Knowl Inf Syst (2012) 30:405–433
DOI 10.1007/s10115-011-0395-3

REGULAR PAPER

# Privacy-preserving hybrid collaborative filtering on cross distributed data

**Ibrahim Yakut · Huseyin Polat**

Received: 19 December 2009 / Revised: 21 February 2011 / Accepted: 6 March 2011 /
Published online: 7 April 2011


**Abstract** Data collected for collaborative filtering (CF) purposes might be cross distributed between two online vendors, even competing companies. Such corporations might want to integrate their data to provide more precise and reliable recommendations. However, due to privacy, legal, and financial concerns, they do not desire to disclose their private data to each other. If privacy-preserving measures are introduced, they might decide to generate predictions based on their distributed data collaboratively. In this study, we investigate how to offer hybrid CF-based referrals with decent accuracy on cross distributed data (CDD) between two e-commerce sites while maintaining their privacy. Our proposed schemes should prevent data holders from learning true ratings and rated items held by each other while still allowing them to provide accurate CF services efficiently. We perform real data-based experiments to evaluate our proposals in terms of accuracy. The results show that the proposed methods are able to provide precise predictions. Moreover, we analyze our schemes in terms of privacy and supplementary costs. We demonstrate that our schemes are secure, and online overhead costs due to privacy concerns are insignificant.

**Keywords** Privacy · Cross distributed data · Hybrid collaborative filtering · Accuracy · Performance

## 1 Introduction

Collaborative filtering (CF) is a relatively new technique for filtering and recommendation purposes. With increasing popularity of the Internet and e-commerce, CF techniques are increasingly becoming a central theme of online shopping sites. Those sites leverage CF systems to provide predictions to their customers. Shoppers prefer those online vendors providing truthful and trustworthy recommendations. Furthermore, they search more products to

I. Yakut · H. Polat (✉)
Computer Engineering Department, Anadolu University,
26470 Eskisehir, Turkey
e-mail: polath@anadolu.edu.tr

purchase. Thus, e-commerce sites can increase their sales and/or profits through successful CF systems.

The major idea behind CF is that an active user ($a$) will prefer those items that like-minded users prefer, or that dissimilar users do not. CF systems provide a prediction to $a$ about an item, referred to as the target item ($q$), that she has never purchased before based on the preferences of a community of users ([15]). After collecting customers' preferences about products they bought, CF systems construct an $n \times m$ user-item matrix ($D$), where $n$ and $m$ represent the number of users and items, respectively. Using $D$, $a$'s known ratings, and her query (for which item she is looking for prediction, $q$), the systems then provide predictions for $a$ on $q$, referred to as $p_{aq}$.

In case of inadequate data, it becomes a challenge to produce recommendations for all items, which leads to very low coverage. Similarly, some vendors might face with the cold start problem in which they might not able to provide predictions due to insufficient data. Since similarities between any two users or items are estimated over commonly rated items by both users or the set of users who rated both items, respectively, insufficient data make it difficult to find large enough commonly rated items. Similarities computed over a small number of commonly rated items then can be considered unreliable. Moreover, to have a large enough neighborhood, data owners should have sufficient number of users. Inadequate data might lead to poor neighborhoods. Thus, when companies own scarce data for CF services, they want to integrate their data. However, due to privacy, legal, and financial reasons, they do not want to reveal their data. Likewise, some organizations might not have the expertise of data mining so that they want to outsource data mining tasks to external service providers while protecting their business intelligence and customer privacy ([39]).

Online vendors collect ratings from various users for different items. Customers' preferences about products held by companies are considered online vendors' private data. Such companies make effort to keep that data private. Users' preferences about different products may help companies to profile their customers in such a way to increase their sales and/or profits. Online retailers offer different discounts and coupons to their customers based on their profiles. Since revealing users' preferences may cause financial losses, data collected for CF purposes are considered valuable asset. Each company is responsible for protecting the collected ratings about their customers, and data transfers may not be possible due to legal reasons. Due to aforementioned reasons, it becomes imperative to propose privacy-preserving schemes for producing predictions based on distributed data between two companies.

Providing services on distributed data is very attractive in e-commerce, collaborative research, and health care. Distributed computations might be inevitable in solving life threading problems such as efficient disease control, effective public safety, facilitating e-commerce, and sharing scientific research data ([8,4]). Similarly, since existing secure information sharing protocols and technologies fail to provide sufficient incentives for government agencies to share information, there is serious information sharing problem among the federal government agencies, and this problem can cause substantial deficiencies in terrorism attack detection ([29]).

Customers' preferences about various products might be distributed between two parties, even rival companies. This partition can be horizontal or vertical. In practice, such data distribution can be combination of both horizontal and vertical partitioning, referred to as cross partitioning. Data owners might end up with cross distributed data (CDD), as shown in Fig. 1, where $n_1 + n_2 = n$ and $m_1 + m_2 = m$. Suppose that there are two e-commerce companies, $A$ and $B$, where both companies sell the same products to the same community of customers. As seen in Fig. 1, $A$ holds the ratings of the users from $u_1$ to $u_{n_1}$ for the items from $i_1$ to $i_{m_1}$ and the ratings of the users from $u_{n_1+1}$ to $u_n$ for the items from $i_{m_1+1}$ to $i_m$, while $B$

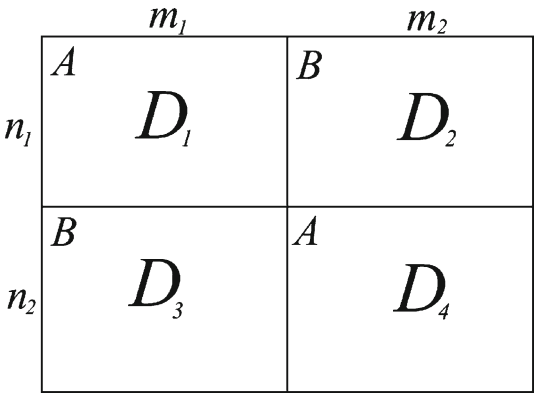


**Fig. 1** CDD: cross distributed data

owns the remaining ratings. $A$ and $B$ can end up with CDD, shown in Fig. 1, under one of the following conditions:

- $A$ makes discounts for the items from $i_1$ to $i_{m_1}$, while, at the same time, $B$ makes discounts for the items from $i_{m_1+1}$ to $i_m$. During these sales campaigns, users from $u_1$ to $u_{n_1}$ buy and rate corresponding discounted items from $A$ and $B$, respectively. After these discounts are over, $A$ makes discounts for the items from $i_{m_1+1}$ to $i_m$, while, at the same time, $B$ makes discounts for the items from $i_1$ to $i_{m_1}$. Another group of customers, users from $u_{n_1+1}$ to $u_n$, then buy and rate corresponding discounted items from $A$ and $B$, respectively. Such sales offerings then lead to CDD between $A$ and $B$, as shown in Fig. 1.
- Customers choose vendors to buy various products depending on some parameters and intuition ([7,50]). This fact causes profiling of users. There are two types of user profiles, $U_1$ and $U_2$, which have different purchasing behaviors. Users in $U_1$ (including $n_1$ users from $u_1$ to $u_{n_1}$) select companies $A$ and $B$ to buy items from $i_1$ to $i_{m_1}$ and items from $i_{m_1+1}$ to $i_m$, respectively. Unlike users in $U_1$, users in $U_2$ (including $n_2$ users from $u_{n_1+1}$ to $u_n$) select e-commerce sites $A$ and $B$ to buy items from $i_{m_1+1}$ to $i_m$ and items from $i_1$ to $i_{m_1}$, respectively. Such purchasing profiles then result CDD between $A$ and $B$, as shown in Fig. 1.
- Huang et al. [16] and Shapira et al. [42] show that web users' online exploration behavior is highly correlated and informative. Thus, customers may select different e-commerce sites at different times to purchase various products. They do not want to give their data to one online vendor only. In this way, they can have more control over their purchasing history, ratings, interested items, viewed items, view duration, and so on. A group of customers ($n_1$ users) give their ratings for $m_1$ and $m_2$ items to $A$ and $B$, respectively, while another group of users ($n_2$ customers) provide their ratings for $m_2$ and $m_1$ items to $A$ and $B$, respectively. Such types of privacy concerns lead to CDD between $A$ and $B$, as shown in Fig. 1.

There are memory- and model-based CF algorithms. Each group of algorithms has its own advantages and disadvantages ([44]). Since memory-based algorithms operate on entire data online, for large datasets, online performance degrades and scalability problems occur. Accuracy and coverage might get worse when data are sparse because they depend on the existence of ratings for co-rated items. However, they are easily implemented, and new users and/or items can be easily and incrementally added. Moreover, accuracy is better compared to model-based ones. Model-based methods generate a model off-line. Thus, their online performance is better. Furthermore, methods in this type better address sparsity, scalability, and

coverage problems. Some computations can be done off-line to improve online performance while predictions should be generated using memory-based methods. Hybrid schemes are proposed to gain the advantages of both memory- and model-based algorithms while decreasing the effects of disadvantages mentioned above ([44]). Pennock et al. [34] experimentally show that the hybrid scheme they proposed outperforms both types of schemes in accuracy while ensuring scalability. Their proposed method is in the hybrid manner, which realizes some computations off-line to improve online performance while generates predictions using memory-based methods.

In this study, we focus on how to produce high-quality referrals on hybrid CF approaches efficiently from CDD while ensuring data holders' privacy. Our scheme first estimates a model off-line, and then predictions are provided based on the model using a memory-based approach. Since accuracy, privacy, and efficiency are conflicting goals, we propose schemes providing equilibrium among them. Intuitively, the key requirement is to obtain preferable accuracy gain due to data integration despite accuracy losses due to privacy measures. Supplementary costs due to privacy concerns should be small and still make it possible to offer loads of referrals to many users efficiently. In other words, the proposed schemes must run and respond to each query in acceptable time with little online extra costs.

## 2 Related work

The evolution of data mining (DM) has brought up some problems like *privacy* in DM applications. To overcome that problem, many studies have been proposed in the area of privacy-preserving data mining (PPDM). Evfimievski [11] proposes randomization-based solutions to ensure privacy and shows that they can be utilized in some DM applications. Pinkas [35] presents how cryptographic techniques can be used for PPDM. In addition to randomization and cryptographic techniques, anonymity-based schemes have been developed to protect individuals' privacy especially in statistical databases. Sweeney [45] proposes $k$-anonymity, which utilizes generalization and suppression techniques to guarantee anonymity in set having size of $k$ individuals. Aggarwal and Yu [1] present several schemes that are used to preserve privacy in the PPDM literature. Generally speaking, different PPDM studies have applied above-mentioned techniques for privacy preservation. In this study, we apply fake or default ratings-based randomization techniques and some cryptographic tools such as homomorphic encryption and 1-out-of-$n$ oblivious transfer protocol (OT). Since we focus on hiding the actual rating values and rated products, we do not utilize anonymization-based techniques like $k$-anonymity, which depends on hiding identifiers and quasi-identifiers to ensure privacy.

Privacy-preserving distributed data mining (PPDDM) has been receiving increasing attention. Yi and Zhang [55] propose a privacy-preserving association rule mining protocol based on a new semi-trusted mixer model. Using secure summation and logarithm, Kantarcioglu and Vaidya [22] present schemes for learning distributed naïve Bayes classifier (NBC) securely. Merugu and Ghosh [31] present privacy-preserving framework for clustering distributed data in unsupervised and semi-supervised scenarios. Su et al. [43] propose a two-party $k$-means via secure approximation. The authors achieve both security and completeness together with good efficiency. Duan and Canny [9] propose an effective zero knowledge tools for PPDDM. They also present a practical private vector addition, which is a surprisingly general tool for implementing many algorithms prevalent in distributed data mining. Teng and Du [46] propose a hybrid multi-group approach, where their hybrid approach combines the strength of the secure multi-computation party approach and the randomization approach to achieve

both high accuracy and efficiency. Liu et al. [28] explore the possibility of using multiplicative random projection matrices for PPDDM. For fully distributed data configuration, Lin et al. [27] propose a method controlling data sharing, preventing disclosure of individual data items or any results that can be traced to an individual site. The authors in [23] present a multi-party game, where the parties try to maximize their own goals.

