Privacy Preserving Collaborative Filtering by Distributed Mediation

ALON BEN HORIN and TAMIR TASSA*, The Open University of Israel, Israel

Recommender systems have become very influential in our everyday decision making, e.g., helping us choose a movie from a content platform, or offering us suitable products on e-commerce websites. While most vendors who utilize recommender systems rely exclusively on training data consisting of past transactions that took place through them, the accuracy of recommendations can be improved if several vendors conjoin their datasets. Alas, such data sharing poses grave privacy concerns for both the vendors and the users. In this study we present secure multi-party protocols that enable several vendors to share their data, in a privacy-preserving manner, in order to allow more accurate Collaborative Filtering (CF). Shmueli and Tassa (RecSys 2017) introduced privacy-preserving CF protocols that rely on a mediator; namely, a third party that assists in performing the computations. They demonstrated the significant advantages of mediation in that context. We take here the mediation approach into the next level by using several independent mediators. Such distributed mediation maintains all of the advantages that were identified by Shmueli and Tassa, and offers additional ones, in comparison with the single-mediator protocols: stronger security and dramatically shorter runtimes. In addition, while all prior art assumed limited and unrealistic settings, in which each user can purchase any given item through only one vendor, we consider here a general and more realistic setting, which encompasses all previously considered settings, where users can choose between different competing vendors. We demonstrate the appealing performance of our protocols through extensive experimentation.

ACM Reference Format:
Alon Ben Horin and Tamir Tassa. 2021. Privacy Preserving Collaborative Filtering by Distributed Mediation. In *Fifteenth ACM Conference on Recommender Systems (RecSys '21), September 27-October 1, 2021, Amsterdam, Netherlands.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3460231.3474251

1 INTRODUCTION

Collaborative Filtering (CF) is one of the main methods used by recommender systems in order to assist users to navigate through the dazzling abundance of items (products, services, information) available to them, and find the most suitable ones for their tastes and needs [10]. In that method, predictions about the interests of a user are based on aggregated information on preferences of a large set of users. One of the most widely used approaches in CF is to base predictions on characteristics of items. In this so-called *item-based* approach, one uses the collaborative data in order to learn a model of similarity between items; consequently, users are offered items that are similar to previous items that they had already purchased and liked.

To improve the quality of predictions, larger volumes of training data are needed. Hence, it is in the interest of vendors to collaborate and conjoin their historical data in order to issue more accurate recommendations [6]. However, such collaboration may jeopardize the privacy of users, who trust the vendors through whom they purchased or rated items to keep that information confidential. Additionally, the vendors themselves may wish to keep their historical data for themselves, as it has commercial value that they would not like to share with potential competitors. Privacy-Preserving Collaborative Filtering (PPCF) addresses those issues by enabling the use of CF without disclosing private information.

Manuscript submitted to ACM

In this study we propose secure protocols of Multi-Party Computation (MPC) [26] for item-based CF. Our protocols enable the computation of the similarities between items, and then to issue predictions of two types: how a given user would rate a given item, and what are the currently top items to be recommended to a given user.

Shmueli and Tassa [21, 22] have presented such protocols in the mediated model. In that model [2], there exists a mediator that assists in performing intermediate computations, but he is prevented from accessing the actual data due to privacy concerns. Their protocols rely on a *single* mediator, to whom the vendors provide the user-item ratings under homomorphic encryption. Owing to the homomorphic property, the mediator is capable of performing computations on the encrypted data, but thanks to the encryption he remains oblivious to the plaintext underneath. As explained in [22, Section 3], mediation is most advantageous for PPCF: it frees the vendors from the need to communicate with each other, and the need to be constantly online in order to assist other vendors in their recommendation queries; it reduces communication and computational costs; and it enables an economically-realistic collaboration model between vendors that differ in their contribution to the CF training data (as each vendor offers a different set of items to a different set of users), and in their demand from the CF system (in terms of the number and type of queries that they submit).

In this study we also present PPCF protocols in the mediated model. We implement the very same item-based CF technique as in [22]. However, while the protocols in [22] used a single mediator, ours rely on several independent mediators. The advantages of distributed mediation are very significant, as we proceed to explain.

