parallel. Without a P2P network, such extra computations should be performed entirely by a central server.

To sum up, supplementary costs due to our schemes are inevitable, because privacy, accuracy, and performance are conflicting goals. Extra costs are negligible and our schemes are still able to provide predictions efficiently. Moreover, the peers can determine the values of data disguising parameters in such a way to reach required levels of privacy and performance.

3.3. Accuracy analysis: experimental results

Due to privacy concerns, accuracy might become worse. To evaluate how various privacy concerns and their solutions on randomization affect the accuracy of our P2P NBC-based schemes, we performed various experiments using real data sets, Jester and EachMovie (EM). Jester [29] database has 100 jokes and records of 17,988 users. The ratings range from -10 to +10, and the scale is continuous. The DEC Systems Research Center collected EM [30] contains ratings of 72,916 users for 1,628 movies. User ratings were recorded on a numeric six-point scale, ranging from 0 to 1. Although there are other data sets available for CF, our results based on these two sets can be generalized. Jester is denser and almost 50% of all ratings are available. To measure the accuracy of our schemes, we used classification accuracy (CA) and F1 metric (F1). CA is the ratio of the number of correct classifications to the number of classifications. F1 combines precision (P) and recall (R) with an equal weight, where precision and recall are used for information retrieval tasks, where F1 = $(2 \times R \times R)/(P + R)$. Larger is CA and F1, better are the results.

We first transformed numerical ratings into binary. Using the similar methodology in [7], we labeled items as 1, if the numerical rating for the item is bigger than 0.5, or 0 otherwise in EM, while we labeled them as 1, if the numerical rating for the item is above 2.0, or 0 otherwise in Jester. We first selected those users who rated more than 60 items from both data sets. We randomly divided each data set into two disjoint sets, train and test. For each experiment, we then randomly selected the required numbers of train and test users from train and test sets, respectively, based on the experiment requirements. For each test user (active user or peer), we randomly picked five rated items, replaced their entries with null, and tried to predict their votes. We

compared predicted votes with true withheld ratings. We performed each trial 100 times to obtain trustworthy results. After computing CA and F1 values, we demonstrated the final overall outcomes. As stated in [28], using the latest ratings makes accuracy better. After a certain point, old ratings reflect the current ratings and leads to lower predictions quality. Therefore, using recent ratings enhance accuracy [28]. In our schemes, we propose to use the most up to date ratings and remove the dated ratings to prevent privacy attacks. Since the effects of using current ratings are explained in [28], we did not perform experiments for this purpose.

To show how filling some of a's unrated items' cells with fake ratings affects accuracy, we performed trials using both data sets. We used 1,000 and 500 train and test users, respectively, from both data sets. We varied α_a from 0 to 100 to show how different δ_a values affect accuracy. Note that when α_a is 0, a does not fill any unrated items' cells. Once we determine α_a, we uniformly randomly selected δ_a over the range [0, α_a]. After calculating CA and F1 values for both data sets with varying α_a values, we displayed the outcomes in Figure 1 and Figure 2, respectively.

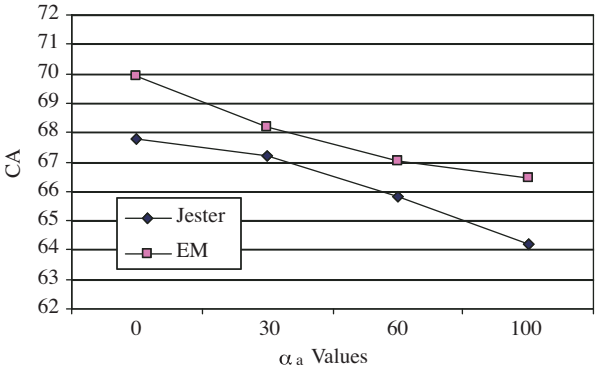

Figure 1. CA with varying α_a values.