Kantarcioglu and Clifton [20] address secure mining of association rules over horizontally partitioned data (HPD). They also investigate how to compute $k-nn$ classification from distributed data while preserving data sources' privacy ([21]). Kaya et al. [24] present a distributed clustering protocol for HPD based on a very efficient homomorphic additive secret sharing scheme with privacy. The authors in [17] propose a privacy-preserving clustering method based on the dissimilarity matrix construction using a secure comparison protocol on HPD for numerical, alphanumeric, and categorical type attributes. Similar works are proposed at the same topic by Yang and Huang [53]. Yi and Zhang [54] consider privacy-preserving NBC for HPD and propose a two-party protocol and a multi-party protocol to achieve it. Luo et al. [30] study the problem of the distributed training of a model-based classifier when the data records are stored by multiple data sites as horizontal partitioned manner, each of which requires its local data records to be maintained private. They claim that their scheme is more efficient than the previously proposed ones.

Vaidya and Clifton [48] present privacy-preserving scheme to mining association rules over vertically partitioned data (VPD). Alternative approach for the same subject is proposed by Rozenberg and Gudes [40]. In order to learn the Bayesian network structure for distributed heterogeneous data, a privacy-preserving scheme is proposed by Wright and Yang [51]. In his thesis, Vaidya [49] extensively studies how to perform various data mining functionalities based on VPD with privacy. Vaidya et al. [47] introduce a generalized privacy-preserving variant of the ID3 algorithm for VPD distributed over two or more parties. Amirbekyan and Castro [2] propose a private scheme for $k-nn$ queries over VPD. They prove and claim that their solution is more efficient and practical than the previously proposed ones.

Jagannathan and Wright [18] introduce the concept of "arbitrarily partitioned data (APD)," which is the hybrid version of the vertical and horizontal partitioning. In their study, they present privacy-preserving protocol for $k$-means clustering in this type of setting of data. For the similar clustering goal, Prasad and Rangan [36] develop a privacy-preserving BIRCH algorithm on APD. Han and Ng [14] propose a privacy-preserving decision tree induction algorithm on APD among multi-parties. Bansal et al. [3] present a privacy-preserving algorithm for the neural network learning when the data set is arbitrarily partitioned between two parties. In this study, our data configuration is simpler and special case of APD, which we call it "cross distributed data-CDD."

Distributed data-based CF with privacy is becoming popular with increasing popularity of e-commerce. Polat and Du [38] show how to offer top-$N$ recommendation based on HPD or VPD between two parties without deeply violating data owners' privacy. Our study is different from their work. They consider HPD or VPD, while we consider CDD. We consider numeric ratings, while they utilize binary ratings. Moreover, they offer sorted list of referrals, while our schemes produce predictions for single items. Polat and Du [37] discuss how to provide predictions for single items based on VPD between two parties while preserving their privacy. The authors consider VPD, while we investigate how to offer predictions with privacy based on CDD. Furthermore, they consider all users in the database as neighbors and utilize entire users' data for prediction computations, while we propose privacy-preserving schemes to select $a$'s neighbors given a set of $n$ users. Since they utilize all users' data, they are able to conduct some computations off-line. Although we perform prediction computations on selected users' data, the parties can still be able to conduct some computations like

selecting the best neighbors off-line to improve online performance. Yakut and Polat [52] investigate how to produce singular value decomposition (SVD)-based private recommendations on HPD or VPD between two data owners. The authors utilize SVD-based CF scheme to produce referrals based on partitioned data between two parties only while protecting their privacy. Unlike our study, they consider VPD or HPD. Kaleli and Polat [19] investigate how to achieve NBC-based CF tasks on distributed data with privacy. The authors utilize binary ratings and NBC-based CF algorithm to generate referrals, where the scheme determines whether $a$ will like $q$ or not, while our scheme determines how much $a$ will like or dislike $q$. Unlike their schemes, we consider CDD.

## 3 Background

CF systems first calculate the similarities between $a$ and each user in $D$ using a similarity measure. Users $a$ and $i$ can be thought of as two vectors in an $m$ dimensional item-space. The similarity between them ($w_{ai}$) can be calculated by computing the cosine of the angle between these two vectors, as follows ([41]):

$$w_{ai} = \cos(\vec{r_a}, \vec{r_i}) = \frac{\vec{r_a} \cdot \vec{r_i}}{\|\vec{r_a}\| \times \|\vec{r_i}\|}, \tag{1}$$

where the $\cdot$ represents the dot-product operation, $\|\vec{r_x}\|$ represents the Euclidean length of the vector $\vec{r_x}$, which is the square root of the dot product of the vector with itself, and $\vec{r_a}$ and $\vec{r_i}$ are users $a$'s and $i$'s ratings vectors, respectively. To select a subset of the users as neighbors of $a$, Herlocker et al. [15] propose the following scheme, referred to as the best-$N$: select the best $N$ correlates for a given $N$ as neighbors. The prediction for $a$ on item $q$ ($p_{aq}$) can be computed, as follows ([15]), where $v_{\text{norm}_{iq}} = v_{iq} - \overline{v_i}$, $n_a$ shows the number of $a$'s neighbors who rated $q$, $\overline{v_a}$ is the average of $a$'s ratings, and $w_{ai}$ is the similarity weight between users $a$ and $i$:

$$p_{aq} = \overline{v_a} + \frac{\sum_{i=1}^{n_a} v_{\text{norm}_{iq}} \times w_{ai}}{\sum_{i=1}^{n_a} w_{ai}}. \tag{2}$$

In addition to default ratings-based randomization, we employ homomorphic encryption schemes to achieve privacy. Suppose that $\xi$ is an encryption function, $e$ is a public key, and $q_1$ and $q_2$ are private data values. Homomorphic encryption property allows an addition operation to be conducted on the encrypted data without decrypting them, as follows: $\xi_e(q_1) \times \xi_e(q_2) = \xi_e(q_1 + q_2)$. Homomorphic cryptosystems are useful. We utilize one of them in our framework, which is proposed by Paillier [33]. We also utilize OT protocol, which refers to a protocol, where at the beginning of the protocol one party, Bob, has $n$ inputs $X_1, X_2, \ldots, X_n$ and at the end of the protocol the other party, Alice, learns one of the inputs $X_i$ for some $1 \leq i \leq n$ of her choice, without learning anything about the other inputs and without allowing Bob to learn anything about $i$ ([10]). An efficient OT protocol is proposed by Naor and Pinkas [32], where it could be achieved with polylogarithmic (in $n$) communication time.

## 4 CDD-based recommendations with privacy

All online vendors' catalogs and items in their stocks can be easily reached from the Internet publicly. Thus, item labels make no sense for privacy. Before providing CF services together,

the parties agree on unique customer IDs that they hold and share the same unique item labels. We assume that the parties are semi-honest following the proposed protocols' steps. However, they can save and evaluate any obtained information, partial results, or final outcomes. Although it is not an easy task to define privacy succinctly, we can describe it, as follows: The parties should not be able to learn the actual rating values held by each other and the rated items in each other's databases.

Without privacy concerns, $A$ and $B$ can integrate their own data and provide predictions. However, due to privacy, legal, and financial concerns, they do not want to disclose their data to each other. First of all, revealing private data might leak information about the company. Collected ratings are considered the companies' private data. Secondly, customers do not want their data to be transferred. It is e-commerce sites' responsibility to preserve and keep users' data in private. Otherwise, data transfers may cause legal problems. Finally, ratings collected for CF purposes are valuable asset, which can be used to profile customers for developing marketing and advertising strategies. Revealing rated or unrated items poses financial risks, because other company can put those unrated items on sales or send discount coupons to customers. Therefore, on one hand, it is imperative for data owners to protect their data against each other. On the other hand, the companies should be able to produce precise and reliable predictions efficiently to their customers, because users prefer those sites with true recommendations and they search more items to purchase. Providing truthful and dependable referrals is possible only if the sites have sufficient data. Even though two companies might be rivals, they might decide to collaborate to get competing advantages over other companies if their privacy is preserved. Besides privacy and accuracy, efficiency is also vital. Since off-line costs are not critical for overall performance, our schemes perform some works off-line.

Before giving the details of our proposed scheme, we can briefly define the problem that we want to solve, as follows: *When data collected for CF purposes are distributed between two parties, how do such companies offer recommendations based on their integrated data without revealing true ratings and the rated items in their databases to each other?* Our proposed solution consists of off-line model construction and online prediction estimation. Off-line computation functions include mean, deviation from mean normalization, and cosine similarity for determining similar items. Similarly, during online phase, cosine similarity is utilized for estimating user–user similarities. In addition, active users' neighbors are determined. Finally, using a prediction algorithm, a referral is computed. Our main privacy constraint is that any party learns nothing about individual rating values and the rated items held by the other company. Auxiliary confidentiality restriction can be defined as no exchanged intermediate computation value allows parties inferring information conflicts the main privacy control. While searching for a solution to the above-mentioned problem, there are major challenges that should be considered. The first one is providing high-quality predictions while preserving privacy. Second, the proposed scheme must ensure online efficiency. Finally, besides protecting individual votes, rated items should also be guarded.

### 4.1 Off-line process

The first task in off-line phase is data normalization using deviation from mean method ([15]). To do that, the parties need the mean ($\overline{v_i}$) of each user $i$'s ratings for all $i = 1, 2, \ldots, n$. Suppose that user $i$ rated $h$ items. Then, $\overline{v_i}$ of such ratings can be computed, as follows:

$$\overline{v_i} = \frac{\sum_{j=1}^{h} v_{ij}}{h}. \tag{3}$$

Since data are cross distributed between $A$ and $B$, the parties need to find a way to compute $\overline{v_i}$ values for all users securely. The mean is an example of algebraic measures, which can be computed by applying two or more distributive measures. An important property of distributive measures like sum, count, and so on, is that they can be computed by partitioning the entire set into smaller subsets, computing the measure for each subset, and merging their results. Note that, as seen from Eq. 3, $\overline{v_i} = sum/count$. Therefore, the parties are able to compute $\overline{v_i}$ for all $i = 1, 2, \ldots, n$, as follows in a distributive manner:

$$\overline{v_i} = \frac{\sum_{j=1}^{M_i} v_{ij}}{M_i} = \frac{\sum_{j=1}^{M_{i_A}} v_{ij} + \sum_{j=1}^{M_{i_B}} v_{ij}}{M_{i_A} + M_{i_B}}, \tag{4}$$

where $M_i$ shows the number of items rated by user $i$, $M_{i_A}$ and $M_{i_B}$ represent the number of user $i$'s ratings for items held by $A$ and $B$, respectively, and $M_i = M_{i_A} + M_{i_B}$.

