Privacy-Preserving Collaborative Filtering based on Horizontally Partitioned Dataset

Arjan Jeckmans, Qiang Tang, and Pieter Hartel
Distributed and Embedded Security
University of Twente
Enschede, the Netherlands
{a.j.p.jeckmans,q.tang,pieter.hartel}@utwente.nl

***Abstract*—Nowadays, recommender systems have been increasingly used by companies to improve their services. Such systems are employed by companies in order to satisfy their existing customers and attract new ones. However, many small or medium companies do not possess adequate customer data to generate satisfactory recommendations. To solve this problem, we propose that the companies should generate recommendations based on a joint set of customer data. For this purpose, we present a privacy-preserving collaborative filtering algorithm, which allows one company to generate recommendations based on its own customer data and the customer data from other companies. The security property is based on rigorous cryptographic techniques, and guarantees that no company will leak its customer data to others. In practice, such a guarantee not only protects companies' business incentives but also makes the operation compliant with privacy regulations. To obtain precise performance figures, we implement a prototype of the proposed solution in C++. The experimental results show that the proposed solution achieves significant accuracy difference in the generated recommendations.**

***Keywords*—Recommender System; Collaborative Filtering; Privacy; Homomorphic Encryption**

I. Introduction

Web based shopping companies often provide customers with recommendations for other products that they might be interested in. Good recommendations point customers to interesting and useful products and can increase sales. Consider the following scenario: A customer has made several purchases in an online bookshop and he has also given feedback about what other books he likes and dislikes (for example on books that he has purchased elsewhere). Upon returning to the bookshop, the customer is given a personalized list of books that he may be interested in and has not purchased or rated before. Such a personalized list of interesting items is normally generated using a recommender system, which often is a collaborative filtering algorithm, that uses data from the entire customer database. Companies, which have a lot of customers, are more likely to have enough data to generate good recommendations. However, other companies do not necessarily have enough data to do so, as noted in [1]. In any case, more customer data can only lead to better recommendations.

For companies to gain access to more customer data and provide more meaningful recommendations for their customers, they can: (1) request the aid of another company which has a large customer database, or (2) collaborate with multiple other companies which contribute their relatively small customer databases to create a large one. The issue is that companies may not be able to simply share, or give each other full access to, their customer databases. This will result in an undesirable loss of control over their customer database, which is basically their main asset. In addition, privacy regulations may prohibit such data sharing activities. Companies may be suggested to rely on a third party to generate recommendations. However, this requires companies to share all their data with the third party, and is undesirable as well. The challenge is to find an efficient privacy-preserving mechanism which allows companies to generate recommendations based on their joint sets of databases, while preserving the privacy of their individual customer database respectively.

A. Contribution

There are two approaches to design collaborative filtering algorithms. One is neighborhood-based (e.g. [2]), and the other one is latent factor based (e.g. [3], [4]) which often rely on matrix factorization techniques. In practice, latent based approach is advantageous over the neighborhood-based because it allows implicit feedbacks from customers. However, due to the technical difficulty of integrating cryptographic mechanisms with latent based approach, we focus on neighborhood-based collaborative filtering algorithms in this paper.

We first propose two secure two-party protocols, namely secure absolute value protocol ABS and secure division protocol DIV. We then construct a privacy-preserving collaborative filtering algorithm for the two-company setting, where company A requests the aid of company B to get recommendations for its customer. In our solution, company A uses a homomorphic encryption scheme to hide its customer's data and share the encrypted data with company B, which computes its contributions to the final recommendations in the encrypted domain. From company B, company A only obtains aggregated and anonymized data, which however allow it to generate the top X recommendations for its customer. In the honest-but-curious model (where companies adhere to the protocol, but try to learn additional information), our solution guarantees that: (1) company A has only access to an aggregated and randomized version of company B' database; (2) company B does not learn any information about company A's customer data. We then build a prototype implementation of our solution and present performance (computation/communication costs and accuracy) results based on the prototype. We show a linear relation between the number of customers and execution time, which is the best that can be achieved. We also show a larger accuracy gain for company A as the difference in customer population between the companies increases.

There are two approaches to achieve privacy-preserving data mining. One is perturbation and anonymization based following the work of Agrawal and Srikant [5], and the other is cryptography-based following the work of Lindell and Pinkas [6]. In dealing with a large data set, the perturbation and anonymization based approach is generally efficient and flexible, however this approach usually does not provide rigorous security guarantees. For example, Narayanan and Shmatikov have demonstrated serious de-anonymization attacks against the Netflix Prize dataset [7]. Recently, McSherry and Mironov [8] applied the concept of differential privacy to recommender systems, and achieved rigorous security. However, this approach in general may reduce the computation accuracy as it will modify the original data. In contrast, the cryptography-based approach can provide rigorous security guarantees and will not affect computation accuracy, but it is often too complex to be practical. Our work demonstrates that, for (at least some) recommender algorithms, the cryptographic approach can be feasible, which means both efficiency and rigorous privacy protection can be achieved at the same time.

B. Organization

In Section II, we formally specify the recommendation scenario. In Section III, we present our solution and analyze its security. In Section IV, we report on the performance of our prototype implementation. In Section V, we review the related work, and in Section VI we conclude the paper.

II. Problem Statement and Security Model

In this section, we describe the research problem in the two-company setting, and present our security model.

