Deriving Private Information from Randomly Perturbed Ratings

Sheng Zhang, James Ford, Fillia Makedon
{clap, jford, makedon}@cs.dartmouth.edu
Department of Computer Science
Dartmouth College, Hanover, NH 03755

Abstract

Collaborative filtering techniques have become popular in the past several years as an effective way to help people deal with information overload. An important security concern in traditional recommendation systems is that users disclose information that may compromise their individual privacy when providing ratings. Randomized perturbation schemes have been proposed to disguise user ratings while still producing accurate recommendations. However, recent research has suggested that perturbation schemes might not be able to preserve privacy as much as has been believed. We propose two data reconstruction methods that derive original private information from disguised data in existing perturbation collaborative filtering schemes. One method is based on k-means clustering and the other uses singular value decomposition (SVD). We have conducted theoretical and experimental analysis on the difference between original data and reconstructed data. Our experiments show that both methods can derive a considerable amount of original information. This study helps to determine an empirical trade-off between recommendation accuracy and user privacy in perturbation schemes.

Keywords: collaborative filtering, privacy, randomized perturbation, data reconstruction.

1 Introduction

Collaborative Filtering (CF) is a way to serve up products or services to a particular user based on what other users with similar tastes have preferred. People uses CF systems to cope with information overload by reducing the number of alternatives they need to consider. However, traditional CF systems are a serious threat to individual privacy because data collected from customers covers personal information about places and things they do, watch, and purchase [7, 8, 17]. Customer data is valuable, and some companies have sold theirs upon suffering bankruptcy. While some people might be willing to selectively provide personal information if they can benefit in return, as a recent survey [9] indicates, a significant number of people are not willing to do that. Therefore, a research challenge for CF systems is to provide accurate recommendations without compromising customers' privacy.

To the best of our knowledge, there are three methods of preserving user privacy that have been used in recommendation systems. The first is to use anonymization techniques that allow users to provide personal information while keeping their identities private [1, 19]. The main problem with this approach is that the quality of the collected data cannot be guaranteed because the identities of data contributors cannot be verified. The second approach is to design secure multi-party computation protocols for CF algorithms to ensure that privacy is not compromised during communications between the server and users [7, 8]. As cryptographic operations are frequently used in this approach, computational cost is a concern. The last method uses randomized perturbation techniques to disguise user data while still allowing the server to estimate the aggregate information with reasonable accuracy [17, 18].

Recently, Kargupta *et al.* [16] pointed out that randomization techniques might not preserve privacy as much as had been believed. They proposed a random matrix-based spectral filtering technique to recover the original data from the perturbed data, and their results showed that recovered data can be close to the original data. Motivated by this work, Huang *et al.* [15] proposed using two data reconstruction methods (principal component analysis (PCA) and Bayes estimation) that are based on data correlations in the original data. Their experiments showed that the original data can be reconstructed more accurately when correlations are high.

While methods from the above studies can be used generally for various applications, the goal of this paper is to design specific data reconstruction techniques for randomized perturbation CF schemes (in particular, schemes proposed in [17, 18]). We intend to answer the following questions: What is the empirical trade-

off between recommendation accuracy and customer privacy? Can randomized perturbation CF schemes preserve enough privacy in customers' data? What kinds of rating data sets are most applicable for the application of randomized perturbation?

In this paper, we develop two data reconstruction methods to recover information about the original ratings from randomly perturbed rating data. Our first method is based on k-means clustering, which exploits data distributions within a rating profile (vector) to reconstruct original ratings from any perturbed rating vector. Our second method is based on singular value decomposition (SVD), which exploits data correlations among different rating vectors to reconstruct original z-scores (normalized rating data) from a whole perturbed rating matrix.

We test the performance of these two methods in real data sets, where experimental results show that both methods are able to reconstruct a considerable amount of information on the original data. Results also clearly reveal an empirical trade-off between recommendation accuracy and user privacy in perturbation schemes. Moreover, we provide several suggestions drawn from these experiments for the future use of these schemes.

The rest of the paper is organized as follows. Section 2 summarizes related work. Section 3 gives an overview of randomized perturbation schemes in collaborative filtering. In Section 4 and 5, we present new k-means based and SVD-based data reconstruction methods. Experimental results are presented in Section 6. Finally, we conclude our work in Section 7.

2 Related Work

Canny proposed a privacy-preserving scheme for truncated SVD and factor analysis in collaborative filtering [7, 8]. In this scheme, users retain ratings themselves; a community of users can then compute a public aggregate of their data to generate recommendations without disclosing their individual ratings. The idea is to reduce the SVD computation to an iterative calculation requiring only the addition of vectors of user data, and use homomorphic encryption to allow the sums of encrypted vectors to be computed and decrypted securely. There are two concerns regarding this scheme. First, both the computational cost and the communication cost for each user in each round of iterations are $O(km \log n)$ (in which m is the number of items, n is the number of users, and k is the rank of the low-dimensional linear model). According to the author's estimation in [7], each round may take about 15 hours and 40-60 rounds are usually needed. The second concern is that the number of total users should be known in advance and all

users need to synchronize to make sure they work on the same data during the whole process.

Polat and Du proposed two random perturbation CF schemes for a Pearson correlation-based CF algorithm [17] and an SVD-based CF algorithm [18]. The random perturbation technique was first introduced into privacy-preserving data mining by Agrawal and Srikant [3] and was extended by Agrawal and Aggarwal [2]. Since a trade-off between accuracy and privacy exists in random perturbation techniques, several studies have focused on privacy analysis [11, 15, 16]. Evfimievski *et al.* presented a formula for privacy breaches and a methodology to limit breaches in the field of association rule mining [11]. Kargupta *et al.* pointed out that arbitrary randomization is not safe [16]. Huang *et al.* studied why and how correlations affect privacy and identified other potential factors that can influence privacy [15]. Inspired by this body of work, we give a privacy analysis of random perturbation schemes in the application of collaborative filtering.

3 Random Perturbation Collaborative Filtering Schemes

In this section, we give an overview of random perturbation CF schemes proposed in [17, 18] and introduce the notation that is used in this paper.

In order to disguise a number a using random perturbation techniques, one can simply add a random value r to it; $a + r$ instead of a will then be revealed to the data miner. Although the data miner does not know the actual value of a, since it is disguised, it can still estimate many quantities that depend on aggregated data (*e.g.*, the expectation of a).