First, while the single-mediator protocols relied on homomorphic encryption, the distributed mediation-based protocols that we present here rely on secret sharing as the cryptographic protection shield. As secret sharing is a linear operation, while homomorphic encryption requires expensive modular exponentiations, the effects on runtime costs are overwhelming. Another aspect that contributes to reducing runtime and communication costs is the size of the underlying arithmetic. Secret sharing can be executed on standard arithmetic (say, 64-bit) because it can be executed over any field which is large enough to represent all possible secret values. However, homomorphic encryption requires arithmetic of at least 512 and preferably also 1024 bits.

Second, the protocols in [22] are vulnerable to malicious collusion, in the sense that if the single mediator colludes with one of the vendors, all private information is revealed to them. In the secret sharing-based protocols that we present here, a similar privacy collapse occurs only if at least half of the mediators betray the trust vested in them and collude. In case the group of mediators has an honest majority, the private information remains fully protected.

Another advantage that our protocols offer is that they free the vendors from any need to communicate with each other. While non-mediated PPCF protocols require a constant communication between the collaborating vendors, and the single-mediator protocols [22] reduced the communication demands only to the offline (and less frequent) phase, our protocols free the vendors from any need of communicating with each other, or even being aware of the number or the identity of other vendors. They only need to communicate with the mediators.

The last contribution that we offer is with regard to the collaborative setting, in the sense of how the user-item rating data is distributed among the vendors. All existing works on PPCF assumed distribution scenarios in which each entry in the user-item rating matrix is owned by just one vendor. Such exclusivity can be found only in unrealistic markets of zero competition: a user who wants to purchase a specific item can do so only through a single vendor. In reality, however, users usually have a choice between competing vendors. We introduce here a much more realistic distribution scenario, which generalizes all previously-considered scenarios, that allows such competition.

The outline of the paper is as follows. We begin with an overview of related work in Section 2, and preliminary discussions and necessary background in Section 3. We present our protocols in Section 4, demonstrate their performance in Section 5, and conclude in Section 6.

2 RELATED WORK

The literature of PPCF may be classified according to the approach that they take towards achieving privacy.

Obfuscation-based methods. In such methods, the user-item data is transformed in some way that prevents individual users from being identified, while maintaining a close level of accuracy in the generated recommendations.

Polat and Du [18] used randomized perturbation techniques to introduce randomness in the data so that users could not be identified with certainty. Similarly, Yakut and Polat [25] proposed a privacy preserving implementation of item-based CF based on the cosine similarity score. They achieve privacy by injecting fake ratings into the distributed matrix of user-item ratings. As a result of this practice, the predictions that their algorithm issues differ from those that the underlying CF method would issue. Another example is the study of Weinsberg et al. [24]; they designed techniques for adding ratings to a user's profile in order to prevent inference of that user's properties (e.g., gender), while having an insignificant effect on the recommendations provided to that user.

Obfuscation-based methods are scalable since the transformations usually only need to be applied to the data at the point of origin, after which the obfuscated data can be used directly. However, the security of these techniques is harder to prove since they rely on randomness and anonymity. Also, they issue inaccurate recommendations as those are computed from obfuscated data.

Clustering-based methods. Such methods rely on grouping the users into clusters and then extracting a representation of that cluster to be used in the CF process. Such methods provide anonymity for the users within each cluster. Another advantage of this approach is scalability, since a reduced representation of the data is used. Larger clusters enable smaller representations, but, on the other hand, they provide less accurate filtering. Examples of studies that suggest clustering-based PPCF methods are [8, 11, 13, 23].

Cryptography-based methods. In that approach, cryptographic means are used in order to protect the sensitive data. Typically, such methods use homomorphic encryption, since such encryption allows performing arithmetic computations on the encrypted values. The study of [22] that we discussed in the Introduction falls under that category. It delegates all CF computations to a mediator, who receives the user-item rating data under homomorphic encryption, and then performs the needed CF computations on the encrypted data. Another cryptographic-based PPCF method was presented by Basu et al. [3]. They proposed a privacy-preserving algorithm that is based on the Slope One predictor [12]. They base the security of their algorithm on additively homomorphic public-key encryption. Their algorithm has an offline pre-computation phase, followed by an online prediction phase. When a vendor wishes to get a predicted rating, all of the vendors must participate in the computation of that prediction.

