Privacy-preserving Hybrid Recommender System

Qiang Tang
Luxembourg Institute of Science and Technology
5, Avenue des Hauts-Fourneaux
L-4362, Esch-sur-Alzette, Luxembourg
tonyrhul@gmail.com

Husen Wang
Luxembourg Institute of Science and Technology
5, Avenue des Hauts-Fourneaux
L-4362, Esch-sur-Alzette, Luxembourg
wanghs.thu@gmail.com

ABSTRACT

Privacy issues in recommender systems have attracted the attention of researchers for many years. So far, a number of solutions have been proposed. Unfortunately, most of them are far from practical as they either downgrade the utility or are very inefficient. In this paper, we aim at a more practical solution, by proposing a privacy-preserving hybrid recommender system which consists of an incremental matrix factorization (IMF) component and a user-based collaborative filtering (UCF) component. The IMF component provides the fundamental utility while it allows the service provider to efficiently learn feature vectors in plaintext domain, and the UCF component improves the utility while allows users to carry out their computations in an offline manner. Leveraging somewhat homomorphic encryption (SWHE) schemes, we provide privacy-preserving candidate instantiations for both components. Our experiments demonstrate that the hybrid solution is much more efficient than existing solutions.

CCS Concepts

•Security and privacy → Privacy-preserving protocols;

Keywords

Recommender, Homomorphic Encryption, Privacy

1. INTRODUCTION

Recommender system predicts the preferences that users would give to an item, so that it enables users to make the most appropriate choices from the immense variety of items that are available. Today, recommender systems play an important role in every corner of our daily life. Generally speaking, two representative types of recommender systems are neighborhood-based and model-based. With a neighborhood-based recommender system, in order to predict a user Alice's rating for an item i, the system first chooses a neighborhood for Alice or the item i then computes the prediction based on data from the neighborhood. With a model-based recommender system, the system first trains a model using all available data then computes the predictions based on the model. Due to their respective (dis)advantages, recommender service providers often adopt a hybrid approach, which combines neighborhood-based method and model-based method in different ways [Burke 2002].

In real-world deployment, most existing recommender systems are centralized in the sense that a service provider will collect the inputs from all users and compute recommendations for them. The collected data range from explicit inputs such as ratings to implicit behavior data such as browsing histories and locations. This makes recommender systems very privacy invasive to individual users. For instance, Weinsberg et al. [Weinsberg et al. 2012] demonstrated that what has been rated by a user can already potentially help an attacker identify this user. Calandrino et al. [Calandrino et al. 2011] pointed out inference attacks which allow an attacker with some auxiliary information to infer a user's transactions from temporal changes in the public outputs of a recommender system. Today, privacy has become a troublesome issue for both the service provider and end users.

1.1 Related Work

In the past decade, researchers have actively investigated privacy-preserving recommender systems. Existing solutions can be categorized into two groups. The cryptographic solutions (e.g. [Canny 2002, Kim et al. 2016, Nikolaenko et al. 2013, Veugen et al. 2015]) often aim at securing the procedure of underlying recommender protocols, by using cryptographic tools such as homomorphic encryption schemes and zero-knowledge proof protocols. The data-obfuscation solutions (e.g. [Berlioz et al. 2015, Guerraoui et al. 2015, Liu et al. 2015, McSherry and Mironov 2009]) adopt the concept of differential privacy and rely on adding noise to the original data or computation results to restrict the information leakage from recommender outputs. These two types of solutions are somehow complementary as they try to prevent information leakage from different sources.

Due to the large underlying user population, neighborhood selection and model training procedures often make the cryptographic solutions very inefficient even with efficient cryptographic building blocks. Some cryptographic solutions assume more than one semi-trusted servers. A serious risk for these solutions (e.g. [Kim et al. 2016, Nikolaenko et al. 2013]) is that if one server is compromised then all users' private data may be completely exposed because the data is protected by a single key from this server. The

SCC'17, April 02 2017, Abu Dhabi, United Arab Emirates
 ISBN 978-1-4503-4970-3/17/04…$15.00
DOI: http://dx.doi.org/10.1145/3055259.3055268

obfuscation-based solutions usually assume a fully trusted third party, which is responsible for calibrating noises into the computation. Technically, this third party can be replaced by a secure multiparty computation protocol. But, the cost can be too high to be practical. Moreover, even though differential privacy is a mathematically rigorous concept, it has many limitations in practical usage, e.g. [Dankar and Emam 2013, Haeberlen et al. 2011]. It is worth noting that there exist relatively more efficient cryptographic solutions (e.g. [Jeckmans et al. 2013, Tang and Wang 2015a]), which are based on additional setup assumptions. Unfortunately, these solutions do not generalize to broader settings.

1.2 Our Contribution

In this paper, we propose a privacy-preserving hybrid recommender system, which accommodates both neighborhood-based and model-based components. The final predictions for a user Alice is a combination of the predictions from the individual components. To facilitate privacy protection, we instantiate the neighborhood-based component via user-based collaborative filtering (UCF), and instantiate the model-based component via incremental matrix factorization (IMF). The IMF allows recommender service provider (RecSys) to perform matrix factorization based on expert rating dataset, and a user Alice can perform a privacy-preserving IMF protocol to learn her feature vector.

Relying on somewhat homomorphic encryption (SWHE) schemes, we propose privacy-preserving protocols for both components. We first propose a privacy-preserving UCF protocol by introducing a third party, named Proxy. Different from most existing works, it is not necessary to require the Proxy to be semi-honest. We then propose a privacy-preserving IMF protocol, which pushes most of the computation workload to the server. We finally evaluate the privacy properties of the hybrid system with respect to a number of attack scenarios, ranging from pure semi-honest attackers to malicious ones. As the privacy-preserving UCF protocol will be less frequently executed than the privacy-preserving IMF protocol in the hybrid solution, we mainly study the performances of the IMF protocol, based on Microsoft SEAL library [Library 2016]. The results show that the protocol is extremely efficient for the end users, while it is also more efficient for the server than other existing solutions.