The parties first compute the corresponding aggregate data ($sum$ and $count$). They then exchange them and finally compute the final outcomes. However, they want to achieve such tasks without deeply jeopardizing their privacy. Note that since $a$ can ask predictions from either party, each party should be able to return a prediction to $a$, based on the distributed computation with the other party. Also note that both parties should exchange data during data normalization. Otherwise, they might not join the distributed CF process. Therefore, we propose the following privacy-preserving scheme, in which we add fake or default ratings to data holders' databases, to calculate the $\overline{v_i}$ values in a secure manner:

1. Each party $j$ selectively or uniformly randomly chooses a $\gamma_j$ value over the range $(1, 100]$.
2. Each party $j$ then uniformly randomly generates a random value $\delta_j$ over the range $(1, \gamma_j]$.
3. The companies selectively or uniformly randomly choose some of their unrated items' cells to fill with fake ratings ($f_r$) based on $\delta_j$. Note that $C_g^f = C_{f-g}^f$, where $C_g^f$ shows the number of ways picking $g$ unordered outcomes out of $f$ possibilities. Therefore, if $\delta_j < (100 - 2d_j)/2$, then the party sets $\delta_j$ at the chosen value; otherwise, it sets it at $100 - d_j - \delta_j$, where $d_j$ shows the density, as percent, of the data that company $j$ holds. The parties selectively or uniformly randomly choose $\delta_j$ percent of their unrated items' cells to fill with $f_r$.
4. Each party might decide to use one of the following methods to find $f_r$ in each sub-part of $D$. Note that $D$ might be considered to be split into four sub-parts, where each party holds two of them:
   (a) Generate $f_r$ using the distribution of the ratings residing in that sub-part. After determining the distribution of the ratings and their characteristics in each sub-part, the parties generate $f_r$ using that distribution. Users' preferences about many products usually show Gaussian distribution with mean ($\mu$) and standard deviation ($\sigma$). Therefore, the parties compute the mean and the standard deviation of the ratings in each sub-part and generate $f_r$ using the corresponding values for each sub-part. Note that number of $f_r$ depends on how many cells to be filled, which are determined before.
   (b) Rather than using a distribution to generate $f_r$, determine non-personalized or default ratings ($v_d$) by computing local averages in each sub-part. The parties can utilize one of the following methods to find $v_d$ in each sub-part:
      – Local overall mean: Calculate the overall mean of the ratings in that sub-part as a non-personalized rating.

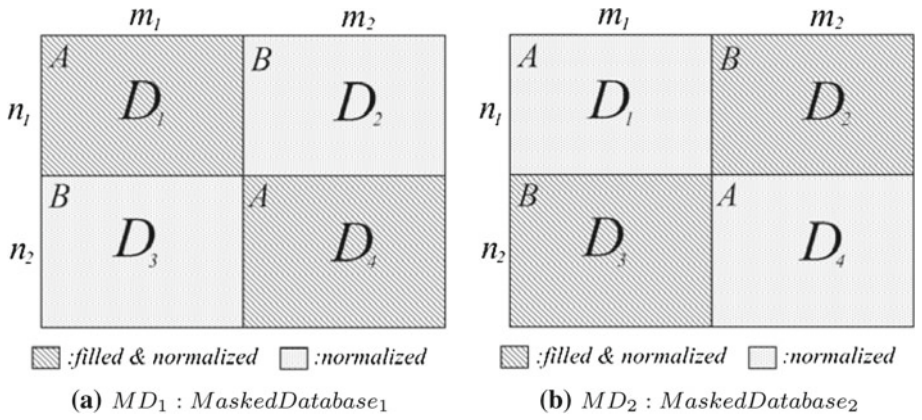


**Fig. 2** Masked normalized databases

- User (row) mean: Compute the mean ratings of each user using their ratings in that sub-part as non-personalized ratings
- Item (column) mean: Compute the mean ratings of each item using their corresponding ratings in that sub-part as $v_d$.

5. After deciding which method to use for determining $f_r$, they generate or find them and fill chosen unrated items' cells with corresponding $f_r$. Besides original databases, the parties end up with filled matrices.
6. Since $a$ can ask prediction from $A$ or $B$, both parties should normalize their data either using deviation from mean method. Moreover, both of them should give aggregate results of its data to the other party and receive results. Therefore, they follow the following steps to normalize their data, where for simplicity, we explain the procedure in terms of $A$ only:
   (a) $A$ finds aggregate values ($\sum_{j=1}^{M_{i_A}} v_{ij}$ and $M_{i_A}$) on filled database for all $i = 1, 2, \ldots, n$ and sends them to $B$.
   (b) After receiving such values, $B$ computes $\sum_{j=1}^{M_{i_B}} v_{ij}$ and determines $M_{i_B}$ values based on its original database.
   (c) $B$ now can estimate $\overline{v_i}$ values for all users using the received aggregate data on filled database from $A$ and the aggregate data on its original data. It then sends $\overline{v_i}$ values to $A$.
   (d) $A$ and $B$ compute $v_{\text{norm}_{ij}} = v_{ij} - \overline{v_i}$ values of filled and original databases, respectively. Therefore, the parties obtain normalized values based on deviation from mean method. They then save such normalized databases, as shown in Fig. 2a.
7. The parties simultaneously follow the same steps, where they switch their roles. Therefore, they end up with normalized databases, as shown in Fig. 2b.

Data holders mask their data by filling some of the unrated item cells with non-personalized votes. It is possible to estimate sum values in Eq. 4 using some secure multi-party computation (SMC) protocols. However, it is hard to implement such cryptographic protocols compared to our schemes. Such cryptographic solutions introduce additional computation costs so that efficiency becomes lower. Moreover, in addition to hiding true rating values, cooperating parties want to hide the rated and/or unrated items, as well. Filling some of the empty cells helps

companies conceal their rated items. Finally, adding default ratings into user-item matrices increases amount of available ratings, which helps provide more dependable predictions and improve coverage.

After data normalization, each party ends up with two normalized databases; one is based on filled, while one is based on original ratings. Since the parties collect ratings over time, they update their data, and off-line process is repeated per particular time. To prevent data deriving about each other's data through such continual computations, data holders remove some of the old ratings and insert the newer ones. To improve the overall performance of the algorithm proposed by Herlocker et al. [15], we propose to utilize hybrid approaches. For this purpose, after data normalization, the parties then construct a model off-line, where the best similar items to each item are found by using cosine similarity measure. When $a$ asks predictions from $A$, the parties work on the model generated from the database ($MD_1$) shown in Fig. 2a, which is called $Model_1$. If $a$ asks predictions from $B$, the parties work on the model constructed from the database ($MD_2$) shown in Fig. 2b, which is called $Model_2$. Then, predictions are estimated based on such model using the proposed memory-based algorithm. Two items $i$ and $j$ can be thought of as two vectors in an $n$ dimensional item-space ([41]). As in Eq. 1, the similarity between two items ($w_{ij}$) can be similarly calculated by computing the cosine of the angle between two vectors. The best similar items to each item are then determined based on such similarity measures. After choosing such similar items to each item off-line, the model is constructed. It is then utilized for prediction computations. Equation 1 can be simplified, as follows:

$$w_{ij} = \cos(\vec{r_i}, \vec{r_j}) = \sum v'_{ik} \times v'_{jk}, \tag{5}$$

where $v'_{ik} = v_{ik} / \sqrt{\sum_{k=1}^{n} v_{ik}^2}$ and $v'_{jk} = v_{jk} / \sqrt{\sum_{k=1}^{n} v_{jk}^2}$.

As seen from Eq. 5, to select the best similar users, normalized ratings are needed. To determine $v'_{ik}$ and $v'_{jk}$ values, the parties need to compute $\sum_{k=1}^{n} v_{ik}^2$ and $\sum_{k=1}^{n} v_{jk}^2$ aggregates, where

$$\sum_{k=1}^{n} v_{ik}^2 = \sum_{k=1}^{n_1} v_{ik}^2 + \sum_{k=1}^{n_2} v_{ik}^2, \tag{6}$$

where $n_1$ and $n_2$ show the number of users held by $A$ and $B$, respectively. Similar equation can be written for $\sum_{k=1}^{n} v_{jk}^2$. $A$ and $B$ need to compute sub-aggregates, $\sum_{k=1}^{n_1} v_{ik}^2$ and $\sum_{k=1}^{n_2} v_{ik}^2$, respectively and exchange them. Since their databases are already masked or filled during data normalization, they compute such aggregates for all items and exchange them. They then determine $v'_{ik}$ and $v'_{jk}$ values. Due to filled databases, they cannot derive information about each other's ratings. Now, they can estimate $w_{ij}$ values, as follows:

1. For similarities between those items in the left or the right halves, the parties can compute $w_{ij}$ values, as follows:
   (a) $A$ finds $\xi_{K_{A_1}}(\sum_{k=1}^{n_1} v'_{ik} v'_{jk}) = \xi_{K_{A_1}}(A_i)$ values for the first item and sends them to $B$, where $\xi$ shows homomorphic encryption function while $K_{A_1}$ is $A$'s public key.
   (b) $B$ similarly finds $\xi_{K_{A_1}}(\sum_{k=1}^{n_2} v'_{ik} v'_{jk}) = \xi_{K_{A_1}}(B_i)$ values for the first item.
   (c) $B$ uses homomorphic encryption and gets $\xi_{K_{A_1}}(A_i + B_i)$ values. Note that there are $m_1 - 1$ such values.
   (d) $B$ permutes them using a permutation function $f_b$ and sends them back to $A$. After receiving them, $A$ decrypts them and determines those satisfying $\tau_1$, where $\tau_1$ is a threshold value.

(e) It then sends those values satisfying $\tau_1$ to $B$. Since $B$ knows the $f_b$, it determines which items whose similarities with the first item satisfy $\tau_1$.
(f) $B$ informs $A$. Now, $A$ and $B$ know the items that are among the best similar items to the first one.
(g) For the other items, the parties follow the similar steps. However, since some of the similarities are already computed, $A$ sends bogus data for known similarities and uses different keys to encrypt the sub-aggregates. $B$ similarly adds bogus data for already computed similarities.
(h) For the items in the right half, the parties follow the same steps; however, they switch their roles.

2. For similarities between those items in the different halves, the parties can compute $w_{ij}$ values, as follows:

(a) For the upper half, the parties perform the followings:
- $A$ permutes $m_1$ items. It hides each item column vector into $X-1$ random vectors. In other words, it creates $X-1$ random vectors for each item vector and hides each true vector into them. It then sends the first group of items vectors to $B$.
- $B$ computes the scalar product between each received vector and each column vector in its database. It then finds $\xi_{K_B}(\sum_X)$ values for all $X$, where $\sum$ shows the scalar product of two vectors; $K_B$ represents $B$'s public key.
- $A$ uses OT to get the required results or sub-aggregates for that item. $A$ follows the same steps for all items.
- $A$ now has $\xi_{K_B}(w_{1m_1+1}), \xi_{K_B}(w_{1m_1+2}), \ldots, \xi_{K_B}(w_{1m})$ and $\xi_{K_B}(w_{2m_1+1}), \xi_{K_B}(w_{2m_1+2}), \ldots, \xi_{K_B}(w_{2m}), \ldots, \xi_{K_B}(w_{m_1m_1+1}), \xi_{K_B}(w_{m_1m_1+2}), \ldots, \xi_{K_B}(w_{m_1m})$.

(b) For the lower half, the parties perform the followings:
- $A$ permutes $m_2$ items. It hides each item column vector into $Y-1$ random vectors. In other words, it creates $Y-1$ random vectors for each item vector and hides each true vector into them. It then sends the first group of items vectors to $B$.
- $B$ computes the scalar product between each received vector and each column vector in its database. It then finds $\xi_{K_B}(\sum_Y)$ values using homomorphic encryption.
- $A$ uses OT to get the required results or sub-aggregates for that item. $A$ follows the same steps for all items.
- $A$ now has the similar values as in the upper half. $A$ now uses homomorphic encryption to get the similarity values.
- It then creates random numbers $r_{A_{ij}}$ and finds $\xi_B(r_{A_{ij}})$. It uses homomorphic encryption to get $\xi_B(w_{ij}+r_{A_{ij}})$ values and sends them to $B$.
- $B$ decrypts them and sends them back to $A$. It now can find $w_{ij}$ values by getting rid of random numbers. It then determines those satisfying $\tau_1$ and informs $B$.

Note that $A$ cannot derive data if $n > m$, because there are $n$ unknowns but $m$ equations. It cannot find a unique solution from given equations. If $n < m$, then the parties should follow the following solution: $B$ adds random numbers to disguised similarity values by $A(w_{ij}+r_{A_{ij}}+r_{B_{ij}})$ generated from the range $[-\tau_1 \times c\%, \tau_1 \times c\%]$, where $c$ is a security parameter. $A$ then correspondingly adjusts $\tau_1$.