The studies of [4, 5] present a practical implementation of a PPCF system, based on the Google App Engine for Java (GAE/J) cloud platform. They designed algorithms that rely on a homomorphic encryption scheme to preserve the privacy of user data in the cloud.

While cryptography-based techniques provide strong security guarantees, without sacrificing the accuracy of their results, they do not scale well. As practical systems involve millions of users and items, such techniques may have impractical runtimes, especially in online settings where a short response time is needed.

Ahmad et al. [1] introduced the notion of *distributed trust*, which is similar to the distributed mediation model of computation that underlies our protocols. They aimed at increasing the trust of users in the system: instead of relying on a single server, they distribute the trust among several servers. They achieve privacy by using a threshold homomorphic cryptosystem. As we show here, while trust/security is the primary motivation for distributing mediation, such distribution ushers in an even more meaningful advantage: it enables to replace expensive homomorphic encryption

with much more efficient cryptographic techniques, and thus enable significantly shorter runtimes that could be a key factor in making PPCF a viable practice.

3 PRELIMINARIES

We begin this section by providing the necessary background on item-based CF (Section 3.1). We proceed to describe distributed scenarios, one of which is the general distribution scenario that we consider here (Section 3.2). Next, we describe the setting of distributed mediation (Section 3.3), and conclude with an overview of secret sharing (Section 3.4).

3.1 Item-based collaborative filtering

We provide here a brief introduction to item-based CF [9]. In what follows, we use the following notation agreements:

(1) If r is a non-negative integer then $\xi(r) = 0$ if $r = 0$ and $\xi(r) = 1$ otherwise.
(2) If $\mathbf{x}$ is a vector and f is any scalar function, then $f(\mathbf{x})$ is the vector in which $f(\mathbf{x})(\cdot) = f(\mathbf{x}(\cdot))$.
(3) If $\mathbf{x}$ and $\mathbf{y}$ are two N-dimensional vectors then $\mathbf{x} \cdot \mathbf{y}$ is the N-dimensional vector in which $(\mathbf{x} \cdot \mathbf{y})(n) = \mathbf{x}(n) \cdot \mathbf{y}(n)$, $n \in [N] := \{1, \ldots, N\}$.
(4) Inner products between vectors will be denoted by $\langle \cdot, \cdot \rangle$; the induced norm will be denoted by $\| \cdot \|$.

Let $U = \{u_1, \ldots, u_N\}$ be a set of users (consumers) and $B = \{b_1, \ldots, b_M\}$ be a set of items (products or services). The user-item rating matrix, R, is an $N \times M$ matrix where $R(n, m)$ is either a positive integer which indicates a rating that u_n had given to b_m, or zero if u_n had not rated b_m. In item-based CF, one uses the rating information, as given in R, in order to infer similarities between the items. This similarity model is then used in order to predict how users would rate items that they still had not purchased, or to determine the potentially most appealing items for a given user.

Let S be a symmetric $M \times M$ matrix where $S(\ell, m)$ is the similarity score between items b_ℓ and b_m, $\ell, m \in [M] := \{1, 2, \ldots, M\}$. Then the similarity scores are defined in Definition 1.

Definition 1. *Let $\mathbf{c}_m = (R(n, m) : n \in [N])$ denote the m-th column in the user-item rating matrix R, where $m \in [M]$. Given indices of two items, $\ell, m \in [M]$, let $\mathbf{c}_{\ell|m} := \mathbf{c}_\ell \cdot \xi(\mathbf{c}_m)$ denote the projection of the ℓ-th column of the user-item rating matrix R on the subset of users that rated both items b_ℓ and b_m. Then the cosine similarity score is*

$$S(\ell, m) = \frac{\langle \mathbf{c}_\ell, \mathbf{c}_m \rangle}{\|\mathbf{c}_{\ell|m}\| \cdot \|\mathbf{c}_{m|\ell}\|}, \tag{1}$$

where if $\mathbf{c}_{\ell|m} = \mathbf{0}$ or $\mathbf{c}_{m|\ell} = \mathbf{0}$, $S(\ell, m)$ is set to zero.