To preserve user privacy, the server should not know any true rating. Random values are drawn from Gaussian or uniform distributions by the user to add to original ratings. In a uniform distribution, users create uniform random values from a range $[-\alpha_R, \alpha_R]$. In a Gaussian distribution, users generate random values with zero mean and standard deviation σ_R. Users then disguise their ratings before sending them to the server. The steps of data disguise are as follows (see Figure 1):

1. The server decides whether only rated entries are used or all entries are used, chooses a random value distribution (Gaussian or uniform), and gives users the related parameter (α_R or σ_R).

2. If only rated entries are used, each user i computes her rating average and standard deviation σ_i, and then calculates the z-scores for those items that she has rated. If all entries are used, each user first fills unrated entries of her rating vector with her rating average, and then calculates the standard deviation

items

users | rating matrix A

the server decides on a uniform or a Gaussian distribution, and on the value of α or σ, respectively

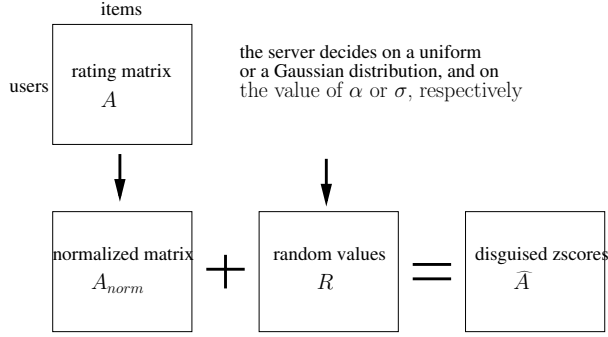normalized matrix A_{norm} $+$ random values R $=$ disguised zscores $\widehat{A}$

Figure 1: The flow chart for random perturbation collaborative filtering schemes.

σ_i and z-scores for all items. Denote the computed z-score matrix (users by items) as A_{norm}. If only rated entries are used, then there are some missing entries in A_{norm}.

3. Each user creates random values drawn from the selected distribution and adds these random values to her z-score values to generate disguised z-scores. Each user then sends the disguised z-scores to the server. Denote the matrix of generated random values as R and the disguised z-score matrix as $\widehat{A} = A_{norm} + R$.

After the server obtains the disguised z-scores, it can apply Eqs. (4–5) in [17] to make predictions using the Pearson correlation-based CF algorithm proposed in [13]. Alternatively, it can apply Eqs. (3–6) in [18] to make predictions using the SVD-based CF algorithm proposed in [20].

There are two versions of Step 2. The first version uses only rated entries, and is designed for the Pearson correlation-based CF algorithm because the algorithm needs to explicitly know what items are rated by a certain user. Users may have a concern regarding this version in some applications because whether items are rated or not is often a privacy issue. The second version, which uses all entries, is designed for the SVD-based CF algorithm. It is worth pointing out that if users do not have the above concern, the first version can bring performance gains in predictions and recommendations when used with the SVD-based algorithm. This point will be further explained in Section 5.2. In this paper the first version of Step 2 is considered for both CF algorithms. Readers should also note that for a user i the standard deviations σ_i computed in these two versions are different; the one computed in the first version is typically much larger than the one computed in the second version (as determined by the sparsity of rating vectors).

4 k-means Clustering based Reconstruction

Given a row vector of the disguised z-score matrix $\widehat{A}$ (corresponding to the ratings of one user), we now propose to use k-means clustering to derive estimates of the original ratings.

4.1 Discrete-valued Ratings We start from the simple case where ratings are discrete-valued and only rated entries are used in the scheme. Assume that there are k possible ratings $(a_1, \cdots, a_k)$ used in the recommendation system, so that the number of possible z-scores in a user's profile after normalization should also be k; denote these k z-scores as $z_1, \cdots, z_k$. Denote the disguised z-scores from this user as $d_1, \cdots, d_h$ if she has given ratings to h items. In order to derive the original rating corresponding to d_i, the server intends to find an a^* from a_1 to a_k that maximizes the posterior probability $\Pr(a|d_i)$. Since $\Pr(a_j|d_i) = \Pr(z_j|d_i)$ for each j, the server can first compute a z^* from z_1 to z_k that maximizes $\Pr(z|d_i)$ and then derive a^* from z^*. If the server knows the values of $z_1, \cdots, z_k$, the solution to maximizing $\Pr(z|d_i)$ can easily be obtained by finding the z^* that is the nearest to d_i.

Recall that every d_i is the sum of a z-score and a random value generated from either a Gaussian distribution or a uniform distribution. Therefore, every d_i can be thought of as a value generated by mixture models, in which the number of components is k and the jth component is $N(z_j, \sigma_R^2)$ (in the Gaussian case) or $\text{Uniform}(z_j - \alpha_R, z_j + \alpha_R)$ (in the uniform case). In general, the Expectation-Maximization (EM) algorithm [10] is one way to compute the parameters of mixture models ($z_1, \cdots, z_k$ in this case). In this paper, we use a simplified and constrained version—the k-means clustering algorithm. After k-means is conducted on $d_1, \cdots, d_h$, the original k z-scores will be approximated by the sorted k cluster centroids. An entry with the disguised z-score assigned into the jth cluster will be reconstructed as the jth possible rating a_j. For the initial cluster centroids assignment, a naive method is to estimate the first cluster centroid as the mean of the bottom $l\%$ of h disguised z-scores and the kth centroid as the mean of the top $l\%$, and then interpolate the other centroids.

Figure 2 shows a detailed example taken from MovieLens. It plots the original ratings, the disguised z-scores (using a Gaussian distribution with $\sigma_R = 0.5$), and the cluster centroids (after k-means) of one user's rating profile.