### 4.2 Online process

There are three major steps in online phase: calculating $w_{ai}$ values between $a$ and each user $i$ for all $i = 1, 2, \ldots, n$, determining $a$'s neighbors, and computing $p_{aq}$ using $a$'s neighbors' ratings for $q$ and the corresponding $w_{ai}$ values. When $a$ asks a prediction for $q$ from one of the companies, that company acts as a master party (MP). It determines $a$'s ratings mean vote ($\overline{v_a}$). It then normalizes her ratings using deviation from mean approach. Finally, it sends her normalized ratings and $q$ to the other party. For simplicity, we assume that $A$ acts as MP. In other words, the parties work on Model$_1$. When $a$ asks prediction from $B$, the parties follow the same steps by switching their roles, where they work on Model$_2$. The parties answer the query securely, as follows:

1. **Computing $w_{ai}$ values:** Note that $w_{ai}$ values can be computed using the cosine measure in a secure manner. Eq. 1 can be simplified, as follows:

$$w_{ai} = \cos(\vec{r_a}, \vec{r_i}) = \sum_k v'_{ak} \times v'_{ik}, \tag{7}$$

where $v'_{ik} = v_{ik} / \sqrt{\sum_{k=1}^{M_i} v_{ik}^2}$ and $v'_{ak} = v_{ak} / \sqrt{\sum_{k=1}^{M_a} v_{ak}^2}$ and $k$ shows the commonly rated items by both $a$ and $i$. Note that $M_a$ and $M_i$ show the number of rated items by $a$ and user $i$, respectively. Suppose that $a$ asks prediction from $A$. Therefore, it acts as the MP. To find $w_{ai}$ values, the parties need to exchange aggregate data. Due to hybrid distribution, ratings of $q$ are held by both parties. Therefore, each party $j$ computes $\sum_{k \in k_j} v'_{ak} \times v'_{ik}$ values for all users $i = 1, 2, \ldots, n$ and sends the aggregate data, of those users whose ratings for $q$ are held by the other party, to the other party. For example, suppose that the first $n_1$ users' ratings of $q$ are held by $B$ and the last $n_2$ users' ratings of $q$ are held by $A$. In other words, $q$ is one of the last $m_2$ items in Fig. 1. Then, $A$ sends the corresponding similarity weight values or aggregate data for the first $n_1$ users to $B$, while $B$ sends the corresponding similarity weight values for the last $n_2$ users to $A$. They find similarities using cosine measure in a secure manner, as follows:

   (a) $A$ finds $v'_{ak}$ values because it knows $a$'s ratings and sends them to $B$.

   (b) To determine $v'_{ik}$ values, on the other hand, they need to compute $\sum_{k=1}^{M_i} v_{ik}^2$ aggregates, where

$$\sum_{k=1}^{M_i} v_{ik}^2 = \sum_{k=1}^{M_A} v_{ik}^2 + \sum_{k=1}^{M_B} v_{ik}^2, \tag{8}$$

   where $M_A$ and $M_B$ show user $i$'s rated items held by $A$ and $B$, respectively. The parties need to exchange such sub-aggregates to determine $v'_{ik}$ values. They compute such values based on the filled matrices. Note that they masked their original databases using fake ratings during data normalization off-line. Therefore, they cannot derive information about each other's data while exchanging such sub-aggregates. Each party $j$ then computes $\sum_{k=1}^{M_j} v_{ik}^2$ values for all $i = 1, 2, \ldots, n$. They then exchange them and find $v'_{ik}$ values.

   (c) As explained before, since scalar dot product is a distributive measure, the parties can compute $w_{ai}$ values using cosine measure, as follows:

$$w_{ai} = \cos(\vec{r_a}, \vec{r_i}) = \sum_{k \in k_A} v'_{ak} \times v'_{ik} + \sum_{k \in k_B} v'_{ak} \times v'_{ik}. \tag{9}$$

Each party can act as $a$ in multiple scenarios to derive information about the other party's ratings. Hence, we propose the following privacy-preserving scheme to securely calculate $w_{ai}$ values. The parties basically insert some fake normalized ratings into $a$'s normalized ratings vector corresponding part like they do when normalizing their ratings. Considering the same example mentioned above, $A$ fills some of $a$'s unrated items' cells among the first $m_1$ items, while $B$ fills some of $a$'s unrated items' cells among the last $m_2$ items, as follows:

(a) Each party $j$ selectively or uniformly randomly chooses $\alpha_j$ over the range (1, 100].
(b) Each party $j$ uniformly randomly generates a random value $\beta_j$ over the range $(1, \alpha_j]$.
(c) They selectively or uniformly randomly choose $\beta_j$ percent of $a$'s unrated items' cells from the corresponding part to fill with fake normalized ratings ($f_n$).
(d) Each party might decide to use one of the following methods to find $f_n$:
  – Generate $f_n$ values using the distribution of $a$'s normalized ratings.
  – The parties can use default normalized ratings ($f_d$), which are based on non-personalized ratings, as $f_n$: Use item (column), user (row), or local overall mean approach to determine them. They compute the mean normalized ratings of each item using their corresponding values held by that party.
(e) Each party finally fills selectively or uniformly randomly chosen $a$'s corresponding unrated items' cells with $f_n$.

Note that the parties perform those steps whenever $a$ asks a prediction because each party can act as $a$ to derive data. However, the parties can compute the item mean values off-line because they already have the required data to find them. That improves online performance. After adding $f_n$ into $a$'s vector, the parties can securely compute $w_{ai}$ values in a distributed manner. Each party sends the required aggregate data to the other party, which holds the ratings of $q$. The party then computes the final $w_{ai}$ values by adding aggregate data values.

2. **Determining $a$'s neighbors:** In order to determine $a$'s neighbors using the best-$N$ approach, decreasingly sorted order of the $w_{ai}$ values are needed. Therefore, the parties follow the following steps:

(a) We assume that $A$ acts as the MP. The parties exchange the sub-aggregates needed to find $w_{ai}$ values. $A$ and $B$ then compute similarity weights between $a$ and the $n_1$ and $n_2$ users, respectively.
(b) $B$ permutes $n_2$ similarity weights using a permutation function ($f_B$) and sends them to $A$.
(c) $A$ does not know the order of similarity weights due to permutation. However, for $A$, the probability of guessing the correct order of such weights is 1 out of $n_2!$. $A$ then determines the best-$N$ users as neighbors, where $N_1$ and $N_2$ users are selected from the users held by $A$ and $B$, respectively, where $N = N_1 + N_2$. $A$ then sends the chosen $N_2$ weights back to $B$ who can determine which users are selected as neighbors because it knows $f_B$.
(d) The parties now have $a$'s neighbors; however, they do not know which users are selected neighbors by the other party and which neighbors already rated $q$. Those neighbors' ratings who rated $q$ are used in prediction computation.

To further improve privacy or make it more difficult for the parties to learn the selected neighbors, the parties can perform the following: After determining the best $N_1$ and $N_2$ users among the users they hold, respectively, $A$ and $B$ uniformly randomly choose $\psi_A$

and $\psi_B$ percents of those users, who are member of the best-$N$ users but not rated $q$, as neighbors, where $\psi_j$ is an integer between 0 and $d_j$. Since such users do not rate $q$, the parties insert default values for $q$ into such cells and use their data for prediction computation, too.

3. **Computing $p_{aq}$:** $p_{aq}$ can be estimated by using the algorithm proposed by Herlocker et al. [15], as follows: Since $A$ knows $\overline{v_a}$, it only needs the following value to compute $p_{aq}$:

$$P_{aq} = \frac{\sum_{i=1}^{n_a} v_{\text{norm}_{iq}} \times w_{ai}}{\sum_{i=1}^{n_a} w_{ai}}, \tag{10}$$

where $p_{aq} = \overline{v_a} + P_{aq}$ and $v_{\text{norm}_{iq}} = v_{iq} + \overline{v_i}$. Similarly, since $P_{aq}$ can be computed in a distributive manner, it can be written, as follows:

$$P_{aq} = \frac{\sum_{i=1}^{s_A} v_{\text{norm}_{iq}} \times w_{ai} + \sum_{i=1}^{s_B} v_{\text{norm}_{iq}} \times w_{ai}}{\sum_{i=1}^{s_A} w_{ai} + \sum_{i=1}^{s_B} w_{ai}}, \tag{11}$$

where $s_A$ and $s_B$ shows the number of $a$'s neighbors who rated $q$ held by $A$ and $B$, respectively, and $n_a = s_A + s_B$. It can be simply written that $P_{aq} = (X_A + X_B)/(Y_A + Y_B)$. Note that $A$ does not know $v_{\text{norm}_{iq}}$, $w_{ai}$, and $s_B$ values owned by $B$ and which users selected as $a$'s neighbors held by $B$. Similarly, $B$ does not know $v_{\text{norm}_{iq}}$, $w_{ai}$, and $s_A$ values that $A$ holds and which users selected as $a$'s neighbors held by $A$. After computing $X_B$ and $Y_B$ values, $B$ sends them to $A$, which calculates $P_{aq}$. It then can compute $p_{aq}$ by de-normalizing $P_{aq}$ and sends it to $a$.

## 5 Privacy analysis

Our main privacy constraint states that the proposed schemes should hide actual ratings and rated items residing in data owners' databases against each other. Likewise, auxiliary confidentiality restriction affirms that no exchanged intermediate computation value allows parties inferring information conflicts the main privacy control. Collaborating vendors are semi-trusted ones. Furthermore, they can also act as $a$ in multiple scenarios to learn the other party's private data. In the proposed scheme, publicly known values are user and item IDs, each party's public key, partial aggregate results like sum and count, and the final prediction because each party can act as $a$. Similarly, confidential data include both true rating values and rated items.

Our proposed solution consists of two major functions, off-line model generation and online prediction estimation. The output of off-line phase is a model. The parties share the neighbors of each item without exchanging actual weights. On the other hand, the output of online phase, called $p_{aq}$, might be known by both parties because any party can act as $a$. Off-line phase includes protocols like mean, similarity, and neighborhood formation. The output of mean protocol is user mean ratings. Similarity weights between any two pairs of items are obtained after similarity protocol. Finally, the best items are determined for each item during neighborhood formation. Online phase contains three protocols: estimating user-user similarities whose output is similarity weights between $a$ and each user in $D$, forming $a$'s neighborhood whose output is the best $N$ similar users to $a$, and finally estimating $p_{aq}$ whose output is the prediction returned to $a$. In the following, we demonstrate that the parties are not able to derive any data during both off-line and online phases. More specifically, we show that our schemes preserve data owners' privacy during normalization and model

construction conducted off-line and computing $w_{ai}$ values, determining $a$'s neighbors, and calculating $p_{aq}$, which are performed online.

During normalization, the parties estimate user mean values by exchanging partial aggregate results computed on filled databases. Thus, they are not able to derive information about each other's private data due to default ratings-based randomness and exchanged aggregate data. Even if they know the number of ratings during computing $\overline{v_i}$ values, they cannot learn true rating values. Moreover, they do not know the $v_d$ values, because such ratings are estimated from data residing in each party's database. However, the parties can guess the rated or unrated items in each other's databases because they exchange the number of ratings involved in mean or normalization protocol. For example, for the party $A$, guessing the correct $\gamma_i$ is 1 out of 100. After guessing it, the probability of guessing the correct $\delta_B$ values is 1 out of $\delta_B$. Since $A$ knows $n_B$, it can guess the number of filled unrated items' cells with probability 1 out of $(100 \times \gamma_B)$. Then, it can guess the rated items by a single user with the probability of 1 out of $(100 \times \gamma_B \times C_{B_r}^{m_B})$, where $B_r$ represents the number of rated items among the items $B$ owns by a user and $m_B$ shows the number of items held by $B$. The probability for $B$ can be determined similarly. During similarity computations and neighborhood formations, privacy is achieved via permutation, randomization, homomorphic encryption, and OT protocol. The probability of guessing the correct value from perturbed ones is 1 out of $(m-1)!$. Similarly, after permutation, column vectors are hidden into random vectors. The probability of guessing the true column vector from disguised ones is 1 out of $X$ (or $Y$). The value of $X$ or $Y$ can be determined based on how much privacy and off-line performance the parties want. Due to homomorphic encryption and OT protocol, which happen to be secure ([10,32]), the parties cannot derive information about each other's data. At the end, the parties share the best similar items to each item without revealing similarity weights.