The similarity scores are used to predict u_n's rating of b_m as follows. Let:

- $q < M$ be a preset (typically small) integer.
- $N_q(m)$ be the set of indices of the q nearest neighbors of b_m (those with highest $S(\cdot, m)$).
- $N_q^+(m) := \{\ell \in N_q(m) : S(\ell, m) > 0\}$.
- $\mathbf{s}_m$ be the M-dimensional vector for which $\mathbf{s}_m(\ell) = S(\ell, m)$ if $\ell \in N_q^+(m)$ and $\mathbf{s}_m(\ell) = 0$ otherwise.
- $\overline{R(b_m)} = \left[\sum_{n \in [N]} R(n, m)\right] / \left[\sum_{n \in [N]} \xi(R(n, m))\right]$ be the average rating given to item b_m.
- $\mathbf{r}_n$ be the n-th row of the user-item rating matrix R.
- $\bar{\mathbf{r}}_n$ be the vector of u_n's adjusted ratings, i.e. $\bar{\mathbf{r}}_n(\ell) = (R(n, \ell) - \overline{R(b_\ell)}) \cdot \xi(R(n, \ell))$, $\ell \in [M]$.
- $\xi(\mathbf{r}_n)$ be the row vector in which $\xi(\mathbf{r}_n)(m) = \xi(\mathbf{r}_n(m))$, $m \in [M]$; namely, it is the binary vector that identifies all items that u_n had rated.

Then the predicted rating $P(u_n, b_m)$ is the weighted average over the adjusted ratings that u_n made thus far,

$$P(u_n, b_m) := \overline{R(b_m)} + \frac{\langle \mathbf{s}_m, \bar{\mathbf{r}}_n \rangle}{\langle \mathbf{s}_m, \xi(\mathbf{r}_n) \rangle} \,. \tag{2}$$

The weighted average is taken over at most q items, and it is based only on items that have a positive similarity to b_m. The quotient in Eq. (2) is undefined if the denominator equals zero (i.e., if none of the items that u_n had rated in the past is in $N_q^+(m)$). In that case, $P(u_n, b_m)$ is set to $\overline{R(b_m)}$. (There exist other variants of those similarity scores and prediction formulas. We focus here on the version that is suggested in [9]. The modification of our protocols to other variants is straightforward.)

Sometimes, instead of showing to u_n his predicted rating on some item, the goal is to present to him the h items which are most likely to appeal to him, without predicted ratings. To that end, one produces a *ranking* of all items that u_n had not rated so far in order to extract from it the top h items, for some $h \geq 1$. Following the discussion in [9, Section 2.3], we implement herein the following ranking procedure. Let $J(n)$ be the subset of indices of items that u_n already rated. Define for each $m \in [M] \setminus J(n)$ the score

$$\hat{s}(m) = \sum_{\ell \in J(n) \cap N_q(m)} S(m, \ell) \,. \tag{3}$$

Namely, $\hat{s}(m)$ is the sum of similarities between b_m and all those items that u_n already rated and fall within $N_q(m)$, the q-neighborhood of b_m. Then, the top h items to be recommended to u_n are those with the highest values of $\hat{s}(m)$.

3.2 Distributed scenarios

We consider distributed scenarios in which there are K vendors, $V_1, \ldots, V_K$, where each one of them offers a subset of items to some subset of users. Let $I_k \subseteq [N]$ denote the set of indices of users that V_k serves, $J_k \subseteq [M]$ denote the set of indices of the items that V_k offers, and $N_k := |J_k|$ and $M_k := |I_k|$ be their corresponding sizes, $k \in [K]$. The subsets I_k and J_k, $k \in [K]$, are publicly known.

The study of privacy-preserving collaborative filtering considered one of the following three distributed scenarios:

Horizontal: All vendors offer all items in B but they serve disjoint subsets of users from U. Namely, $J_k = [M]$ for all $k \in [K]$, but the sets $I_1, \ldots, I_K$ are all disjoint and their union is $[N]$. In particular, if R is the $N \times M$ user-item matrix, then V_k owns the subset of R's rows that corresponds to I_k, $k \in [K]$.

Vertical: All vendors serve all users in U but they offer disjoint subsets of items from B. Namely, $I_k = [N]$ for all $k \in [K]$, but the sets $J_1, \ldots, J_K$ are all disjoint and their union is $[M]$. Here, V_k owns the subset of R's columns that corresponds to J_k, $k \in [K]$.

Hybrid: In that model, the user-item matrix is distributed in a general manner between the vendors (not necessarily by rows or columns), so that each entry in the matrix is held by a single vendor.