There are two specific cases in which some ratings may be missed in a user's rating profile. For the first case, assume that certain ratings except the lowest and the highest have never appeared. As a result, certain
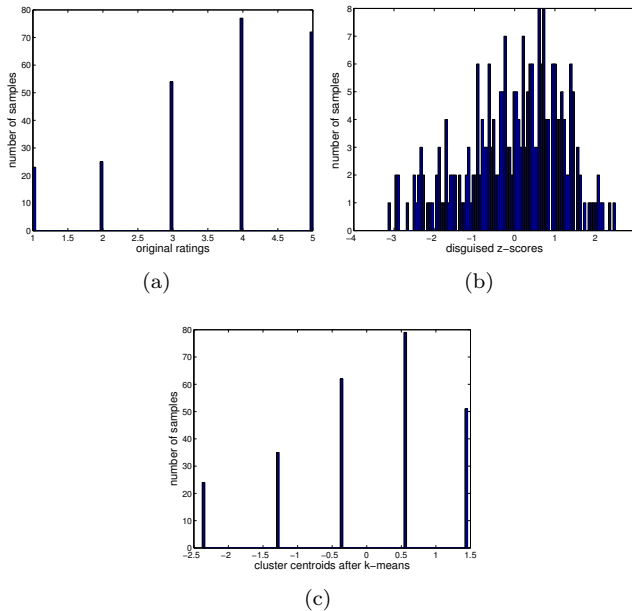
Figure 2: This example shows how k-means is used to reconstruct original ratings from disguised z-scores in the perturbation scheme when only rated entries are used. Subfigure (a) is a histogram of the original ratings from one user in MovieLens. Subfigure (b) is a histogram of the disguised z-scores in which random perturbations are incorporated. Subfigure (c) shows the histogram of the assigned cluster centroids after k-means is used.

clusters might become empty in the first iteration of k-means given the estimation of $z_1, \cdots, z_k$. For example, if a_j is missed, then the jth cluster may become empty. If we still assign disguised z-scores to the jth cluster, an initially "correct" cluster of disguised z-scores will probably be divided into two sub-clusters. To prevent this situation, we drop any empty cluster during the clustering process. In the second case, the lowest or the highest rating is missing; given this, the estimation of $z_1, \cdots, z_k$ may deviate considerably from the true values. One can prove that the probability of accurately reconstructing original ratings by any method in this case can be no more than 50%. If two rating profiles result in the same z-score vector (see Table 1 for an example), there is no better way than flipping a coin to find the original profile from only the disguised z-score vector. When the lowest or the highest rating is missing in a rating profile, we can always find at least one different profile that results in the same z-score vector as the current one does.

We now consider the other version of the perturbation scheme (Step 2 above), in which all entries are

Table 1: Two rating profiles that result in the same z-score vector in a five point rating scale system.

Item	I1	I2	I3	I4	I5	I6	I7	I8
Profile1	2	2	-	3	3	3	-	5
Profile2	1	1	-	2	2	2	-	4

used. In order to derive ratings from a disguised z-score vector in this version, we can first attempt to determine which entries are rated and which entries are unrated, and then apply k-means clustering on "rated" entries only. Since the z-score of each unrated entry in A_{norm} is 0 according to Step 2 in the scheme given in Section 3, the corresponding disguised z-score will lie in the range of $[-\alpha_R, \alpha_R]$ if a uniform distribution is used and in the range of $[-3\sigma_R, 3\sigma_R]$ with a probability of 99.7% if a Gaussian distribution is used. Therefore, in our classification method, entries with disguised z-scores that lie in these two ranges are marked as unrated entries and all the other entries are marked as rated. This classification method may identify many rated entries as unrated[1]; however, we suppose that an attacker is more interested in precision (the ratio of the number of entries correctly marked rated to those marked rated) than recall (the ratio of the number of entries correctly marked rated to those actually rated). Figure 3 shows the disguised z-scores, the disguised z-scores of the entries marked rated, and the cluster centroids (after k-means) from the same user's profile used previously in Figure 2.

4.2 Continuous-valued Ratings The above analysis is based on the assumption that ratings are discrete-valued. For the case where ratings are continuous-valued, we discretize ratings by dividing the rating range into k isometric segments; if the disguised z-score of an entry is assigned to the jth cluster after k-means, we reconstruct the rating of this entry as the median value of the jth segment. There are two kinds of errors when measuring the difference between a true rating and a reconstructed rating. One is the assignment of an entry to the wrong cluster, *e.g.*, an entry with a rating from the jth segment may be assigned to the $j + 1$th cluster. The other is the representation error caused by using the median value as the reconstructed rating. Deciding the value of k is actually balancing a trade-off between these two errors; the clustering penalty becomes larger when k is larger and the representation error becomes larger when k is smaller. In our experiments, k is chosen empirically.

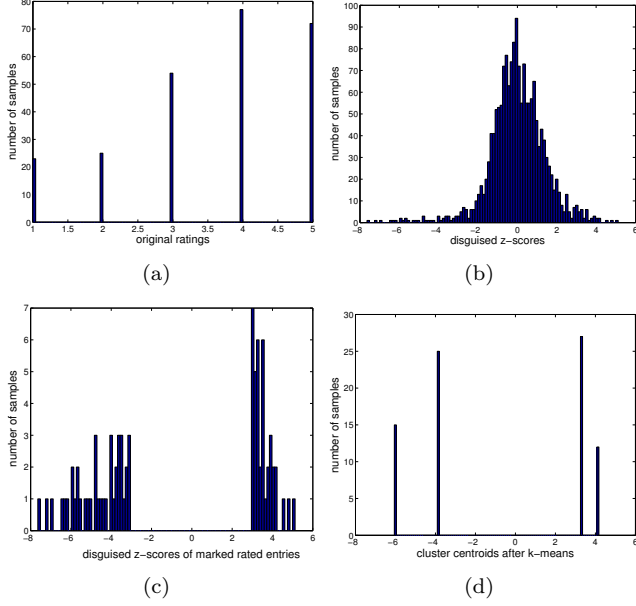[1]Unrated entries may also be marked as rated if a Gaussian distribution is used, though with a small probability.

Figure 3: This example shows how k-means is used to reconstruct original ratings from disguised z-scores in the perturbation scheme when all rated entries are used. The histogram of the original ratings is plotted in (a) and is identical to Figure 2(a). Subfigure (b) plots the histogram of all disguised z-scores. Subfigure (c) is a histogram of the disguised z-scores of the entries identified as rated. Subfigure (d) shows the histogram of the assigned cluster centroids after k-means is used. Note that one cluster becomes empty in the clustering process.

5 SVD-based Reconstruction

Attackers may also be interested in reconstructing the z-scores in A_{norm} as they also disclose important information relevant to user privacy. In this section, we show how to estimate the original z-scores in each rating profile from the entire perturbed rating matrix using correlations among the rating profiles.

