SVD-based Collaborative Filtering with Privacy [*]

Huseyin Polat
Department of Electrical
Engineering and Computer Science
Syracuse University, 121 Link Hall
Syracuse, NY 13244-1240, USA
Phone: +1 315 443 4124
hpolat@ecs.syr.edu

Wenliang Du
Department of Electrical
Engineering and Computer Science
Syracuse University, 121 Link Hall
Syracuse, NY 13244-1240, USA
Phone: +1 315 443 9180
wedu@ecs.syr.edu

ABSTRACT

Collaborative filtering (CF) techniques are becoming increasingly popular with the evolution of the Internet. Such techniques recommend products to customers using similar users' preference data. The performance of CF systems degrades with increasing number of customers and products. To reduce the dimensionality of filtering databases and to improve the performance, Singular Value Decomposition (SVD) is applied for CF. Although filtering systems are widely used by E-commerce sites, they fail to protect users' privacy. Since many users might decide to give false information because of privacy concerns, collecting high quality data from customers is not an easy task. CF systems using these data might produce inaccurate recommendations. In this paper, we discuss SVD-based CF with privacy. To protect users' privacy while still providing recommendations with decent accuracy, we propose a randomized perturbation-based scheme.

Categories and Subject Descriptors

K.4.4 [**Computers and Society**]: Electronic Commerce—*Security*; H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Security, Performance, Experimentation

Keywords

Privacy, collaborative filtering, SVD, randomization

1. INTRODUCTION

Information overload is becoming a major problem for users with the evolution of the Internet. Different approaches are used to separate the interesting and the valuable information from the rest in order to cope with this problem. Collaborative filtering (CF) is a recent technique that helps users to cope with information overload by using the preferences of other users. With the growth of E-commerce, there is increasing commercial interest in CF technology. Some commercial web sites like Amazon.com, CDNow.com, and MovieFinder.com have made successful use of CF.

CF systems work by collecting ratings for items and matching together users who share the same interest or tastes. Such systems help new users to better decide which items to buy. The goal of CF is to predict how well a user, referred to as the *active user*, will like an item that he did not buy before based on the preferences of a community of users [10]. While it was shown that the memory-based (correlation-based) CF schemes perform well [10], they suffer from some limitations [4, 13]. Therefore, in addition to correlation-based CF algorithms, SVD is applied for CF to address such limitations [4, 13].

Although it has been shown that CF systems have many successful applications in several domains, such systems have a number of disadvantages [6, 7]. The most important is that they are a serious threat to individual privacy. Most online vendors collect preferences of their customers, and make efforts to preserve their customers' privacy. However, several schemes are extremely vulnerable and can be mined for preferences of users [6]. In addition, customer data is a valuable asset and it has been sold when some E-companies suffered bankruptcy. The courts have supported the rights of liquidators to sell off data about their customers' personal information as an asset. Since data from users are needed for CF purposes and many users have concerns about their privacy, providing privacy measures is a key to the success of both data collection and producing recommendations with decent accuracy.

Some people might be willing to selectively divulge information if they can get benefit in return [14]. However, according to a survey conducted in 1999 [8], a significant number of people are not willing to divulge their information because of privacy concerns. The challenge is *how can users contribute their private information for CF purposes without greatly compromising their privacy?*

Anonymous techniques [1, 12] are widely used to achieve privacy. Such techniques allow users to divulge their data without disclosing their identities. However, it is difficult for

[*]This work was supported in part by Grant IIS-0219560 and IIS-0312366 from the United States National Science Foundation.

the database owner to guarantee the quality of the database because a malicious user could send random data and render the database useless, or a competing company could send a great deal of made-up information to make their products the most favorable ones. It is important for the database owner to verify the identities of the data contributors to guarantee the quality.