A. Problem Statement

In the two-company setting, company A collaborates with company B in order to get better recommendations for its customers. We assume that company A has n' customers and company B has $n - n'$ customers, so that they have n customers in total. We further assume that customers from both companies have provide some ratings on the same set of m items. For the simplicity of description, we assume that there is no common customer between company A and company B. Let a rating be an integer from a domain $[v_{min}, v_{max}]$. The ratings of customer y, for $1 \leq y \leq n$, are denoted as a vector $V_y = (v_{y,1}, v_{y,2}, \cdots, v_{y,m})$ where $v_{y,i}$, for any $1 \leq i \leq m$, represents customer y's rating for item i. Company A holds the rating vectors V_y $(1 \leq y \leq n')$, and company B holds the rating vectors V_y $(n'+1 \leq y \leq n)$. Let the average rating of customer y be denoted by $\overline{v}_y = \frac{\sum_{i=1}^{m} v_{y,i}}{m}$. The research problem is to design a privacy-preserving collaborative filtering algorithm such that: for customer x, where $1 \leq x \leq n'$, company A can compute the top X unrated items (by customer x) with the highest predictions, which are computed from its own database and that of company B.

B. Security Model

We assume that both company A and company B are honest-but-curious, which means that they will adhere to the protocol specification but will try to infer information from the protocol execution transcripts. The rationale behind this assumption is that the companies are expected to have signed a service level agreement when engaging in a collaboration. Malicious behaviors will be deterred due to the potential monetary penalties and legal actions. Customer feedback can be used to test the validity of the recommendations. For example, when a number of customers of company A receive useless recommendations, company B might have acted maliciously.

Before describing the privacy requirements, we note an asymmetry between the roles of company A and company B: company A will make use of company B's database to generate recommendations, therefore company A will be able to learn (or, infer) some information about company B's database; on the other side, there is no need for company B to know anything about company A's database because it will not generate anything. We distinguish two cases for privacy protection.

1) Privacy of Company A: Company A should leak no information about its customer database to company B, namely company B should learn nothing from a protocol execution.

2) Privacy of Company B: We observe that, if company A learns the predictions for a customer, then it is able to recommend those with high predictions to the customer. Note that the predictions are generated based on the databases from both company A and company B. Based on this observation, we require that, in a protocol execution, company A learns only the information that can be inferred from the predictions, but nothing else.

Depending on the application scenario, the requirement for the *privacy of company B* can be enhanced. For instance, instead of learning the predictions, we can require that company A only learns the top X items with the highest predictions. Achieving such a strong privacy guarantee may result in an intolerable complexity of the solution. We leave further discussions of such specific scenarios as future work.

III. The Proposed Solution

In this section we first present the collaborative filtering algorithm in plaintext domain, and then transform the operations into the encrypted domain.

A. Recommendation without Encryption

Following the framework proposed in [2], a collaborative filtering algorithm generally operates in three steps:

1) The customer similarity computation step: the similarities between customer x and all other customers are computed based on their ratings.
2) The neighborhood selection step: the most similar customers to customer x are selected. This step aims to improve recommendation efficiency and accuracy.
3) The prediction generation step: the predictions for customer x are computed.

In this subsection, we detail the formulas that are used in our solution for each step. There is no privacy protection.

1) Computing Customer Similarity: Herlocker et al. [2] provide a comparison of different similarity weighting techniques. They conclude that the Pearson correlation is the best correlation metric to use. The formula for the Pearson correlation is given by:

$$sim_{x,y} = \frac{\sum_{i=1}^{m}(v_{x,i} - \overline{v}_x)(v_{y,i} - \overline{v}_y)}{\sqrt{\sum_{i=1}^{m}(v_{x,i} - \overline{v}_x)^2 \cdot \sum_{i=1}^{m}(v_{y,i} - \overline{v}_y)^2}} \quad (1)$$

The result of this formula $sim_{x,y}$ is the similarity between customers x and y. The range of $sim_{x,y}$ is $[-1, 1]$. For our convenience, we rewrite the formula as follows.

$$sim_{x,y} = \sum_{i=1}^{m} c_{x,i}c_{y,i} \quad (2)$$

$$c_{x,i} = \frac{v_{x,i} - \overline{v}_x}{\sqrt{\sum_{j=1}^{m}(v_{x,j} - \overline{v}_x)^2}}, c_{y,i} = \frac{v_{y,i} - \overline{v}_y}{\sqrt{\sum_{j=1}^{m}(v_{y,j} - \overline{v}_y)^2}} \quad (3)$$

Clearly, for $1 \leq i \leq m$, the range of $c_{x,i}$ or $c_{y,i}$ is $[-1, 1]$, and only V_x (V_y) is needed to compute $c_{x,i}$ ($c_{y,i}$). Define the vector $C_x = (c_{x,1}, c_{x,2}, \cdots, c_{x,m})$. Then the similarity can be computed by taking the inner product of the vectors C_x and C_y, where C_y is defined in the same way as C_x.

2) Selecting Neighborhood: Instead of selecting a neighborhood of similar customers, we select the entire customer population as the neighborhood. We make this choice because it will increase the performance in the encrypted domain. This choice results in a slightly lower accuracy for items that were already covered in the neighborhood selection scheme [2]. However, it enables us to use dissimilar customers through negative correlation and increase the the coverage.

3) Generating Predictions: To generate a recommendation, Herlocker et al. [2] suggest using a prediction algorithm that uses the deviation from mean approach to normalization. We use the following formula, introduced by Resnick et al. [9], to compute predications:

$$pred_{x,i} = \overline{v}_x + \frac{\sum_{y=1}^{n}(v_{y,i} - \overline{v}_y)sim_{x,y}}{\sum_{y=1}^{n}|sim_{x,y}|} \quad (4)$$

The result of this formula $pred_{x,i}$ is a predicted rating for item i by customer x. The range of $pred_{x,i}$ is $[2 \cdot v_{min} - v_{max}, 2 \cdot v_{max} - v_{min}]$. Since we only need the relative order of the predictions to compute the top X recommendations, we use a simplified formula, namely $pred'_{x,i} = \frac{E_{x,i}}{D_{x,i}}$, where

$$\begin{gathered} E_{x,i} = E^A_{x,i} + E^B_{x,i}, D_{x,i} = D^A_{x,i} + D^B_{x,i}, \\ E^A_{x,i} = \sum_{y=1}^{n'}(v_{y,i} - \overline{v}_y)sim_{x,y}, D^A_{x,i} = \sum_{y=1}^{n'}|sim_{x,y}|, \\ E^B_{x,i} = \sum_{y=n'+1}^{n}(v_{y,i} - \overline{v}_y)sim_{x,y}, D^B_{x,i} = \sum_{y=n'+1}^{n}|sim_{x,y}| \end{gathered} \quad (5)$$

Intuitively, company A can compute $E^A_{x,i}$ and $D^A_{x,i}$, and, given C_x, company B can compute $E^B_{x,i}$ and $D^B_{x,i}$. Together, they can compute the order of the predictions $pred'_{x,i}$.