In online computations, the parties can act as $a$ in multiple scenarios to learn data residing in the other party's database. In such attacks, the party acting as $a$ changes only one rating in its ratings vector to learn the true votes and rated and/or unrated items in the other party's database during $w_{ai}$ computations. However, to defend themselves against such types of attacks, the parties perturb $a$'s ratings vector, as explained previously. Since $a$'s vector is masked like the parties disguise their databases using default ratings, privacy analysis can be similarly done. During similarity estimation, the parties exchange partial aggregates computed on filled data. Without knowing the number of ratings involved in such calculations, it is not feasible to determine true ratings from such aggregates. Our neighborhood formation protocol for $a$ is also secure. The parties are not able to derive data while determining $a$'s neighbors, because they do not exchange anything. First of all, they do not know how many and which users are selected by the other party as $a$'s neighbors. Second, they do not know which neighbors have already rated $q$, because only those neighbors' data who already rated $q$ are used in prediction estimation. Due to the similar reasons, our $p_{aq}$ estimation protocol also does not violate main confidentiality constraint. Since one of the parties sends aggregate values to the MP during $P_{aq}$ computation; and the MP does not know the values involved in $P_{aq}$ computation and which users selected as neighbors held by the other party, it cannot derive information about the other party's data while computing $P_{aq}$.

## 6 Supplementary costs analysis

Our schemes consist of both off-line and online phases. Unlike online costs, off-line costs are not critical for overall performance. Due to privacy measures, additional costs are inevitable. However, such extra online costs should be small enough to provide predictions efficiently.

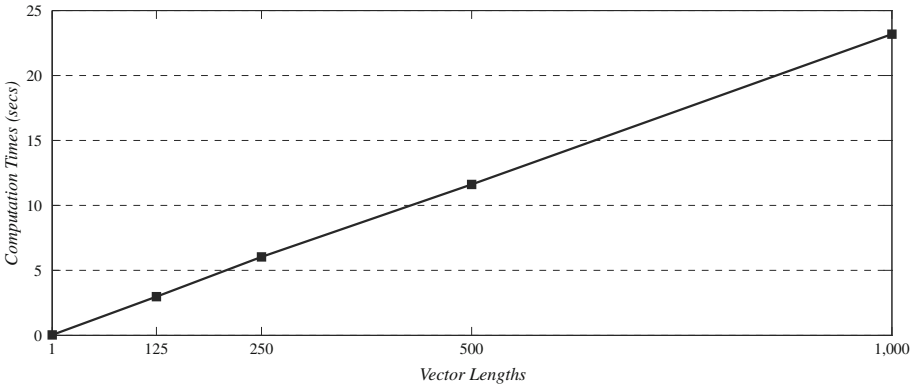


**Fig. 3** Effects of encryption on computation time spent on scalar product

In the following, we analyze our schemes in terms of supplementary storage, computation, and communication (number of communications and amount of data to be transmitted) costs.

Off-line phase consists of two major tasks: data normalization and model construction. In data normalization, the parties fill their sets with fake ratings and estimates user mean votes. Since fake votes are non-personalized ratings, it takes constant time to determine them for each item or user. Thus, due to determining fake ratings, computation costs are in the order of $O(m)$ or $O(n)$ for item or user default votes, respectively. Other computations like randomly selecting some parameters, empty cells, calculating sum and count values, and filling empty cells for normalization are negligible. In off-line process, the parties end up with two matrices. Due to them, supplementary storage costs are in the order of $O(nm)$. Due to fake or default ratings, there is no additional storage cost, because they can be derived. However, each party can determine $f_r$ values off-line and saves them to improve online performance. In this case, due to $f_r$ values, additional storage costs are in the order of $O(m)$ or $O(n)$ for item or user default votes, respectively. To normalize data, the parties communicate with each other. For each matrix, number of communications is two only. Thus, total number of communications is four only for normalization.

Model construction includes estimating similarities between items and determining those items satisfying a pre-determined threshold. Given an $n \times m$ matrix, without privacy concerns, computation costs for calculating similarities between $m$ items are in the order of $O(m^2n)$. Those items satisfying the threshold are selected as the best similar items. With privacy concerns, our proposed model construction scheme includes homomorphic encryption and OT protocol. Other than encryption, decryption, homomorphic encryption, and OT, computations like permutation, creating random vectors, addition, and so on are negligible. Additional off-line computation costs during model creation due to privacy concerns are as follows: The amounts of encryptions and decryptions are in the order of $O(m^2)$. Similarly, number of homomorphic encryptions is in the order of $O(m^2)$. To determine the running times of cryptographic algorithms, benchmarks for the CRYPTO++ toolkit from http://www.cryptopp.com/ can be used ([6]). We performed an experiment for testing the computational time spent on estimating the scalar product of two vectors using homomorphic encryption. We used randomly selected two vectors with varying length from MovieLens Million (MLM) data set, which is described in Experiments section. We varied the vector length from 1 to 1,000 and displayed the computation time in Fig. 3. As expected, with increasing vector length, computation time augments. We did not show the computation times when there

is no encryption, because even if the vector lengths are 1,000, the time to find their scalar product is less than 1 millisecond. As seen from the figure, encryption significantly affects the computation times needed to estimate scalar product of two vectors. Since homomorphic encryption is utilized off-line, such costs are not that critical for the overall performance.

The parties utilize $m_1 + m_2$ number of OT. Thus, number of OT employed is in the order of $O(m)$. For each OT, one of the parties (the sender) performs $2(X + Y)$ double exponentiations, while the other party (the receiver) conducts six exponentiations ([32]). An efficient OT protocol proposed by Naor and Pinkas [32] could be achieved with polylogarithmic (in $n$) communication complexity. During model construction, number of communications between parties is $5(m + 1)$ except communications due to OT. In a prediction process, $a$ sends a message containing her ratings and a query to the CF system, which returns a result. Thus, number of communications is two only. In our scheme, since the parties communicate with each other online, the number of communications is six in total. Due to our proposed schemes, additional amounts of data to be transferred are as follows: First, $a$'s ratings are sent to both parties rather than one of them. Second, on average, $A$ and $B$ exchange $n/2$ aggregate values. Finally, one of the parties sends two aggregate values to the MP.

It is expected that privacy concerns cause extra computation costs. Additional costs due to selection of $\alpha_j$ and $\beta_j$, determining $a$'s unrated items, and filling her vector with $f_n$ values online can be considered negligible. However, when computing $w_{ai}$ values, number of multiplications increases due to inserted $f_r$ values in each company's database and $f_n$ values into $a$'s vector. Due to inserted $f_n$ values, on average, total number of additional multiplications is $(1/2) \times (\alpha_j/2) \times (1/100) \times (m'_a/2) \times d_j$, where $d_j$ shows density rate of the database held by company $j$ and $m'_a$ represents number of $a$'s unrated items. Similarly, due to inserted $f_r$ values, on average, total number of additional multiplications is $(\gamma_i/2) \times (1/100) \times m'_u \times d_j$, where $m'_u$ represents number of unrated items in $D$. Note that $d_j$ is very low and $\alpha_j$ and $\gamma_i$ values are constants less than 100. Note also that our schemes do not cause any auxiliary computation costs while determining $a$'s neighbors and $p_{aq}$ computation. Thus, extra online computation costs due to privacy concerns are acceptable and still make it possible to generate predictions efficiently.

## 7 Experiments

After examining our schemes in terms of privacy and supplementary costs, we want to evaluate the proposed schemes in terms of accuracy and online performance, as well. To achieve such goal, we performed a variety of experiments using two well-known real data sets. We showed that the proposed scheme is secure and theoretically does not introduce significant additional costs. We also need to demonstrate that the scheme is able to offer accurate predictions efficiently while privacy measures are in place.

When data holders own inadequate data, they are limited to generate recommendations for some items. If they agree to collaborate to offer predictions on their distributed data when privacy-preserving measures are introduced, they are more likely to provide referrals to more items. Therefore, we perform experiments to demonstrate how coverage changes through collaboration with varying $n$ values. We also hypothesize that the parties are able to generate more truthful recommendations if they decide to collaborate. To verify this, we conduct experiments based on split data only and integrated data. We plan to show how collaboration between two e-commerce sites affects the quality of the CF services. Hybrid approaches enhance online performance because some works are done off-line. We also utilize a hybrid approach, where the best similar items to each item are selected. Since a smaller number of

items are involved in prediction computations, online performance is expected to improve. Moreover, selecting some of the items for generating predictions affects accuracy, as well. We conduct trials to demonstrate the effects of hybrid approach on online performance and accuracy.

In order to add randomness into data held by each party and active users' data, the parties might use different non-personalized values. In one hand, density of the data increases due to filled default values, which might make accuracy better. On the other hand, such values might not represent users' true preferences, which may make accuracy worse. To understand the effects of various default values and determine the best choice, which gives the most accurate results, we run a variety of trials. As explained previously, privacy and accuracy are two clashing goals. Due to privacy protection measures applied by the vendors, accuracy is expected to be worse. We define some privacy parameters like $\gamma_j$ and $\alpha_j$. They are among the factors that might have an effect on accuracy. To show how the quality of the predictions changes with varying $\gamma_j$ and $\alpha_j$ values, we perform different experiments. Finally, after determining the optimum values of each parameter like $N$, $\gamma_j$, and $\alpha_j$, we perform a final set of experiments to evaluate the overall performance of our proposed schemes.

### 7.1 Data sets and evaluation criteria

We performed various experiments using well-known real data sets Jester and MLM. The results on these data sets can be generalized. Jester ([13]) is a web-based joke recommendation system. MLM was collected by GroupLens at the University of Minnesota (www.cs.umn.edu/research/Grouplens). MLM contains discrete votes, while Jester has continuous ratings. The ratings range from -10 to 10 and 1 to 5 in Jester and MLM, respectively. Although Jester has 100 jokes, MLM has 3,592 movies. On the other hand, MLM and Jester have 7,463 and 73,421 users, respectively. In Jester, almost 44% of the ratings are available. Each user in MLM, on the other hand, has rated at least 20 movies. Jester is much denser than MLM.

First of all, we used examples of the most common criteria for CF like Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE) ([5,12]) as evaluation criteria to evaluate the overall performance in terms of accuracy. To obtain NMAE, we normalize MAEs by dividing them by the difference between the maximum and the minimum ratings. MAE and NMAE measure how close the predictions provided by our proposed schemes with privacy concerns to the ones produced without privacy concerns. The lower they are, the more accurate the results are. Thus, they should be minimized. Second, to show how integrating split data affects the number of items for which predictions could be provided, we utilized coverage metric. Coverage is the percentage of items for which a CF algorithm can provide referrals. And finally, in order to show how online performance varies with hybrid approach, we defined $T$, in seconds, as online time required to generate predictions.

