Privacy-Preserving Collaborative Filtering

Huseyin Polat and Wenliang Du

ABSTRACT: Collaborative filtering (CF) techniques are becoming very popular on the Internet and are widely used in several domains to cope with information overload. E-commerce sites use filtering systems to recommend products to customers based on the preferences of like-minded customers, but their systems do not protect user privacy. Because users concerned about privacy may give false information, it is not easy to collect high-quality user data for collaborative filtering, and recommendation systems using poor data produce inaccurate recommendations. This means that privacy measures are key to the success of collecting high-quality data and providing accurate recommendations.

This article discusses collaborative filtering with privacy based on both correlation and singular-value decomposition (SVD) and proposes the use of randomized perturbation techniques to protect user privacy while producing reasonably accurate recommendations. Such techniques add randomness to the original data, preventing the data collector (the server) from learning private user data, but this scheme can still provide accurate recommendations. Experiments were conducted with real datasets to evaluate the overall performance of the proposed scheme. The results were used for analysis of how different parameters affect accuracy. Collaborative filtering systems using randomized perturbation techniques were found to provide accurate recommendations while preserving user privacy.

KEY WORDS AND PHRASES: Accuracy, collaborative filtering, privacy, randomized perturbation, SVD.

Collaborative Filtering

With the evolution of the Internet, information overload is becoming a major problem for users. Different approaches are used to separate what is interesting and valuable from other information so as to cope with this problem. Information filtering and recommendation schemes have become increasingly important elements of good service to users. Collaborative filtering (CF), or social filtering, is a recent technique for filtering and recommendation purposes.

The relatively new concept of collaborative filtering is descended from work done in the area of information filtering. 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 [16]. With the growth of e-commerce, there has been increasing commercial interest in filtering technology. Collaborative filtering techniques are widely used in e-commerce, direct recommendations, and search engines. As the number of users accessing the Internet grows, such techniques are becoming ever more popular parts of on-line shopping sites. They are used to leverage knowledge...
about the known preferences of users to recommend items of interest to other users [26]. Web sites like Amazon.com, CDNow.com, and MovieFinder.com have made successful use of collaborative filtering [20]. The collaborative filtering system on Amazon.com (www.amazon.com), for example, suggests books to customers based on other books they have told Amazon they enjoy.

Collaborative filtering systems work by collecting numerical ratings for items and matching users whose ratings indicate that they share the same interests or tastes. The goal of such systems is to predict how well a user, referred to as the active user, will like an item that he/she has never purchased before, based on the preferences of a community of users [19, 26]. For example, given the active user’s ratings for several movies and a database of other users’ ratings, the system predicts how the active user would rate movies not yet seen. The key idea is that the active user will prefer items that like-minded users prefer or that dissimilar users do not prefer. The fundamental assumption holds that if User1 and User2 rate h items similarly, they share similar tastes, and therefore will rate other items congruently [17]. Approaches to CF differ in the ways they define “rating,” “h,” and “similarly.” The task is to predict the votes of the active user based on a database of user votes (the user database) from a sample or population of other users [7]. The user database consists of a set of votes, v_uj, corresponding to the vote for user u on item j.

Resnick and Varian assume that a good way to find interesting content is to find other people who have similar interests and then recommend titles that they like [28]. Other filtering systems like PHOAKS and Referral Web were developed in order to recommend Web resources [21, 35]. The Fab system combines content and collaborative filtering by identifying user interests [5]. It then forwards highly rated documents to users with similar profiles. Miyahara and Pazzani discuss an approach to collaborative filtering based on the Naive Bayesian Classifier and apply their model to two variants of collaborative filtering [24]. Lemire modifies a wide range of filtering systems to make them scale and translation invariant, and generally improves their accuracy without increasing computational cost [23].

**Privacy-Preserving Collaborative Filtering Problem**

With the help of collaborative filtering, users can get recommendations about many of their daily activities, including but not limited to movies, books, news, music CDs, and things to do in a city [9]. Collaborative filtering systems have been applied successfully in several domains but also have a number of disadvantages [9, 10]. The most serious is the threat to individual privacy. They offer great potential for individuals to share all sorts of information about places and things to do, see, and buy, but they also entail severe privacy risks. Most on-line vendors try to preserve the privacy of the customers whose preferences they collect. However, several schemes are extremely vulnerable and can be mined for user preferences [10]. In addition, customer data are a valuable asset, and e-companies in financial difficulties have sometimes sold them. The fragility of the digital marketplace has forced many e-companies into bankruptcy, and the courts have ruled that in such cases the liquidators have
the right to sell off data about customers’ personal information as an asset. Note, for instance, Amazon.com’s policy: “In the unlikely event that Amazon.com, Inc., or substantially all of its assets are acquired, customer information will of course be one of the transferred assets.” This policy is typical of many other companies, and despite Amazon’s language, the papers are filled with news of “unlikely events” leading to bankruptcies and transfers of customer information. Because data from users are needed for collaborative filtering purposes and many users have concerns about their privacy, providing privacy measures is a key to the success of collaborative filtering services.

E-commerce sites collect buying information about their customers and use it to build personal profiles. These profiles pose several privacy risks [11]. One is the problem of unsolicited marketing. Users are concerned that information they provide for use in personalized e-commerce may be utilized to send them targeted advertising or may be sold to other companies that will advertise to them. Many customers are also concerned about the possibility of a computer figuring things out about them. A further risk that most people do not consider is that the information in their profile may be subpoenaed in a criminal case or in civil litigation. Individuals are also concerned that companies will profile them in order to facilitate price discrimination. Finally, computer users are afraid of being subject to government investigation.

Some people are willing to selectively divulge information if they receive benefits in return [36]. Examples of the benefits provided include discounts on purchases, useful recommendations, and information filtering. However, a survey in 1999 showed that privacy concerns make a significant number of people unwilling to divulge information [12]. Among the survey respondents, 17 percent were privacy fundamentalists—people so concerned about misuse of their data that they were unwilling to provide data even with privacy protection measures in place. However, 56 percent of respondents comprised a pragmatic majority. They too were concerned about data usage, but less so than the fundamentalists, and their concerns were significantly reduced by the presence of privacy-protection measures. The remaining 27 percent were only marginally concerned and therefore willing to provide data under almost any condition, although they expressed mild general concerns about privacy. One of the main bottlenecks faced by collaborative filtering systems is collecting truthful data. Users concerned about privacy may decide to give false rather than true information. Filtering systems using false data produce inaccurate predictions. The goal of introducing privacy is to assure users who are willing to participate in data collection that the filtering system can be trusted to provide truthful information, and not to induce all users, including the privacy fundamentalists, to participate. Computer users feel more comfortable about sharing their personal preferences about products when privacy measures are in place. Therefore, providing privacy protection measures is the key to success both in collecting high-quality data and providing accurate recommendations. The challenge is to find a way that users can contribute their personal information for collaborative filtering purposes without greatly compromising their privacy. How can we improve the chances of collecting the truthful data necessary for collaborative filtering while preserving user privacy?
Anonymous techniques are widely used to achieve privacy [1, 27, 34]. Such techniques allow users to divulge their data without disclosing their identities. The major problem with anonymous techniques, however, is that there is no way to ensure the quality of the dataset—malicious users could send random data and render the database useless, and competing companies could send made-up information to make their products seem more desirable. Attacks of this kind could render the database useless. If the communication is truly anonymous, the database owner may be unable to control the quality of the data. It is imperative for the database owner to verify the identities of the data contributors in order to guarantee data quality.