B. Cryptographic Preliminaries

In this subsection, we first review our main cryptographic primitive, namely the Paillier encryption scheme [10], then show how to encrypt negative values.

1) Paillier Encryption: The $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ algorithms of Paillier encryption scheme [10] are as follows.

- KeyGen: This algorithm generates a tuple (N, p, q, g, λ), where p and q are two primes, $N = pq$, $\lambda = lcm(p - 1, q - 1)$, and $g \in_R \mathbb{Z}^*_{N^2}$. The private key is $SK = \lambda$, and the public key is $PK = (N, g)$.
- Enc: The ciphertext for a message $m \in \mathbb{Z}_N$ is $c = g^m r^N \bmod N^2$, where $r \in_R \mathbb{Z}_N$. For simplicity, we denote $\mathsf{Enc}(m, PK)$ as $[m]$.
- Dec: This algorithm computes the message as $m = L(c^\lambda \bmod N^2)/L(g^\lambda \bmod N^2) \mod N$, where $L(u)$ is defined as $(u - 1)/N$.

The scheme is semantically secure under the decisional composite residuosity assumption [10]. Based on the description, it is straightforward to verify that Paillier scheme possesses the following homomorphic properties.

$$[m_1] \cdot [m_2] = [m_1 + m_2], \ \ ([m_1])^{m_2} = [m_1 \cdot m_2].$$

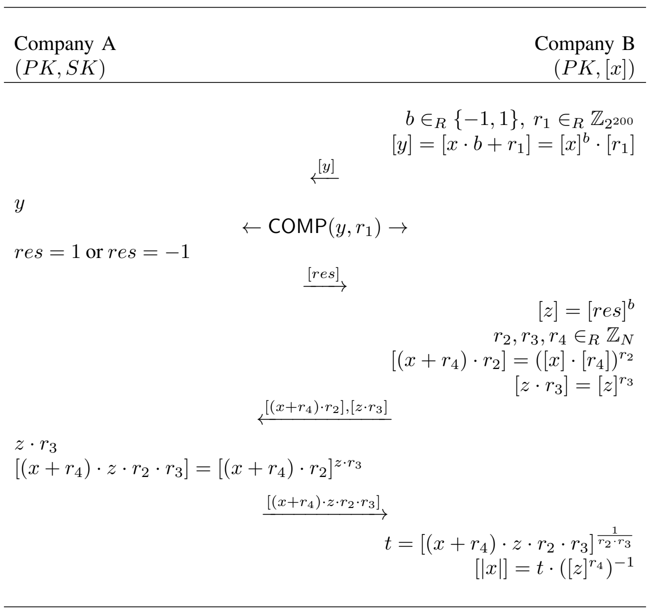
2) Encrypting Negative Integers: To represent negative integers we make use of the cyclic property of the cryptosystem. The top half of the message space will represent negative numbers. When the message space is $m \in \mathbb{Z}_N$, we represent $-m$ by $N - m$, as $N - m \equiv -m \pmod{N}$. We have to be careful of overflows so that a negative number does not suddenly become a positive number or vice versa.

C. Cryptographic Sub Protocols

In this subsection, we describe the sub protocols for secure comparison, secure absolute value, and secure division.

1) Secure Comparison Sub Protocol: The secure comparison protocol, denoted by $\mathsf{COMP}(x, y)$, is run between company A and company B, where company A has x and company B has y. At the end, company A should learn 1 if $x \geq y$ and -1 otherwise while company B learns nothing. Since Yao [11], a lot of solutions have been proposed, including [12], [13], [14], [15]. In this paper, we use that of Veugen [15].

2) Secure Absolute Value Sub Protocol: The protocol, shown in Fig. 1, is run between company A and company B, where company A has a Paillier key pair (PK, SK) and company B has $[x]$. We require that $-2^{50} \leq x \leq 2^{50}$. At the end, company B should learn $[|x|]$ while company A learns nothing. Let the protocol be denoted by $\mathsf{ABS}(x)$.



Company A
(PK, SK)
Company B
$(PK, [x])$
$b \in_R \{-1, 1\},\ r_1 \in_R \mathbb{Z}_{2^{200}}$
$[y] = [x \cdot b + r_1] = [x]^b \cdot [r_1]$
$\xleftarrow{[y]}$
y
$\leftarrow \mathsf{COMP}(y, r_1) \rightarrow$
$res = 1$ or $res = -1$
$\xrightarrow{[res]}$
$[z] = [res]^b$
$r_2, r_3, r_4 \in_R \mathbb{Z}_N$
$[(x + r_4) \cdot r_2] = ([x] \cdot [r_4])^{r_2}$
$[z \cdot r_3] = [z]^{r_3}$
$\xleftarrow{[(x+r_4)\cdot r_2], [z \cdot r_3]}$
$z \cdot r_3$
$[(x + r_4) \cdot z \cdot r_2 \cdot r_3] = [(x + r_4) \cdot r_2]^{z \cdot r_3}$
$\xrightarrow{[(x+r_4)\cdot z \cdot r_2 \cdot r_3]}$
$t = [(x + r_4) \cdot z \cdot r_2 \cdot r_3]^{\frac{1}{r_2 \cdot r_3}}$
$[|x|] = t \cdot ([z]^{r_4})^{-1}$