Due to the space limit, in the full paper [Tang and Wang 2016], we show that the hybrid system provides more accurate recommendation results than the individual components. In contrast, many existing solutions sacrifice the accuracy by using tailored recommendation techniques.

1.3 Organization

In Section 2, we present the preliminaries. In Section 3, we give a high-level description of our privacy-preserving hybrid system. In Section 4 and 5, we describe the privacy-preserving UCF and IMF protocols respectively. In Section 6, we provide extended security and efficiency analysis. In Section 7, we conclude the paper.

2. PRELIMINARY

Suppose the item set is denoted by $\mathbf{I} = (1, \cdots, M)$. For a user x, its ratings are denoted by $\mathbf{R}_x = (r_{x,1}, \cdots, r_{x,M})$. The rating value is often an integer from $\{0, 1, 2, 3, 4, 5\}$. If item i has not been rated, then $r_{x,i}$ is set to be 0. User x's average rating is denoted by $\overline{r_y}$. We use bold letters such as $\mathbf{u}, \mathbf{v}$ to denote vectors, and use $\langle \mathbf{u}, \mathbf{v} \rangle$ to denote the inner product.

2.1 User-based Collaborative Filtering

For two rating vectors $\mathbf{R}_x, \mathbf{R}_y$, their Cosine similarity is denoted as $Sim_{x,y}$, where

$$Sim_{x,y} = \frac{\langle \mathbf{R}_x, \mathbf{R}_y \rangle}{\|\mathbf{R}_x\| \times \|\mathbf{R}_y\|} = \langle \frac{\mathbf{R}_x}{\|\mathbf{R}_x\|}, \frac{\mathbf{R}_y}{\|\mathbf{R}_y\|} \rangle$$

Suppose we want to compute predictions for a user x based on the rating data from a user group $\mathcal{G}$, then the formula is as follows.

$$p_{x,i} = \overline{r_x} + \frac{\sum_{y \in \mathcal{G} \wedge r_{y,i} \neq 0} Sim_{x,y}(r_{y,i} - \overline{r_y})}{\sum_{y \in \mathcal{G} \wedge r_{y,i} \neq 0} Sim_{x,y}}$$

We select UCF as a building block for our hybrid system, because it provides better accuracy than other algorithms in case of a small user population.

2.2 Matrix Factorization

Given a user set $\mathcal{U} = \{1, 2, \cdots, N\}$ and their rating vectors $\mathbf{R}_x$ for $x \in \mathcal{U}$, let $\mathcal{R}$ denote the set of (x, j) such that $r_{x,j} \neq 0$. One of the most popular collaborative filtering algorithms is based on low-dimensional factor models, which derive two feature matrices $\mathbf{U}$ and $\mathbf{V}$ from the rating dataset. The feature vector $\mathbf{u}_x$ denotes user x's interest and the feature vector $\mathbf{v}_j$ denotes item j's characteristics. Every feature vector has very low dimension k, which is often a much smaller integer than M and N.

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_N \end{bmatrix} \quad \text{and} \quad \mathbf{V} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_M \end{bmatrix}$$

In practice, $\mathbf{U}$ and $\mathbf{V}$ are often computed by minimizing the following Regularized least Squares Error (RSE) function:

$$\min_{\mathbf{U},\mathbf{V}} \frac{1}{|\mathcal{R}|} \sum_{(x,j) \in \mathcal{R}} (r_{x,j} - \langle \mathbf{u}_x, \mathbf{v}_j \rangle)^2 + \lambda \sum_{x \in \mathcal{U}} ||\mathbf{u}_x||_2^2 + \mu \sum_{j \in \mathbf{I}} ||\mathbf{v}_j||_2^2 \quad (1)$$

for some positive parameters λ, μ. Using the standard gradient descent method, $\mathbf{U}$ and $\mathbf{V}$ can be learned through recursively applying the updating rules. Every round of updating is called an *epoch*, and the total number of epochs is denoted as MAX_{epoch}.

$$\mathbf{u}_x^{(t)} = \mathbf{u}_x^{(t-1)} - \gamma \nabla_{\mathbf{u}_x} F(\mathbf{U}^{(t-1)}, \mathbf{V}^{(t-1)}) \quad (2)$$

$$\mathbf{v}_j^{(t)} = \mathbf{v}_j^{(t-1)} - \gamma \nabla_{\mathbf{v}_j} F(\mathbf{U}^{(t-1)}, \mathbf{V}^{(t-1)}) \quad (3)$$

where $\gamma > 0$ is a small gain factor and

$$\nabla_{\mathbf{u}_x} F(\mathbf{U}, \mathbf{V}) = -2 \sum_{j:(x,j) \in \mathcal{R}} \mathbf{v}_j (r_{x,j} - \langle \mathbf{u}_x, \mathbf{v}_j \rangle) + 2\lambda \mathbf{u}_x \quad (4)$$

$$\nabla_{\mathbf{v}_j} F(\mathbf{U}, \mathbf{V}) = -2 \sum_{x:(x,j) \in \mathcal{R}} \mathbf{u}_x (r_{x,j} - \langle \mathbf{u}_x, \mathbf{v}_j \rangle) + 2\mu \mathbf{v}_j \quad (5)$$

2.3 Somewhat Homomorphic Encryption

Since the breakthrough work of Gentry [Gentry 2009], many somewhat homomorphic encryption (SWHE) schemes

have been proposed (e.g. BGV scheme [Brakerski et al. 2012] and YASHE scheme [Bos et al. 2013]). A SWHE scheme can be described by three algorithms (Keygen, Enc, Dec).

- Keygen(λ, L): With the security level λ, the multiplication depth L, this algorithm outputs a key tuple (PK, SK, EVK).
- Enc(PK, m): this algorithm outputs a ciphertext c.
- Dec(SK, c): this algorithm outputs a plaintext m.

Throughout the paper, given a key pair (PK_u, SK_u) for some user u, we use $[m]_u$ to denote a ciphertext of the message m under public key PK_u. When $\mathbf{m}$ is a vector of messages, we use Enc$(PK_u, \mathbf{m})$ to denote the vector of ciphertexts, where encryption is done for each element independently. Given two ciphertexts $[m_1]_u$ and $[m_2]_u$, we have the following notations.