**Outline of a Solution**

The goal of collaborative filtering with privacy is to ensure user privacy and provide the most accurate recommendations possible. Privacy and accuracy are conflicting goals, however. Improving one of them decreases the other. The discussion that follows proposes a technique that can achieve a good balance between privacy and accuracy such that collaborative filtering systems are able to produce accurate recommendations but still preserve user privacy. The server is prevented from learning how much users like or dislike items they rated earlier and what items each user has rated. In other words, the proposed technique addresses the problem of knowing what items a user has already voted. After all, users might feel more violated if it is revealed that they voted a given item (e.g., a pornographic site or magazine) than if their specific ratings are revealed. Therefore, the proposal addresses the full privacy issue while providing accurate recommendations.

The proposed scheme to allow privacy-preserving collaborative filtering is depicted in Figure 1. Each user disguises his/her private data and sends it to the server. The server cannot derive truthful information about the user’s private information, but the data-perturbing scheme still allows the server to conduct collaborative filtering from perturbed data. Private data are disguised by means of randomized perturbation (RP) techniques. These techniques are useful if one is interested in aggregated data rather than individual data items, because when the numbers of users and items are significantly large, the aggregate information can be estimated with decent accuracy. Since collaborative filtering is based on the aggregate values of a dataset rather than on individual data items, one may hypothesize that by combining randomized perturbation techniques with collaborative filtering algorithms, it is possible to achieve a decent degree of accuracy for the collaborative filtering with privacy.

This hypothesis was verified by implementing the randomization technique for the correlation-based and SVD-based collaborative filtering algorithms [19, 31]. A series of experiments using the Jester and MovieLens (ML) datasets was then conducted to ascertain the accuracy of the results. The overall performance of the proposed scheme was measured based on disguised data. The results show that the predictions found on randomized data are very close to the original ratings.
Related Work

Collaborative Filtering with Privacy

Canny proposes alternative schemes for privacy-preserving collaborative filtering [9, 10]. In his schemes, users control all of their own private data, and a community of users can compute a public “aggregate” of their data without disclosing the data of individual users. The aggregate allows personalized recommendations to be computed by members of the community or by outsiders. His methods reduce the collaborative filtering task to an iterative calculation of the aggregate, requiring only the addition of vectors of user data. He then uses homomorphic encryption to allow the sums of encrypted vectors to be computed and decrypted without exposing individual data. His scheme is based on distributed computation of a certain aggregate of the data of all the users. The aggregate is treated as public data. Each user constructs the aggregate and uses local computation to get personalized recommendations. The scheme can be implemented with untrusted servers, or with additional infrastructures, such as a fully peer-to-peer (P2P) system. The P2P architecture allows users to create and maintain their own recommender groups.

The scheme proposed here differs from Canny’s. His work focuses on a peer-to-peer framework in which users actively participate in a collaborative filtering process, whereas the present proposal focuses on another framework in which users send their data to a server and do not participate in the

Figure 1. Privacy-Preserving Collaborative Filtering
filtering process—only the server needs to conduct the collaborative filtering. The server collects data from many customers and conducts collaborative filtering using an existing database. Both frameworks have their applications. The peer-to-peer framework is most suitable in community-based filtering systems, but the framework proposed in this paper is more appropriate for systems that provide on-line collaborative filtering services, such as Amazon and Yahoo.

**Randomized Perturbation Techniques**

Agrawal and Srikant used RP techniques to achieve privacy [4]. A simple way to disguise a number $a$ is to add a random value $r$ to it. Thus $a + r$, rather than $a$ alone, will appear in the database, where $r$ is a random value drawn from some distribution. Although nothing can be done to $a$ because it is disguised, certain computations are possible if one is interested in the aggregate of data.

The basic idea of RP techniques is to perturb the data in such a way that the server can only know the range of the data, and the range is broad enough to preserve user privacy. The information from each individual user is scrambled, but if there are enough users and items, the aggregate information of the users can be estimated with decent accuracy. Such a property is useful for computations based on aggregate information. One can still generate meaningful outcomes for these computations without knowing the exact values of individual data items, because the needed aggregate information can be estimated from the scrambled data. Because collaborative filtering is also based on aggregate information rather than individual data items, RP techniques can be applied to them. Evfimievski proposes some methods and results in randomization for numerical and categorical data and tries to recover the aggregate properties of the data [14].

**Background**

**Correlation-Based Collaborative Filtering Algorithm**

GroupLens introduced an automated collaborative filtering system using the correlation-based algorithm to provide personalized predictions for Usenet news articles [22, 29]. An extension of the GroupLens algorithm was proposed by Herlocker, Konstan, Borchers, and Riedl [19]. They compare the performance of different normalization techniques, such as bias-from-the-mean, $z$-scores, and nonnormalized rating. The $z$-scores perform better than the nonnormalized rating approach. The mean and the standard deviation of the $z$-scores are 0 and 1, respectively. If the $v_{ij}$ is user $u$’s vote on item $j$, $\bar{v}_u$ is the mean vote for user $u$, and $\sigma_u$ is the standard deviation for user $u$, then the $z$-score ($z_{uj}$) can be defined as $z_{uj} = (v_{uj} - \bar{v}_u)/\sigma_u$. Herlocker et al. account for the differences in spread between users’ rating distributions by converting ratings to $z$-scores. They compute a weighted average of the $z$-scores:
where \( n \) is the total number of users, and \( w_{au} \) is calculated as follows:

\[
\begin{align*}
    w_{au} &= \frac{\sum_{d=1}^{n} (v_{ad} - \overline{v}_a)(v_{ud} - \overline{v}_u)}{\sigma_a \sigma_u},
\end{align*}
\]

The summation over \( d \) is over the items for which both active user \( a \) and user \( u \) gave ratings. \( \sigma_a \) and \( \sigma_u \) are the standard deviations of the ratings by active user \( a \) and user \( u \), respectively. \( p_{aq} \) is the predicted vote for active user \( a \) for item \( q \).