Figure 1. Secure ABS Sub Protocol

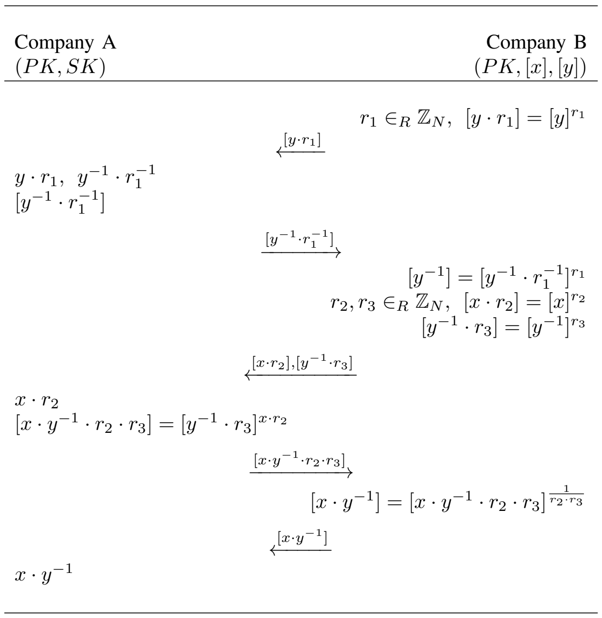
In more detail, the protocol acts as follows:

1) Company B selects $b \in_R \{-1, 1\}$, $r_1 \in_R \mathbb{Z}_{2^{200}}$. The domain of r_1 is chosen in such a way that it can hide x with statistical security. Then, company B computes $y = x \cdot b + r_1$ and sends $[y]$ to company A.
2) Company A decrypts y and runs the secure comparison sub protocol with company B who has r_1. Company A obtains res and sends $[res]$ to company B.
3) Company B computes $[z] = [res]^b$. Clearly, $z = 1$ if $x > 0$ and $z = -1$ if $x < 0$. It then selects $r_2, r_3, r_4 \in_R \mathbb{Z}_N$, and sends $[(x + r_4) \cdot r_2]$ and $[z \cdot r_3]$ to company A.
4) Company A decrypts $[z \cdot r_3]$, and sends $[(x + r_4) \cdot r_2]^{z \cdot r_3}$ to company B.
5) Company B computes $[|x|] = [(x + r_4) \cdot z \cdot r_2 \cdot r_3]^{\frac{1}{r_2 \cdot r_3}} \cdot ([z]^{r_4})^{-1}$.

3) Secure Division Sub Protocol: The protocol, shown in Fig. 2, is run between company A and company B, where company A has a Paillier key pair (PK, SK) and company B has $[x]$ and $[y]$. We assume $y \neq 0$. At the end, company A should learn $\frac{x'}{y'}$ while company B learns nothing, where $\frac{x'}{y'} = \frac{x}{y}$ and $GCD(x', y') = 1$. Note that the equation $\frac{x'}{y'} = \frac{x}{y}$ holds in the integer domain instead of $\mathbb{Z}_N$.

In more detail, the protocol acts as follows:

1) Company B selects $r_1 \in_R \mathbb{Z}_N$ and sends $[y \cdot r_1]$ to company A.



Company A
(PK, SK)
Company B
$(PK, [x], [y])$
$r_1 \in_R \mathbb{Z}_N,\ [y \cdot r_1] = [y]^{r_1}$
$\xleftarrow{[y \cdot r_1]}$
$y \cdot r_1,\ y^{-1} \cdot r_1^{-1}$
$[y^{-1} \cdot r_1^{-1}]$
$\xrightarrow{[y^{-1} \cdot r_1^{-1}]}$
$[y^{-1}] = [y^{-1} \cdot r_1^{-1}]^{r_1}$
$r_2, r_3 \in_R \mathbb{Z}_N,\ [x \cdot r_2] = [x]^{r_2}$
$[y^{-1} \cdot r_3] = [y^{-1}]^{r_3}$
$\xleftarrow{[x \cdot r_2], [y^{-1} \cdot r_3]}$
$x \cdot r_2$
$[x \cdot y^{-1} \cdot r_2 \cdot r_3] = [y^{-1} \cdot r_3]^{x \cdot r_2}$
$\xrightarrow{[x \cdot y^{-1} \cdot r_2 \cdot r_3]}$
$[x \cdot y^{-1}] = [x \cdot y^{-1} \cdot r_2 \cdot r_3]^{\frac{1}{r_2 \cdot r_3}}$
$\xleftarrow{[x \cdot y^{-1}]}$
$x \cdot y^{-1}$

Figure 2. Secure DIV Sub Protocol

2) Company A decrypts $[y \cdot r_1]$ and inverts it to obtain $y^{-1} \cdot r_1^{-1}$. It then sends $[y^{-1} \cdot r_1^{-1}]$ to company B.
3) Company B computes $[y^{-1}] = [y^{-1} \cdot r_1^{-1}]^{r_1}$. It then selects $r_2, r_3 \in_R \mathbb{Z}_N$ and sends $[x \cdot r_2]$ and $[y^{-1} \cdot r_3]$ to company A.
4) Company A decrypts $[x \cdot r_2]$ and sends $[x \cdot y^{-1} \cdot r_2 \cdot r_3]$ to company B.
5) Company B computes $[x \cdot y^{-1}] = [x \cdot y^{-1} \cdot r_2 \cdot r_3]^{\frac{1}{r_2 \cdot r_3}}$, and sends $[x \cdot y^{-1}]$ to company A.
6) Company A decrypts to retrieve $x \cdot y^{-1}$, which is in the domain of $\mathbb{Z}_N^*$. Suppose $-T \leq x, y \leq T$ and $T << N$, then company A can build a list of pairs $(x \cdot y^{-1}, \frac{x'}{y'})$, where $\frac{x'}{y'} = \frac{x}{y}$ and $GCD(x', y') = 1$. Company A then looks up the list and obtains $\frac{x'}{y'}$.

