Privacy-Preserving Collaborative Filtering on Vertically Partitioned Data*

Huseyin Polat and Wenliang Du

Department of Electrical Engineering and Computer Science,
Syracuse University, CST 3-114, Syracuse, NY 13244-1240, USA
{hpolat, wedu}@ecs.syr.edu

Abstract. Collaborative filtering (CF) systems are widely used by E-commerce sites to provide predictions using existing databases comprised of ratings recorded from groups of people evaluating various items, sometimes, however, such systems' ratings are split among different parties. To provide better filtering services, such parties may wish to share their data. However, due to privacy concerns, data owners do not want to disclose data. This paper presents a privacy-preserving protocol for CF grounded on vertically partitioned data. We conducted various experiments to evaluate the overall performance of our scheme.

1 Introduction

Collaborative filtering (CF) is a recent technique that helps users cope with information overload using other users' preferences. It is widely used by E-commerce, direct recommendations, and search engines [1,2]. The goal is to predict how well a user (*an active user*) will like an item that he/she did not buy before based on other users' preferences [4].

Data collected for CF purposes might be vertically partitioned between different parties where the parties hold disjoint sets of items' ratings collected from the same users. An individual's preferences for products might be split among different E-commerce companies such as Amazon.com and MovieFinder.com. Online vendors can produce better referrals if they share information about their customers with other vendors. Joint data is beneficial for E-commerce sites because customers prefer returning to stores with better referrals and they search for more products to purchase. Shared information will also benefit customers by making it more likely to receive more accurate and reliable recommendations. Combining vertically partitioned data (VPD) is helpful when CF systems have limited rated items. To find more reliable matchings and provide more accurate referrals, the overlap between users should be large enough; this might be achieved by integrating VPD. However, due to privacy concerns, data owners do not want to collaborate and disclose their data to each other.

* This work was supported by Grants ISS-0219560 and ISS-0312366 from the United States National Science Foundation.

VPD-based CF is essential and can be achieved if privacy measures are introduced to data owners. We study the privacy-preserving collaborative filtering (PPCF) on VPD problem: *To maximize mutual advantages, two online vendors (A and B) holding disjoint sets of items' ratings gathered from the same users, want to provide CF services to their future customers using the joint data. How can they provide such services using the integrated data while preserving their privacy?*

We propose a protocol to achieve PPCF on VPD. Privacy, accuracy, and efficiency are conflicting goals. Therefore, the proposed protocol should achieve a good balance between them. Our scheme consists of off-line and online computation components. We conduct some computations off-line to achieve data exchange between parties with privacy. During the online computation, a new customer (an active user a) communicates with both parties. They then perform data exchange through a with privacy. The company that does not hold ratings of the target item (the item that a is looking for a prediction) finds prediction and tells a . Since data exchanges are required whenever a new customer wants a prediction and either party can act as an active user in multiple scenarios to learn about other party's data, the proposed protocol should be secure against such attacks coming from both parties.

Canny proposes two schemes for PPCF [1,2]. A community of users can compute personalized recommendations without exposing individual data using such schemes. Polat and Du use randomized perturbation techniques for PPCF [6,7]. Vaidya and Clifton [8,9,10] present privacy-preserving methods for association rule mining, naïve Bayes classifier, and K -means clustering based on VPD. We used the CF algorithm proposed by [4]. If v_{ij} is user i 's vote on item j , and $\overline{v_i}$ and σ_i are the mean vote and the standard deviation of the user i 's ratings, respectively, then the z-scores (z_{ij}) can be defined as $z_{ij} = (v_{ij} - \overline{v_i})/\sigma_i$. Herlocker et. al find predictions as follows where n is the number of users:

$$p_{aq} = \overline{v_a} + \sigma_a \cdot \frac{\sum_{i=1}^n w_{ai} \cdot z_{iq}}{\sum_{i=1}^n w_{ai}} \quad w_{ai} = \sum_k z_{ak} \cdot z_{ik} \quad (1)$$

where k is the item set both a and the user i have rated and q is the target item. σ_a and σ_i are standard deviations of a 's ratings and i 's ratings, respectively. p_{aq} is the prediction for a on q and w_{ai} is similarity between a and i . We used homomorphic property for our proposed protocol: $E_k(x) * E_k(y) = E_k(x + y)$. Many such systems exist, and an example is the system by Paillier [5]. A useful property of homomorphic encryption schemes is that an addition operation can be conducted based on the encrypted data without decrypting them.