**Singular Value Decomposition**

Singular value decomposition (SVD) is a well-known matrix factorization technique [31] that factors an \( n \times m \) matrix \( A \) into three matrices as \( A = USV^T \), where \( U \) and \( V \) are two orthogonal matrices of size \( n \times y \) and \( m \times y \), respectively; \( y \) is the rank of matrix \( A \). \( S \) is a diagonal matrix of size \( y \times y \) having all singular values of matrix \( A \) as its diagonal entries. The \( y \times y \) matrix \( S \) can be reduced to have only the \( k \) largest diagonal values to obtain a matrix \( S_k \), \( k < y \). The basic steps of SVD are as follows:

1. Find the eigenvalues \((\lambda_i)\) and the number of nonzero eigenvalues of the matrix \( A^T A \).
2. Find the orthogonal eigenvectors of the matrix \( A^T A \) corresponding to the obtained eigenvalues and form the column-vectors of the matrix \( V \).
3. Form the diagonal matrix \( S \) by placing on the leading diagonals the square roots of eigenvalues, \( s_i = \sqrt{\lambda_i}, i = 1, \ldots, y \).
4. Find the column-vectors of matrix \( U \) using \( b_i = s_i^{-1} A v_i, i = 1, \ldots, y \), and arrange them to form the matrix \( U \) with size \( n \times y \).

**SVD-Based Collaborative Filtering Algorithm**

Scalability is one of the disadvantages of correlation-based collaborative filtering algorithms. As the number of users and items increases, so too does the computation cost. Alternative methods are tried for recommendation purposes. The SVD-based collaborative filtering algorithm is the best choice for large and sparse databases.
Sarwar, Karypis, Konstan, and Riedl propose an SVD-based algorithm for collaborative filtering [31]. The sparse user-item ratings matrix \( A \) is filled using the average ratings for users to capture a meaningful latent relationship. The filled matrix is normalized by converting ratings to z-scores. The normalized matrix \( A_{\text{norm}} \) is factored into \( U \), \( S \), and \( V \) using SVD. Then the matrix \( S_k \) is obtained by retaining only \( k \) largest singular values. Accordingly, the dimensions of matrices \( U \) and \( V \) are also reduced. Then \( U_k \sqrt{S_k} \) and \( S_k V_k^T \) are computed. These matrices can be used to compute the prediction for any user \( u \) and item \( q \). To compute a prediction for a user \( u \) for item \( q \), the scalar product of the \( u \)th row of \( U_k \sqrt{S_k} \) [denoted as \( U_k \sqrt{S_k} (u) \)] and the \( q \)th column of \( \sqrt{S_k V_k^T} \) [denoted as \( \sqrt{S_k V_k^T} (q) \)] is calculated, and the result is denormalized as follows:

\[
p_{uq} = \bar{v}_u + \sigma_u \left( U_k \sqrt{S_k} (u) \cdot \sqrt{S_k V_k^T} (q) \right),
\]

where \( \bar{v}_u \) and \( \sigma_u \) are the mean rating and the standard deviation for user \( u \), respectively. Since the user \( u \) that is looking for prediction will do the denormalization, we can define \( p_{uq} = \bar{v}_u + \sigma_u p \), where \( p \) is defined as follows:

\[
p = U_k \sqrt{S_k} (u) \cdot \sqrt{S_k V_k^T} (q).
\]

**Privacy-Preserving Collaborative Filtering Using RP Techniques**

**Randomized Perturbation**

Randomized perturbation techniques can be applied to computations based on aggregate information. Such techniques make it possible to produce meaningful outcomes if one is interested in aggregate data. Scalar product and sum are aggregate-based computations and are used in collaborative filtering algorithms.

**Scalar Product**

The proposed scheme for privacy preserves scalar product using randomized perturbation techniques. The scheme does not use an interactive protocol. That is, the scalar product is directly computed from the disguised data without any involvement of the data owner. On the flip side, the scheme does not achieve 100 percent accuracy because of the underlying randomized perturbation technique.

Let \( A \) and \( B \) be the original vectors and given, where \( A = (a_1, \ldots, a_n) \) and \( B = (b_1, \ldots, b_n) \). \( A \) is disguised by \( R = (r_1, \ldots, r_n) \), and \( B \) is disguised by \( V = (v_1, \ldots, v_n) \), where \( r_i \)'s and \( v_i \)'s are random values drawn from some distribution with the mean \( \mu \) being 0. Let \( A' = A + R \) and \( B' = B + V \) be the disguised data that are known; the scalar product of \( A \) and \( B \) can now be estimated from \( A' \) and \( B' \)
As seen from the equation, the last three summations are the effects of the random values. The expected value of $A' \cdot B'$ can be written as follows:

$$E(A' \cdot B') = E[(A + R) \cdot (B + V)] = E(A \cdot B) + E(A \cdot V) + E(B \cdot R) + E(R \cdot V).$$

Because $R$ and $V$ are independent and random values ($r_i$'s and $v_i$'s) drawn from some distribution with mean $\mu$ being 0, we have $E(A \cdot B) = A \cdot E(V) = A \cdot 0 = 0$; similarly, $E(B \cdot R) = 0$; and $E(R \cdot V) = E(R) \cdot E(V) = 0 \cdot 0 = 0$. Therefore, $E(A' \cdot B') = E(A \cdot B)$. Finally, one can approximately write that $A' \cdot B' = A \cdot B$, which is $A' \cdot B' = \sum_{i=1}^{n} a_i b_i$.

Sum

Let $A$ be the original vector with $n$ values and given, where $A = (a_1, \ldots, a_n)$. $A$ is disguised by $R = (r_1, \ldots, r_n)$, where $r_i$'s are random values drawn from some distribution with mean $\mu$ being 0. Let $A' = A + R$ be the disguised data that are known. It can now be shown how the sum of $A$ can be estimated from $A'$:

$$\sum_{i=1}^{n} a'_i = \sum_{i=1}^{n} (a_i + r_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} r_i.$$

The expected value of $A'$ can be written as follows:

$$E(A') = E(A + R) = E(A) + E(R).$$

Since random values ($r_i$'s) are drawn from some distribution with the mean $\mu$ being 0, $E(R) = 0$. Therefore, $E(A') = E(A)$. Finally, one can approximately write that $A' \approx A$, which is $\sum_{i=1}^{n} a'_i \approx \sum_{i=1}^{n} a_i$.