It is argued in [8] that a low-dimensional linear model is a powerful method for describing the rating matrix. The assumption that the rating matrix can be modeled linearly is also used in [5, 6, 7, 12, 20, 21]. Based on this assumption, A_{norm} can be written as the sum of a low-dimensional linear model (denoted as X) and a Gaussian distributed noise matrix (denoted as Z), where $z_{ij} \sim N(0, \sigma^2)$. Therefore, with random perturbations R we have

$$(5.1) \qquad \widehat{A} = A_{norm} + R = X + Z + R.$$

In making predictions, the server wants to construct the model X based on perturbed ratings $\widehat{A}$. If the server knows X, then since $A_{norm}^{ij} \sim N(X_{ij}, \sigma^2)$, the server can use X_{ij} as the prediction for an unrated entry A_{norm}^{ij}. The variance of this prediction error is σ^2. Meanwhile, for a rated entry A_{norm}^{ij}, the server can also reconstruct it as X_{ij}. The variance of this reconstruction error is also σ^2. Therefore, using X for predictions and reconstructions leads to an equivalence in the variance of the prediction error and the reconstruction error. This explicitly shows a trade-off between prediction accuracy and privacy.

5.1 Deriving the Model Now the problem is how to find X given $\widehat{A}$. If R is generated by a Gaussian distribution, i.e., $R_{ij} \sim N(0, \sigma_R^2)$, Eq. (5.1) can be further extended as

$$(5.2) \qquad \widehat{A} = X + Z + R = X + U,$$

where $U_{ij} \sim N(0, \sigma^2 + \sigma_R^2)$. Therefore, the log-likelihood of the disguised z-score matrix given the model, $\log \Pr(\widehat{A}|X)$, can be expressed as

$$(5.3) \log \Pr(\widehat{A}|X) = -\frac{1}{2(\sigma^2 + \sigma_R^2)} \sum_{ij} (\widehat{A}_{ij} - X_{ij})^2 + C,$$

where C is a constant. By the above equation, finding an X that maximizes $\log \Pr(\widehat{A}|X)$ is reduced to a low rank approximation problem, which can be solved by conducting SVD on $\widehat{A}$. If the rank of the linear model is k, the solution is $X = \widehat{A} H_k H_k^T$, in which H_k is the matrix consisting of the top k right singular vectors of $\widehat{A}$.

Another approach, usable for both random variable distributions, is to find an X that maximizes $\log \Pr(A_{norm}|X)$. This maximization problem can also be reduced to a low rank approximation problem using an induction similar to the above. Then we have $X = A_{norm} V_k V_k^T$, where V_k is the matrix of the top k right singular vectors of A_{norm}. It is easy to see that the column vectors in V_k are also the first k eigenvectors of $A_{norm}^T A_{norm}$. Moreover, $A_{norm}^T A_{norm}$ can be estimated from $\widehat{A}^T \widehat{A}$ using the following two equations. As $E(R_{ij}) = 0$, by Eqs. (3–4) in [18], we have

$$(5.4) \quad E((\widehat{A}^T \widehat{A})_{fg}) = (A_{norm}^T A_{norm})_{fg}, f \neq g,$$
$$(5.5) \quad E((\widehat{A}^T \widehat{A})_{ff}) = (A_{norm}^T A_{norm})_{ff} + n\sigma^2,$$

where n is the number of users. So given $\widehat{A}$, $A_{norm}^T A_{norm}$ and V_k can be sequentially computed, and X can be approximated as $\widehat{A} V_k V_k^T$. The error between this approximation and X is

$$\begin{aligned}
(5.6) \quad \widehat{A} V_k V_k^T - X &= (\widehat{A} - A_{norm}) V_k V_k^T \\
&= R V_k V_k^T.
\end{aligned}$$

Using the proof of Theorem 5.2 in [15], the mean square error of $RV_kV_k^T$ can now be quantified as $var(R_{ij})\frac{k}{m}$, where m is the number of items. When a Gaussian distribution is used, the mean square error is $\sigma_R^2\frac{k}{m}$; when a uniform distribution is used, it is $\alpha_R^2\frac{k}{3m}$. Note that using $\widehat{A}V_kV_k^T$ as the approximation of X was also proposed in the SVD-based CF algorithm for randomly perturbed ratings in [18], but here we give a clear relation between $\widehat{A}V_kV_k^T$ and the model X that maximizes $\log\Pr(A_{norm}|X)$.

Comparing the above two approaches, if R is generated from a Gaussian distribution, the first approach is preferred because it can directly compute X to maximize $\log\Pr(\widehat{A}|X)$ without any approximation error. However, for a uniform distribution, only the second approach can be used.

5.2 EM Procedure In practice, the rating matrix A is usually sparse. Consequently, the z-score matrix A_{norm} obtained by filling unrated entries with user averages may not lead to a good linear model X. A more reasonable objective function is to find an X that maximizes $\log\Pr(\widehat{A}^o|X)$ or $\log\Pr(A_{norm}^o|X)$, where $\widehat{A}^o$ and A_{norm}^o denote the disguised z-scores and the original z-scores of those rated entries, respectively. Note that even when all entries are used in the perturbation scheme, the server can still use the classification method discussed in Section 4.1 to identify some rated entries.

The maximization of $\log\Pr(\widehat{A}^o|X)$ or $\log\Pr(A_{norm}^o|X)$ can be accomplished using an EM procedure. The details of similar inductions can be found in [21] and in Section 2 of [22]. In iteration t of the EM procedure, in the Expectation step, each unrated entry in $\widehat{A}$ ($\widehat{A}_{ij}$) is replaced with $X_{ij}^{(t-1)}$ to form a filled-in matrix $\widehat{A}^{(t)}$. In the Maximization step, the server either computes $X^{(t)}$ as $\widehat{A}^{(t)}H_k^{(t)}(H_k^{(t)})^T$ using the first approach (if a Gaussian distribution is used) or approximates $X^{(t)}$ by $\widehat{A}^{(t)}V_k^{(t)}(V_k^{(t)})^T$ using the second approach (if a uniform distribution is used). For the first iteration, unrated entries in $\widehat{A}$ are filled with 0 in the Expectation step.

6 Experiments

We used two publicly available data sets, MovieLens and Jester, in our experiments. MovieLens is a web-based recommender system for movies based on the GroupLens project.[2] User ratings for movies are discrete-valued on a five-star scale. Jester is a web-based joke recommendation system [12]. The dataset contains 100 jokes, with continuous-valued user ratings ranging from

from -10 to 10. For our experiments, we used a 943-by-1862 (users-by-items) rating matrix from the standard 100k MovieLens data set and a 3000-by-100 rating matrix from Jester. For each rating matrix, 80% of the observed ratings were chosen randomly as available training cases and the other observed ratings were reserved as test cases.