- We use $[m_1]_u \oplus [m_2]_u$ and $[m_1]_u \otimes [m_2]_u$ to denote the homomorphic addition and multiplication respectively.
- We use the notation $\sum_{1 \le i \le N} [m_i]_u$ to denote the result of sequentially applying $\oplus$ to the cipheretxts.
- We use **ReRand** to denote a rerandomization algorithm. Given a ciphertext $[m]_u$, no attacker can determine whether **ReRand**$([m]_u)$ and $[m]_u$ encrypt the same plaintext or not.
- With respect to $\oplus$, we use the notation $\ominus$ to denote the homomorphic substraction operator. It is clear that we can implement $\ominus$ based on $\oplus$.
- We also use $\oplus$, $\otimes$, and $\ominus$ to denote partial homomorphic operations, in which case one of the arguments is in plaintext form. The distinction should be clear from the contexts. Unlike $\otimes$, the partial $\otimes$ has less noise increase and doesn't need the costly relinearization, which makes it much faster.

2.4 Approximate Integer Division

Algorithm 1: Approximate Integer Division

$t = L$
while $t \ge 0$ **do**
  $([b_t]_a; \emptyset) = \mathsf{COM}([X]_a, [2^t Y]_a; SK_a)$
  $[X]_a = [X]_a \ominus (2^t \otimes [b_t]_a \otimes [Y]_a)$
  $t = t - 1$
$[\lfloor \frac{X}{Y} \rfloor]_a = \sum_{0 \le t \le L} 2^t \otimes [b_t]_a$

Assume a two-party setting, where Alice holds a SWHE public/private key pair (PK_a, SK_a) and the RecSys holds two ciphertexts $[X]_a$ and $[Y]_a$ satisfying $X < 2^{L+1} Y$ for some integer L. We design an algorithm (i.e. Algorithm 1) for the RecSys to compute $[\lfloor \frac{X}{Y} \rfloor]_a$. The algorithm is based on a secure integer comparison protocol COM [Tang and Wang 2015b]: given inputs $([A]_a, [B]_a; SK_a)$ from RecSys and Alice respectively, COM outputs $([b]_a, \emptyset)$ to the participants respectively. Note that $b = 1$ if $A \ge B$ and $b = 0$ otherwise.

The complexity of this protocol is briefly summarized in Table 1. Note that we use par. $\otimes$ to denote partial $\otimes$.

	$\oplus$	$\ominus$	$\otimes$	par. $\otimes$	COM
Alice	0	0	0	0	L+1
Server	L	1	L+1	2L+1	L+1

Table 1: Division Complexity

3. PRIVACY-PRESERVING SYSTEM

There are two types of entities in the recommender systems: recommender service provider (RecSys) and users. The overall system structure is shown in Figure 1.

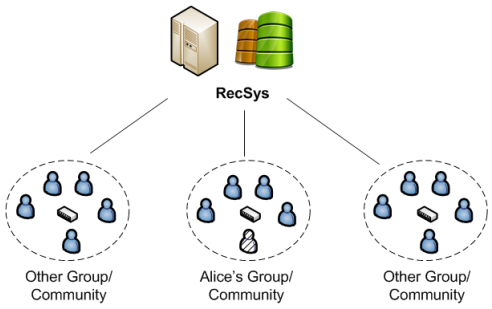

Figure 1: Hybrid System Framework

The RecSys is a commercial organization who collects item ratings from professionals and build models to help the users find their best recommendations. The users are customers of the Recsys, and they want to receive good recommendations without sacrificing their privacy (i.e. disclosing their rating vectors to the RecSys or others). We further assume the users are clustered into groups, depending on demographic information such as locations. For each user group, we assume the existence of a third party named Proxy, which will facilitate the protocol execution.

To compute recommendations for a user Alice of a certain group, we will take into account both local influences and global influences. The local influence comes from the users from Alice's group, and we employ the UCF to benefit from the similarities between users in a group. For each group, the execution of UCF is coordinated by the Proxy. The global influence comes from the expert dataset from the RecSys, and we employ IMF to learn item features from the dataset. Note that the expert dataset is from professional critics who may make a living for their opinions. In addition, they usually disclose their dataset to RecSys for some monetary reward. Finally, the local and global influences are combined to provide hybrid recommendations.

3.1 Privacy-preserving System

We assume RecSys possesses a SWHE key pair (PK_s, SK_s). Every user, say Alice, registers at RecSys and the Proxy of her group with an identifier ID_a and SWHE key pair (PK_a, SK_a). We assume that the communication among users is integrity protected, and the communication between any user and the Proxy is through an anonymous network and confidentiality protected. Hence, it is difficult for the Proxy to identify users purely from the data sources.

Suppose a user Alice wants to compute the predictions for all the items, then the protocol is as follows.

1. Suppose the user group that Alice is involved in is denoted as $\mathcal{G}$. With the privacy-preserving UCF protocol from Section 4, at the end of the protocol, the Proxy in Alice's group sends $[p_{a,i}^{(ucf)}]_a$ for every item $1 \le i \le M$ to RecSys.

2. For Alice, let the indices of the unzero ratings (in the form (a, j)) be denoted by $\mathcal{R}_a$. Alice initiates the privacy-preserving IMF protocol from Section 5. At the end of the protocol, RecSys obtains $[p_{a,i}^{(imf)}]_a$ for every $1 \le i \le M$. Note that a non-private form of the IMF protocol is shown in Algorithm 2.

3. With both $[p_{a,i}^{(imf)}]_a$ and $[p_{a,i}^{(ucf)}]_a$, RecSys chooses a contribution factor α, which denotes the percentage that IMF contributes to the final prediction, and computes the hybrid predictions $[p_{a,i}]_a$ $(1 \le i \le M)$.

$$[p_{a,i}]_a = \alpha \otimes [p_{a,i}^{(imf)}]_a \oplus (1-\alpha) \otimes [p_{a,i}^{(ucf)}]_a$$

   This can be regarded as the simplest way to generate hybrid predictions. We can adapt it for better security, as shown in Section 6.1.