**PPCF Using Randomized Perturbation**

The goal of collaborative filtering using randomized perturbation is to achieve privacy and produce recommendations with high accuracy. Users may send false data instead of their actual data to achieve perfect privacy. But producing accurate recommendations is impossible from false data. On the other hand, if users send their actual data to the server, finding high-quality recom-
mendations is possible, but user privacy is not preserved. The proposed technique achieves a good balance between privacy and accuracy.

Without privacy concerns, users send their private data to the server, which creates a central database containing ratings from all users. Once the database for collaborative filtering purposes is created, the server can provide recommendations to users based on their queries (i.e., on the items about which they are looking for predictions) using Equations 1 and 2. However, with privacy concerns, users perturb their private data before they send it to the server. The proposed technique achieves data disguise by using randomized perturbation techniques. Each user easily disguises data without the help of a third party. The server should not be able to find out the true values of the ratings for items the users have rated before and about which they have voted because of the noise data. The scheme consists of three main steps: data disguising and collection, estimation using the disguised data, and provision of the predictions to active users. The details of these steps are as follows.

**Data Disguising and Collection**

Users fill the missing ratings with their mean votes, normalize the votes by converting them to $z$-scores, add a random value drawn from some distribution to each of the $z$-score values, and send the disguised data to the server. The server is unable to find out the true $z$-scores and the items users rated because of the noise data. Because the users provide disguised $z$-scores to the server without letting it know the standard deviation of the ratings and the mean vote, estimating the original ratings from disguised $z$-scores would be difficult. The data disguising is summarized below:

1. The server decides on distributions of perturbing data (uniform or Gaussian), parameters ($\sigma$ and $\mu$), and methods to select the parameters, and lets each user know.
2. Each user $u$ fills the empty cells in the ratings vector using the mean vote and calculates the $z$-scores.
3. Each user $u$ creates $m$ random values $r_{uj}$ drawn from some distribution, where $m$ is the total number of items. The user $u$ adds the random values to its $z$-score values and generates the disguised $z$-scores $z'_{uj} = z_{uj} + r_{uj}$, $j = 1, \ldots, m$. Finally, each user sends $z'_{uj}$ values to the server, which creates the disguised user–item matrix ($A'$).

**Estimation Using Perturbed Data**

*Correlation-Based Collaborative Filtering with Privacy.* To show how the approach works for the correlation-based algorithm, the $z$-score notation ($z_{uj}$) is used to simplify Equations 1 and 2. Therefore, we get the following from Equation 2:

$$w_{au} = \sum_d z_{ad} z_{ud}. \quad (9)$$
When Equation 1 is simplified using the $z$-score notation, we get the following:

$$p_{aq} = \overline{v}_a + \sigma_a \sum_{u=1}^{n} w_{au} z_{aq} = \overline{v}_a + \sigma_a p'_{aq}$$

(10)

where $p'_{aq}$ is defined as follows:

$$p'_{aq} = \frac{\sum_{u=1}^{n} \sum_{d} z_{ad} \overline{z}_{ud}}{\sum_{u=1}^{n} \sum_{d} z_{ad} z_{ud}} = \frac{\sum_{d} \sum_{u=1}^{n} z_{ad} z_{aq} - \sum_{d} \sum_{u=1}^{n} z_{ad} \overline{z}_{aq}}{\sum_{d} \sum_{u=1}^{n} z_{ad} z_{ud}}$$

(11)

Since all users have filled their missing ratings with their mean rating, the counter, $d$, is same for all users. The numerator part consists of a scalar product between vector $Z_d = (z_{1d}, \ldots, z_{nd})$ and vector $Z_q = (z_{1q}, \ldots, z_{nq})$. If the server can compute the scalar products for all $d$'s, it can send the results of the scalar products to the active user, who can easily compute the numerator part. The denominator part is even simpler. All the server needs to do is to send the result of $\sum_{u=1}^{n} z_{ud}$ for each $d$ to the active user.

After it gets all the disguised $z$-scores $z'_{ud}$ from many users, the server can provide collaborative filtering services to active users based on the following facts derived from privacy preserving scalar product and sum:

$$\sum_{u=1}^{n} z_{ud} z_{aq} = \sum_{u=1}^{n} z_{ud} z'_{aq}$$

and

$$\sum_{u=1}^{n} z'_{ud} \approx \sum_{u=1}^{n} z_{ud}.$$

**SVD-Based Collaborative Filtering with Privacy.** To provide recommendation services to users using the SVD-based algorithm, the server first computes the SVD of matrix $A'$. As explained before, once the server computes $A'^T A'$, it can find the $S'$ and $V'$ matrices based on $A'^T A'$. Each entry of $A'^T A'$ is computed by calculating the scalar product of the rows of matrix $A'^T$ and the columns of matrix $A'$ using the facts of the privacy-preserving scalar product and sum.

The entries other than the diagonal ones are computed as follows:
\[
(A^T A)'_{jj} = \sum_{u=1}^{n} (z_{uf} + r_{uf})(z_{ug} + r_{ug}) = \sum_{u=1}^{n} z_{uf}^2 + \sum_{u=1}^{n} z_{uf} r_{ug}
+ \sum_{u=1}^{n} z_{ug} r_{uf} + \sum_{u=1}^{n} r_{uf} r_{ug} \approx \sum_{u=1}^{n} z_{uf} z_{ug},
\]

where \(n\) is the total number of users, \(f\) and \(g\) show the row and column numbers, respectively, and \(f \neq g\). Since random values \(r_{uf}'s\) and \(r_{ug}'s\) are independent and drawn from some distribution with \(\mu = 0\), the expected value of \(\sum_{u=1}^{n} r_{uf} r_{ug}\) is 0. Similarly, the expected values of \(\sum_{u=1}^{n} z_{uf} r_{ug}\) and \(\sum_{u=1}^{n} z_{ug} r_{uf}\) are 0.