For each data set, 20 trials were conducted using each version of the perturbation schemes (one based on rated entries only and the other based on all entries) for each random noise distribution (Gaussian or uniform). We selected three kinds of metrics detailed below for the evaluation.

1. To evaluate prediction and recommendation accuracy, *Mean Absolute Error (MAE)* and *ROC-4 area* are used. MAE measures the average absolute deviation between a predicted rating and the user's true rating. The receiver operating characteristic (ROC) model attempts to measure the extent to which an information filtering system can successfully distinguish between signal (relevance) and noise. The ROC-4 area is the area underneath an ROC curve when ratings of 4 and above are considered signal and those below are considered noise. In our experiments, ROC-4 area averaged per-user is computed. According to the experiments in [14], MAE and ROC-4 per-user do not correlate well; using both metrics thus helps to evaluate recommendation accuracy more objectively and comprehensively.

2. For evaluating reconstruction accuracy when k-means is used for the data reconstruction, we use MAE to measure the average absolute deviation between a reconstructed rating and the true rating; this is denoted as *R-MAE* to distinguish it from MAE above. We also compute the ratio of discrete-valued ratings (MovieLens) that are reconstructed correctly; this is denoted as *Accuracy* for simplicity. For the SVD-based data reconstruction method, MAE is used to measure the deviation between reconstructed z-scores and the original z-scores; this is denoted as *zscore-MAE*. For ease of comparison between these two reconstruction methods, *P-MAE* is computed to evaluate the deviation between the true ratings and the potential ratings that can be recovered by the SVD-based method. Potential ratings are computed using the derived z-scores together with averages and standard deviations of users' rating vectors (assuming they are known).

3. Finally, *Precision* and *Recall* are used to evaluate

the accuracy of identifying entries (as rated) when all entries are used in the perturbation scheme.

6.1   MovieLens Table 3 first gives the prediction and recommendation accuracy of two naive CF algorithms, "Item averages" and "User averages"; they are considered as the baselines for future comparisons. Since both algorithms are based on the sum operation, it is easy to design privacy-preserving schemes for them. Therefore, we believe that it is unacceptable if the prediction performance of a certain CF algorithm is worse than the performance of these two algorithms. Note that the ROC model is meaningless when "User averages" is used because for a certain user all items will receive the same prediction.

We now look at the case of using only rated entries in the perturbation scheme. Both the Pearson correlation-based algorithm in [17] and the SVD-based (with an EM procedure) algorithm proposed in Section 5 are used for predicting ratings. We use our SVD-based algorithm instead of the original algorithm proposed in [18] because our algorithm can get better performance according to our preliminary experiments. The number of dimensions is set to 10 in the SVD-based algorithm. Table 2 lists the results obtained from MovieLens for different values of σ_R or α_R. When $\sigma_R, \alpha_R = 0$, no random noise is actually added to the original z-scores. We observe that the Accuracy obtained by the k-means based reconstruction method is not 100% at that time; the reason is that some possible ratings have never appeared in some users' rating profiles, as discussed in Section 4.1. When $\sigma_R = 1$ or $\alpha_R = \sqrt{3}$, the MAE and the ROC-4 area of both CF algorithms are worse than those from using "Item averages" for predictions. If reasonable random values are used (*e.g.* $1/3 \le \sigma_R \le 2/3$ or $\sqrt{3}/3 \le \alpha_R \le 2\sqrt{3}/3$), the k-means based reconstruction method has a 45–67% accuracy in correctly deriving the orginal ratings.

Shown in Figure 4 is the comparison of our SVD-based reconstruction method and the PCA-based reconstruction method proposed in [15].[3] It demonstrates that the reconstruction error obtained by our method is smaller than that of the previous approach. Figure 5 compares the k-means based reconstruction method with the SVD-based method. It shows that the reconstruction error obtained from either data reconstruction method is always smaller than the prediction error. For a smaller σ_R or α_R, the k-means based reconstruction method outperforms the SVD-based method; for a

---

[3]Note that an EM procedure is incorporated in the proposed SVD-based method while it is absent in the PCA-based method in [15].

Table 3: MAE and ROC-4 area of two CF algorithms, "Item averages" and "User averages", in MovieLens. Note that the ROC-4 area is meaningless when "User averages" is used because for a certain user all items will receive the same prediction.

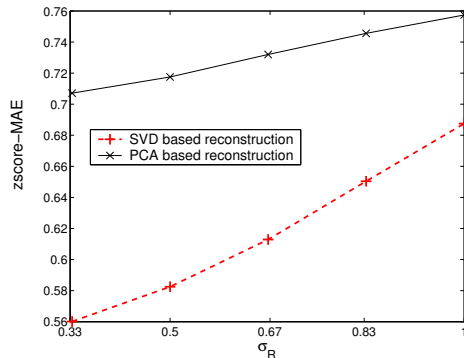| CF algorithms	MAE	ROC-4
Item averages	0.8154	0.7578
User averages	0.8350	-

Figure 4: Comparison of our SVD-based reconstruction method and the PCA-based reconstruction method proposed in [15] in MovieLens when only rated entries are used in the perturbation scheme.

larger variance, the SVD-based method has a smaller reconstruction error.

Table 4 displays prediction and reconstruction errors when all entries are used in the perturbation scheme for MovieLens. The baseline prediction performance (when z-scores are not perturbed) of the SVD-based algorithm in this case is much worse than the baseline prediction performance in the version where only rated entries are used. When $\sigma_R = 3$ or $\alpha_R = 3\sqrt{3}$, the prediction performance becomes worse than the performance of the naive CF algorithm "Item averages". If a Gaussian distribution (with $1 \le \sigma_R \le 2$) is used to generate random perturbations, 20–40% of rated entries are recalled with a precision at 80–90%. Better results in terms of precision and recall are obtained when random values are generated from a uniform distribution. The k-means based reconstruction method derives the original ratings correctly for more than 55% of these recalled rated entries. Figure 6 furthers the comparison of the two data reconstruction methods for this case. The k-means based reconstruction method always obtains a smaller reconstruction error than the SVD-based reconstruction method for both random noise distributions.

65

Table 2: Prediction performance and reconstruction performance in MovieLens if only rated entries are used in the perturbation scheme.