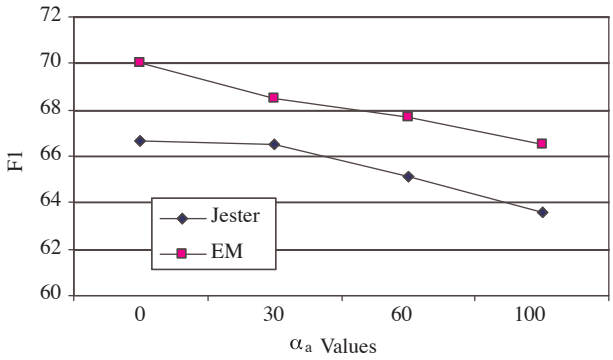

Figure 2. F1 with varying α_a values.

As seen from Figure 1 and Figure 2, accuracy slightly becomes worse with increasing α_a values. Although we insert the same number of 1s and 0s into some of the unrated items' cells, such fake ratings might not represent a's true preferences about those unrated items. Therefore, we can say that randomness boosts with augmenting α_a values; and it is expected that the quality of the predictions becomes poorer. However, accuracy losses due to inserting fake ratings into some randomly selected unrated items' cells are negligible and it is still possible to offer referrals with decent accuracy. For Jester, when α_a is 60, CA losses are 2% only. When it is 100, they are more than 3%. In terms of F1 metric, the results are slightly better compared to CA. Our results are similar for EM data set.

In our second set of experiments, we demonstrated how inserting bogus votes into the train users' randomly selected unrated items' cells affects our results. For the same reasons explained in the first set of experiments, quality of the predictions might be affected by inserting false votes into train users' unrated items' cells. To show such affects, we performed trials using both data sets. We again used the same 1,000 and 500 train and test users, respectively, from both sets. We varied α_i from 0 to 100 to show how different δ_i values affect accuracy. Once again, after determining α_i, we uniformly randomly selected δ_i over the range [0, α_i]. After calculating CA and F1 values for both data sets, we demonstrated our outcomes in Figure 3 and Figure 4, respectively.

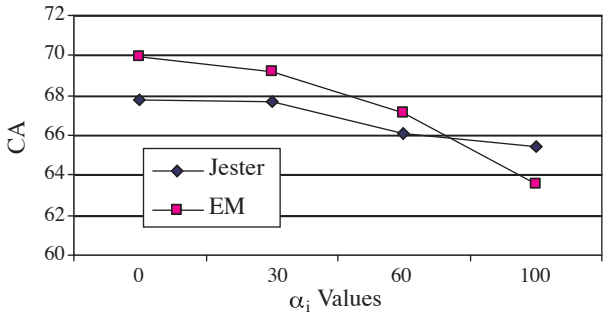

Figure 3. CA with varying α_i values.

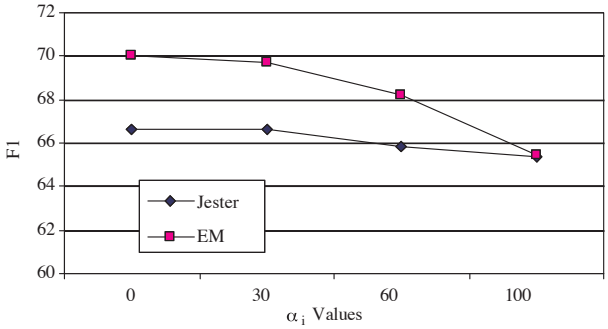

Figure 4. F1 with varying α_i values.

As seen from Figures 3 and 4, and as expected, the quality of recommendations gets worse with increasing α_i values. For both data sets, our results, in terms of CA and F1, are similar and get somewhat worse. Due to the same reasons explained previously, randomness increases with augmenting α_i values. As expected, accuracy worsens with increasing randomness. However, as seen from Table 2, accuracy losses due to privacy concerns are small and our proposed schemes still make it possible to offer accurate referrals.

Finally, we performed experiments to show how different numbers of the peers involving the prediction computations affect the overall performance of our schemes. Remember that the peers decide whether to join the recommendation process or not based on the comparison of λ_i and r_i values. We performed experiments using both data sets while varying numbers of train peers (n_p) from 100 to 2,000 for both data sets. We used 500 test users. We calculated overall averages of CA and F1 values. Since the results are similar, we displayed CA values only in Figure 5. Note that without privacy concerns, those peers who rated q involve in prediction generation. However, with privacy concerns, the peers join the recommendation process based on the comparison of λ_i and r_i values. If they did not rate q, they fill its cell with their default vote (v_{id}).