However, since the scalar product is computed between the same vectors for the diagonal entries \((f = g)\), they can be estimated as follows:

\[
(A^T A)'_{ff} = \sum_{u=1}^{n} (z_{uf} + r_{uf})(z_{uf} + r_{uf}) = \sum_{u=1}^{n} z_{uf}^2 + 2 \sum_{u=1}^{n} z_{uf} r_{uf}
+ \sum_{u=1}^{n} r_{uf}^2 \approx \sum_{u=1}^{n} z_{uf}^2 + \sum_{u=1}^{n} r_{uf}^2.
\]

Again, the expected value of \(\sum_{u=1}^{n} z_{uf} r_{uf}\) is 0. However, as \(\sum_{u=1}^{n} z_{uf}^2\) values are only needed for diagonal entries, it is necessary to get rid of \(\sum_{u=1}^{n} r_{uf}^2\) in Equation 13 as follows:

\[
(A^T A)'_{ff} \approx \sum_{u=1}^{n} z_{uf}^2 + \sum_{u=1}^{n} r_{uf}^2 - n\sigma_r^2 = \sum_{u=1}^{n} z_{uf}^2,
\]

where \(\sigma_r\) is the standard deviation of random numbers. After estimating the matrix \(A^T A'\), the server can find the rank and the eigenvalues, which are used to find eigenvectors that form the matrix \(V'\). It then finds matrix \(S'\) using the eigenvalues estimated from \(A^T A'\).

Finally, the server needs to calculate the first \(y\) column-vectors of \(U\) using \(b_i = s_i^{-1} A v_i\) for \(i = 1, \ldots, y\), where \(v_i's\) are column-vectors of \(V\). Similarly, \(b_i\) vectors can be estimated using \(A', s'_i\), and \(v_i\) vectors, where \(v_i''s\) and \(s_i''s\) are estimated from the matrix \(A^T A'\). The entries of \(b_i\) vectors are estimated as follows:

\[
b_i(j) = s_i^{-1} \sum_{l=1}^{m} (z_{jl} + r_{jl}) v_{il}' = s_i^{-1} \sum_{l=1}^{m} z_{jl} v_{il}' + s_i^{-1} \sum_{l=1}^{m} r_{jl} v_{il}' = s_i^{-1} \sum_{l=1}^{m} z_{jl} v_{il}',
\]

where \(j = 1, \ldots, n\) and the expected value of \(\sum_{j=1}^{m} r_{jl} v_{il}'\) is 0.

**Providing Predictions**

**Correlation-Based Collaborative Filtering with Privacy.** To get a prediction for item \(q\), the active user computes the z-scores \(z_{ad}\) for the items already rated. Then the server sends the results of \(\sum_{u=1}^{n} z_{uf} z_{ud}\) and \(\sum_{u=1}^{n} z_{uf} r_{ud}\) for all \(d's\) to the active user, who uses Equation 11 and Equation 10 to compute \(p'_{aq}\) and \(p_{aq}\), the predicted rating for the user on item \(q\). Even if we let the privacy-protected user
be the active user, the server does not know the active user’s mean vote and standard deviation and which items the active user rated before. In consequence, it can only learn the estimated value of $p'_{aq}$. The active user is the one who is going to estimate $p_{aq}$ using the estimated data received from the server.

**SVD-Based Collaborative Filtering with Privacy.** After estimating $U'$, $S'$, and $V^{T}$ from disguised data, the server finds $S'_k$ by retaining only $k$ largest singular values and computes $U'_k \sqrt{S'_k}$ and $\sqrt{S'_k} V'_k^{T}$ matrices. To get a prediction for item $q$, the user $u$ sends a query (about the item for which a prediction is sought) to the server. The server computes the $p'$ by calculating the scalar product of the $u$th row of $U'_k \sqrt{S'_k}$ and the $q$th column of $\sqrt{S'_k} V'_k^{T}$ and sends it to the user $u$, who can now calculate the $p'_{aq}$. The server will only be able to know the estimated value of $p'$ because it does not have the mean rating and the standard deviation of user $u$ who is looking for a prediction. Therefore, it will not be able to learn how much user $u$ likes or dislikes item $q$.

**Query Disguising**

Users looking for predictions may have concerns about revealing their queries to the server. A naive solution to the problem of preventing it from learning what item the active user is looking for a prediction about would be to send $N$ queries, one of which is a true query and the remainder are randomly selected queries. The probability of the server’s guessing the true query is 1 out of $N$. However, because this solution makes it possible for the active users to get $N$ predictions instead of one, it is not feasible from the server’s point of view.

Instead, the use of a 1-out-of-$N$ oblivious transfer protocol is proposed [6, 13]. At the beginning of this protocol, one party, Bob, has $N$ inputs $X_1, \ldots, X_N$ and at the end of the protocol the other party, Alice, learns one of the inputs, $X_i$ for some $1 \leq i \leq N$ of her choice, without learning anything about the other inputs and without allowing Bob to learn anything about $i$. Naor and Pinkas proposed an efficient 1-out-of-$N$ oblivious transfer protocol [25]. Combining this protocol with the scheme proposed by Cachin, Micali, and Stadler [8] would achieve a 1-out-of-$N$ oblivious transfer protocol with polylogarithmic (in $n$) communication complexity.

When active users use 1-out-of-$N$ oblivious transfer protocol to hide their queries, the server needs to compute $N$ scalar products instead of one for an SVD-based scheme. Computation for the denominator will be the same for a correlation-based scheme, but it will be $N$ times more for the numerator. The computation performance can be improved by doing some computations off-line, which is not critical to the performance. In the case of an SVD-based scheme, decomposition and computing the reduced matrices of $U'_k \sqrt{S'_k}$ and $\sqrt{S'_k} V'_k^{T}$ can be done off-line. In the case of a correlation-based scheme, computing the required data for the denominator can be done off-line. The server can compute the data required for the numerator for all items off-line and store them. Based on the query, it sends the corresponding data to the active user.
Experiments

Datasets

The experiments use MovieLens (ML) and Jester datasets to evaluate the overall performance of the randomized perturbation-based collaborative filtering scheme. Jester is a Web-based joke recommendation system developed at the University of California, Berkeley [18]. The database has 100 jokes and records of 17,988 users. The ratings range from −10 to +10, and the scale is continuous. MovieLens data were collected by the GroupLens Research Project at the University of Minnesota (www.cs.umn.edu/research/GroupLens). There are two datasets available. The first, MovieLens Public Data (MLP), consists of 100,000 ratings for 1,682 movies by 943 users. The second, MovieLens Million Data (MLM), consists of approximately 1 million ratings for approximately 3,500 movies made by 7,463 users. Each user has rated at least 20 movies. Ratings are made on a five-star scale.

Evaluation Criteria

Several evaluation criteria for collaborative filtering have been used in the literature [10, 19, 33]. The most common criteria are the mean absolute error (MAE) and the standard deviation ($\sigma$). These two criteria are also used to evaluate the accuracy of the proposed schemes. The lower the MAE, the more accurate the scheme is. The standard deviation of the errors should also be minimized. The lower the deviation, the more consistently accurate the scheme is.