We propose a scheme to allow SVD-based CF with privacy. Our goal is to ensure users' privacy and to provide accurate predictions. However, privacy and accuracy are conflicting goals; improving one of them decreases the other. We propose a technique to achieve a balance between them. We want to prevent the server from learning which items that the users rated before and how much they like or dislike those rated items. In our scheme (Fig. 1), each user first disguises his private data, and sends it to the data collector (the server), such that the server cannot derive the truthful information about the user's private information. However, the data disguising scheme should still be able to allow the server to conduct CF from the disguised data. We use randomized perturbation (RP) techniques [3] to disguise private data. These techniques are useful if we are interested in aggregate data rather than individual data items because when the number of users and items are significantly large, the aggregate information of these users can be estimated with decent accuracy. Since SVD-based CF is based on aggregate values of a dataset, we hypothesize that *by combining the RP techniques with SVD-based CF algorithms, we can achieve a decent degree of accuracy for SVD-based CF with privacy.*

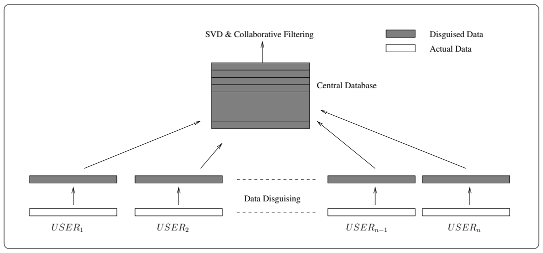


Figure 1: Privacy preserving CF with SVD

To verify this hypothesis, we implemented the RP technique for the SVD-based CF algorithm [13]. We then conducted a series of experiments to show how accurate our results are. We measured the overall performance of our scheme based on disguised data. Our results show that the predictions we have found on randomized data are very close to the original ratings.

2. RELATED WORK

Canny proposes alternative models for privacy-preserving collaborative filtering (PPCF) in which users control all of their data [6, 7]. A community of users can compute a public "aggregate" of their data that does not expose individual users' data. He iteratively calculates the aggregate requiring only addition of vectors of user data. He then uses homomorphic encryption to allow sums of encrypted vectors to be computed and decrypted without exposing individual data.

Our work here differs from Canny's work. While his work focuses on the P2P framework, in which users actively participate in the CF process, our work focuses on another framework, in which users send their data to a server and they do not participate in the CF process; only the server needs to conduct the CF. Polat and Du [11] used randomized perturbation for privacy-preserving correlation-based CF. While their work focuses on correlation-based CF with privacy, our work here focuses on SVD-based CF with privacy. In our scheme, the server creates a database (user-item matrix, A), and uses SVD to factor A into three matrices that are used for predictions.

3. SVD-BASED CF

SVD is a well-known matrix factorization technique that factors an $n \times m$ matrix A into three matrices [13] 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 the matrix A. S is a diagonal matrix of size $y \times y$ having all singular values of matrix A as its diagonal entries. It is possible to reduce the $y \times y$ matrix S to have only k largest diagonal values to obtain a matrix S_k, $k < y$.

Sarwar et al. [13] propose an SVD-based CF algorithm. 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_{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 $\sqrt{S_k}V_k^T$ are computed. These resultant matrices can be used to compute the prediction for any user u on item q. To compute the prediction, 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:

$$p_{uq} = \overline{v}_u + \sigma_u \Big[U_k\sqrt{S_k}(u) \cdot \sqrt{S_k}V_k^T(q)\Big] \quad (1)$$

where $\overline{v}_u$ and σ_u are mean rating and standard deviation for user u, respectively. Since the user u who is looking for prediction will do the denormalization, we can define $p_{uq} = \overline{v}_u + \sigma_u p$ where

$$p = U_k\sqrt{S_k}(u) \cdot \sqrt{S_k}V_k^T(q) \quad (2)$$

4. PRIVACY-PRESERVING SVD-BASED CF

Randomized perturbation techniques were first used by [3] to achieve privacy. In order to disguise a number a, a simple way is to add a random value r to it. $a+r$, rather than a, will appear in the database, where r is a random value drawn from some distribution. Although we cannot do anything to a since it is disguised, we can conduct certain computations if we are interested in the aggregate data rather than individual data items. The basic idea of randomization is to perturb the data in such a way that the server can only know the range of the data, and such range is broad enough to preserve users' privacy.