4. After receiving $[p_{a,i}]_a$ $(1 \le i \le M)$, Alice can decrypt them and obtain the plaintext predictions.

Algorithm 2: Incremental Matrix Factorization

INPUT: $\mathbf{R}_a$, $\mathbf{U}$ and $\mathbf{V}$
Initialize $\mathbf{u}_a = \mathbf{u}_a^{(0)}$
$\mathbf{U}^{(0)} = \binom{\mathbf{U}}{\mathbf{u}_a}$, $\mathbf{V}^{(0)} = \mathbf{V}$, $\mathcal{R} = \mathcal{R} \cup \mathcal{R}_a$
t=1
while $t \le MAX_{epoch}$ **do**
  $\mathbf{u}_a^{(t)} = \mathbf{u}_a^{(t-1)} - \gamma \nabla_{\mathbf{u}_a} F(\mathbf{U}^{(t-1)}, \mathbf{V}^{(t-1)})$
  $\mathbf{U}^{(t)} = \binom{\mathbf{U}}{\mathbf{u}_a^{(t)}}$
  $\mathbf{V}^{(t)} = \mathbf{V}$
  $t = t + 1$
OUTPUT: $\mathbf{u}_a = \mathbf{u}_a^{(MAX_{epoch})}$

It is common that individual users do not update their ratings frequently. In fact, adding a few ratings values does not affect the final predictions (or improve the recommendation accuracy). This means that, it is unnecessary to frequently execute the privacy-preserving UCF protocol in the hybrid system. Note also that the protocol does not involve RecSys, so Alice's group can carry out the computation in an offline manner. Therefore, when evaluating the computational complexity of the hybrid system, we put more emphasis on step (2).

3.2 Security Model(s)

If we assume RecSys and the Proxy are *semi-honest* (as most other works do), then the security model will be essentially the same as the IND-CPA security for encryption. Consequently, the IND-CPA security of the underlying encryption scheme will provide the necessary guarantee. We, as in the case of [Kim et al. 2016, Nikolaenko et al. 2013], avoid a straightforward description for the model. If the RecSys and the Proxy are compromised or malicious, then things become very complex and tedious to describe. Our setting is almost identical to that in [Tang and Wang 2015b], so that we refer the reader to Section III of [Tang and Wang 2015b] for detailed description of the security models.

4. PRIVACY-PRESERVING UCF

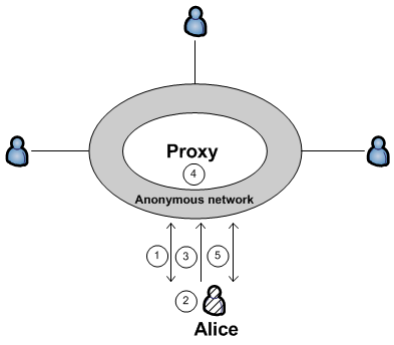

Figure 2: User-based CF

The protocol flow is shown in Figure 2 where the numbers correspond to the steps below. Suppose the users in Alice's group consists of user i for $1 \le i \le T$. Referring to the setup information in Section 3.1, the protocol runs as follows. *For the sake of simplicity, in the first three steps, we only describe the computations for Alice while implicitly assuming all other users perform similarly.*

1. Alice broadcasts her identifier ID_a together with the encrypted vectors $[\frac{\mathbf{R}_a}{\|\mathbf{R}_a\|}]_a$ and $[(r_{a,1} - \overline{r_a}, \cdots, r_{a,M} - \overline{r_a})]_a$. Note that $\frac{\mathbf{R}_a}{\|\mathbf{R}_a\|}$ is the normalized rating vector.

2. After receiving the messages from other users, for every user $1 \le y \le T$, Alice computes encrypted similarity $[Sim_{a,y}]_y$, where $[Sim_{a,y}]_y = \langle \frac{\mathbf{R}_a}{\|\mathbf{R}_a\|}, [\frac{\mathbf{R}_y}{\|\mathbf{R}_y\|}]_y \rangle$.

3. For every item i $(1 \le i \le M)$, Alice does the following.

   - When $r_{a,i} \ne 0$, Alice computes her contribution $Con_{a \xrightarrow{i} y}$ to every user $1 \le y \le T$.

$$NOM_{a \xrightarrow{i} y} = \mathsf{ReRand}([Sim_{a,y}]_y \otimes (r_{a,i} - \overline{r_a}))$$

$$DEN_{a \xrightarrow{i} y} = \mathsf{ReRand}([Sim_{a,y}]_y)$$

$$Con_{a \xrightarrow{i} y} = (ID_y,\ i,\ NOM_{a \xrightarrow{i} y},\ DEN_{a \xrightarrow{i} y})$$

   - When $r_{a,i} = 0$, Alice sets $Con_{a \xrightarrow{i} y}$ for every user $1 \le y \le T$, as follows.

$$Con_{a \xrightarrow{i} y} = (ID_y, i, NOM_{a \xrightarrow{i} y} = [0]_y, DEN_{a \xrightarrow{i} y} = [0]_y)$$

   Alice sends the computed values to the Proxy. In addition, Alice sends $ID_a, [\overline{r_a}]_a$ as well.

4. After receiving the values from Alice and all other users, the Proxy computes the encrypted prediction for every user and item *in a partial form*. In total, there are $M \cdot (T+1)$ partial predictions. For instance, the partial prediction for Alice with respect to item i is computed as follows.

$$([\overline{r_a}]_a, \sum_{1 \le x \le T} NOM_{x \xrightarrow{i} a}, \sum_{1 \le x \le T} DEN_{x \xrightarrow{i} a})$$

5. The Proxy runs the approximate division protocol from Section 2.4 to compute the complete predictions for Alice and all other users. In total, there are $M \cdot (T+1)$ predictions. For instance, for Alice with respect to the item i, the Proxy runs Algorithm 1 from Section 2.4 to compute

$$\lfloor \frac{\sum_{1 \le x \le T} NOM_{x \xrightarrow{i} a}}{\sum_{1 \le x \le T} DEN_{x \xrightarrow{i} a}} \rfloor,$$

which abuses the notation to denote a ciphertext of the approximated division result. Then, the final encrypted prediction $[p_{a,i}^{(ucf)}]_a$ is simply

$$[p_{a,i}^{(ucf)}]_a = [\overline{r_a}]_a \oplus \lfloor \frac{\sum_{1 \le x \le T} NOM_{x \xrightarrow{i} a}}{\sum_{1 \le x \le T} DEN_{x \xrightarrow{i} a}} \rfloor$$

The Proxy sends all encrypted predictions to RecSys.