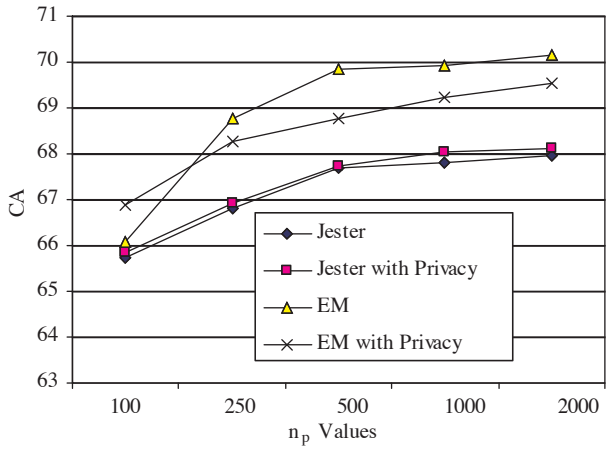

Figure 5. CA with varying n_p values.

Without privacy concerns, for both data sets, the quality of the recommendations turns out to be better with increasing n_p values. Although with increasing number of peers involving in recommendation computations, CA enhances; however, after 500 peers, such improvements become stable. Once there are enough peers to generate referrals, increasing the number of train peers does not enhance the results too much. Although the

results barely become worse with privacy concerns for EM data set, accuracy somewhat improves for Jester. It is expected that the quality of the predictions worsens with privacy concerns due to randomness. Improvements in Jester with privacy concerns might be explained by the density of Jester. EM data set is sparse while Jester is a dense set; and 50 % of all ratings are available. Therefore, v_{id} values can be estimated with decent accuracy and they may represent peers' true preferences. Moreover, number of peers involving recommendation processes increase. To sum up, our results are promising; and they make it possible to generate truthful predictions.

4. Discussion

We propose a privacy-preserving P2P CF scheme, which allows peers to generate recommendations without deeply violating their privacy. Our scheme's most important advantage is that it eliminates the use of a central server. Since workload is distributed, our scheme is scalable. Moreover, the peers are able to disguise their private data variably. Finally, our scheme is more suitable in community-based CF systems. In other words, for close communities like academicians, students, people in the same social class, and so on, our proposed scheme is more appealing. On the other hand, our scheme is not suitable for systems providing online predictions based on a central server like many online vendors utilize. Furthermore, since users actively participate in CF process in our scheme, it is not appropriate for those users who do not want to actively join the process. Finally, the proposed scheme is based on binary ratings rather than numerical ones. Although this seems to be a weakness, numerical ratings can be converted into binary ones.

In our proposed scheme, the peers utilize the CF algorithm proposed by Miyahara and Pazzani [7] in a distributed manner. The authors do not take privacy concerns into account. We provide predictions while preserving users' privacy. Although their scheme's performance degrades with increasing amount of data, since workload is shared between the peers, our scheme is able to perform CF tasks more efficiently. In terms of accuracy, due to privacy measures, the quality of predictions decreases in our scheme compared to their scheme. This phenomenon is expected because privacy-preserving measures add randomness to private data. Note that in the figures displayed in Section 4, the results for α being 0 show the outcomes without privacy concerns. In other words, such results are obtained using the algorithm proposed by Miyahara and Pazzani [7] without any privacy-preserving measure. Canny [24] proposes a distributed CF scheme in which the peers generate a model off-line in a distributive manner. Unlike his scheme, in our scheme, the peers do not construct a model. Since his scheme is model-based, inserting new users' data is not an easy task. The model should be updated periodically. However, any new user can easily join into our proposed scheme. Online performance of Canny's scheme is better than ours because the model is constructed off-line. His scheme is based on numerical ratings and the mean absolute error of his scheme is 0.96 for EM, which happens to be the best performance by any scheme on EM [24]. Our scheme is based on binary ratings and accuracy diminishes due to privacy concerns. We utilize RRT for data disguising while Canny uses homomorphic encryption for privacy. In the scheme proposed in [25], users do not hide the target item's ratings. They send the true ratings of the target item. Moreover, in their scheme, users do not disguise their entire ratings; they only obfuscate some of the ratings by inserting some default votes. They then send the masked ratings. Unlike their scheme, in our method, the peers disguise every rating including the target item's ratings; and we utilize RRT for data masking. The normalized mean absolute error is between 0.15 and 0.18 in their method for Jester. This implies that the generated predictions are similar to their real ratings [25]. In our scheme, for Jester, when n_p is 1,000, CA slightly improves from 67.80 to 68.04. However, for EM, when n_p is 1,000, CA slightly decreases from 69.92 to 69.24. As shown in Figure 5,