With the privacy concerns, the server should not know the true ratings of each user and which items that are rated. We created random numbers using uniform and Gaussian distributions. In uniform distribution, all users create uniform random values from a range $[-\alpha, \alpha]$ where α is a constant number. For Gaussian distribution, each user generates random values using normal distribution with mean (μ) being 0 and standard deviation (σ). Users disguise their data before they send it to the server. The steps of data disguising are as follows:

1. The server decides on the distributions of perturbing data (uniform or Gaussian) and parameters (α, σ, and μ), and let each user know.
2. Each user u fills empty cells of his ratings vector using his 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. Then each user u adds those random values to his z-score values and generates the disguised z-scores $z'_{uj} = z_{uj} + r_{uj}$ for $j = 1 \dots m$. Finally, each user sends z'_{uj} values to the server who creates the disguised user-item matrix (A').

To provide CF services, the server first computes the SVD of matrix A'. As explained before, once the server computes $A'^T A'$, it can find S' and V' matrices based on $A'^T A'$ where S' and V' are estimated matrices of S and V, respectively. Each entry of $A'^T A'$ is estimated by calculating the scalar product of rows of matrix A'^T and the columns of the matrix A'. The entries other than the diagonal ones are estimated as follows:

$$(A'^T A')_{fg} = \sum_{u=1}^{n} (z_{uf} + r_{uf})(z_{ug} + r_{ug}) = \sum_{u=1}^{n} z_{uf} z_{ug}$$
$$+ \sum_{u=1}^{n} z_{uf} r_{ug} + \sum_{u=1}^{n} z_{ug} r_{uf} + \sum_{u=1}^{n} r_{uf} r_{ug} \approx \sum_{u=1}^{n} z_{uf} z_{ug} \quad (3)$$

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 ($f = g$), we can estimate them as follows:

$$(A'^T A')_{ff} = \sum_{u=1}^{n} (z_{uf} + r_{uf})(z_{uf} + r_{uf}) =$$
$$\sum_{u=1}^{n} z_{uf}^2 + 2 \sum_{u=1}^{n} z_{uf} r_{uf} + \sum_{u=1}^{n} r_{uf}^2 \approx \sum_{u=1}^{n} z_{uf}^2 + \sum_{u=1}^{n} r_{uf}^2 \quad (4)$$

Again, the expected value of $\sum_{u=1}^{n} z_{uf} r_{uf}$ is 0. However, since we only need $\sum_{u=1}^{n} z_{uf}^2$ values for diagonal entries, we need to get rid of $\sum_{u=1}^{n} r_{uf}^2$ in Eq. 4 as follows:

$$(A'^T A')_{ff} \approx \sum_{u=1}^{n} z_{uf}^2 + \sum_{u=1}^{n} r_{uf}^2 - n{\sigma_r}^2 \approx \sum_{u=1}^{n} z_{uf}^2 \quad (5)$$

where σ_r is the standard deviation of random numbers. After estimating the matrix $A'^T A'$, the server can now compute the eigenvalues from $A'^T A'$, which are used to find eigenvectors that form the matrix V'. It then finds the 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 $i = 1 \dots y$ 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:

$$b'_i(j) = {s'_i}^{-1} \sum_{l=1}^{m} (z_{jl} + r_{jl}) v'_{il} =$$
$${s'_i}^{-1} \sum_{l=1}^{m} z_{jl} v'_{il} + {s'_i}^{-1} \sum_{l=1}^{m} r_{jl} v'_{il} \approx {s'_i}^{-1} \sum_{l=1}^{m} z_{jl} v'_{il} \quad (6)$$

where $j = 1 \dots n$ and the expected value of $\sum_{l=1}^{m} r_{jl} v'_{il}$ is 0.

After estimating U', S', and V'^T from disguised data, the server forms S'_k 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 (for which item he is looking for prediction) to the server who computes 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 the result to the user u who can now calculate the p'_{uq} using Eq. 1.