D. Recommendation with Encryption

As specified in Section II-A, company A's customer database size is n' and company B's database size is $n - n'$. For customer x, the ratings are $v_{x,i}$ $(1 \leq i \leq m)$, where $v_{x,i} = 0$ means that the customer has not rated i. We assume that company A creates a Paillier key pair by running KeyGen and the bit-length of modulo N is 1024.

1) Scaling, Rounding, and Inner Product: Paillier cryptosystem deals with encryption/decryption of integers, however, in the recommender system we work with non-integer values. Therefore, in the rest of this paper, we assume that the values of $c_{x,i}$ and $c_{y,i}$, for all x, y, i, have been scaled by 100 and rounded to integers. In addition, when computation is done with respect to Equation (5), we assume company A and company B have already scaled the values $v_{y,i} - \overline{v}_y$ by 100 and rounded the results, for all y, i.

For our recommender algorithm, the basic operation required is an inner product between two vectors, say C_x and C_y, with different data owners. Given an encrypted vector $[C_x]$ (meaning each element of the vector is encrypted $[c_{x,i}]$) and an unencrypted vector C_y, anyone can compute the encrypted similarity $[sim_{x,y}]$ following Equation (2):

$$[sim_{x,y}] = [\sum_{i=1}^{m} c_{x,i}c_{y,i}] = \prod_{i=1}^{m}[c_{x,i}c_{y,i}] = \prod_{i=1}^{m}[c_{x,i}]^{c_{y,i}} \quad (6)$$

2) Privacy-Preserving Recommendation Generation: If customer x requires recommendations, company A and company B engage in the protocol shown in Fig. 3.

$$\begin{array}{lcr}
\text{Company A} & & \text{Company B} \\
(x, V_y, 1 \leq y \leq n') & & (V_y, n'+1 \leq y \leq n) \\
(PK, SK) = \mathsf{KeyGen}(1^k) & & \\
\hline
sim_{x,y}, 1 \leq y \leq n' & & \\
\text{For } 1 \leq i \leq m, \text{ compute:} & & \\
[c_{x,i}],\ [E^A_{x,i}],\ [D^A_{x,i}] & & \\
 & \xrightarrow{[c_{x,i}],[E^A_{x,i}],[D^A_{x,i}]\ (1 \leq i \leq m)} & \\
 & & [sim_{x,y}] = \prod_{i=1}^{m}[c_{x,i}]^{c_{y,i}}, n'+1 \leq y \leq n \\
 & & [E^B_{x,i}] = \prod_{y=n'+1}^{n}[sim_{x,y}]^{v_{y,i}-\overline{v}_y}, 1 \leq i \leq m \\
 & \leftarrow \mathsf{ABS}([sim_{x,y}]), n'+1 \leq y \leq n \rightarrow & \\
 & & \text{For } 1 \leq i \leq m, \text{ compute:} \\
 & & [D^B_{x,i}] = \prod_{y=n'+1}^{n}[|sim_{x,y}|] \\
 & & [E_{x,i}] = [E^A_{x,i}] \cdot [E^B_{x,i}] \\
 & & [D_{x,i}] = [D^A_{x,i}] \cdot [D^B_{x,i}] \\
 & \leftarrow \mathsf{DIV}([E_{x,i}], [D_{x,i}]), 1 \leq i \leq m \rightarrow & \\
pred'_{x,i}, 1 \leq i \leq m & &
\end{array}$$

Figure 3. Collaborative Filtering in Two-Company Setting

In more detail, the protocol is detailed as follows.

1) Company A computes the Pearson correlations, namely $sim_{x,y}$ $(1 \leq y \leq n', y \neq x)$, between customer x and all other customers in its own database. For $1 \leq i \leq m$, company A computes $[c_{x,i}]$, $[E^A_{x,i}]$ and $[D^A_{x,i}]$ according to Equations (3) and (5), and sends them to company B. Note that the encryption is done with PK.
2) Company B computes $[sim_{x,y}]$ following Equation (6) for $n'+1 \leq y \leq n$, and then computes $[E^B_{x,i}]$ for $1 \leq i \leq m$ following Equation (5). Company B then runs the ABS sub protocol with company A to obtain $[|sim_{x,y}|]$ for $n'+1 \leq y \leq n$. Company B uses $[|sim_{x,y}|]$ $(n'+1 \leq y \leq n)$ to compute $[D^B_{x,i}] = \prod_{y=n'+1}^{n}[|sim_{x,y}|]$ for $1 \leq i \leq m$. Company B computes $[E_{x,i}] = [E^A_{x,i}] \cdot [E^B_{x,i}]$ and $[D_{x,i}] = [D^A_{x,i}] \cdot [D^B_{x,i}]$.
3) Company A and company B run the DIV protocol for company A to retrieve $pred'_{x,i}$ for $1 \leq i \leq m$. Company A then chooses the top X predicted items among the unrated ones, and sends them to customer x.

E. Security Analysis

We briefly analyse the privacy properties of the protocol, and leave detailed analysis in a future version of this paper. The sub protocols in Section III-C2 and III-C3 are secure. Intuitively, in the ABS sub protocol in Section III-C2, company A learns nothing about x because of the randomization resulted from b, r_1, r_2, r_3, r_4 and company B learns nothing about x because everything is encrypted under company A's public key. In particular, we let $r_1 \in_R \mathbb{Z}_{2^{200}}$, so that r_1 can statistically hide x from company A. At the same time, r_1 will not cause x to overflow, which would lead to an incorrect result. Intuitively, in the DIV sub protocol in Section III-C3, company A learns nothing about x, y due to the randomization resulted from r_1, r_2, r_3 and company B learns nothing about x, y because everything is encrypted under company A's public key.