2 PPCF on VPD

Without privacy as a concern, data owners exchange their data to provide CF services. However, with privacy as a concern, the companies should not be able

to learn each other's data. We can write Eq. (1) as $p_{aq} = \overline{v_a} + \sigma_a \cdot P$ where P can be defined as follows:

$$P = \frac{\sum_{i=1}^n \left[\sum_k z_{ak} z_{ik} \right] z_{iq}}{\sum_{i=1}^n \sum_k z_{ak} z_{ik}} = \frac{\sum_k z_{ak} \left[\sum_{i=1}^n z_{ik} z_{iq} \right]}{\sum_k z_{ak} \left[\sum_{i=1}^n z_{ik} \right]} \quad (2)$$

CF systems can tell whether a will like q or not, rather than telling how much he/she will like it. To do this, p_{aq} is compared with a threshold (τ). If $p_{aq} \geq \tau$, q is recommended as like, otherwise it is recommended as dislike. If the ratings vary from 1 to 5, τ is set to 3.5 while it is set to 2 if they range from -10 to 10. Since A 's and B 's data is used to calculate P , Eq. (2) can be written as:

$$P = \frac{\sum_{k_A} z_{ak_A} \left[\sum_{i=1}^n z_{ik_A} z_{iq} \right] + \sum_{k_B} z_{ak_B} \left[\sum_{i=1}^n z_{ik_B} z_{iq} \right]}{\sum_{k_A} z_{ak_A} \left[\sum_{i=1}^n z_{ik_A} \right] + \sum_{k_B} z_{ak_B} \left[\sum_{i=1}^n z_{ik_B} \right]} = \frac{A_N + B_N}{A_D + B_D} \quad (3)$$

where $k = k_A + k_B$, and k_A and k_B represent the item sets both a and i have rated among the items held by A and B , respectively.

2.1 Off-Line Computation

The denominator part in Eq. (3) can be easily computed because A and B can find A_D and B_D using their own data. However, the party who does not own q needs to have z_{iq} values for all $i = 1, \dots, n$ to compute $\sum_{i=1}^n z_{ik_j} z_{iq}$ necessary for the numerator. Since A and B follow the same steps, we only explain the procedure for A . A horizontally divides its $n \times m_A$ data matrix into c_A sub-matrices where each sub-matrix consists of n/c_A users and their ratings for items held by A , where m_A is the number of items A owns. It then disguises data in each sub-matrix independently. For $i = 1, \dots, c_A$, A performs the followings:

Step 1. Permutes m_A column vectors using a permutation function Π_{Ai} .

Step 2. For $j = 1, \dots, m_A$, divides the permuted column vector $\Pi_{Ai}(I_{ij})$ into d_{ij} random vectors where $\Pi_{Ai}(I_{ij}) = \sum_{z=1}^{d_{ij}} X_{ijz}$ and d_{ij} is an integer chosen with uniform random distribution over the range $[1, \beta_A]$.

Step 3. Permutes $X_{i11}, X_{i12}, \dots, X_{i1d_{i1}}, X_{i21}, X_{i22}, \dots, X_{i2d_{i2}}, \dots, X_{im_A1}, X_{im_A2}, \dots, X_{im_Ad_{im_A}}$ random vectors found in step 2 using π_{Ai} .

Step 4. A sends D_{Ai} permuted random vectors to B where $D_{Ai} = d_{i1} + d_{i2} + \dots + d_{im_A}$. B computes the scalar products between these permuted random vectors and its m_B column vectors using the corresponding parts of them and finds $D_{Ai}m_B$ scalar product results.

Step 5. B encrypts the scalar product results using a homomorphic encryption scheme and its public key e_b and sends $D_{Ai}m_B$ encrypted values to A .