5. EXPERIMENTS

We used two datasets in our experiments. Jester is a web-based joke recommendation system, developed at University of California, Berkeley [9]. The database has 100 jokes and records of 17,988 users. The ratings range from -10 to +10, and the scale is continuous. MovieLens (ML) data were collected by the GroupLens Research Project at the University of Minnesota (www.cs.umn.edu/research/Grouplens). Our ML data consists of 100,000 ratings for 1,682 movies by 943 users. Ratings are made on a 5-star scale. We used the *Mean Absolute Error (MAE)* and the *standard deviation (σ)* as criteria for accuracy analysis.

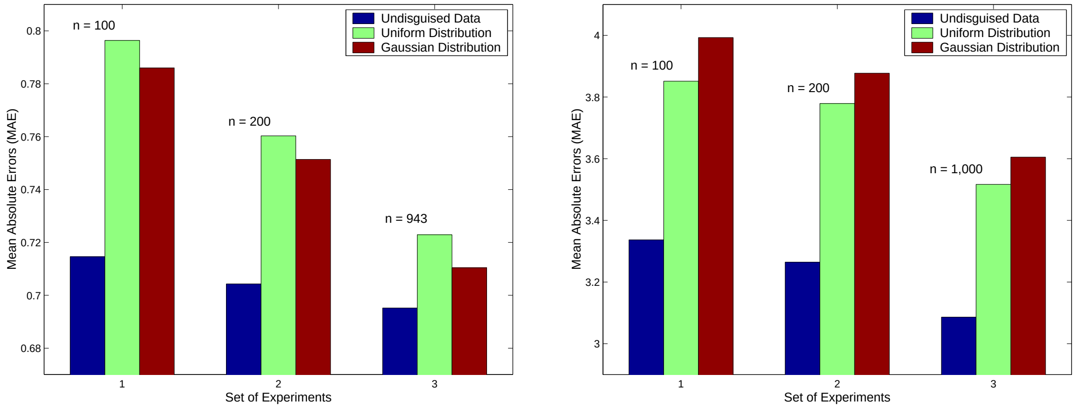
For privacy analysis, we used the method suggested in [2], which takes into account the distribution of original data. The privacy measure should indicate how closely the original value of an item can be estimated from the perturbed data. Agrawal and Aggarwal [2] propose a privacy measure based on the *differential entropy (h)* of a random variable (X). They propose $2^{h(X)}$ as a measure of privacy inherent in the random variable X and denote it by $\Pi(X)$. The average conditional privacy of X given Z is defined 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 $P(X|Z) = 1 - 2^{-I(X;Z)}$, 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 ($Z = X + R$), X has privacy $\Pi(X|Z) = \Pi(X)\big[1 - P(X|Z)\big]$.

5.1 Methodology

We filled the null entries in the user-item matrix (A) by replacing each null entry with the user mean votes for the corresponding rows. We normalized matrix A by replacing each entry with z_{uj} ($z_{uj} = (v_{uj} - \overline{v}_u)/\sigma_u$), where $\overline{v}_u$ is the user average vote and σ_u is the standard deviation of user u. Then, we created m random values r_{uj} using uniform or Gaussian distributions for each user and added them to the z-score values.

Although we used all users in ML dataset, we randomly selected 1,000 users for training from Jester dataset. 10% of the users that we used in our experiments were randomly selected as test users.

We conducted two classes of experiments in terms of available number of ratings. In the first class, we withheld a sin-



(a) MovieLens dataset (rating range: 1–5)

(b) Jester dataset (rating range: -10–10)

Figure 2: Number of users and quality of predictions

gle randomly selected rated item for each user in the test set, and tried to predict its value given all other votes the user rated (*All but 1* protocol) [5]. In the second class, we randomly selected 5 rated items from each test user as test items, and attempted to predict for those items (*All but 5* protocol). We replaced the entries for test items as null. Then we used our SVD-based scheme to predict ratings on selected items for that user. We compared the predictions that we found based on disguised data with the withheld ratings. We ran this procedure 50 times for each test user and found the MAEs and the standard deviations. Then we averaged them over all test users.