### 7.2 Methodology

Given the whole data sets, we first determined those users who have rated more than 100 and 60 items from MLM and Jester, respectively. We then randomly divided the selected users into training and test sets. Training and test users were randomly chosen from training and test sets, respectively. Number of users and/or items selected for training and testing for each set of experiments might be diverse since different experiments might need different requirements. After selecting test users, five rated items were randomly selected as test items from each test user ratings vector. We withheld the rated items' ratings, replaced their entries with null, and predicted their ratings using our privacy-preserving scheme. Due to randomness,

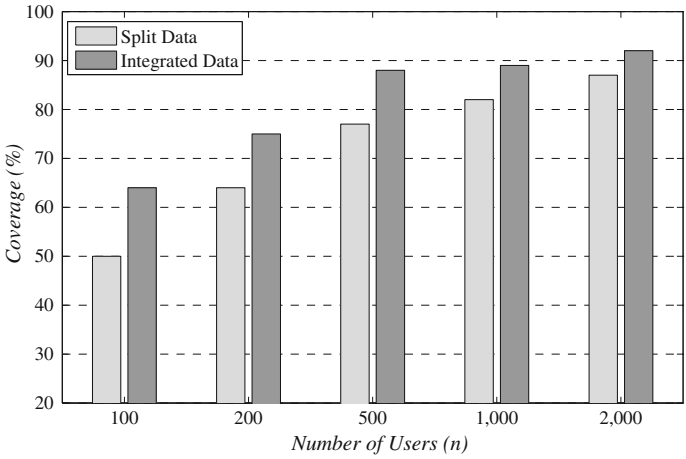


**Fig. 4** Coverage with varying $n$ values

we ran each experiment 100 times to reliably catch the effects of uncertainty. We compared the recommendations provided with privacy concerns with the true user-specified ratings. After calculating MAE, NMAE, and $T$ values, we displayed overall averages.

## 7.3 Experimental results

We explain the details and the outcomes of the experiments that we performed based on two real data sets in the following.

### *7.3.1 Experiment 1-effects of collaboration on coverage and accuracy*

Collaboration between two e-commerce sites having limited amount of data definitely affects coverage. Insufficient data might result very low coverage. With increasing quantity of data, coverage is expected to improve. Therefore, we first conducted experiments to confirm the effects of partnership on coverage and to demonstrate how coverage changes with varying amount of data. For these experiments, we uniformly randomly selected users from given data sets. We utilized both data sets while varying $n$ from 100 to 2,000, where such users were randomly chosen from given data sets. We assumed that predictions can be generated if $q_t$ and $r_c$ are at least one and two, respectively, where $q_t$ shows the number of users who have already rated $q$ and $r_c$ represents the number of commonly rated items between $a$ and the user who has rated $q$. For MLM, we displayed the outcomes in Fig. 4. Since Jester is denser than MLM, when $n$ is 10 or bigger, coverage for integrated data is always 100%. However, when $n$ is 10 and 20, coverages for split data are 89 and 95%, respectively. Although Jester is dense, when $n$ is small (less than or equal to 20), the parties are not able to offer recommendations for all items using their split data only. Through collaboration, however, they can generate predictions for all items.

Our experiment results confirm our premise that coverage improves when vendors offer referrals on CDD. Since MLM is sparse data set, as expected, coverage significantly recovers through collaboration. Also note that it develops with increasing $n$ values. Due to collaboration and increasing $n$ values, amount of ratings available for CF increases, which makes

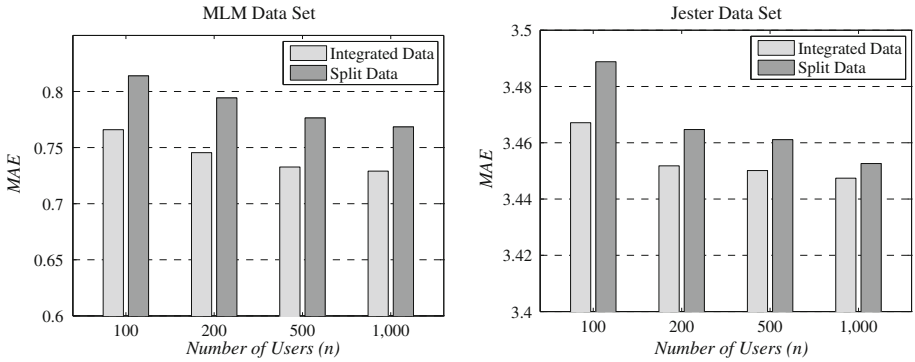


**Fig. 5** MAEs with varying $n$ values

coverage better. Therefore, CF on CDD helps online vendors provide recommendations for more items, even if they have dense data sets.

To give an idea about how integrating split data affects the quality of the referrals, we performed experiments while varying amount of data that each party holds. We ran experiments using both data sets, where we varied $n$ from 100 to 1,000. Note that data are split between two parties, as shown in Fig. 1. We used 500 users for testing while we computed predictions for five rated items for each test user. We first found predictions using the data held by each party only. After calculating MAE and NMAE values for each party, we then averaged them. We finally integrated split data and computed predictions for the same test set. After computing overall outcomes, we compared the results in order to show how collaboration between vendors affects accuracy. Since MAE and NMAE values show similar trends, we showed MAEs in Fig. 5 for both data sets.

As seen from Fig. 5, the quality of the predictions improves with collaboration between parties for both data sets. For MLM data set, which is sparse, MAE values significantly enhance when data owners integrate their split data. However, for Jester, such improvements are very small. This phenomenon can be explained the density of Jester. Since it is very dense compared to MLM, each party is able to offer accurate and dependable recommendations using their own data. As expected, with increasing $n$ values, accuracy improves, as well. When there are 100 users, MAE values are 0.8139 and 0.7659 for split and integrated data, respectively, for MLM. Thus, accuracy improves by 4.80%. For the same cases, NMAE values are 0.2034 and 0.1914, respectively. Similarly, MAE enhances from 0.7684 to 0.7290 through collaboration for MLM when $n$ is 1,000. For Jester, on the other hand, improvements are less than 1% due to data integration. When data collected for CF purposes are sparse, combining CDD between two parties definitely makes accuracy better. We performed $t$-test with significance level being 0.05 to determine whether the improvements due to collaboration are statistically significant or not. For MLM with $n$ being 200, the value of $t$ is 27.06, which is greater than the value of $t$ for 0.05 in the $t$-table. Although the improvements are small for Jester compared to MLM, the enhancements are still statistically significant. Similarly, for Jester with n being 100, the value of $t$ is 2.198, which is still greater than the value of $t$ for 0.05 in the $t$-table.

### *7.3.2 Experiment 2-effects of hybrid approach*

We conducted trials to demonstrate the effects of hybrid method on both performance and accuracy. For these experiments, we fixed $n$ and set it at 500 while varying $N$ from 3,592

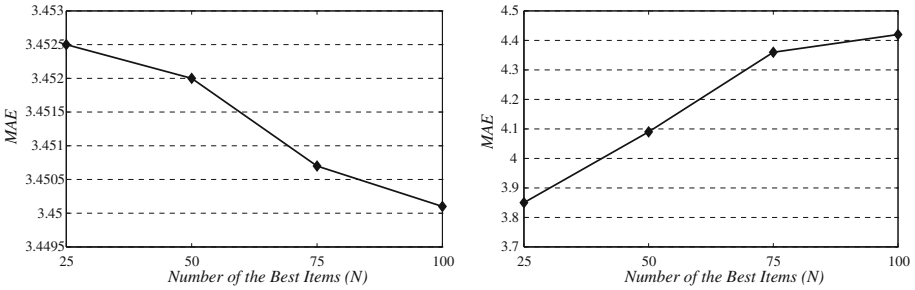


**Fig. 6** MAE and performance with varying $N$ values (Jester Data Set)

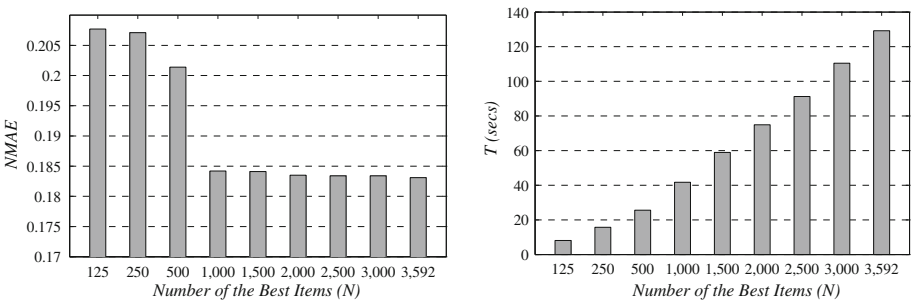


**Fig. 7** NMAE and performance with varying $N$ values (MLM data set)

to 125 and from 100 to 25 for MLM and Jester, respectively. We again used 500 test users and five rated items for each active user as test items. In other words, we generated 2,500 recommendations. First of all, we demonstrated that how the quality of the recommendations and online performance change by applying the hybrid approach for Jester data set. Although we computed MAE and NMAE values, we demonstrated MAE and $T$ values with varying $N$ values for Jester data set in Fig. 6. Similarly, we conducted trials to show how accuracy and performance vary with different $N$ values for MLM. However, since MAE and NMAE show similar tendencies, we displayed NMAE only together with $T$ values for MLM in Fig. 7.

MAE values slightly become worse with decreasing $N$ values for Jester, as seen from Fig. 6. However, such accuracy losses are insignificant because when $N$ values are 100 and 25, corresponding MAE values are 3.4501 and 3.4525, respectively. The loss in MAE values due to varying $N$ from 100 to 25 is 0.0024 only. Unlike MAE, $T$ values improve with decreasing $N$ values, as expected. In order to generate 2,500 predictions using Jester, 4.42 seconds are spent online when $N$ is 100. On the other hand, when $N$ is 25, 3.85 seconds are needed to offer the same number of recommendations. If $N$ is varied from 100 to 50, performance gain is 0.33 seconds while accuracy loss is 0.0019 only. For Jester, we determined 50 as the optimum value of $N$ based on accuracy and online performance. Without sacrificing on accuracy, online performance improves by using the best $N$ items for prediction generation. Since there are 100 jokes only in Jester, improvements due to selecting the best items are limited.

As seen from Fig. 7, NMAE values become worse with decreasing $N$ values. Although NMAE values significantly degrade while varying $N$ from 1,000 to 500 and so on, accuracy almost becomes stable while varying $N$ from 3,592 to 1,000. The same quality can be achieved using the best 1,000 items instead of using the entire items' ratings. As expected,

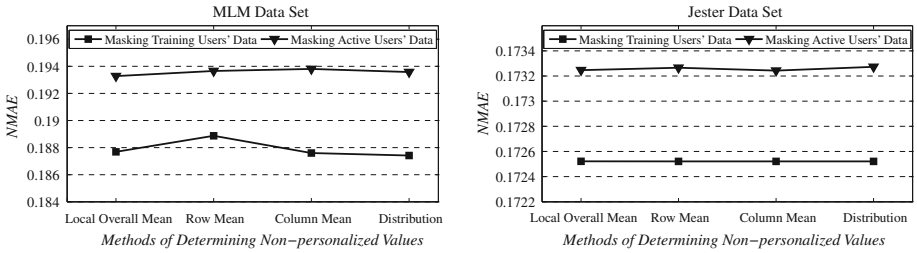


**Fig. 8** NMAE versus methods of determining non-personalized values

online performance improves with decreasing $N$ values because amount of data involved in recommendation computations decreases. Since $T$ enhances significantly when $N$ is 1,000 compared to $N$ being 3,592 and almost the same accuracy is achieved, 1,000 can be considered as the optimum value of $N$ for MLM. For values of $N$ less than 1,000, accuracy losses are expected due to the sparsity of MLM. It becomes a challenge to find enough commonly rated cells with decreasing $N$ values. That leads to inaccurate results.

### *7.3.3 Experiment 3-effects of different non-personalized values*

In order to generate fake values, we propose to use various methods to determine non-personalized votes, which are utilized to fill sparse data sets and $a$'s ratings vector. We first performed experiments for evaluating the effects of default values when they are inserted into train sets. We then ran similar tests to give an idea about how our results change when $a$'s ratings vector is filled with different non-personalized values. We conducted various trials using both data sets. For both data sets, we used 500 users for training and testing, respectively, where we set $\gamma_j$ and $\alpha_j$ values at 50. Since $\beta_j$ and $\delta_j$ values and unrated items' cells are chosen randomly, we ran our experiments 100 times. After calculating overall averages, we displayed NMAE values with various methods of determining non-personalized ratings in Fig. 8.