	MAE(SVD)	ROC-4(SVD)	MAE(Pearson)	ROC-4(Pearson)	R-MAE	Accuracy	zscore-MAE
$\sigma_R, \alpha_R = 0$	0.7493	0.7477	0.7694	0.7544	0.0795	0.9246	-
$\sigma_R = 1/3$	0.7582 ± 0.0020	0.7349 ± 0.0044	0.7749 ± 0.0014	0.7497 ± 0.0028	0.3393 ± 0.0040	0.6712 ± 0.0037	0.5601 ± 0.0005
$\sigma_R = 2/3$	0.7850 ± 0.0026	0.7051 ± 0.0063	0.7932 ± 0.0031	0.7401 ± 0.0041	0.6204 ± 0.0037	0.4565 ± 0.0029	0.6129 ± 0.0012
$\sigma_R = 1$	0.8192 ± 0.0024	0.6716 ± 0.0060	0.8234 ± 0.0025	0.7259 ± 0.0039	0.7850 ± 0.0056	0.3776 ± 0.0026	0.6875 ± 0.0015
$\alpha_R = \sqrt{3}/3$	0.7591 ± 0.0013	0.7343 ± 0.0026	0.7748 ± 0.0020	0.7500 ± 0.0021	0.4167 ± 0.0031	0.5898 ± 0.0030	0.5603 ± 0.0007
$\alpha_R = 2\sqrt{3}/3$	0.7855 ± 0.0029	0.7049 ± 0.0076	0.7928 ± 0.0031	0.7410 ± 0.0039	0.6138 ± 0.0049	0.4474 ± 0.0025	0.6131 ± 0.0013
$\alpha_R = \sqrt{3}$	0.8179 ± 0.0036	0.6730 ± 0.0073	0.8218 ± 0.0036	0.7273 ± 0.0063	0.7983 ± 0.0034	0.3629 ± 0.0018	0.6877 ± 0.0013

Table 4: Prediction performance and reconstruction performance in MovieLens if all entries are used in the perturbation scheme.

|	MAE(SVD)	ROC-4(SVD)	Precision	Recall	R-MAE	Accuracy	zscore-MAE
$\sigma_R, \alpha_R = 0$	0.7971	0.7129	-	-	-	-	-
$\sigma_R = 1$	0.7986 ± 0.0003	0.7036 ± 0.0033	0.8952 ± 0.0017	0.4338 ± 0.0010	0.2943 ± 0.0044	0.7151 ± 0.0043	2.7362 ± 0.0065
$\sigma_R = 2$	0.8048 ± 0.0010	0.6759 ± 0.0038	0.7801 ± 0.0024	0.1797 ± 0.0008	0.4167 ± 0.0056	0.6059 ± 0.0055	3.1580 ± 0.0182
$\sigma_R = 3$	0.8179 ± 0.0017	0.6432 ± 0.0062	0.6285 ± 0.0046	0.0879 ± 0.0010	0.4654 ± 0.0074	0.5847 ± 0.0056	3.7506 ± 0.0189
$\alpha_R = \sqrt{3}$	0.7988 ± 0.0005	0.7046 ± 0.0043	1	0.6486 ± 0.0011	0.2432 ± 0.0034	0.7629 ± 0.0034	2.4828 ± 0.0036
$\alpha_R = 2\sqrt{3}$	0.8051 ± 0.0012	0.6774 ± 0.0061	1	0.4227 ± 0.0016	0.3923 ± 0.0048	0.6194 ± 0.0044	2.7192 ± 0.0111
$\alpha_R = 3\sqrt{3}$	0.8178 ± 0.0022	0.6417 ± 0.0074	1	0.3018 ± 0.0011	0.4533 ± 0.0062	0.5670 ± 0.0058	3.0480 ± 0.0110
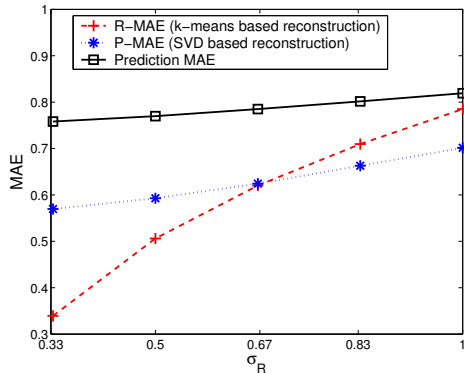
(a) Gaussian distribution
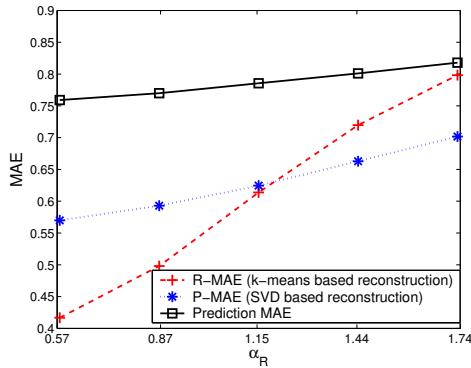
(b) Uniform distribution

Figure 5: Comparison of two data reconstruction methods in MovieLens when only rated entries are used in the perturbation scheme.
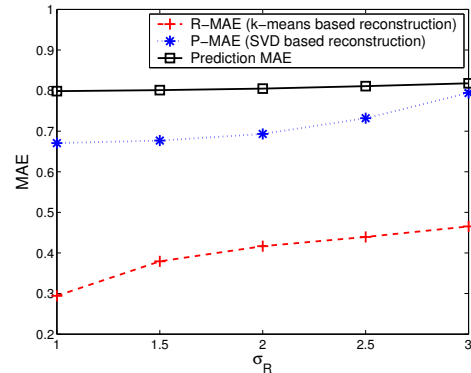
(a) Gaussian distribution
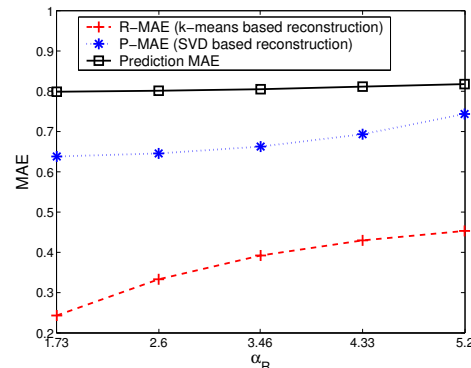
(b) Uniform distribution

Figure 6: Comparison of two data reconstruction methods in MovieLens when all entries are used in the perturbation scheme.

Table 5: MAE and ROC-4 area of two CF algorithms, "Item averages" and "User averages", in Jester.

CF algorithms	MAE	ROC-4
Item averages	4.0703	0.7314
User averages	3.8795	-

6.2 Jester For the Jester data set, the prediction results of using "Item averages" and "User averages" are listed in Table 5; using "User averages" brings a smaller MAE than using "Item averages" in this data set.