4.1 Performance Analysis

Complexity. Let $\ell = M\rho$ be the total number of non-zero ratings, where ρ is the rating density, and let AID denote the complexity numbers from Table 1. The complexity of this protocol is briefly summarized in Table 2. We note that the Enc operations for Alice can be done offline.

	par. $\otimes$	$\oplus$	Enc
Alice	$(\rho+1)MT$	$T(M-1)$	$2(1-\rho)MT - 2M$
Proxy	0	$M(2T^2+T-1)$	0
	AID	ReRand	
Alice	$M(T+1)$	$2\rho MT$	
Proxy	$M(T+1)$	0	

Table 2: UCF Complexity

Security. We consider two scenarios.

- Suppose that the Proxy is a semi-honest player, then it learns no information about the users' rating values since everything is encrypted. No user learns anything about another user due to the encryption. Referring to Step (3), the anonymous communication network prevents the Proxy from learning who has sent the contribution. The application of ReRand is essential to hide which items Alice has rated. Without this, the Proxy might be able to use the technique from [Weinsberg et al. 2012] to re-identify Alice in some scenarios.

- Suppose the Proxy is malicious (or compromised) and colludes with some of the users in the group, then it is still difficult for it to recover and link Alice's rating values. The anonymous communication network makes it hard for the Proxy alone to link Alice's contributions, say $Con_{a \xrightarrow{i} y}$, to her identifier ID_a. If the Proxy colludes with user y, it will be able to decrypt $Con_{a \xrightarrow{i} y}$ for all $1 \le i \le M$. However, it is not immediate that the Proxy can conclude that these $Con_{a \xrightarrow{i} y}$ values belong to the same user Alice. As long as there are other honest users who have rated similar items to Alice, then the Proxy still needs to try all combinations to determine whether two contribution values belong to the same honest user. In the extreme situation that all users collude with the Proxy, then they can learn learn which items have been rated by Alice and the rating deviations $r_{a,i} - \overline{r_a}$. Subsequently, they may succeed in the re-identifying Alice. We know that unless calibrating noise into the computation, there is nothing more we can do when all participants collude. In the context of our hybrid solution, we present more analysis in Section 6.1.

5. PRIVACY-PRESERVING IMF

We present a new protocol based on the following observation. For two column vectors $\mathbf{X}, \mathbf{Y}$, we have $\mathbf{X}\mathbf{Y}^t\mathbf{X} = \mathbf{X}\mathbf{X}^t\mathbf{Y}$. In the following, for Alice, if the rating $r_{a,j} = 0$ we set $d_{a,j} = 0$, otherwise we set $d_{a,j} = 1$. The updating rule can be rewritten as follows.

$$\begin{aligned} \mathbf{u}_a^{(t)} &= (1 - 2\gamma\lambda)\mathbf{u}_a^{(t-1)} + 2\gamma \sum_{j:(a,j)\in\mathcal{R}} \mathbf{v}_j (r_{a,j} - \langle \mathbf{u}_a^{(t-1)}, \mathbf{v}_j \rangle) \\ &= \Delta \mathbf{u}_a^{(t-1)} + 2\gamma \sum_{1 \le j \le M} \mathbf{v}_j r_{a,j} \end{aligned}$$

where $\Delta = ((1 - 2\gamma\lambda)\mathbf{I}_k - 2\gamma(\sum_{1 \le j \le M} d_{a,j}\mathbf{v}_j\mathbf{v}_j^t))$ and $\mathbf{I}_k$ is a $k \times k$ identity matrix. Let $e = MAX_{epoch}$. To further optimize the training, we can expand all the epoches to get the final vector.

$$\begin{aligned} \mathbf{u}_a^{(e)} &= \Delta^e \mathbf{u}_a^{(0)} + 2\gamma \sum_{1 \le j \le M} (\sum_{i \in [0,e)} \Delta^i \mathbf{v}_j^t) r_{a,j} \\ &= \Delta^e \mathbf{u}_a^{(0)} + (2\gamma \sum_{i \in [0,e)} \Delta^i) \sum_{1 \le j \le M} \mathbf{v}_j^t r_{a,j} \end{aligned}$$

Alice		**RecSys**
$(\overline{PK_a}, \overline{SK_a})$		$(\overline{PK_s}, \overline{SK_s})$
$[\mathbf{v}_j]_s$, $[\mathbf{v}_j\mathbf{v}_j^t]_s$ $(1 \le j \le M)$		
$[\Delta_1]_s = [\mathbf{v}_1\mathbf{v}_1^t]_s \otimes d_{a,1}$		
...		
$[\Delta_M]_s = [\mathbf{v}_M\mathbf{v}_M^t]_s \otimes d_{a,M}$		
$[\Delta]_s = \sum_{j:(a,j)\in\mathcal{R}} [\Delta_j]_s$		
	$\xrightarrow{[\Delta]_s}$	
		$\Delta = (1 - 2\gamma\lambda)\mathbf{I}_k - 2\gamma\Delta$
$[\mathbf{v}^t]_s = \sum_{j:(a,j)\in\mathcal{R}} [\mathbf{v}_j^t]_s \otimes r_{a,j}$		
Set a random vector $\mathbf{r}_a$		
$[\mathbf{v'}^t]_s = [\mathbf{v}^t]_s \oplus \mathbf{r}_a$		
	$\xrightarrow{[\mathbf{v'}^t]_s}$	
		Set a random vector $\mathbf{r}_s$
		$\mathbf{v''}^t = \mathbf{v'}^t + \mathbf{r}_s$
	$\xleftarrow{\mathbf{v''}^t}$	
$[\mathbf{v'}^t]_a = [\mathbf{v''}^t - \mathbf{r}_a]_a$		
	$\xrightarrow{[\mathbf{v'}^t]_a}$	
		$[\mathbf{v}^t]_a = [\mathbf{v'}^t]_a \ominus \mathbf{r}_s$
		$[\mathbf{v}^t]_a = (2\gamma \sum_{i \in [0,e)} \Delta^i)[\mathbf{v}^t]_a$
		$[\mathbf{u}_a^{(e)}]_a = \Delta^e \mathbf{u}_a^{(0)} \oplus [\mathbf{v}^t]_a$