As seen from Fig. 8, although all four methods give very similar results for Jester, column mean approach achieves the best results for both disguising train and $a$'s data. Without any disguising, NMAE is 0.1727 for Jester for 500 train users. When we disguise train data with non-personalized ratings generated from using column mean approach, NMAE is about 0.1732. Since Jester is a dense set, due to train data disguising with non-personalized votes, accuracy losses can be considered insignificant. Similarly, when we disguise active user's data with non-personalized normalized ratings generated from using column mean approach, NMAE is about 0.1725. In this case, accuracy slightly improves. However, such improvement is trivial. For MLM, utilizing users' data distribution to produce non-personalized ratings to disguise train data achieves the best results compared to other methods. Without any data disguising, NMAE value is 0.1831 for MLM for 500 train users. When we disguise train data with non-personalized ratings generated from using data distribution approach, NMAE is about 0.1874. Although accuracy becomes worse with data perturbation, losses in NMAE values due to data masking are very small. Local overall average method gives the best results for masking active user's data for MLM. When we perturb $a$'s data with non-personalized values generated from using local overall average method, NMAE is about 0.1932. Since MLM is a sparse data set compared to Jester, accuracy losses due to data masking are larger. However, such losses are very small and still make it possible to offer accurate predictions.

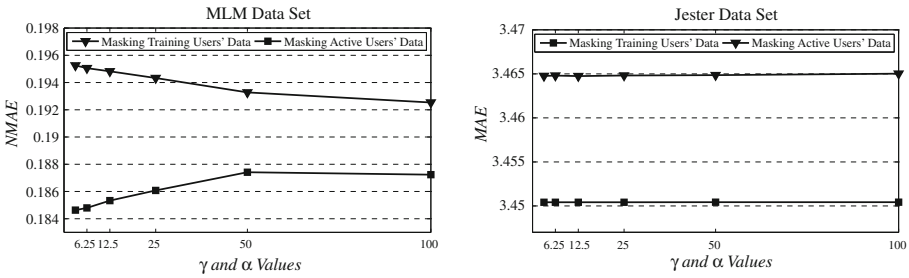


**Fig. 9** NMAE and MAE with varying $\gamma$ and $\alpha$ values

### *7.3.4 Experiment 4-effects of varying $\gamma_j$ and $\alpha_j$ values*

The parties fill some of the blank cells in their databases based on $\gamma_j$ values. In order to show how this affects our results, we performed experiments using both data sets while varying $\gamma_j$ values. We used 500 users for training and testing, respectively, where we used the method that gives the best results to generate non-personalized values to fill unrated items' cells. As we determined in the previous experiments, generating such values based on data distribution and column mean methods for masking train users' data for MLM and Jester, respectively, achieve the best results. Therefore, we used them for data masking. We ran our experiments 100 times. Note that the parties hide active users' data by filling some of their empty cells with default values based on $\alpha_j$ values. After assessing how $\gamma_j$ values affect the outcomes, we performed experiments to evaluate our schemes with varying $\alpha_j$ values because such $\alpha_j$ is another factor that might affect accuracy. We conducted experiments using both data sets while varying $\alpha_j$ values. We used 500 users as train and test users, respectively, where we used column mean and local overall average methods to determine default values for filling some of $a$'s unrated items' cells for Jester and MLM, respectively, because they achieve better results. We ran our experiments 100 times. After computing overall averages, we displayed the outcomes for both data sets in Fig. 9.

In Fig. 9, we showed how accuracy changes with varying $\gamma$ and $\alpha$ values for both data sets. For MLM, when train users' data are perturbed, NMAE values slightly become worse with increasing $\gamma$ values from 3.125 to 100. The best results are obtained when $\gamma$ is 3.125. Although preciseness becomes worse due to data masking, accuracy losses are small as seen from Fig. 9. In case of perturbing active users' data, accuracy slightly enhances with increasing $\alpha$ values. Unlike masking train users' data, $\alpha$ being 100 gives the best results. Unlike MLM, the results are very similar all $\gamma$ and $\alpha$ values for Jester. This phenomenon can be explained due the density of Jester. In order to show the minor differences due to varying $\gamma$ and $\alpha$ values, we displayed MAE values for Jester. Although the outcomes are very similar, the best results are obtained when $\gamma$ is 12.5 and $\alpha$ is 3.125. For both data sets, the results are very promising for all values of $\gamma$ and $\alpha$. The parties can decide their values based on how much accuracy and privacy they want.

### *7.3.5 Experiment 5-overall performance of our proposed schemes*

After evaluating the effects of various factors separately, we finally conducted trials to assess the overall performance of our schemes. In other words, we wanted to give an idea about the joint effects of privacy and accuracy parameters with varying $n$ values. We compared our results with the ones based on split and integrated data without privacy concerns. We

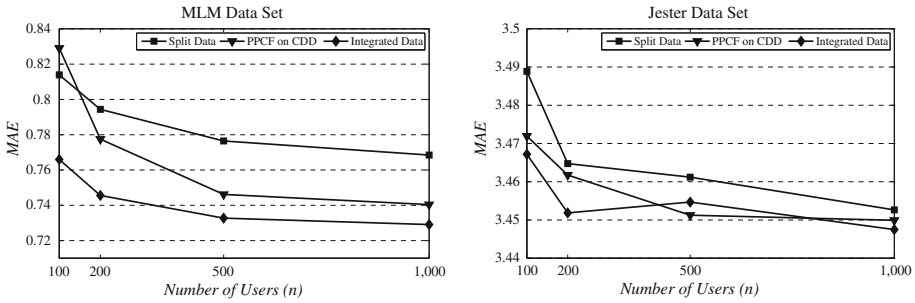


**Fig. 10** Overall performance of PPCF on CDD with varying $n$ values

used both data sets, where 500 users were utilized as test users. We set the values of $\gamma_j$ and $\alpha_j$ at their optimum values that we determined in the previous experiments. We used those methods to determine non-personalized values that give the best results to mask private data. We also fixed $N$ at its optimum values. We ran our experiments 100 times. After computing overall averages, we displayed MAE values for both data sets in Fig. 10.

With increasing $n$ values, the quality of recommendations improves for MLM data set. Although MAE values for $n$ values less than 200 are worse when privacy is protected, PPCF on CDD schemes achieve better results for larger $n$ values. As seen from Fig. 10, the results for split data are the worst due to the insufficient amount of ratings. On the contrary, the outcomes for integrated data are the best, as expected. Due to collaboration, accuracy is expected to become better. However, if data holders offer predictions on their integrated data while preserving their privacy, preciseness becomes worse. For $n$ values larger than or equal to 200, the parties are able to offer predictions with decent accuracy using the proposed PPCF on CDD schemes when they own sparse data. Those companies having insufficient data are able to provide more accurate results when they collaborate while preserving their privacy. The improvements are statistically significant because for $n$ being 500, the value of $t$ is 15.46, which is still greater than the value of $t$ for 0.01 in the $t$-table.

We obtained the similar results for Jester, as well. MAE values become better with increasing $n$ values for all cases for Jester data set. Improvements in accuracy become stable after 500 users. As shown in Fig. 10, collaboration between parties improves accuracy. On the other hand, due to privacy protection measures, the quality of the predictions slightly becomes worse. Compared to the results on split data only, the quality of the referrals on integrated data without privacy concerns and the outcomes of our privacy-preserving scheme are better. Therefore, the parties are able to offer more accurate referrals based on CDD while preserving their privacy than predictions on split data alone. To determine how significant such accuracy gains, we performed $t$-test. When $n$ is 100, the value of $t$ is 1.94, which is still greater than the value of $t$ for 0.1 in the $t$-table.

### 7.4 Discussion

The goal of our study was to improve the quality of the recommendations generated based on a hybrid scheme by using more ratings via distributed configuration while expecting smaller losses in accuracy due to privacy preservation. We mainly focus on the relative accuracy rather than the absolute accuracy. Thus, the empirical results should be examined with the light of this fact. Our results show that our privacy-preserving scheme has achieved our goal.

In the literature, there are various studies comparing different CF approaches. Some examples of such works are as follows: [26] compare user-based, item-based, and hybrid

CF algorithms experimentally. In their experimental configuration, they obtain MAEs of 0.7980, 0.7896, and 0.7791 for user-based, item-based, and hybrid techniques, respectively. According to netflixprize.com, the winner algorithm has root mean square error (RMSE) of 0.8567. The scheme is the improved version of algorithm, combined from the matrix factorization and neighborhood methods, proposed by [25]. [12] find NMAE of 0.187 for Jester. In our scheme, for example, when there are 200 users in MLM, the MAE for split data is about 0.7943, while it is 0.7455 for combined data. For the same case, if privacy concerns are taken into account, the MAE is about 0.7760, as seen from Fig. 10. Similarly, we obtain RMSEs of 1.0120, 0.9421, and 0.9627 for split data, integrated data, and privacy-preserving scheme, respectively. As seen from Fig. 8, for Jester, NMAE is about 0.173, while we obtain NMAEs of about 0.193 and 0.187 for masking $a$'s data and train data, respectively for MLM. Thus, our scheme is able to offer accurate predictions with privacy. Generally speaking, as seen from our $t$-test results, accuracy gains due to collaboration are significant. Although privacy concerns cause accuracy losses, accuracy gains due to collaboration outweigh such losses. Thus, the parties are able to produce more accurate predictions on integrated data without deeply jeopardizing their privacy than the predictions provided on split data only.

## 8 Conclusions and future work

Distributed data-based computations while protecting data owners' privacy are increasingly becoming popular. We presented privacy-preserving schemes to provide predictions with decent accuracy on CDD between two companies. We first explained hybrid partitioning (cross distributed data-CDD) between two online vendors, which is the combination of vertical and horizontal partitioning. Similar partitioning called arbitrary partitioning is defined by Jagannathan and Wright [18]. Such partitioning can be considered as a combination of numerous horizontal and vertical partitioning models. Our data partitioning model is simpler and special version of arbitrary partitioning, where CDD consists of just a vertical partitioning of only two different horizontal partitioned databases, or vice versa. We scrutinized the proposed methods in terms of privacy. Our schemes prevent the data holders from deriving information about each other's databases. The vendors are not able to learn exact ratings and rated items held by each other. Due to privacy protection measures, additional costs are inevitable. Although off-line costs are not that critical for the success of CF systems, we analyzed our schemes in terms of both supplementary off-line and online costs. Our methods cause negligible extra costs, which makes them offer predictions efficiently. To evaluate our schemes in terms of accuracy, we performed a variety of experiments using well-known real data sets. Our results show that they still make it possible to produce precise predictions. The results also confirm that integrating data improves both accuracy and coverage. Through experimental results, we determined the optimum values of privacy and accuracy parameters. We also demonstrated their effects on accuracy. The parties can determine the values of such parameters based on how much accuracy and privacy they want. Each party can also variably mask their data. To sum up, our schemes provide accurate predictions efficiently without greatly violating data owners' privacy.

Generally speaking, to be satisfactory in terms of utility, our proposed scheme should $(i)$ provide pleasing accuracy and coverage, $(ii)$ be efficient especially in terms of online response time, and $(iii)$ ensure privacy. First of all, we empirically demonstrated that the proposed scheme promises accuracy improvements due to contribution of collaborating parties' data even if there are accuracy losses arisen from privacy-preserving process. Accuracy improvements due to collaboration outweigh the losses due to privacy concerns. We also

showed that the improvements are significant using $t$-test analysis. Additionally, the proposed scheme contributes the coverage of the system. Secondly, we theoretically analyzed the scheme in terms of supplementary off-line and online costs. Off-line costs are not that critical for the overall success of the CF systems, and the off-line tasks can be done in decent time. The scheme brings out negligible online overheads due to privacy concerns. About efficiency of off-line process, the proposed computations can be performed in decent time. Finally, we analyzed the scheme in terms of privacy and showed that the parties can offer predictions on their distributed data without jeopardizing their privacy.