although the quality of the predictions generally diminishes with privacy measures, with increasing n_p values, our results become stable and close to their original values. In [26], the peers can infer each other's profiles based on the received recommendations because such predictions are exchanged between peers. Moreover, in their scheme, some users may collude to deduce the opinions of the target user. Compared to their scheme, our method is more secure because predictions are known by active peers only; and it is secure against colluding attacks. When there are 1,000 users, there are no accuracy losses due to privacy concerns in their scheme [26], where they utilize sparse data set. However, when number of users is less than or equal to 500, accuracy losses increase with decreasing number of users. For example, accuracy loss is around 1.8% for 500 users while it is 5% for 100 users. In our scheme, for EM, which is sparse, due to privacy concerns, accuracy losses are around 1% and 1.6% when there are 1,000 and 500 users, respectively. On the other hand, for Jester, which is dense, accuracy slightly improves when privacy protection measures are in place with increasing number of users.

5. Conclusions and future work

In this work, we investigated how to provide accurate P2P network-based referrals efficiently using NBC. Moreover, we showed that it is still possible to generate such recommendations while preserving the users' privacy. We demonstrated that NBC-based predictions can be provided without a central server. Close community groups are able to construct P2P network for various purposes including CF. Without a central authority, the control of the central server is broken; and the recommendation computation workload is split among the peers. Therefore, performance is expected to enhance significantly. We performed real data-based experiments to evaluate the overall performance of our schemes. Our results show that accuracy losses due to privacy concerns are negligible and make it possible to offer accurate recommendations. We also analyzed our schemes in terms of privacy and overhead costs. We showed that our schemes are secure and the overhead costs are small.

Although we can easily extend our schemes to generate top-N recommendations, we believe that detail work should be conducted. In addition to NBC, we will study whether we can employ P2P network-based schemes to other CF filtering algorithms or not. We will search other data disguising techniques to utilize in order to further improve the computation costs.

References

[1] J.L. Herlocke, J.A. Konstan, J.T. Riedl, "Explaining Collaborative Filtering Recommendations", *Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work*, pp. 241–250, USA, 2000.

[2] D. Goldberg, D. Nichols, and B.M. Oki, "Using collaborative filtering to weave an Information Tapestry," *Communications of the ACM,* 35, 12, pp. 61–70, 1992.

[3] D.M. Pennock, E. Horvitz, S. Lawrence, C.L. Giles, "Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-Based Approach," *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence,* pp. 473–480, Stanford, CA, USA, July 2000.

[4] M. Grcar, "User profiling: collaborative filtering", *Proceedings of the SIKDD 2004 at Multiconference IS*, Ljubljana, Slovenia, 2004.

[5] J. S. Breese, D. Heckerman, C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 43–52, USA, 1998.

[6] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl," Item-based collaborative filtering recommendation algorithms," *Proceedings of the 10th International World Wide Web Conference*, pp. 285–295, Hong Kong, 2001.

[7] K. Miyahara, M. J. Pazzani, "Improvement of collaborative Filtering with the simple Bayesian classifier", *IPSJ Journal*, Vol. 43(11), 2002.

[8] C. Basu, H. Hirsh, W. W. Cohen, "Recommendation as classification: Using social and content-based information in recommendation," *Proceedings of the Recommender System Workshop'* 98, pp. 714–720, 1998.

[9] S. Meruge, J. Ghosh, "Privacy-preserving distributed clustering using generative models", *Proceedings of the 3rd IEEE International Conference on Data Mining,* USA, pp. 211–218, 2003.