The results of testing perturbation CF algorithms and data reconstruction methods for the scheme in which only rated entries are used are shown in Table 6. The number of clusters is set to 10 for k-means. The prediction performance of the SVD-based CF algorithm is worse than that of "User averages" when $\sigma_R = 1.5$ or $\alpha_R = 3\sqrt{3}/2$. However, the prediction performance loss of the Pearson correlation-based algorithm is much smaller. We believe that this is caused by two differences of this data set from MovieLens: many fewer items and a much higher density (fraction of the rating matrix that is filled). Figure 7 compares our two data reconstruction methods in Jester when only rated entries are used in the perturbation scheme. As was noted in Figure 5, the k-means based method obtains a smaller reconstruction error when the random noise variance is small while the SVD-based reconstruction method performs better when the noise variance is larger.

Finally, Table 7 lists the results obtained in Jester when all entries are used in the perturbation scheme. We observe that when a Gaussian distribution is used to generate noise, only a small percentage of the rated entries can be recalled (e.g., 7% when $\sigma_R = 1.4$). In contrast, when a uniform distribution is used, more than 20% of the rated entries are recalled when $\alpha_R = 1.4 \cdot \sqrt{3}$. Figure 8 shows that in this case the k-means based data reconstruction method always obtains a smaller reconstruction error than the SVD-based method for both random noise distributions.

6.3 Discussion The following are several observations drawn from our experimental results that can be taken as references for the future use of perturbation schemes in collaborative filtering. First, when ratings are discrete-valued, we consistently observed a prediction error larger than the reconstruction error for both versions of the scheme. This implies that the average amount of information that users need to disclose is larger than the average amount of information they receive in return. Moreover, the k-means based reconstruction method can derive a significant fraction of
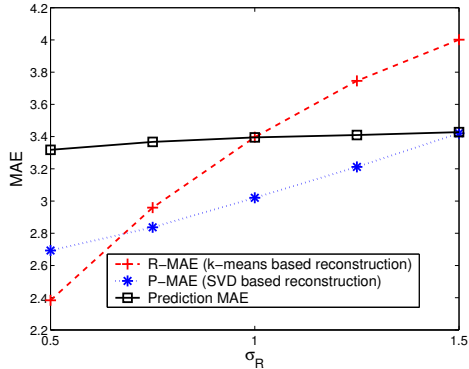
(a) Gaussian distribution
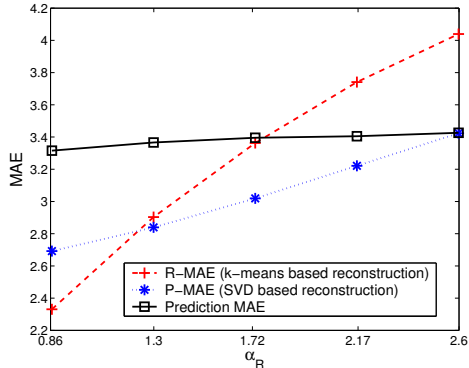
(b) Uniform distribution

Figure 7: Comparison of two data reconstruction methods in Jester when only rated entries are used in the perturbation scheme.

the original ratings exactly for discrete-valued ratings. These two points suggest that continuous-valued ratings are better suited for privacy-preserving perturbation scheme.

Second, there is a trade-off in choosing whether only rated entries or all entries should be used in the perturbation scheme. The prediction performance of the former choice is better than that of the latter; however, in the former choice the server or an attacker can reconstruct data for all rated entries, while they can do so for only a limited number of rated entries in the latter.

Finally, when all entries are used in the perturbation, a Gaussian distribution is preferable for generating random values. This helps to decrease both the precision and recall when identifying rated entries and sequentially increase the reconstruction error.

A flexible randomization scheme was proposed in [17] in which users randomly generate σ_R or α_R within a range decided by the server. For example, if the server sets the range of σ_R as $[0,1]$, each user will choose a

Table 6: Prediction performance and reconstruction performance in Jester if only rated entries are used in the perturbation scheme.

	MAE(SVD)	ROC-4(SVD)	MAE(Pearson)	ROC-4(Pearson)	R-MAE	zscore-MAE
$\sigma_R, \alpha_R = 0$	3.2272	0.7522	3.3798	0.7480	-	-
$\sigma_R = 1/2$	3.3180 ± 0.0069	0.7376 ± 0.0027	3.3813 ± 0.0029	0.7481 ± 0.0005	2.3839 ± 0.0080	0.6113 ± 0.0005
$\sigma_R = 1$	3.5674 ± 0.0110	0.7066 ± 0.0028	3.3952 ± 0.0053	0.7461 ± 0.0010	3.3964 ± 0.0079	0.6852 ± 0.0012
$\sigma_R = 3/2$	3.9306 ± 0.0195	0.6739 ± 0.0041	3.4279 ± 0.0075	0.7422 ± 0.0016	4.0016 ± 0.0089	0.7764 ± 0.0021
$\alpha_R = \sqrt{3}/2$	3.3153 ± 0.0069	0.7382 ± 0.0019	3.3810 ± 0.0028	0.7479 ± 0.0008	2.3309 ± 0.0062	0.6112 ± 0.0005
$\alpha_R = \sqrt{3}$	3.5644 ± 0.0115	0.7067 ± 0.0028	3.3954 ± 0.0036	0.7460 ± 0.0011	3.3627 ± 0.0076	0.6851 ± 0.0015
$\alpha_R = 3\sqrt{3}/2$	3.9297 ± 0.0150	0.6729 ± 0.0032	3.4265 ± 0.0068	0.7421 ± 0.0014	4.0393 ± 0.0078	0.7773 ± 0.0019

Table 7: Prediction performance and reconstruction performance in Jester if all entries are used in the perturbation scheme.