Based on the security of sub protocols in Section III-C1, III-C2 and III-C3, the recommendation algorithm in Section III-D2 is secure with respect to the security model in Section II-B. Given the security of the sub protocols, the algorithm is secure for company A because everything sent to company B is encrypted under the public key PK. Similarly, the algorithm is secure for company B based on the security of the ABS and DIV sub protocols. As all information that company B sends to company A is in the sub protocols. Here, we have a minor note on using DIV sub protocol in the recommendation algorithm in Section III-D2. If $D_{x,i} = 0$ for any $1 \leq i \leq m$, then the DIV protocol will not work. Note the fact that $D_{x,i} = 0$ means that the similarities $|sim_{x,y}| = 0$ for all $1 \leq y \leq n$ that rated i, which can be assumed to be negligible due to the randomness in customers' ratings and the size of customer population. Our implementation in Section IV partially validates this assumption. Should this assumption be untrue, company A would be unable to generate a prediction for item i. But, this would also happen in the unsecured version of the protocol.

Note that the security model is focussed on the privacy of the companies' information. The protocol does not prevent unreliable recommendations. As the companies have access to the customers ratings and recommendations, nothing changes compared to an unsecured single company recommender. Regular methods for preventing unreliable recommendations apply. To prevent unreliable recommendations within the collaboration between the companies, a company can look at recommendation differences between: (1) two different runs of the protocol with the same input (2) a run of the protocol and the recommendations without collaboration. If there are significant differences, this can be an indication of unreliable recommendations.

IV. Performance Evaluation

We have created a prototype implementation in C++. The prototype uses the GNU Multiple-Precision (GMP) library and consist of roughly 750 lines of code. To test this prototype, we use the MovieLens dataset (http://grouplens.org/), which contains 1 million ratings for 3900 movies by 6040 users. The ratings are on an integer scale from 1 to 5. We split the rating dataset in two parts by randomly selecting users as either a customer of company A or company B. All tests are carried out on an Intel Xeon at 3GHz, with 2GB of RAM.

A. Computation Cost

Referring to the proposed protocol, the computational complexity is related to both the number of customers of company B and number of items. Theoretically, the computational complexities is $O(m(n-n'))$ for both companies. To obtain concrete numbers for running time, we investigate two cases, in which the total number of items is fixed.

1) Case 1: In this case, we want to investigate the running time with respect to the total customer population. We take a fixed population distribution as example, where company A has 20% of the total population and company B has 80% of the total population.

We compute the running time values for ten different total populations, namely $604 \times i$ for $1 \le i \le 10$. The running time figures are shown in Fig. 4, where the x-axis denotes the total customer population and the y-axis denotes the running time. The solid line indicates the total running time for company A and company B, while the dashed line indicates only the running time for company A. As expected, the graph shows a linear relation between the number of customers and the running time of the algorithm. The running time for both company A and B individually increases linearly with the customer population. When the total population is 6040 (the full dataset), the running time is 354 seconds.

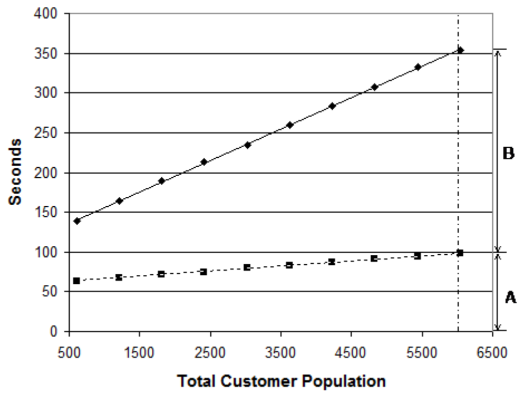


Figure 4. Running Time w.r.t. Total Population

2) Case 2: In this case, we want to investigate the running time with respect to the population distribution between company A and company B. The total population is 6040.

We compute the running time values for eight different population proportions for company A, namely 1%, 2%, 5%, 10%, 20%, 30%, 40%, 50%. The running time figures are shown in Fig. 5, where the x-axis denotes the population distribution and the y-axis denotes the running time. Again, the solid line indicates the total running time and the dashed line indicates the running time for company A. The dashed vertical line shows the intersection with Fig. 4. In particular, when company A only has 1% of the population the running time is 414 seconds, when company A is given 50% of the population the running time is 270 seconds. As stated in Case 1, the running time for company A and B increases linearly with the customer population of company B. As expected, we see a linear relationship between the running time and the distribution of the customers.

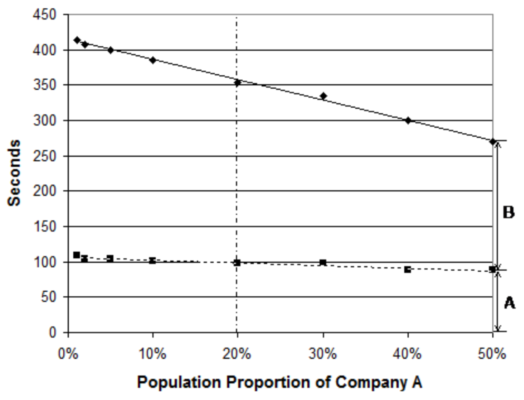


Figure 5. Running Time w.r.t. Population Distribution

B. Communication Cost

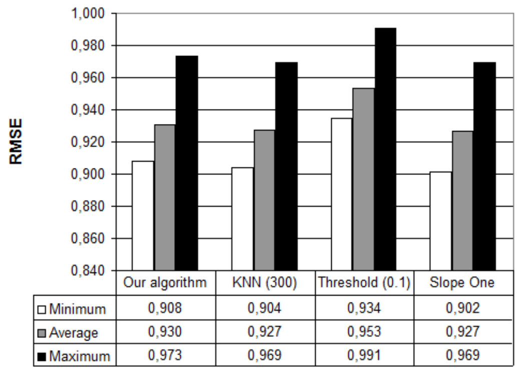
The communication cost between company A and company B is proportional to the number of items and the number of customers held by company B. We give an estimate of the communication cost for the scenario where the full customer population is split between company A and B at a 20% 80% ratio. Given the composite number N of 1024 bits, an encryption, which is essentially a number modulo N^2, has a maximum size of 2048 bits. When adding everything up this results in a transmission of 15.2 MB of data.