[10] L. H. Ungar, D. P. Foster, "Clustering methods for collaborative filtering," *Proceedings of the Workshop on Recommendation Systems at the 15th National Conference on Artificial Intelligence*, USA, 1998.

[11] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, "Analysis of recommendation algorithms for e-commerce", *Proceedings of the 2nd ACM conference on Electronic commerce*, pp. 158–163, Minnesota, USA, 2000.

[12] J. Canny, "Collaborative Filtering with privacy via factor analysis", *Proceedings of the* 25^{th} *ACM SIGIR'02*, pp. 238–245, Finland, 2002.

[13] A. F. Westin, "Freebies and privacy: What net users think." Technical report Opinion Research Corporation, 1999.

[14] L. F. Cranor, "'I didn't buy it for myself' privacy and e-commerce personalization," *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*, pp. 111–117, USA, 2003.

[15] L. F. Cranor, J. Reagle, M. S. Ackerman, "Beyond concern: Understanding net users' attitudes about online privacy," Technical report, AT T Labs-Research, 1999.

[16] A. Tveit, "Peer-to-Peer Based Recommendations for Mobile Commerce," *Proceedings of the International Workshop on Mobile Commerce*, pp. 26–29, Italy, 2001.

[17] T. Olsson, "Decentralized Social Filtering based on Trust," *Proceedings of the National Conference of the American Association of Artificial Intelligence Recommender Systems Workshops*, Madison, WI, 1998.

[18] B. M. Sarwar, J. A. Konstan, J. Riedl, "Distributed Recommender Systems: New opportunities for Internet Commerce," *Internet Commerce and Software Agents: Cases, Technologies and Opportunities*, Idea Group Publishers, 2001.

[19] P. Han, B. Xie, F. Yang, R. Shen, "A Scalable P2P Recommender System Based on Distributed Collaborative Filtering," *Expert Systems with Applications*, Vol. 27, pp. 203–210, 2004.

[20] S. Berkovsky, Y. Eytani, T. Kuflik, F. Ricci, "Collaborative Filtering over Distributed Environment," *Proceedings of the Workshop on Decentralized, Agent-Based and Social Approaches to User Modeling (DASUM), at the International Conference on User Modeling (UM)*, pp. 1–11, UK, 2005.

[21] H. Polat, W. Du, "Privacy-preserving collaborative filtering," *International Journal of Electronic Commerce*, 9 (4), pp. 9–36, 2005.

[22] H. Polat, W. Du, "Achieving private recommendations using randomized response techniques", *Lecture Notes in Computer Science*, 3918, pp. 637–646, 2006.

[23] C. Kaleli, H. Polat, "Providing Private Recommendations Using Naïve Bayesian Classifier," *Advances in Intelligent Web Mastering*, Vol. 43, pp. 168–173, 2007.

[24] J. Canny, "Collaborative filtering with privacy", *Proceedings of the IEEE Symposium on Security and Privacy (IEEE S&P'02)*, pp. 45–57, 2002.

[25] S. Berkovsky, Y. Eytani, T. Kuflik, F. Ricci, "Privacy-Enhanced Collaborative Filtering," *Proceedings of the Workshop on Privacy-Enhanced Personalization (PEP), at the International Conference on User Modeling (UM)*, pp. 75–83, UK, 2005.

[26] N. Lathia, S. Hailes, L. Capra, "Private Distributed Collaborative Filtering Using Estimated Concordance Measures," *Proceedings of RecSys'07*, USA, 2007.

[27] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, Vol. 60(309), pp. 63–69, 1965.

[28] J. K. Kim, H. K. Kim, Y. H. Cho, "A user-oriented contents recommendation system in peer-to-peer architecture," *Journal of Expert Systems with Applications*, 34, pp.. 300-312, 2008.

[29] D. Gupta, M. Digiovanni, H. Narita, K. Goldberg, "Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes," *Proceedings of the Workshop on Recommender Systems: Algorithms and Evaluation*, 22nd Annual International ACM SIGIR Conference, Berkeley, CA, USA, August 1999.

[30] P. McJonese, "EachMovie collaborative filtering data set", DEC Systems Research Center, 1997.