Step 6. Since A knows Π_{Ai} and π_{Ai} and homomorphic encryption is used, it finds the scalar product results of its m_A and B 's m_B column vectors in encrypted forms using homomorphic encryption property. After conducting these steps for all $i = 1, \dots, c_A$, A gets encrypted scalar product results for its all c_A sub-matrices. Since A 's data is horizontally divided, it again uses homomorphic encryption property to find the final scalar product results in encrypted forms.

A creates a matrix Σ_A consisting of $e_b(\Sigma_{ij})$ for $i = 1, \dots, m_A$ and $j = 1, \dots, m_B$ where $e_b(\Sigma_{ij})$ represents the encrypted scalar product between i^{th} column vector of B and the j^{th} column vector of A . It generates large enough v_{ij} random numbers for $i = 1, \dots, m_B$ and $j = 1, \dots, m_A$, encrypts them using e_b , and adds them to the $e_b(\Sigma_{ij})$ values using homomorphic encryption property. It finds matrix Σ'_A consisting of $e_b(\Sigma'_{ij})$ where $\Sigma'_{ij} = \Sigma_{ij} + v_{ij}$ and stores v_{ij} values in a matrix V_A . It sends Σ'_A to B that decrypts the encrypted values and finds the matrix Σ''_A consisting of Σ'_{ij} values and stores it. By following the same procedure, B finds matrices Σ_B , Σ'_B , and V_B . B stores V_B and A finds Σ''_B , and then stores it. A and B compute the item mean votes and store them in $m_A \times 1$ and $m_B \times 1$ matrices, respectively.

2.2 Online Computation

Since either party can act as an active user in multiple scenarios, online component should be secure against such attacks. The steps are as follows:

Step 1. a sends his/her data and a query to the company that owns q . Assume that B owns q . B computes B_N and B_D . However, since A can act as an active user in multiple scenarios to learn them, B uses private B_N & B_D computation protocol, which is explained in the following, to compute them.

Step 2. B can compute A'_N value using the data from the q^{th} row of the matrix Σ''_A and a 's corresponding data where $A'_N = A_N + R_q$. The data from the q^{th} row of the matrix Σ''_A represents $\sum_{i=1}^n z_{ik_A} z_{iq}$ values disguised by v_{qk_A} random numbers for all k_A . A can compute $R_q = \sum_{k_A} z_{ak_A} v_{qk_A}$ where k_A represents the items rated by a among the items held by A .

Step 3. B computes $A_N + R_q + B'_N$ and B'_D and sends them together with a 's new mean vote, standard deviation, and the z-scores for those items rated by a among items held by A to A through a . A computes R_q , finds $A_N + B'_N = A'_N + B'_N - R_q$ and A_D , and estimates P' using Eq. 3 based on the query.

A computes p'_{aq} , tells a whether he/she will like q or not by comparing p'_{aq} with τ . Since B can act as an active user, A uses a random threshold to prevent B from learning A_D and A_N . It generates a uniform random number ($r_{A\tau}$) from a range $[-\alpha_A, \alpha_A]$, finds $\tau + r_{A\tau}$, and uses it as a random threshold.

Our scheme can be extended to multi-party. Each vendor exchanges data off-line with others and stores it as in two-party scheme. During online phase, a sends his/her data to the party that owns q . That party computes the required data like it does in two-party scheme and sends results to a . Then one party acts as a master site. Other parties compute the values required for numerator and denominator parts. Each company creates a large enough uniform random num-

ber from a range $[-\gamma, \gamma]$, adds it to the values for numerator and denominator parts, and sends them through a to master site, which estimates the prediction.

Private B_N & B_D Computation Protocol. We explain the protocol only for B . After B gets a' data, it finds the number of rated items (C_B) that a rated among the items it holds. If C_B is less than $\lfloor m_B/2 \rfloor$, then B finds the items that a did not rate among the items B owns. B generates a uniform random integer S_{Ba} from the range $[1, m_B - C_B]$, randomly selects S_{Ba} unrated items among the items it owns, and fills their cells in the a 's ratings vector with their mean votes. If C_B is bigger than $\lfloor m_B/2 \rfloor$, B finds the items that a rated among the items B owns and creates a uniform random integer S_{Br} from the range $[1, C_B]$. It randomly selects S_{Br} rated items and removes their ratings from a 's ratings vector. B computes B'_N and B'_D using the new ratings vector of a and finds a ratings' new mean and standard deviation and computes the z-scores.