Figure 3: Privacy-preserving Algorithm 2

In the proposed protocol, we choose to precalculate Δ interactively, then do the training. The protocol is as follows.

1. RecSys computes $\mathbf{U}$ and $\mathbf{V}$ based on its expert dataset.

2. Alice and RecSys first interactively compute the matrix Δ and then RecSys runs the Algorithm 2 in the encrypted form, as shown in Figure 3.

3. RecSys can compute the predictions for Alice in the encrypted form. For instance, for the item i, the prediction is $[p_{a,i}^{(imf)}]_a = \langle [\mathbf{u}_a]_a, \mathbf{v}_i \rangle$.

5.1 Performance Analysis

Complexity. Note that Alice can compute $[d_{a,j}]_a, [r_{a,j}]$ $(1 \leq j \leq M)$ in the offline manner, and RecSys can compute $\mathbf{U}, \mathbf{V}$ and encrypt $\mathbf{V}$ in an offline manner. Let k, e, and ℓ be be defined as before, the number of offline and online computations are briefly summarized in Table 3 and 4.

	$\oplus/\ominus$	par. $\otimes$	Enc
Alice	$(k^2+k)\cdot(\rho M-1)$	$(k^2+k)\cdot\rho M$	0
RecSys	0	0	$(k^2+k)M$

Table 3: Complexity (offline)

	$\oplus/\ominus$	par. $\otimes$	Enc	Dec
Alice	$2k$	0	k	0
RecSys	k^2+k	k^2	0	k

Table 4: Complexity (online)

Security. With respect to security, RecSys does not leak information to Alice except what can be inferred from the predictions. On the other hand, Alice will only leak the information that can be inferred from Δ. In the worst case, $d_{a,j}$ $(1 \leq j \leq M)$ might be leaked to RecSys. However, we argue that the leakage would be much smaller in reality. For a practical recommender system (e.g. our evaluation in Section 6.2 or Netflix), we can always assume that $M \gg k^2$. In this case, for a semi-honest RecSys, computing $d_{a,j}$ $(1 \leq j \leq M)$ from Δ is equivalent to some variant of sparse subset problem, which is shown to be hard in [Nguyen and Stern 1999], unless that the density $M/\log(max(\Delta)) < 0.94$[Coster et al. 1992], which is around several hundreds in our case. Besides, the small k may not give a unique solution. When the RecSys is malicious (i.e. it can randomly set the $\mathbf{v}_j$ $(1 \leq j \leq M)$), it will be easy for RecSys to recover $d_{a,j}$ $(1 \leq j \leq M)$. However, we do not see any incentive for this kind of attack because it will disrupt the service and expose the attack to Alice, thus lose RecSys's customers.

In theory, it is easy to require Δ to be encrypted under PK_a so that RecSys needs to compute the Δ^t $(1 \leq t \leq e)$ in the encrypted form. However, consider the practical size of k and e, this will significantly increase the circuit depth for the SWHE scheme and the complexity will dramatically increase accordingly. In our design, we have made a tradeoff between security and efficiency on purpose.

6. EVALUATING THE SOLUTION

In this section, we present additional security and efficiency analysis for the hybrid system from Section 3.

6.1 Security Analysis

For the hybrid recommender system, we are interested in the security for both an end user Alice and RecSys. With respect to Alice, referring to the system structure in Figure 1, it is clear that the behavior of the users from other groups do not affect Alice's security. We consider four attack scenarios, shown in Figure 4, which are concerned with Alice's rating values and the received predictions.

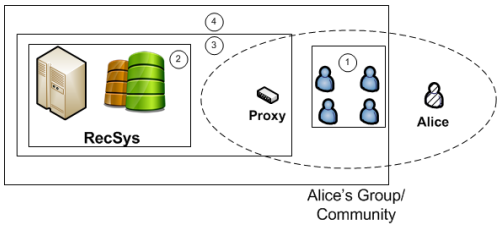

Figure 4: Attack Scenarios against Alice

- In the first scenario, all other users in Alice's group collude to figure out Alice's information. In this case, the malicious users will only have access to the encrypted values sent by Alice in the first step of the privacy-preserving UCF protocol, plus the final recommendation results. Based on the IND-CPA security of the encryption scheme, this means that the attacker only gains what can be inferred from the legitimate predictions they receive.

- In the second scenario, the attacker is RecSys. In this case, the only possible place for information leakage is the privacy-preserving IMF protocol execution. Suppose that RecSys is semi-honest, then it learns very little in the server-centric protocol, according to the analysis in Section 5. If RecSys is malicious which means it can deviate from the protocol execution, then it can learn which items Alice has rated. As indicated at the end of Section 5, RecSys is not incentivized to be malicious.

- In the third scenario, RecSys and the Proxy collude. Note the fact that the Proxy only aggregate the encrypted data in the privacy-preserving UCF protocol, we have the same conclusion as in the second scenario.

- In the fourth scenario, all players except for Alice collude. In this case, even if these players follow the protocol specification, they will learn both Alice's rating values and the received predictions simply by decrypting the encrypted data in step (1) of the privacy-preserving UCF protocol. This is an extreme scenario, and the consequence is implied by the recommendation utility.