In this study we consider a **general distribution scenario**. Like the hybrid scenario, each vendor owns some sub-matrix, but the sub-matrices could be overlapping. Namely, it is possible that a user u_n who wishes to purchase an item b_m could do so through more than just one vendor, as is the underlying assumption in all above described scenarios. That is the typical case in real markets – users usually have a choice between different vendors. Hence, the general distribution scenario is more realistic than those considered so far in prior art, and it includes all of them as private cases.

3.3 The mediated setting

We assume $D > 1$ independent mediators, T_d, $d \in [D]$, that assist the vendors in performing the computations. They are assumed to be semi-honest, i.e., they follow the prescribed protocols, but try to glean from the messages received during those protocols information on the user-item rating matrix. We let $\mathbf{T} := \{T_1, \ldots, T_D\}$ denote the set of all mediators. Our working assumption is that of an *honest majority*: if a subset of $\mathbf{T}$ colludes and combines the information that they got, in order to extract information on private user-item rating data, the subset's size is less than $D/2$.

3.4 Secret Sharing

Secret sharing [20] methods are protocols that enable to distribute a secret among a group of participants, such that each of them is allocated a piece of information, called *a share*, so that some subsets of those shares enable the reconstruction of the secret. In its most basic form, called *Threshold Secret Sharing*, the secret can be reconstructed only when a sufficient number of shares are combined together, while smaller sets of shares reveal no information at all on the secret. In our context, the secret holders will be the vendors, and the group of participants among whom the secrets will be shared are the mediators. The domain of secrets is known in advance and it will be a finite field $\mathbb{Z}_p$, where p is some sufficiently large prime (in particular, larger than the set of participants as well as the number of possible secrets).

We will use the Shamir threshold secret sharing scheme [20]. It is a D'-out-of-D scheme in the following sense: if D' is an integer in the range $1 \le D' < D$, then the shares that are distributed in that scheme allow the recovery of the secret x from *any* subset of D' shares, while any subset of $D' - 1$ shares reveals no information on the secret. The scheme has two procedures: Share and Reconstruct:

- $\text{Share}_{D',D}(x)$. The procedure samples a uniformly random polynomial $g(\cdot)$ over $\mathbb{Z}_p$, of degree at most $D' - 1$, where the free coefficient is x. That is, $g(t) = x + \alpha_1 t + \alpha_2 t^2 + \ldots + \alpha_{D'-1} t^{D'-1}$, where α_j, $1 \le j \le D' - 1$, are selected uniformly at random from $\mathbb{Z}_p$. The procedure outputs D values – $g(1), \ldots, g(D)$ – where $x_d = g(d)$ is the share given to T_d, $d \in [D]$. It is easy to see that any selection of $D' - 1$ shares reveals nothing about the secret x, whereas any subset of D' shares fully determines x, by means of polynomial interpolation, as we describe next.

- $\text{Reconstruct}_{D'}(x_1, \ldots, x_D)$. The procedure is given any selection of D' shares out of $\{x_1, \ldots, x_D\}$, and it then interpolates a polynomial $g(\cdot)$ of degree at most $D' - 1$ using the given points. It then outputs $x = g(0)$.

In what follows we apply secret sharing on secret vectors and matrices. By that we mean that when a vendor wishes to share a vector, or a matrix, among $\mathbf{T}$, he will compute and distribute shares in each component of the vector or matrix, independently. Consequently, any sufficiently large subset of the mediators will be able to reconstruct the shared vector or matrix by reconstructing each component independently. We also use D'-out-of-D secret sharing with $D' = \lfloor (D+1)/2 \rfloor$. With such threshold, the number of shares needed to reconstruct the secret is at least $D' \ge D/2$. Hence, under our assumption of honest majority (in the sense that if a collusion between some of the mediators occurs, it involves less than half of them), the shared secrets will remain fully protected.

We conclude this section by a note on the selection of the field's size p. Selecting the prime p to be a Mersenne prime (a prime of the form $p = 2^t - 1$ for some integer $t > 1$) is advantageous since multiplication of two field elements in such cases can be done without performing an expensive division (in case the multiplication result exceeds the modulus). We use herein the Mersenne prime $p := 2^{31} - 1$ since it is sufficiently large for our purposes, in the sense that whenever we perform secret sharing, the value of the shared secret is strictly smaller than p.