C. Recommendation Accuracy

To evaluate the overall accuracy property of the protocol on the whole customer population (namely, 6040 customers), we randomly selects 10% of the 6040 customers, denoted as $\mathcal{S}$, and remove 5 ratings for each of them. The removed ratings are used to compare against their predictions to determine the accuracy. We use root mean squared error (RMSE) as the accuracy measure:

$$RMSE = \sqrt{\frac{1}{t} \sum_{x \in \mathcal{S}, 1 \le i \le m} (pred_{x,i} - v_{x,i})^2}, \qquad (7)$$

where t is the total number of predictions. Note that in our case, the prediction formula $pred'_{x,i}$ is only used to predict the ordering of items, we use the formula for $pred_{x,i}$, defined in Equation (4), to normalize back to predictions of actual ratings. This formula will give the same ordering results and meaningful accuracy figures. The computation shows an average RMSE of 0.930. This is compared against k-nearest neighbors, threshold neighborhood, and slope one prediction in Fig. 6. The results show that our algorithm is comparable to these established recommendation methods.



	Our algorithm	KNN (300)	Threshold (0.1)	Slope One
□ Minimum	0,908	0,904	0,934	0,902
▨ Average	0,930	0,927	0,953	0,927
■ Maximum	0,973	0,969	0,991	0,969

Figure 6. Accuracy Comparison with Established Algorithms

To evaluate the recommendation accuracy gained by company A, we consider eight cases, where the population proportion of company A in the whole population are 1%, 2%, 5%, 10%, 20%, 30%, 40%, 50% respectively. When the proportions are 1%, 2%, 5%, 10%, we let company A's customers be from the set $\mathcal{S}$. When the proportions are 20%, 30%, 40%, 50%, we let company A's customers consist of all customers from the set $\mathcal{S}$ plus customers from the rest of the whole population. For every case, we compute the RMSE value for company A, where the computation is only based on company A's customer data, and compute a difference by subtracting the RMSE value based on the data of the whole population. The RMSE differences of all eight cases are shown in Fig. 7, where the x-axis denotes the population distribution and the y-axis denotes the RMSE difference. From the figure, the accuracy difference decreases when company A's population proportion increases. This implies that the accuracy gain becomes less when company A has more customers.

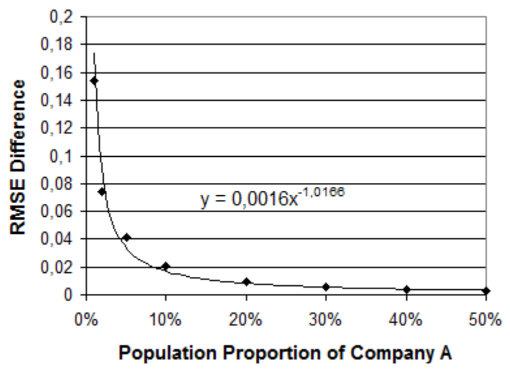


Figure 7. Accuracy Change w.r.t. Population Distribution

D. Improving the Efficiency

We foresee two ways to reduce the computation costs of the proposed algorithm. Instead of using the data from all customers, company B can use those from a subset of its customers. For example, those customers that have rated a lot of items. By using a subset of the customers, company B has to compute fewer similarities between customers and has fewer entries to compute the prediction with. When company B uses a dense subset of the customers, the combined dataset will also have a higher density. In this case, we will expect the recommendation accuracy, compared to the accuracy computed from the full set of B's customers, will show only a very small difference. As to the computational efficiency, both companies will be much more efficient because $[sim_{x,y}]$ and $[|sim_{x,y}|]$, for any y not in the subset, will not be computed. In summary, the performance will depend on how to choose the subset by company B, and we leave it as a future work to perform further investigation.

In the proposed algorithm, if company A discloses the items that customer x has rated, then the computation becomes much less complex. This is reasonable because it does not make sense to make a prediction for an already rated item. Note that company A does not disclose the ratings for the rated items. In more details, for any item i which has been rated by the customer, the values $[E_{x,i}]$, $[D_{x,i}]$, their components, and their division $pred_{x,i}$ do not have to be computed. Furthermore, for company B, computing $[sim_{x,y}]$ can be done faster. In summary, company A can sacrifice a bit of the privacy of its customer for a better computational performance.

V. Related Work

In the literature, the most relevant work to ours is that of Basu et al. [16], [17] and that of Polat and Du [1]. Basu et al. proposed a privacy preserving version of the slope one predictor. They pre-compute the deviation and cardinality matrices under encryption and make the cardinality matrix public. Then the prediction for a single item can be computed under encryption and all parties collaborate to decrypt the result. Making the cardinality matrix public in the case of two parties will leak information. Furthermore, their timing information is based on a single prediction for a single customer and item. When predicting the top X recommendations, this timing information has to be increased proportional to the number of items in the database (predictions can be computed in parallel). The setting for Polat and Du [1] is slightly different: a customer, who is not a member of either company, wants company A and B to compute recommendations for him/her. This customer plays an active role in the protocol and privacy is based on randomizing values, rather than encryption. Another difference in their work is that a rating is either 0 or 1, which makes protocol design easier than our case.