5.2 Experimental Results

To evaluate the overall performance of our scheme, we conducted several experiments. We hypothesize that privacy and accuracy depend on several factors including the total number of users (n) and items (m), the distribution and the range of perturbing data, and the total number of retained singular values (k). We found 10 to be the optimum value of k for both datasets.

5.2.1 Total Number of Users (n) and Items (m)

To show that our scheme works better with increasing n, we conducted three different sets of experiments using both datasets where we fixed the number of items to 1,682 and 100 for ML and Jester, respectively and set $k = 10$ while varying n. We used *All but 5* protocol and showed our results in Fig. 2. We created perturbing data using uniform and Gaussian distributions with $\sigma = 1$. Fig. 2 shows MAEs for undisguised data, uniform and Gaussian perturbed data for both datasets. As we expected, accuracy improves with increasing n. In Eq. 3 and Eq. 4, the scalar products are computed over n. In the long run, the sample mean and variance of perturbing data will converge to their expected values. Therefore, the accuracy of our scheme is getting better with increasing n.

We conducted experiments to show the effects of different total numbers of items and showed our results based on ML data in Table 1. We used *All but 5* protocol where we fixed n while varying m. First, we used all available items, and then we randomly selected 500 and 100 items. When we selected 100 or 500 items, we used the users who rated at least two items among those items. Because of that, there are 779 users in our third group experiments where $m = 100$. As can be seen from the table, accuracy becomes better with increasing m because the scalar product between A' and v_i' is computed over m in Eq. 6. 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.

Table 1: Number of items - prediction quality

	Data Disguise	943x1,682	943x500	779x100
MAE	Uniform	0.7229	0.7815	0.8354
	Gaussian	0.7104	0.7706	0.8015
	Undisguised	0.6952	0.7359	0.7422
σ	Uniform	0.6069	0.5919	0.6907
	Gaussian	0.6172	0.6032	0.6683
	Undisguised	0.5947	0.5683	0.5722

5.2.2 Level of Perturbation

We conducted experiments using ML data while varying the parameters of perturbing data to show how the levels of perturbation affect accuracy. We created random numbers using uniform and Gaussian distributions while varying the standard deviations. We used 943x1,682 user-item matrix. We compared the predictions based on disguised data using our scheme with the predictions on original data. Fig. 3 shows how mean absolute errors change with increasing level of perturbation. As seen from Fig. 3, the level of perturbation is critical for accuracy. The results become better with decreasing levels of perturbation. As we know, when the standard deviation is small, the randomness also becomes smaller; thus accuracy can be improved.

5.2.3 Privacy and Accuracy

To protect the private data, the level of perturbation is critical. If the level is too low, the perturbed data still discloses significant amounts of information; if it is too high, accuracy will be very low. The greater the level of perturbation, the greater the amount of privacy we have. For exam-

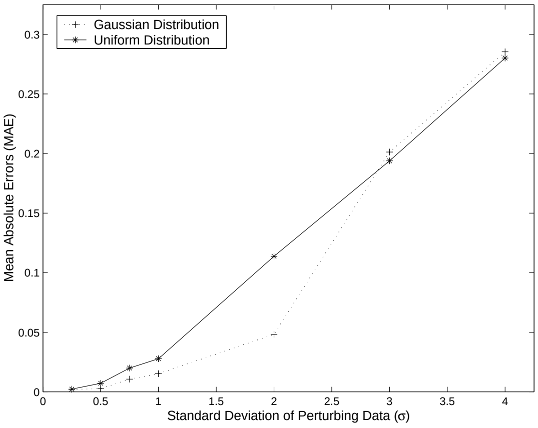


Figure 3: Level of perturbation and MAEs

ple, while $\Pi(X|Z)$ is 1.5491 when $\sigma = 0.5$, it is 2.4561 when $\sigma = 1$ for uniform distributed perturbing data. With increasing levels of perturbation, privacy loss becomes smaller. However, accuracy decreases with increasing levels of perturbation. We showed the tradeoff between privacy loss and accuracy in Fig. 4 for ML data. Although privacy levels increase with increasing levels of perturbation, accuracy becomes worse because accuracy and privacy conflict each other.