If $p_1, p_2, \ldots, p_d$ are predicted values from undisguised data, and $p'_1, p'_2, \ldots, p'_d$ are predicted values from disguised data, then $E = |\xi_1, \xi_2, \ldots, \xi_d| = |p'_1 - p_1, p'_2 - p_2, \ldots, p'_d - p_d|$ represents errors. Therefore, the MAE and the standard deviation of the errors are computed using the following equations:

$$\text{MAE} = \frac{1}{d} \sum_{i=1}^{d} |\xi_i|, \quad \sigma = \frac{1}{d-1} \sum_{i=1}^{d} (\xi_i - E)^2.$$  \hspace{1cm} (16)

Data-Disguising Methods

Uniform Distribution

All users create uniform random values from a range $[-\alpha, \alpha]$, where $\alpha$ is a constant number and $\alpha = \sqrt{3}\sigma$.

Gaussian Distribution

Each user generates random values using normal distribution based on the mean ($\mu$) being 0 and the standard deviation ($\sigma$).
Two different ways are used to select the standard deviation of perturbing data ($\sigma$). The fixed scheme uses a fixed standard deviation to generate random numbers. After the $\sigma$ is decided, random numbers based on it are generated. However, in the random scheme, after the standard deviation of the perturbing data is decided, each user randomly generates a number within the range $[0, \sigma]$ and then uses that value as a standard deviation to generate random numbers.

**Privacy Measure**

The privacy measure should indicate how closely the original value of an item could be estimated from the perturbed data. Different privacy measures are used to quantify privacy [3, 4, 15, 30]. Evfimievski et al. propose a privacy measure when the private data consist of categorical items [15]. Rizvi and Haritsa use a method to quantify the privacy for binary data [30]. Agrawal and Srikant use a measure based on how closely the original values of a modified attribute can be estimated [4]. If it can be estimated with $c$ percent confidence that a value $x$ lies in the interval $[x_1, x_2]$, then the interval width ($x_2 - x_1$) defines the amount of privacy at $c$ percent confidence level. For example, if the random numbers are generated with a uniform distribution from $[-\alpha, \alpha]$, then to achieve 50 percent confidence, the interval width is $\alpha$; to achieve 95 percent confidence, the interval width is $1.9\alpha$. When the random numbers are generated using a Gaussian distribution with the mean ($\mu$) being 0 and the standard deviation $\sigma$, to achieve 95 percent confidence, the interval width is $3.92\sigma$. However, the method does not take into account the distribution of original data. To accurately quantify privacy, the present research used the method suggested by Agrawal and Aggarwal [3], which takes into account the distribution of original data.

Agrawal and Aggarwal propose a privacy measure based on the differential entropy ($h$) of a random variable ($X$) [3]. They propose $2^h(X)$ as a measure of privacy inherent in the random variable $X$ and denote it by $\Pi(X)$. For a general random variable $X$, $\Pi(X)$ denotes the length of the interval, over which a uniformly distributed random variable has the same uncertainty as $X$. They define the average conditional privacy of $X$ given $Z$, which takes into account the additional information available in the perturbed values as $\Pi(X|Z) = 2^h(X/Z)$, where $h(X|Z)$ is the conditional differential entropy of $X$ given $Z$. This motivates the metric $P(X|Z) = 1 - 2^{-h(X;Z)}$, which is the fraction of privacy of $X$ lost by revealing $Z$, where $I(X;Z) = h(Z) - h(Z|X)$. If the original value is $X$, which is disguised by $R$, after revealing $Z$ where $Z = X + R$, $X$ has privacy $\Pi(X|Z) = \Pi(X)[1 - P(X|Z)]$.

**Methodology**

**Selecting the Training Data**

Users and items are randomly selected for training from real datasets. The number of users and the number of items for training data may differ for
different sets of experiments because the number of users and items are varied based on the parameters to be compared. In one set of experiments, the number of users is fixed and the number of items is varied, and the opposite is done in another set of experiments in order to show how different total numbers of users and items affect the overall performance of the proposed scheme.

Selecting the Test Data

After the training data are selected, the test data are randomly selected from real datasets. The numbers of users and items in the test data vary based on different sets of experiments. Both the training data and the testing data are sometimes restricted according to the requirements of the experiments. In other words, the data used in the experiments are sometimes preprocessed.

Prediction for Active Users

A database is created using training users’ data, and the test users’ data are applied to find predictions for test items using both the original algorithm based on original data and the proposed scheme based on disguised data. Predictions are found for withheld rated items. Two different protocols are used, as suggested by Breese, Heckerman, and Kadie [7].

1. In the first protocol, a single randomly selected rated item is withheld for each user in the test set, and an effort is made to predict its value given all the other votes the user rated. This scheme is called the All but 1 protocol.
2. In the second protocol, five rated items are randomly selected from each test user as test items, and an attempt is made to predict for those items. This is called the All but 5 protocol.

Finding Errors

The overall performance of the scheme is evaluated by comparing the predictions found based on disguised data using the randomized-perturbation-based scheme with both the actual ratings for withheld items and the predictions found on original data using the original algorithms. The experiments are run for each active user selected randomly from the test set. Each time, items are selected from the test set, and predictions are found for them based on both the original and the disguised data. The data-disguising scheme is run 100 times, and predictions are found on the disguised data. The predictions for test items for that active user are then averaged. The final results for one active user are then averaged over all items. Finally, the average value is found over all active users selected randomly from the test users, and that final value is displayed.
Experimental Results

It is hypothesized that the privacy and accuracy of the privacy-preserving collaborative filtering scheme depend on several factors, including the total numbers of users \(n\) and items \(m\), the distribution and range of the perturbing data, and the methods of selecting the standard deviation. The overall performance of the SVD-based scheme also depends on the total number of retained singular values \(k\). The experiments for the SVD-based scheme used the Jester and MLP datasets. The experiments for the correlation-based scheme used the Jester and MLM datasets. For withheld ratings, the All but 5 and All but 1 protocols were used for the SVD- and correlation-based schemes, respectively. The numbers of users and items were fixed in the experiments to determine the best value of \(k\), but the values of \(k\) were varied. Since the optimum value of \(k\) was 10 for both the Jester and MLP datasets, \(k\) was set to be 10 in the experiments based on the SVD-based scheme.

Total Number of Users

Three different sets of experiments were conducted to show that the schemes work better as the number of users \(n\) increases. Perturbing data were created using uniform and Gaussian distributions with \(\sigma = 1\). MAEs were shown for undisguised data, uniform, and Gaussian perturbed data. The number of items for the SVD-based scheme was fixed at 1,682 and 100, respectively, for the MLP and Jester datasets, \(k = 10\), and \(n\) varied. Figure 2 shows the results for the Jester dataset. In the first set, \(n = 100\); it was increased to 200 and 1,000 in the second and third sets of experiments, respectively.