Some other privacy-preserving recommender algorithms focus on the privacy of individual customers. Aïmeur et al. [18] provided a framework where customer data are separately stored over two parties, where an agent has access to ratings and the company has access to the items so that they together can generate recommendations for customers. Polat and Du [19] proposed a singular value decomposition predictor based on random perturbation of data. They go on to show the impact on privacy and accuracy, and their inherent tradeoff due to perturbation. Berkovsky et al. [20] proposed to combine random perturbation with a peer-to-peer structure to create a form of dynamic random perturbation. For each request, the customer can decided what data to reveal and how much protection is put on the data. Different perturbation strategies are compared based on accuracy and perceived privacy. The requirement for a peer-to-peer structure makes this approach less suitable for our scenario, where only two parties are involved. McSherry and Mironov [8] proposed collaborative filtering algorithms in the differential privacy framework. Similar to other perturbation and anonymization based approaches, this approach still has a tradeoff between privacy and accuracy. In our approach, we preserve privacy without decreasing accuracy.

Canny [21], [22] uses homomorphic encryption to privately compute intermediate values of the collaborative filtering process. These intermediate values are made public and used in singular value decomposition and factor analysis, which leads to recommendations. However, because the intermediate values are made public, this leaks a lot of information about the customers when all data is held by only two parties (as is our case). Erkin et al. [23] proposed a collaborative filtering algorithm based on homomorphic cryptosystems. This algorithm requires every customer to take part in the protocol execution in order to compute recommendations for a single customer, and this makes the solution unscalable in practice.

VI. Conclusion

We have proposed a privacy-preserving collaborative filtering algorithm for companies to compute recommendations based on a joint set of customer databases. Based on the experiment results from a prototype implementation, we have shown that an individual company can generate more accurate recommendations. The complexity analysis shows that it takes about six minutes to computes recommendations for a customer using a PC. Notice that company A can pre-compute the encryptions and company B can perform most of the computations in a parallel manner. Therefore, the performance can be significantly improved in practical deployment, and the solution is in fact feasible. This paper leaves many lines of future work. First of all, it is interesting to investigate stronger privacy definitions for specific scenarios, and to compare the implications of cryptographic privacy definitions and other definitions such as differential privacy. Secondly, it is interesting to investigate methods of improving efficiency of the proposed algorithm, as mentioned in Section IV-D. Thirdly, it is interesting to consider the situation where companies possess a vertically partitioned dataset. This case seems to have more privacy concerns because we somehow need to link the records in different customer databases.

References

[1] H. Polat and W. Du, "Privacy-preserving top-n recommendation on distributed data," *Journal of the American Society for Information Science and Technology*, vol. 59, pp. 1093–1108, 2008.

[2] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999, pp. 230–237.

[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, pp. 30–37, 2009.

[4] J. D. M. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 713–719.

[5] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 439–450.

[6] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Advances in Cryptology, CRYPTO – 00*, 2000, pp. 36–54.

[7] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, 2008, pp. 111–125.

[8] F. McSherry and I. Mironov, "Differentially private recommender systems: building privacy into the netflix prize contenders," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 627–636.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994, pp. 175–186.

[10] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology, EUROCRYPT – 99*, 1999, pp. 223–238.

[11] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1982, pp. 160–164.

[12] J. Garay, B. Schoenmakers, and J. Villegas, "Practical and secure solutions for integer comparison," in *Public Key Cryptography 2007*. Springer, 2007, pp. 330–342.

[13] V. Kolesnikov, A.-R. Sadeghi, and T. Schneider, "Improved garbled circuit building blocks and applications to auctions and computing minima," in *Cryptology and Network Security (CANS)*, 2009, pp. 1–20.

[14] F. Kerschbaum, D. Biswas, and S. d. Hoogh, "Performance comparison of secure comparison protocols," in *Proceedings of the 20th International Workshop on Database and Expert Systems Application*, 2009, pp. 133–136.

[15] T. Veugen, "Comparing encrypted data," http://isplab.tudelft.nl/sites/default/files/Comparingf, 2011.

[16] A. Basu, H. Kikuchi, and J. Vaidya, "Privacy-preserving weighted slope one predictor for item-based collaborative filtering," in *Proceedings of the international workshop on Trust and Privacy in Distributed Information Processing*, 2011.

[17] A. Basu, J. Vaidya, and H. Kikuchi, "Efficient privacy-preserving collaborative filtering based on the weighted slope one predictor," *Journal of Internet Services and Information Security (JISIS)*, vol. 1, no. 4, pp. 26–46, 11 2011.

[18] E. Aïmeur, G. Brassard, J. Fernandez, and F. Mani Onana, "Alambic: a privacy-preserving recommender system for electronic commerce," *International Journal of Information Security*, vol. 7, pp. 307–334, 2008.

[19] H. Polat and W. Du, "Svd-based collaborative filtering with privacy," in *Proceedings of the 2005 ACM symposium on Applied computing*, 2005, pp. 791–795.

[20] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci, "Enhancing privacy and preserving accuracy of a distributed collaborative filtering," in *Proceedings of the 2007 ACM conference on Recommender systems*, 2007, pp. 9–16.

[21] J. Canny, "Collaborative filtering with privacy," in *IEEE Symposium on Security and Privacy*, 2002, pp. 45–57.

[22] ——, "Collaborative filtering with privacy via factor analysis," in *Proceedings of the 25th annual international conference on Research and development in information retrieval*, 2002, pp. 238–245.

[23] Z. Erkin, M. Beye, T. Veugen, and R. Lagendijk, "Efficiently computing private recommendations," in *International Conference on Acoustic, Speech and Signal Processing-ICASSP*, 2011, pp. 5864–5867.