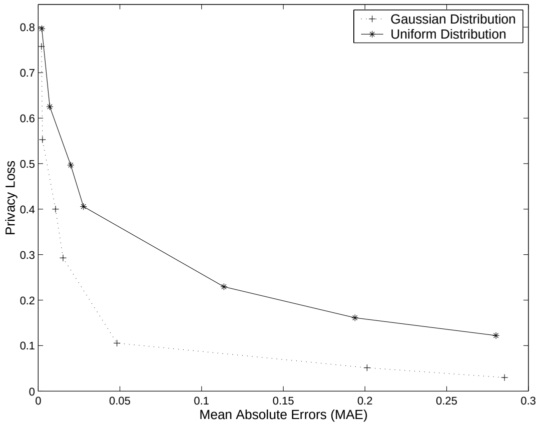


Figure 4: Privacy loss vs. accuracy

5.2.4 Summary

As can be seen from Fig. 2(a), in which we showed results for ML dataset, the MAE is 0.7964 when we use uniform perturbing data for data disguising with $\sigma = 1$ and $n = 100$. However, it is 0.0818 when we compared the predictions from disguised data with the predictions on original data where the MAE is 0.7146. Since the rating range for ML dataset is from 1 to 5, $MAE = 0.0818$ indicates our results are very close to the results generated from the original data. In Fig. 2(b), in which we showed results for Jester dataset, the MAE is 0.5145 for uniform perturbing data with $\sigma = 1$ and $n = 100$ when we compared the predictions from disguised data with the predictions on original data. Since the rating range is from -10 to 10 in Jester dataset, an error of 0.5145 is equivalent to 0.1029 in a 1– 5 scale.

6. CONCLUSIONS AND FUTURE WORK

We have presented a solution to SVD-based CF with privacy. Our solution makes it possible for servers to collect private data without greatly compromising users' privacy. Our experiments have shown that our solution can achieve accurate predictions while preserving privacy. We believe that accuracy of our scheme can be further improved if more aggregate information is disclosed along with the disguised data, especially those whose disclosure does not compromise much of users' privacy. We will study how these kinds of aggregate data disclosures affect accuracy and privacy.

7. 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 *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database System*, Santa Barbara, CA, May 21-23 2001.

[3] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD on Management of Data*, pages 439–450, Dallas, TX, May 15-18 2000.

[4] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proceedings of the 1998 Workshop on Recommender Systems*, August 1998.

[5] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, Madison, WI, July 1998.

[6] J. Canny. Collaborative filtering with privacy. In *IEEE Symposium on Security and Privacy*, pages 45–57, Oakland, CA, May 2002.

[7] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 238–245, Tampere, Finland, August 2002.

[8] L. F. Cranor, J. Reagle, and M. S. Ackerman. Beyond concern: Understanding net users' attitudes about online privacy. Technical report, AT&T Labs-Research, April 1999. Available from `http://www.research.att.com /library/trs/TRs/99/99.4.3/report.htm`.

[9] D. Gupta, M. Digiovanni, H. Narita, and K. Goldberg. Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes. In *Workshop on Recommender Systems Algorithms and Evaluation, 22nd International Conference on Research and Development in Information Retrieval*, Berkeley, CA, 2000.

[10] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. T. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 1999 Conference on Research and Development in Information Retrieval*, August 1999.

[11] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, Melbourne, FL, November 19-22 2003.

[12] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web transaction. *ACM Transactions on Information and System Security*, 1(1):Pages 66–92, 1998.

[13] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of dimensionality reduction in recommender system-a case study. In *ACM WebKDD 2000 Web Mining for E-commerce Workshop*, 2000.

[14] A. F. Westin. Freebies and privacy. Technical report, Opinion Research Corporation, July 1999. Availabe from `http://www.privacyexchange.org/iss/surveys /sr990714.html`.