4 PRIVACY PRESERVING PROTOCOLS

In this section we present our privacy-preserving protocols. In Sections 4.1 we describe the computations and protocols that are performed in the offline phase and involve all of the vendors $V_1, \ldots, V_K$, as well as the mediators $\mathbf{T}$. Then, we describe the online phase in which a given vendor V_k submits queries to $\mathbf{T}$ towards computing the predicted rating of some user u_n for an item b_m, $P(u_n, b_m)$, Eq. (2) (Section 4.2), or getting the top h items for a given user u_n (Section 4.3). The online phase is carried out solely by the relevant vendor, V_k, and the mediators, $\mathbf{T}$. Namely, the participation of all vendors is required only in the offline and less frequent phase.

4.1 Offline model construction

Recall that R is the global user-item rating matrix; for every $n \in [N]$ and $m \in [M]$, $R(n,m)$ is the rating that u_n gave to b_m, or zero if no such rating was given. Let $\mathrm{sq}(R)$ and $\xi(R)$ be matrices of the same dimensions as R, whose entries are as follows:

$$\mathrm{sq}(R)(n,m) = (R(n,m))^2\,, \text{ and } \xi(R)(n,m) = \xi(R(n,m))\,,\ (n,m) \in [N] \times [M]\,.$$

The columns of R were denoted by $\mathbf{c}_m$, $m \in [M]$. Therefore, the columns of $\mathrm{sq}(R)$ and $\xi(R)$ are $\mathbf{c}_m^2$ and $\xi(\mathbf{c}_m)$, under our notation agreements (see Section 3.1). Hence, the similarity score between a given pair of items, say b_ℓ and b_m, $\ell < m \in [M]$, is given by $S(\ell, m) = z_1/\sqrt{z_2 z_3}$, where

$$z_1 = \langle \mathbf{c}_\ell, \mathbf{c}_m \rangle\,,\ z_2 = \|\mathbf{c}_{\ell|m}\|^2 = \langle \mathbf{c}_\ell^2, \xi(\mathbf{c}_m) \rangle\,,\ z_3 = \|\mathbf{c}_{m|\ell}\|^2 = \langle \xi(\mathbf{c}_\ell), \mathbf{c}_m^2 \rangle \tag{4}$$

(see Definition 1).

The user-item matrix in our case is distributed among K parties, the vendors. The vendor V_k, $k \in [K]$, possesses an $N_k \times M_k$ user-item rating matrix, denoted R_k. That matrix holds an entry for each user that V_k serves and for each item that V_k offers. We assume hereinafter that each nonzero entry in R occurs in just a single R_k, $k \in [K]$; such an assumption is natural since if u_n purchased and rated an item b_m through one vendor V_k, he would typically not rate it again through another vendor.[1] Hence, R_k is the $N_k \times M_k$ sub-matrix of the global $N \times M$ matrix R, which corresponds to R's entries whose indices are in the Cartesian product $I_k \times J_k$. Stated differently,

$$R = \sum_{k \in [K]} [\![R_k]\!] \tag{5}$$

where $[\![R_k]\!]$ is the $(N \times M)$-"inflation" of R_k in the sense that

$$[\![R_k]\!]\,(i,j) = \begin{cases} R_k(i,j) & \text{if } (i,j) \in I_k \times J_k \\ 0 & \text{if } (i,j) \in [N] \times [M] \setminus I_k \times J_k \end{cases}.$$

Protocol 1 is executed by the vendors and the mediators, $\mathbf{T}$, towards the goal of $\mathbf{T}$ learning $S(\ell, m) = \frac{\langle \mathbf{c}_\ell, \mathbf{c}_m \rangle}{\|\mathbf{c}_{\ell|m}\| \cdot \|\mathbf{c}_{m|\ell}\|}$, $\forall (\ell, m) \in [M] \times [M]$, $\ell \neq m$ (see Eq. (1)). The input to the protocol is the user-item sub-matrices R_k, $k \in [K]$, that are held by the vendors. The protocol has three phases:

(1) Phase 1: The vendors distribute shares relating to their sub-matrices of user-item ratings.
(2) Phase 2: The mediators compute shares relating to the global user-item matrix.
(3) Phase 3: The mediators compute the similarity scores between every pair of items.