	MAE(SVD)	ROC-4(SVD)	Precision	Recall	R-MAE	zscore-MAE
$\sigma_R, \alpha_R = 0$	3.3303	0.7400	-	-	-	-
$\sigma_R = 0.7$	3.3924 ± 0.0044	0.7198 ± 0.0033	0.9619 ± 0.0009	0.2402 ± 0.0009	2.3532 ± 0.0138	0.8298 ± 0.0031
$\sigma_R = 1.4$	3.5895 ± 0.0117	0.6828 ± 0.0032	0.8780 ± 0.0036	0.0677 ± 0.0006	2.9741 ± 0.0311	1.1448 ± 0.0121
$\sigma_R = 2.1$	3.9047 ± 0.0137	0.6486 ± 0.0033	0.7848 ± 0.0047	0.0341 ± 0.0004	3.5908 ± 0.1363	1.2714 ± 0.0194
$\alpha_R = 0.7 \cdot \sqrt{3}$	3.3926 ± 0.0037	0.7191 ± 0.0020	1	0.4302 ± 0.0011	2.2553 ± 0.0064	0.8651 ± 0.0014
$\alpha_R = 1.4 \cdot \sqrt{3}$	3.5872 ± 0.0113	0.6830 ± 0.0028	1	0.2220 ± 0.0010	2.4507 ± 0.0078	1.0059 ± 0.0045
$\alpha_R = 2.1 \cdot \sqrt{3}$	3.9070 ± 0.0151	0.6495 ± 0.0032	1	0.1480 ± 0.0007	2.6874 ± 0.0145	1.0967 ± 0.0109
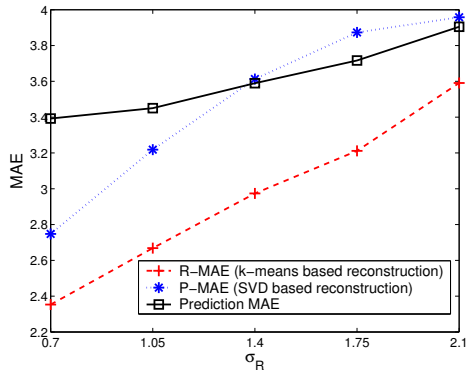
(a) Gaussian distribution
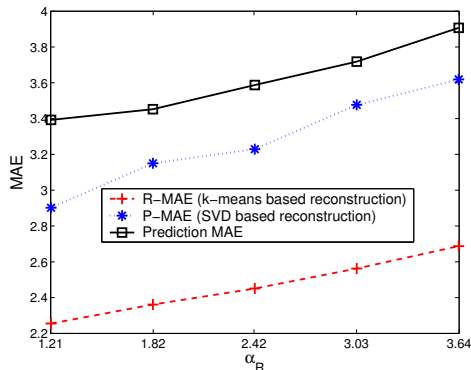
(b) Uniform distribution

Figure 8: Comparison of two data reconstruction methods in Jester when all entries are used in the perturbation scheme.

random value in this range as the standard deviation of the Gaussian distribution he will use. In fact, the server is still able to estimate the variance of perturbations that each user generates in this flexible scheme. Since the variance of each user's normalized z-score vector is always 1, the standard deviation (for a Gaussian distribution) or the range (for a uniform distribution) of added random values can be derived from the variance of a user's perturbed z-score vector. Therefore, our data reconstruction methods can still work in this flexible scheme. The only adjustment is in the SVD-based data reconstruction method: when different standard deviations are used to generate Gaussian distributed noise, each row will have a different scalar in Eq. (5.3). Scalar-weighted SVD in [4] gives an analytical solution of X that maximizes $\log \Pr(\widehat{A}|X)$ for such a case.

7 Summary and Conclusions

Based on our study of how to reconstruct the original data from randomly perturbed data in the randomized perturbation schemes for collaborative filtering, we propose two data reconstruction methods: a k-means based method and an SVD-based method. Our experiments show that both have good performances in deriving original ratings or z-scores in real data sets. The k-means based method is always better than the SVD-based method when only rated entries are used in the perturbation scheme. When all entries are used, the SVD-based method is better when the random value variance is large and the k-means based method is bet-

ter when the random value variance is small.

The proposed data reconstruction methods have two contributions to the general application of perturbation schemes in privacy-preserving data mining. First, the k-means based method can be used to reconstruct original data when data attributes are valued in the same range. Second, the proposed SVD-based method with an EM procedure can help to exploit correlations to derive information from a disguised data set in which only some data entries are available.

There are two directions for our future work. One is to design other data reconstruction methods that derive user ratings more accurately. On the other hand, we will also study new random perturbation techniques that defeat data reconstruction methods and achieve better privacy preservation.

Acknowledgements

References

[1] Anonymizer.com. http://www.anonymizer.com.

[2] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proc. of the 20th ACM PODS*, pages 247–255, 2001.

[3] R. Agrawal and R. Srikant. Privacy preserving data mining. In *Proc. of the ACM SIGMOD*, pages 439–450, 2000.

[4] P. Aguiar and J. Moura. Factorization as a rank 1 problem. In *Proc. of CVPR*, pages 1178–1184, 1999.

[5] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *Proc. of the 33rd ACM STOC*, pages 619–626, 2001.

[6] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proc. of the 15th ICML*, pages 46–54, 1998.

[7] J. Canny. Collaborative filtering with privacy. In *Proc. of the IEEE Symp. on Security and Privacy*, pages 45–57, 2002.

[8] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proc. of the 25th ACM SIGIR*, pages 238–245, 2002.

[9] L. F. Cranor, J. Reagle, and M. S. Ackerman. Beyond concern: Understanding net users' attitudes. Technical report, AT&T Research, available at http://www.research.att.com/projects/privacystudy/, 1999.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Statistical Society*, 39:1–38, 1977.

[11] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proc. of the 22nd ACM PODS*, pages 211–222, 2003.

[12] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[13] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. of the 22nd ACM SIGIR*, pages 230–237, 1999.

[14] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Tran. on Information Systems*, 22(1):5–53, 2004.

[15] Z. Huang, W. Du, and B. Chen. Deriving private information from randomized data. In *Proc. of the ACM SIGMOD*, pages 37–48, 2005.

[16] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *Proc. of the 3rd IEEE ICDM*, pages 99–106, 2003.

[17] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Proc. of the 3rd IEEE ICDM*, pages 625–628, 2003.

[18] H. Polat and W. Du. SVD-based collaborative filtering with privacy. In *Proc. of the 20th ACM Symp. on Applied Computing*, pages 791–795, 2005.

[19] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for web transactions. *ACM Tran. on Information and System Security*, 1(1):66–92, 1998.

[20] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems–a case study. In *ACM WebKDD Workshop*, 2000.

[21] N. Srebro and T. Jaakkola. Weighted low rank approximation. In *Proc. of the 20th ICML*, pages 720–727, 2003.

[22] S. Zhang, W. Wang, J. Ford, F. Makedon, and J. Pearlman. Using singular value decomposition approximation for collaborative filtering. In *Proc. of the 7th IEEE Conf. on E-commerce*, pages 257–264, 2005.