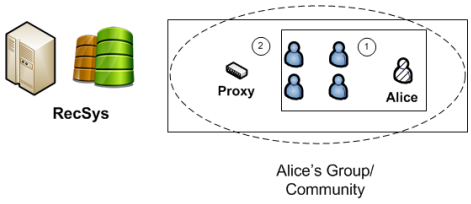

Figure 5: Attack Scenarios against RecSys

With respect to the security for RecSys, we consider two scenarios, shown in Figure 5. For the sake of simplicity, we only consider one user group.

- In one scenario, all users in the group collude. The information leakage comes from the privacy-preserving IMF protocol. At the end, the malicious users can recover the some of the feature vectors $\mathbf{v}_j$ $(1 \leq j \leq M)$. The attack is simple, by trivially setting their user feature vectors $\mathbf{u}_i$ to be from $\mathbf{v}_j$ $(1 \leq j \leq M)$. Given the fact that all users collude, they can compute the UCF's contributions in the hybrid predictions they receive, and then they can obtain the selected $\mathbf{v}_j$ $(1 \leq j \leq M)$. However, when not all users collude, then the attack does not trivially work anymore. Since the malicious users do not know exactly the UCF's contributions in the hybrid predictions they receive, then it is much harder for them to recover some of the $\mathbf{v}_j$ $(1 \leq j \leq M)$. An aditional barrier is that each user can only contribute one user feature vector through the privacy-preserving protocol, therefore, M users need to collude in order to recover all $\mathbf{v}_j$ $(1 \leq j \leq M)$.

- In the other scenario, the Proxy also colludes with the users. In this case, the consequence is the same as the above scenario when all users collude. If the Proxy is a branch of RecSys or has some commercial agreement, then it will not be incentivized to collude and leak RecSys's information.

Two Simple Enhancements. Assuming the Proxy and RecSys are semi-honest, we can introduce some uncertainty in two steps of the hybrid system from Section 3.

- One is in step (1), which executes the privacy-preserving UCF protocol. Referring to the specification in Section 4, the Proxy does not aggregate everything to the predictions. For instance, the partial prediction for Alice with respect to the item i is computed as follows. First randomly choose a set $\mathcal{S}$ to include some percentage of the users, and then compute the prediction as follows.

$$([\overline{r_a}]_a, \sum_{x \in \mathcal{S}} NOM_{x \xrightarrow{i} a}, \sum_{1 \leq x \leq T} DEN_{x \xrightarrow{i} a})$$

  If the percentage is close to 1, then the recommendation accuracy will not be affected. As a result, the colluded users will not be able to know the exact UCF contributions. This enhancement mitigates the attacks against both Alice (by decreasing what other users can infer from the received predictions) and RecSys (making it harder to recover $\mathbf{v}_j$ $(1 \leq j \leq M)$).

- The other is in step (3) of the hybrid system from Section 3. In this case, RecSys can randomly set the contributing factor α for every item and every user. If the derivation of the α values is small, then this will not affect the recommendation accuracy much. As a result, this will make it very difficult for the malicious users to precisely recover $\mathbf{v}_j$ $(1 \leq j \leq M)$.

These two enhancements share some links to differential privacy. It is an interesting topic for further research.

6.2 Complexity Analysis

We use the MovieLens 1M dataset[1]. For 80-bit security, we set the feature dimension to be $k = 10$ and set the MAX_{epoch} to be $e = 30$, $T = 50$. Alice's rating vector has the density $\rho = 0.9\%$ and RecSys's expert dataset has the density $\rho' = 4.19\%$. We evaluate our scheme with Microsoft SEAL library [Library 2016] based on YASHE scheme. We select the ciphertext modulus $q = 2^{226} - 2^{26} + 1$, the polynomial modulus $p(x) = x^{8192} + 1$. Using Chinese Reminder Theorem, we select two 40-bit primes to represent the plaintext space of 2^{80}. The primes are 1099511922689 and 1099512004609. By packing 8192 plaintexts into one ciphertext, we can process 8192 multiplications in one homomorphic multiplication. Based on an Intel(R) Core(TM) i7-5600U CPU 2.60GHz, 8GB RAM, we have the timing complexity for Enc (52.43 ms), Dec (39.63 ms), $\otimes$ (207.76 ms), $par.\otimes$ (70.28 ms), $\oplus/\ominus$ (742 μs).

Since the system requires fix-point computation, we need to round all the float points during the training epochs. For the matrix $\Delta^e \mathbf{u}_a^{(0)}$ (denoted as Δ_e) in the protocol from Figure 3, we transform the float-point vector into an integer one by rounding the matrix to be $\Delta_r = \lfloor 2^{10} \Delta_e \rfloor$. We transform all float point matrices or vectors in the same way to integer ones. The timing costs of the privacy-preserving IMF protocol are summarized in Table 5.

	Alice (offline)	Alice (online)	RecSys (offline)	RecSys (online)
PPIMF	10.10	1.08	1110	14.30

Table 5: Timing Cost (Seconds)

Compared with the schemes from [Kim et al. 2016, Nikolaenko et al. 2013], we solve a cryptographically different problem since we only protect the incremental part rather than all the training process. As a result of the simplification, we get rid of the extra third-party CSP. Regarding the complexity, our solution performs significantly better.

- The communicational cost is reduced from 580 MB [Kim et al. 2016] to 5.73 MB in our protocol.

- The computational cost per round in [Kim et al. 2016] is 1.5 minutes on a PC with 3.4 GHz 6-core 64GB RAM, while the one in [Nikolaenko et al. 2013] costs 170 minutes on two servers with 1.9 GHz 16-cores 128 GB RAM. Our computational cost is much less than theirs.

While the schemes in [Kim et al. 2016, Nikolaenko et al. 2013] deal with factorizing the whole rating matrix, which generates the $\mathbf{U}$ matrix for multiple users, our scheme generates one feature vector for a user Alice by default. However, if Alice is the holder of multiple users' data, it's easy to process them in one time by packing the Δ matrix from multiple users.