Figure 2. Number of Users vs. MAEs (Jester Dataset)
All the items were used for the correlation-based scheme, varying the total number of users. Figure 3 shows the results for the MLM dataset. In the first set, \( n = 50 \); it was increased to 100 and 900 in the second and third sets of experiments, respectively.

As expected, accuracy improves with increasing \( n \) because the scalar product and sum are computed over \( n \). The privacy-preserving scalar product and sum give better results as the amount of data increases. In the long run, the sample mean and variance of the perturbing data will converge to their expected values. Therefore, the results get better as \( n \) increases.

When privacy is not a concern, accuracy improves with increasing \( n \), and after a certain point accuracy becomes worse [32]. When privacy is a concern, the accuracy of the schemes improves with increasing \( n \) when one compares the results based on undisguised data using the original algorithm with the results based on disguised data using the schemes. For example, the difference between the MAEs for undisguised data and disguised data using uniform perturbing data is 0.1602 when \( n = 50 \), whereas it is 0.0536 when \( n = 900 \), as seen from Figure 3.

Figure 3. Number of Users vs. MAEs (MLM Dataset)

Total Number of Items

Experiments were conducted to show how different total numbers of items affect the overall performance of the scheme. Only the ML datasets were used, because the number of items (100) in Jester is too limited for such experiments. Perturbing data were created using uniform and Gaussian distributions with \( \sigma = 1 \), and the experiments used a fixed \( n \) and varying \( m \). Experiments were run based on both schemes, but only the results for the SVD-based scheme are
Table 1. Number of Items vs. Prediction Quality (MLP Dataset).

<table>
<thead>
<tr>
<th>Data disguising</th>
<th>943 × 1,682</th>
<th>943 × 500</th>
<th>779 × 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAE Uniform</td>
<td>0.7229</td>
<td>0.7815</td>
<td>0.8354</td>
</tr>
<tr>
<td>Gaussian</td>
<td>0.7104</td>
<td>0.7706</td>
<td>0.8015</td>
</tr>
<tr>
<td>Undisguised</td>
<td>0.6952</td>
<td>0.7359</td>
<td>0.7422</td>
</tr>
<tr>
<td>( \sigma ) Uniform</td>
<td>0.6069</td>
<td>0.5919</td>
<td>0.6907</td>
</tr>
<tr>
<td>Gaussian</td>
<td>0.6172</td>
<td>0.6032</td>
<td>0.6683</td>
</tr>
<tr>
<td>Undisguised</td>
<td>0.5947</td>
<td>0.5683</td>
<td>0.5722</td>
</tr>
</tbody>
</table>

shown. First, all the available items were used, and then 500 and 100 items were randomly selected. When 100 or 500 items were randomly selected, the experiments used the users who had rated at least two of those items. As a result, there are 779 users in the third group of experiments where \( m = 100 \). Table 1 shows the mean absolute errors and standard deviations of errors for both undisguised and perturbed data.

As can be seen, accuracy improves with increasing \( m \) because the scalar product and sum based on disguised data give better results as the amount of available data increases. As explained before, with increasing \( m \), the sample mean and variance of perturbing data will converge to their expected values. Therefore, accuracy improves with increasing \( m \).

**Level of Perturbation**

Experiments using both datasets were conducted with the parameters of the perturbing data varied to show how different levels of perturbation affect accuracy. Random numbers were created using uniform and Gaussian distributions while varying the standard deviations. The predictions were compared based on disguised data using the scheme with the predictions on original data. The experiments were conducted using both datasets for both schemes, but results are shown only for the Jester and MLM datasets for the SVD- and correlation-based schemes, respectively. The MLM and Jester datasets used \( 1,000 \times 3,500 \) and \( 1,000 \times 100 \) user–item matrices, respectively. Figures 4 and 5 show how the mean absolute errors change with increasing levels of perturbation for the SVD- and correlation-based schemes, respectively.

As seen from the figures, the level of perturbation is critical for accuracy. The results improve with decreasing levels of perturbation. Because the randomness becomes smaller when the standard deviation is small, accuracy can be improved.

**All but 5 vs. All but 1**

To show how different total numbers of available ratings affect the results, experiments using the All but 1 and All but 5 protocols were conducted using
Figure 4. Level of Perturbation vs. MAEs (Jester Dataset)

Figure 5. Level of Perturbation vs. MAEs (MLM Dataset)
the SVD-based scheme. Table 2 shows the mean absolute errors and the standard deviations of the errors for the MLP dataset. The experiments used a $943 \times 1,682$ user–item matrix, and uniform and Gaussian random values were created using $\sigma = 1$.

As can be seen from the table, the results based on the All but 1 protocol are slightly better than the results based on the All but 5 protocol. Accuracy improves as the total number of available ratings increases, because randomized perturbation techniques give better results with a great number of available data.

<table>
<thead>
<tr>
<th>Data disguising</th>
<th>All but 5</th>
<th>All but 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAE Uniform</td>
<td>0.7229</td>
<td>0.7085</td>
</tr>
<tr>
<td>Gaussian</td>
<td>0.7104</td>
<td>0.6967</td>
</tr>
<tr>
<td>Undisguised</td>
<td>0.6952</td>
<td>0.6834</td>
</tr>
<tr>
<td>$\sigma$ Uniform</td>
<td>0.6069</td>
<td>0.5795</td>
</tr>
<tr>
<td>Gaussian</td>
<td>0.6172</td>
<td>0.6006</td>
</tr>
<tr>
<td>Undisguised</td>
<td>0.5947</td>
<td>0.5728</td>
</tr>
</tbody>
</table>

**Table 2. All but 1 vs. All but 5 (MLP Dataset).**

Privacy and Accuracy

The level of perturbation plays a critical role in protecting the private data. If the level is too low, the perturbed data will disclose significant amounts of information, and if it is too high, accuracy will be very low. The greater the level of perturbation, the greater the amount of privacy. Table 3 shows the privacy levels with increasing levels of perturbation for both uniform and Gaussian perturbed data.

The changes in privacy loss $P(X|Z)$ with increasing level of perturbation are shown in Figure 6. The $P(X|Z)$ values were computed while the standard deviation of perturbing data was changed using the idea of privacy measure defined earlier.

As can be seen from Figure 6, the greater the level of perturbation, the smaller the privacy loss. It was hypothesized that accuracy decreases with increasing level of perturbation. This was shown by combining Figure 4, which shows the MAEs for the Jester dataset based on the SVD-based scheme, with Figure 6 (privacy loss vs. level of perturbation); the trade-off between privacy loss and accuracy can be seen in Figure 7. Although privacy levels increase with the level of perturbation, accuracy becomes worse because accuracy and privacy conflict.