3 Privacy and Overhead Costs Analysis

In this section we first investigated privacy.

Claim 1. *After B gets the required data from A , the probability of guessing the A 's data for B is 1 out of $(m_A!D_A!(\beta_A)^{m_A})^{c_A}$.* Since A uses $\Pi_{Ai}s$ for all $i = 1, \dots, c_A$ to permute its m_A column vectors in each sub-matrix, for B , the probability of guessing the correct positions of them is 1 out of $m_A!$. A divides each of its permuted column vectors into random vectors where it decides how many random vectors a permuted vector be divided into based on a uniform random integer from the range $[1, \beta_A]$. Therefore, the probability of guessing the number of random vectors that each vector is divided into is 1 out of $(\beta_A)^{m_A}$. A uses $\pi_{Ai}s$ for all $i = 1, \dots, c_A$ to permute random vectors. Therefore, guessing their correct positions is 1 out of $D_A!$ with the assumption that all D_{Ai} values are same and equal to D_A . Since A horizontally divides its data into c_A parts, the probability of guessing the A 's data for B is 1 out of $(m_A!D_A!(\beta_A)^{m_A})^{c_A}$.

Claim 2. *A is not able to learn B_N and B_D due to the private B_N & B_D computation protocol.* Since B uses random integer S_{Ba} and randomly selects S_{Ba} unrated items among the items it owns, the probability of guessing the correct S_{Ba} and which S_{Ba} unrated items are selected is 1 out of $((m_B - C_B)(m_B - C_B)!)/(S_{Ba}!(m_B - C_B - S_{Ba})!)$. B also fills unrated items' cells in a 's ratings vector with their mean votes, which are only known by it.

Claim 3. *B is not able to learn A_N and A_D when it acts as an active user in numerous scenarios.* Since A tells a that he/she will like or dislike q and produces referrals using a random threshold, B will not learn A_N and A_D .

Claim 4. *A will not learn scalar product results conducted by B due to encryption and B is not able to derive $\Sigma_{ij}s$ from $\Sigma''_{ij}s$ due to random numbers.*

Unlike off-line communication cost, online communication cost is vital and the number of online communications is only 4 for our scheme. The additional storage costs due to privacy issues are $O(m_A m_B + m_A)$ and $O(m_A m_B + m_B)$ for

A and B , respectively. Although off-line computation cost is not critical, online computation cost is essential.

Claim 5. *Additional computation costs due to privacy concerns during the online phase are insignificant.*

Claim 6. *The costs due to the encryptions and the decryptions done off-line are $O(c_B D_B + m_A m_B)$ and $O(c_A D_A + m_A m_B)$ for A and B , respectively where D_A and D_B represent the average number of random vectors for each sub-matrix.*

Claim 7. *The costs due to the multiplications done off-line are $O(n m_A D_B)$ and $O(n m_B D_A)$ for A and B , respectively.*

4 Experiments

We used Jester and MovieLens (ML) data sets. Jester [3] has 100 jokes and records of 17,988 users. The ratings range from -10 to +10. ML (www.cs.umn.edu/research/GroupLens) consists of ratings for 3,592 movies made by 7,463 users. Ratings are made on a 5-star scale. We measured the accuracy of our approach using classification accuracy (CA) and F -measure (FM), which is a weighted combination of precision and recall.

4.1 Methodology

We randomly selected 2,000 users for training from Jester and ML. Since we conducted different sets of experiments with varying number of rated items (M), we found those users who rated certain number of items and randomly selected 400 test users among them for each experiment. For each test user, we randomly selected 5 rated items, withheld a single rated item for each test user, and tried to predict its value given all other ratings. We did this for all 5 test items. We replaced the test item's entry as null. We ran the selection of the subset of rated or unrated items protocol 10 times for each test item. We created r_τ uniform random numbers while varying the range to evaluate random threshold. For each test item, we created 10 uniform random numbers for those experiments testing accuracy with random threshold. We converted the withheld items' ratings into binary ratings. We then compared the recommendations on our scheme with the withheld items' converted ratings and found CA and FM values.