[1] We note that even in such unusual cases of repeated ratings, the outputs of our protocols are still justifiable, without needing to modify the protocols. However, we omit the interesting analysis and discussion of such cases due to space limitation.

In Phase 1 (Steps 1-5) each vendor, V_k, $k \in [K]$, first computes $\mathrm{sq}(R_k)$ and $\xi(R_k)$, where

$$\mathrm{sq}(R_k)(i,j) = (R_k(i,j))^2\,,\ \xi(R_k)(i,j) = \xi(R_k(i,j))\,,\ (i,j) \in I_k \times J_k\,. \tag{6}$$

The vendor V_k then proceeds to generate D'-out-of-D shares in each entry of each of those three matrices, with $D' = \lfloor (D+1)/2 \rfloor$, and distributes those shares to the mediators.

In Phase 2 (Steps 6-10) each mediator accumulates the shares received from all vendors. To that end, T_d (for every $d \in [D]$) first initializes three share matrices, denoted R^d, $\mathrm{sq}(R)^d$ and $\xi(R)^d$, to be $N \times M$ zero matrices. Then, whenever he gets sub-matrices of shares from a vendor, V_k, he adds those shares to the relevant entries of the matrix (R^d, $\mathrm{sq}(R)^d$ or $\xi(R)^d$). In view of Eq. (5) and the linearity of secret sharing, in the end of this phase the collection of all D shares $\{R^d(n,m)\}_{d\in[D]}$ are D'-out-of-D shares in $R(n,m)$ for every $(n,m) \in [N] \times [M]$; similarly, the entries of the matrices $\mathrm{sq}(R)^d$ and $\xi(R)^d$, $d \in [D]$, are D'-out-of-D shares in the entries of $\mathrm{sq}(R)$ and $\xi(R)$, respectively.

Finally, in Phase 3 (Steps 11-22), the mediators use those shares in order to compute the similarity score between every pair of items, b_ℓ and b_m, $1 \le \ell < m \le M$. As explained earlier, this computation is carried out by computing the three inner products z_1, z_2 and z_3 in Eq. (4). This is done in Steps 12, 13 and 14 in Protocol 1, by invoking Protocol 2, which we explain later on. Then, they compute from those three values the final score $S(\ell,m)$, according to Definition 1 (Steps 15-20). Since the similarity scores will be used later on in computations over $\mathbb{Z}_p$, we translate the real-valued scores, as computed in Step 16, to integral scores by multiplying them by a sufficiently large factor Q and rounding to the nearest integer (Step 17). We used in our experiments $Q = 1000$ (a choice that preserves an accuracy of three digits after the decimal point). Finally, as the similarity score matrix S is symmetric, the mediators store the similarity score $S(\ell,m)$ which they just computed also in $S(m,\ell)$ (Step 21).

Protocol 2 computes inner products between vectors in $\mathbb{Z}_p^N$ which are shared by a D'-out-of-D secret sharing scheme among the D mediators, where $D' = \lfloor (D+1)/2 \rfloor$. It serves as a sub-protocol in Steps 12, 13, 14 in Phase 3 of Protocol 1. Note that the goal of Phases 1 and 2 in Protocol 1 is to ensure the initial requirement in Protocol 2.

For each $i \in [2]$ and each $n \in [N]$, the D values $\{\mathbf{v}_{i,d}(n)\}_{d\in[D]}$ are D'-out-of-D shares in $\mathbf{v}_i(n)$. The meaning of that is that there exists a polynomial $f_{i,n}$ of degree $D'-1$ over $\mathbb{Z}_p$, such that the share $\mathbf{v}_{i,d}(n)$ that T_d holds in $\mathbf{v}_i(n)$ equals $f_{i,n}(d)$, $d \in [D]$. Now, consider the polynomial $F := \sum_{n=1}^N f_{1,n} \cdot f_{2,n}$. It is a polynomial of degree $2(D'-1)$ over $\mathbb{Z}_p$. In particular, one needs $2D'-1$ point values in F in order to reconstruct it. In addition, $F(0) = \sum_{n=1}^N f_{1,n}(0) \cdot f_{2,n}(0) = \sum_{n=1}^N \mathbf{v}_1(n) \cdot \mathbf{v}_2(n) = \langle \mathbf{v}_1, \mathbf{v}_2 \rangle$. The value that each T_d computes in Step 1 equals $s_d = \sum_{n\in[N]} \mathbf{v}_{1,d}(n) \cdot \mathbf{v}_{2,d}(n) = \sum_{n\in[N]} f_{1,n}(d) \cdot f_{2,n}(d) = F(d)$. Therefore, the set of shares $\{s_d\}_{d\in[D]}$ is a set of $(2D'-1)$-out-of-D shares in the desired inner product. Our setting of D' ensures that $2D'-1 \le D$. (Specifically, if D is odd then $2D'-1 = D$, while if D is even then $2D'-1 = D-1$.) Hence, any selection of $2D'-1$ mediators can use their s_d-shares that were computed in Step 1 in order to interpolate F (Step 2) and recover $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle$ (Step 3).