We are planning to investigate how to provide predictions using pure model- or memory-based CF algorithms based on CDD with privacy. We will study how to apply our schemes to such algorithms. Although we investigated CDD, we are planning to study other partitioning, as well. In practice, it is possible but not common to have CDD; however, this model is a transition case to the quite convincing arbitrarily partitioning model. We will study producing arbitrarily partitioned data-based CF services without deeply violating data owners' privacy. We will study whether it is possible to offer referrals based on binary ratings, which are cross distributed between two parties while preserving their privacy. Numerical ratings and binary ones may require different approaches to achieve CF services on them when they are distributed between two parties. One important issue that should be addressed is that data overlapping. We assume that ratings are shared distinctly; however, some ratings might be held by two parties. We want to scrutinize how to handle such overlapping and to show performance changes with different amounts of overlapping data. Finally, data might be cross distributed among more than two parties. It should be studied how to offer accurate predictions efficiently when ratings are cross distributed among multiple parties while preserving their privacy.

**Acknowledgments** The work is supported by the Grant 108E221 from The Scientific and Technological Research Council of Turkey (TUBITAK).

## References

1. Aggarwal C, Yu PS (eds) (2008) Privacy-preserving data mining: models and algorithms. Springer Science + Business Media, NY
2. Amirbekyan A, Estivill-Castro V (2009) Practical protocol for Yao's millionaires problem enables secure multi-party computation of metrics and efficient privacy-preserving *k-NN* for large data sets. Knowl Inf Syst 21(3):327–363
3. Bansal A, Chen T, Zhong S (2010) Privacy-preserving back-propagation neural network learning over arbitrarily partitioned data. Neural Comput Appl 20(1):143–150
4. Bhowmick SS, Gruenwald L, Iwaihara M et al. (2006) PRIVATE-IYE: a framework for privacy-preserving data integration. In: Proceedings of the 22nd international conference on data engineering workshops. Atlanta, GA, April 2006, p 91
5. Canny J (2002) Collaborative filtering with privacy via factor analysis. In: Proceedings of the international ACM SIGIR conference. Tampere, Finland, August 2002, pp 238–245
6. Canny J (2002a) Collaborative filtering with privacy. In: Proceedings of the IEEE symposium on security and privacy. Oakland, CA, pp 45–57
7. Chang J, Hung LP, Ho CL (2007) An anticipation model of potential customers' purchasing behavior based on clustering analysis and association rules analysis. Expert Syst Appl 32(3):753–764
8. Clifton C, Doan A, Elmagarmid A et al. (2004) Privacy-preserving data integration and sharing. In: Proceedings of the 9th ACM SIGMOD workshop on research issues in data mining and knowledge discovery. Paris, France, June 2004, pp 19–26
9. Duan Y, Canny J (2008) Practical private computation and zero-Knowledge tools for privacy-preserving distributed data mining. In: Proceedings of SDM 2008 confererence. Atlanta, GA, USA, April 2008, pp 265–276

10. Even S, Goldreich O, Lempel A (1985) A randomized protocol for signing contracts. Commun ACM 28:637–647
11. Evfimievski A (2002) Randomization in privacy-preserving data mining. SIGKDD Explor 4(2):43–48
12. Goldberg K, Roeder T, Gupta D et al (2001) Eigentaste: a constant time collaborative filtering algorithm. Inf Retr 4(2):133–151
13. Gupta D, Digiovanni M, Narita H et al (1999) Jester 2.0: a new linear-time collaborative filtering algorithm applied to jokes. In: Proceedings of the workshop on recommender systems: algorithms and evaluation, international ACM SIGIR conference. Berkeley, CA, USA, August 1999, pp 291–292
14. Han S, Ng WK (2007) Multiparty privacy-preserving decision trees for arbitrarily partitioned data. Int J Intell Control Syst 12(4):351–358
15. Herlocker JL, Konstan JA, Borchers A et al (1999) An algorithmic framework for performing collaborative filtering. In: Proceedings of the ACM SIGIR conference. Berkeley, CA, USA, pp 230–237
16. Huang CY, Shen YC, Chiang IP et al (2007) Characterizing web users' online information behavior. J Am Soc Inf Sci Technol 58(13):1988–1997
17. Inan A, Kaya SV, Saygin Y et al (2007) Privacy-preserving clustering on horizontally partitioned data. Data Knowl Eng 63(3):646–666
18. Jagannathan G, Wright RN (2005) Privacy-preserving distributed *k*-means clustering over arbitrarily partitioned data. In: Proceedings of the 11th ACM SIGKDD international conference on knowledge discovery and data mining. Chicago, IL, USA, August 2005, pp 593–599
19. Kaleli C, Polat H (2007) Providing naïve Bayesian classifier-based private recommendations on partitioned data. Lecture Notes in Computer Science 4702:515–522
20. Kantarcioglu M, Clifton C (2004) Privacy-preserving distributed mining of association rules on horizontally partitioned data. Trans Knowl Data Eng 16(9):1026–1037
21. Kantarcioglu M, Clifton C (2004) privately computing a distributed $k - nn$ classifier. Lecture Notes in Computer Science 3202:279–290
22. Kantarcioglu M, Vaidya JS (2003) Privacy-preserving naïve bayes classifier for horizontally partitioned data. In: Proceedings of the IEEE ICDM workshop on privacy preserving data mining. Melbourne, FL, USA, November 2003, pp 3–9
23. Kargupta H, Das K, Liu K (2007) Multi-party privacy-preserving distributed data mining using a game theoretic framework. Lecture Notes in Computer Science 4702:523–531
24. Kaya SV, Pedersen TB, Savas E et al (2009) Efficient Privacy-preserving Distributed Clustering based on Secret Sharing. Lecture Notes in Computer Science 4819:280–291
25. Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of KDD 2008, Las Vegas, NV, USA, August 2008, pp 426–434
26. Liang Z, Bo X, Jun G (2008) A hybrid approach to collaborative filtering for overcoming data sparsity. In: Proceedings of the 9th international conference on signal processing. Beijing, China, October 2008, pp 1595–1599
27. Lin X, Clifton C, Zhu M (2005) Privacy-preserving clustering with distributed EM mixture modeling. Knowl Inf Syst 8(1):68–81
28. Liu K, Kargupta H, Ryan J (2006) Random projection-based multiplicative data perturbation for privacy-preserving distributed data mining. Trans Knowl Data Eng 18(1):92–106
29. Liu P, Chetal A (2005) Trust-based secure information sharing between federal government agencies. J Am Soc Inf Sci Technol 56(3):283–298
30. Luo H, Fan J, Lin X et al (2009) A distributed approach to enabling privacy-preserving model-based classifier training. Knowl Inf Syst 20(2):157–185
31. Merugu S, Ghosh J (2003) Privacy-preserving distributed clustering using generative models. In: Proceedings of the 3rd IEEE international conference on data mining. Melbourne, FL, USA, November 2003, pp 211–218
32. Naor M, Pinkas B (1999) Oblivious transfer and polynomial evaluation. In: Proceedings of the 31st ACM symposium on theory of computing. Atlanta, GA, USA, May 1999, pp 245–254
33. Paillier P, (1999) Public-key cryptosystems based on composite degree residue classes. Lecture Notes in Computer Science 1592:223–238
34. Pennock DM, Horvitz E, Lawrence S et al (2000) Collaborative filtering by personality diagnosis: a hybrid memory- and model-based approach. In: Proceedings of the 16th conference on uncertainty in artificial intelligence. Stanford, CA, USA, July 2000, pp 473–480
35. Pinkas B (2002) Cryptographic techniques for privacy-preserving data mining. SIGKDD Explor 4(2): 12–19
36. Prasad PK, Rangan CP (2007) Privacy-preserving BIRCH algorithm for clustering over arbitrarily partitioned databases. In: Proceedings of the ADMA 2007 conference. Harbin, China, August 2007, pp 146–157

37. Polat H, Du W (2005) Privacy-preserving collaborative filtering on vertically partitioned data. Lecture Notes in Computer Science 3721:651–658
38. Polat H, Du W (2008) Privacy-preserving top-*N* recommendation on distributed data. J Am Soc Inf Sci Technol 59(7):1093–1108
39. Qiu L, Li Y, Wu X (2008) Protecting business intelligence and customer privacy while outsourcing data mining tasks. Knowl Inf Syst 17(1):99–120
40. Rozenberg B, Gudes E (2006) Association rules mining in vertically partitioned databases. Data Knowl Eng 59(2):378–396
41. Sarwar BM, Karypis G, Konstan JA et al (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international world wide web conference. Hong Kong, May 2001, pp 285–295
42. Shapira B, Elovici Y, Meshiach A et al (2005) PRAW—a PRivAcy model for the web. J Am Soc Inf Sci Technol 56(2):159–172
43. Su C, Bao F, Zhou J et al (2007) Privacy-preserving two-party *k*-means clustering via secure approximation. In: Proceedings of the 21st international conference on advanced information networking and applications workshops. Niagara Falls, Ontario, Canada, May 2007, pp 385–391
44. Su X, Khoshgoftaar TM (2009) A survey of collaborative filtering techniques. Adv Artif Intell Vol:2009
45. Sweeney L (2002) *k*-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl-based Syst 10(5):557–570
46. Teng Z, Du W (2009) A hybrid multi-group approach for privacy-preserving data mining. Knowl Inf Syst 19(2):133–157
47. Vaidya JS, Clifton C, Kantarcioglu M et al (2008) Privacy-preserving decision trees over vertically partitioned data. ACM Trans Knowl Discov Data 2(3):1–27
48. Vaidya JS, Clifton C (2002) Privacy-preserving association rule mining in vertically partitioned data. In: Proceedings of the 8th ACM SIGKDD international conference. Edmonton, Alberta, Canada, July 2002, pp 639–644
49. Vaidya JS (2004) Privacy-preserving data mining over vertically partitioned data. PhD thesis, Purdue University, West Lafayette, IN, USA
50. Van den Poel D, Buckinx W (2005) Predicting online purchasing behavior. Eur J Oper Res 166:557–575
51. Wright RN, Yang Z (2004) Privacy-preserving Bayesian network structure computation on distributed heterogeneous data. In: Proceedings of the 10th ACM SIGKDD international conference, Seattle, WA, USA, August 2004, pp 703–718
52. Yakut I, Polat H (2010) Privacy-preserving SVD-based collaborative filtering on partitioned data. Int J Inf Tech Decis Mak 9(3):473–502
53. Yang W, Huang S (2008) Data privacy protection in multi-party clustering. Data Knowl Eng 67:185–199
54. Yi X, Zhang Y (2009) Privacy-preserving naïve bayes classification on distributed data via semi-trusted mixers. Inf Syst 34(3):371–380
55. Yi X, Zhang Y (2007) Privacy-preserving distributed association rule mining via semi-trusted mixer. Data Knowl Eng 63(2):550–567

## Author Biographies

**Ibrahim Yakut** graduated from Electrical and Electronics Engineering Department at Eskisehir Osmangazi University, Eskisehir, Turkey, in 2005. He received an MSc. degree of Computer Engineering from Anadolu University, Eskisehir, Turkey, in 2008. He is currently a PhD. candidate in the Department of Computer Engineering at Anadolu University. He is an research assistant in the same department. His research interests are privacy-preserving data mining in general, and he specifically focusses on recommender systems and privacy-preserving collaborative filtering.

**Huseyin Polat** is an Assistant Professor in Computer Engineering Department at Anadolu University, Eskisehir, Turkey. He is also department vice chair. He received a BSc. degree from Istanbul Technical University, Istanbul, Turkey in 1997. He got his Master's degree and PhD. from Computer Science Department at Syracuse University in 2001 and 2006, respectively. His research interests are primarily collaborative filtering with privacy, private predictions on selected models, and privacy-preserving data mining in general.