4.2 Experimental Results

Number of Rated Items (M). We hypothesize that since prediction quality improves with increasing M , when two parties conduct CF on the joint data, accuracy improves. To show the effects of different M values, we conducted experiments using ML data while varying M . Table 1 shows CAs and FMs with varying M . Based on the settings of each experiment, we selected those users for testing who rated M number of items. Overall performance increases with increasing M . If there is limited number of rated items, with increasing M values, we gain significant improvement. However, the improvement becomes

Table 1. Prediction Quality vs. M

M	$M < 50$	$40 < M < 100$	$100 \leq M < 200$	$200 \leq M < 400$
CA	0.6645	0.7010	0.7100	0.7140
FM	0.7313	0.7686	0.7713	0.7743

stable when enough ratings are available. CA is 0.6645 when M is less than 50 while it increases to 0.7010 when $40 < M < 100$. Besides CA, FM also increases from 0.7313 when $M < 50$ to 0.7686 when $40 < M < 100$.

Number of Removed Ratings (S_r). We conducted experiments while varying S_r using Jester and ML, and showed CAs in Fig. 1. 400 test users were randomly selected among those users who rated more than 200 and 80 items for ML and Jester, respectively. As seen from Fig. 1, accuracy slightly becomes worse with increasing S_r because the available ratings are decreasing. When we increased S_r from 0 to 100, we lost 0.0065 accuracy for ML. This means that if there are significantly large number of ratings available, removing some of them does not affect accuracy too much.

(a) ML (rating range: 1-5)

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

Fig. 1. Prediction Quality vs. S_r

Number of Appended Ratings (S_a). Since accuracy improves with increasing available ratings, appending more ratings may improve accuracy. However, since empty cells are filled with the item mean votes, which can be considered default votes for a and may not match with his/her true ratings for those items, that might make accuracy worse. We performed experiments using ML with varying S_a . 400 test users were randomly selected among those users who rated more than 40 and less than 100 items. CA improves with increasing S_a up to 60 appended ratings. When S_a is 100, CA becomes worse and it is 0.7035; but it is still better than the CA, which is 0.7010, when S_a is 0.

Range of Uniform Random Values (α). To show how different α values affect our results, we ran experiments while using uniformly created r_τ values from the range $[-\alpha, \alpha]$ with varying α values using ML. 400 test users were randomly selected among those users who rated more than 40 and less than 100 items. The results slightly become worse with increasing α because with increasing range, r_τ values become larger, the random threshold $(\tau + r_\tau)$ fluctuates more and causes a loss in the performance. When we increased α from 0 to 0.1, CA degrades by 0.0015 while FM decreases by 0.0032.

5 Conclusions and Future Work

We have presented a solution to the PPCF based on VPD problem. Our solution makes it possible for two parties to conduct filtering services using their joint data without disclosing their data to each other. Our experiment results have shown that our solution produces accurate referrals compared with the true ratings. We will study multi-party scheme in detail and show how accuracy and privacy change compared to two-party scheme and with varying number of parties.

References

1. J. Canny. Collaborative filtering with privacy. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 45–57, Oakland, CA, USA, May 2002.
2. 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, 2002.
3. 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, USA, August 1999.
4. 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.
5. P. Paillier. Public-key cryptosystems based on composite degree residue classes. In *Advances in Cryptology – EUROCRYPT’99*, pages 223–238, 1999.
6. 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, USA, November 19–22 2003.
7. H. Polat and W. Du. SVD-based collaborative filtering with privacy. In *Proceedings of the 20th ACM Symposium on Applied Computing Special Track on E-commerce Technologies*, Santa Fe, NM, USA, March 13–17 2005.
8. J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD*, pages 639–644, 2002.
9. J. Vaidya and C. Clifton. Privacy preserving k-means clustering over vertically partitioned data. In *Proceedings of the 2003 ACM SIGKDD*, Washington, DC, USA, August 24–27 2003.
10. J. Vaidya and C. Clifton. Privacy preserving naïve bayes classifier for vertically partitioned data. In *Proceedings of the 2004 SIAM Conference on Data Mining*, Orlando, FL, USA, May 2004.