6.3 Complexity Analysis of UCF

Considering the simplicity of operations in UCF, we can select small parameters for the SWHE. We select the ciphertext modulus $q = 2^{190} - 2^{30} + 1$, the polynomial modulus $p(x) = x^{4096} + 1$. The plaintext space is 2^9. Based on an Intel(R) Core(TM) i7-5600U CPU 2.60GHz, 8GB RAM, we have the timing complexity for Enc (41.78 ms), Dec (41.22 ms), $par.\otimes$ (0.398 ms, the plaintext is much smaller), $\oplus/\ominus$ (85 μs). The time cost for Alice is 23046 seconds and for Proxy is 4106 seconds. But considering UCF is performed much less frequently than IMF, this is acceptable.

[1] https://grouplens.org/datasets/movielens/1m/

7. CONCLUSION

This paper has described a hybrid design for privacy-preserving recommender systems. Instead of focusing on a single recommender algorithm, we have proposed a new design in an attempt to facilitating privacy protection and enhancing recommendation accuracy. We have also attempted to improve the efficiency by slightly relaxing the privacy guarantees in the privacy-preserving IMF protocol. It shows that the efficiency can be dramatically improved while the users' privacy might not be affected in practical settings. Besides privacy, other issues such as robustness, are equally important or more important from the perspective of service providers. Note that the typical operations such as data cleaning becomes much more difficult in privacy-preserving systems (usually due to encryption). How to address these issues altogether remains as a challenging future work.

Acknowledgements

Both authors are supported by a CORE (junior track) grant from the National Research Fund, Luxembourg.

8. REFERENCES

[Berlioz et al. 2015] A. Berlioz, A. Friedman, M. A. Kaafar, R. Boreli, and S. Berkovsky. 2015. Applying Differential Privacy to Matrix Factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 107–114.

[Bos et al. 2013] Joppe W Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. 2013. Improved security for a ring-based fully homomorphic encryption scheme. In *Cryptography and Coding*. Springer, 45–64.

[Brakerski et al. 2012] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. 2012. (Leveled) Fully Homomorphic Encryption Without Bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. 309–325.

[Burke 2002] R. Burke. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* 12, 4 (2002), 331–370.

[Calandrino et al. 2011] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov. 2011. "You Might Also Like: " Privacy Risks of Collaborative Filtering. In *32nd IEEE Symposium on Security and Privacy, S&P 2011*. 231–246.

[Canny 2002] J. F. Canny. 2002. Collaborative Filtering with Privacy. In *IEEE Symposium on Security and Privacy*. 45–57.

[Coster et al. 1992] Matthijs J Coster, Antoine Joux, Brian A LaMacchia, Andrew M Odlyzko, Claus-Peter Schnorr, and Jacques Stern. 1992. Improved low-density subset sum algorithms. *Computational complexity* 2, 2 (1992), 111–128.

[Dankar and Emam 2013] F. K. Dankar and K. El Emam. 2013. Practicing Differential Privacy in Health Care: A Review. *Trans. Data Privacy* 6, 1 (2013), 35–67.

[Gentry 2009] C. Gentry. 2009. Fully Homomorphic Encryption Using Ideal Lattices. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*. 169–178.

[Guerraoui et al. 2015] R. Guerraoui, A. Kermarrec, R. Patra, and M. Taziki. 2015. D2P: Distance-based Differential Privacy in Recommenders. *Proc. VLDB Endow.* 8, 8 (2015), 862–873.

[Haeberlen et al. 2011] A. Haeberlen, B. C. Pierce, and A. Narayan. 2011. Differential Privacy Under Fire. In *Proceedings of the 20th USENIX Conference on Security*. 33–33.

[Jeckmans et al. 2013] A. Jeckmans, A. Peter, and P. H. Hartel. 2013. Efficient Privacy-Enhanced Familiarity-Based Recommender System. In *Computer Security - ESORICS 2013 (LNCS)*, Vol. 8134. Springer, 400–417.

[Kim et al. 2016] S. Kim, J. Kim, D. Koo, Y. Kim, H. Yoon, and J. Shin. 2016. Efficient Privacy-Preserving Matrix Factorization via Fully Homomorphic Encryption: Extended Abstract. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. 617–628.

[Library 2016] Microsoft SEAL Library. 2016. https://sealcrypto.codeplex.com/. (2016).

[Liu et al. 2015] Z. Liu, Y. X. Wang, and A. Smola. 2015. Fast Differentially Private Matrix Factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 171–178.

[McSherry and Mironov 2009] F. McSherry and I. Mironov. 2009. 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*. 627–636.

[Nguyen and Stern 1999] Phong Nguyen and Jacques Stern. 1999. The hardness of the hidden subset sum problem and its cryptographic implications. In *Advances in Cryptology – CRYPTO'99*. Springer, 31–46.

[Nikolaenko et al. 2013] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh. 2013. Privacy-preserving Matrix Factorization. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. 801–812.

[Tang and Wang 2016] Q. Tang and H. Wang. 2016. Privacy-preserving Hybrid Recommender System. Cryptology ePrint Archive: Report 2016/1134. (2016).

[Tang and Wang 2015a] Q. Tang and J. Wang. 2015a. Privacy-Preserving Context-Aware Recommender Systems: Analysis and New Solutions. In *Computer Security - ESORICS 2015 (LNCS)*, Vol. 9327. Springer, 101–119.

[Tang and Wang 2015b] Q. Tang and J. Wang. 2015b. Privacy-preserving Friendship-based Recommender Systems. Cryptology ePrint Archive: Report 2015/1152. (2015).

[Veugen et al. 2015] T. Veugen, R. de Haan, R. Cramer, and F. Muller. 2015. A Framework for Secure Computations With Two Non-Colluding Servers and Multiple Clients, Applied to Recommendations. *IEEE Transactions on Information Forensics and Security* 10, 3 (2015), 445–457.

[Weinsberg et al. 2012] U. Weinsberg, S. Bhagat, S. Ioannidis, and N. Taft. 2012. BlurMe: inferring and obfuscating user gender based on ratings. In *Sixth ACM Conference on Recommender Systems*. 195–202.