The proposed schemes make it possible to find a numerical relation between MAE and perturbation level. A relation can easily be drawn between accuracy and privacy using Figure 7, which showed the mean absolute errors with different privacy losses. Users will be able to know what maximum privacy (maximum perturbation) can be kept if accuracy needs to be kept to a certain level.
Selecting the Parameters of Perturbing Data

It is quite certain that accuracy can be improved. This can be achieved by using different methods to select the parameters of perturbing data, including the fixed and random selection methods, as explained before. Experiments were conducted based on the correlation-based scheme using the MLM dataset, where \( n = 900 \), to show how different methods of parameter selection affect overall performance. Because the uniform and Gaussian distributed perturbing data perform similarly, the experiment used only uniform perturbing data. The standard deviation of the perturbing data was varied from 0.25 to 3.00; the mean absolute errors and standard deviations of errors based on fixed and random parameter selection methods are shown in Table 4.

The results show that the random parameter selection scheme yields better performance than the fixed scheme. This is so because the randomness becomes smaller when the parameters are selected randomly, and thus the accuracy improves.

<table>
<thead>
<tr>
<th>Perturbed data</th>
<th>Standard deviation of perturbing data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.25</td>
</tr>
<tr>
<td>II[X,Z]</td>
<td></td>
</tr>
<tr>
<td>Uniform</td>
<td>0.8401</td>
</tr>
<tr>
<td>Gaussian</td>
<td>1.0021</td>
</tr>
</tbody>
</table>

Table 3. Privacy Levels vs. Levels of Perturbation.

Figure 6. Privacy Loss vs. Level of Perturbation

Selecting the Parameters of Perturbing Data
The goal is to achieve user privacy while providing accurate predictions. The results were evaluated by comparing predictions based on disguised data using the proposed scheme with both the original ratings and predictions based on the original data using the original algorithms. As can be seen from Figure 3, which shows the results for the MLM dataset based on the correlation-based scheme, the MAE is 0.9259 when the data disguising uses uniform perturbing data with $\sigma = 1$ and $n = 100$. However, it is 0.0835 when the predictions based on disguised data are compared with the predictions based on original data, where the MAE is 0.8424. Because the rating range for the MLM dataset is from 1 to 5, MAE = 0.0835 indicates that the results are very close to the results generated from the original data. In Figure 5, the MAEs are less than 0.23 for the MLM dataset for the correlation-based scheme, and in Figure 4 they are less than 1.95 for the Jester dataset for the SVD-based scheme when the standard deviation of perturbing data is 3 or less. However, because the ratings

<table>
<thead>
<tr>
<th>Standard deviation of perturbing data</th>
<th>0.25</th>
<th>0.50</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MAE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fixed scheme</td>
<td>0.8166</td>
<td>0.8341</td>
<td>0.8587</td>
<td>0.9251</td>
<td>1.0295</td>
</tr>
<tr>
<td>Random scheme</td>
<td>0.8105</td>
<td>0.8137</td>
<td>0.8371</td>
<td>0.8703</td>
<td>0.8945</td>
</tr>
<tr>
<td><strong>$\sigma$</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fixed scheme</td>
<td>0.4934</td>
<td>0.4966</td>
<td>0.5565</td>
<td>0.6531</td>
<td>0.7683</td>
</tr>
<tr>
<td>Random scheme</td>
<td>0.5086</td>
<td>0.5113</td>
<td>0.5301</td>
<td>0.5419</td>
<td>0.5868</td>
</tr>
</tbody>
</table>

**Table 4. Fixed Scheme vs. Random Scheme (MLM Dataset).**

**Summary**

The goal is to achieve user privacy while providing accurate predictions. The results were evaluated by comparing predictions based on disguised data using the proposed scheme with both the original ratings and predictions based on the original data using the original algorithms. As can be seen from Figure 3, which shows the results for the MLM dataset based on the correlation-based scheme, the MAE is 0.9259 when the data disguising uses uniform perturbing data with $\sigma = 1$ and $n = 100$. However, it is 0.0835 when the predictions based on disguised data are compared with the predictions based on original data, where the MAE is 0.8424. Because the rating range for the MLM dataset is from 1 to 5, MAE = 0.0835 indicates that the results are very close to the results generated from the original data. In Figure 5, the MAEs are less than 0.23 for the MLM dataset for the correlation-based scheme, and in Figure 4 they are less than 1.95 for the Jester dataset for the SVD-based scheme when the standard deviation of perturbing data is 3 or less. However, because the ratings
range from –10 to 10 in the Jester dataset, an error of 1.95 is equivalent to 0.39 in a 1 to 5 scale. Therefore, the randomized perturbation-based scheme provides predictions with decent accuracy.

Conclusions and Future Work

Collaborative filtering techniques are very popular on the Internet. Many e-commerce sites use such techniques to provide predictions using other people’s preference data. Because users with privacy concerns may decide to provide false data rather than true data, collecting truthful data is a challenge. It will be easier to collect truthful data for CF purposes when privacy measures are in place.

The discussion in this paper has presented solutions to both correlation- and SVD-based CF with privacy. The solutions make it possible for servers to collect private data without greatly compromising user privacy. The proposed scheme prevents the server from learning which items users have rated and how much users like or dislike them. Because privacy and accuracy are conflicting goals, it is imperative to find a good balance between them. The proposed scheme achieves this balance. The experiments show that the proposed solutions can achieve accurate predictions without exposing user privacy. An analysis based on the experimental results evaluated how different factors affect the overall performance of the scheme.

Accuracy can be further improved if more information is disclosed along with the disguised data, especially aggregate information whose disclosure does not compromise user privacy to a great extent. The authors intend to study how data disclosures of these kinds affect overall performance and privacy.

REFERENCES

HUSEYIN POLAT (hpolat@ecs.syr.edu) is a Ph.D. candidate in the Department of Electrical Engineering and Computer Science at Syracuse University. He received his master’s degree from the Computer and Information Science Department at Syracuse University. His research interests primarily concern collaborative filtering with privacy and privacy-preserving data mining.

WENLIANG DU (wedu@ecs.syr.edu) is an assistant professor in the Department of Electrical Engineering and Computer Science at Syracuse University. He has a bachelor’s degree in computer science from the University of Science and Technology of China and a Ph.D. in computer science from Purdue University. His primary research interests are privacy-preserving data mining, wireless network security, and computer security.