4.1.1 Privacy. The protocols maintain the privacy of the inputs provided by the vendors as those inputs are communicated to the mediators by D'-out-of-D secret sharing. It means that as long as the set of mediators has an honest majority, those inputs are fully protected, in the *information-theoretic* sense, as opposed to the security of encryption-based protocols that rely on computational assumptions. Note that our protocols are not perfectly secure, in the MPC sense, as they reveal to the mediators not only the final desired outputs (the similarity scores $S(\ell,m)$) but also the intermediate values z_1, z_2, z_3. Hiding even those intermediate values (which do not reveal private information about specific ratings of individual users) would significantly increase the runtimes of the protocols. The design of such enhancements and the evaluation of their toll on runtime are left for future research.

Protocol 1 Computing the similarity matrix

Require: Each V_k, $k \in [K]$, holds a matrix $\{R_k(i,j) : (i,j) \in I_k \times J_k\}$.

1: **for all** $k \in [K]$ **do**
2: V_k computes $\mathrm{sq}(R_k)$ and $\xi(R_k)$ (see Eq. (6)).
3: V_k computes D'-out-of-D shares in each entry in the matrix R_k, where $D' = \lfloor (D+1)/2 \rfloor$. Denoting those shares by $\{R_k^d(i,j) : (i,j) \in I_k \times J_k\}_{d \in [D]}$, V_k proceeds to send the dth set of shares to T_d, for all $d \in [D]$.
4: V_k does similarly with the matrices $\mathrm{sq}(R_k)$ and $\xi(R_k)$.
5: **end for**
6: **for all** $d \in [D]$ **do**
7: T_d initializes three $N \times M$ matrices, denoted R^d, $\mathrm{sq}(R)^d$ and $\xi(R)^d$, to be the zero matrices.
8: When receiving from V_k the set of shares $\{R_k^d(i,j) : (i,j) \in I_k \times J_k\}$, T_d updates the entries of the matrix R^d as follows: $R^d(i,j) \leftarrow R^d(i,j) + R_k^d(i,j)$ for all $(i,j) \in I_k \times J_k$.
9: Similarly, T_d updates the entries of the matrices $\mathrm{sq}(R)^d$ and $\xi(R)^d$ by adding into them the shares received from V_k in $\mathrm{sq}(R_k)$ and $\xi(R_k)$.
10: **end for**
11: **for all** $1 \le \ell < m \le M$ **do**
12: The mediators invoke Protocol 2 to compute $z_1 = \langle \mathbf{c}_\ell, \mathbf{c}_m \rangle$ from the shares that they hold in $\mathbf{c}_\ell$ and $\mathbf{c}_m$.
13: The mediators invoke Protocol 2 to compute $z_2 = \langle \mathbf{c}_\ell^2, \xi(\mathbf{c}_m) \rangle$ from the shares that they hold in $\mathbf{c}_\ell^2$ and $\xi(\mathbf{c}_m)$.
14: The mediators invoke Protocol 2 to compute $z_3 = \langle \xi(\mathbf{c}_\ell), \mathbf{c}_m^2 \rangle$ from the shares that they hold in $\xi(\mathbf{c}_\ell)$ and $\mathbf{c}_m^2$.
15: **if** $z_2 z_3 \neq 0$ **then**
16: The mediators set $S(\ell,m) \leftarrow z_1 / \sqrt{z_2 z_3}$.
17: The mediators set $S(\ell,m) \leftarrow \lfloor QS(\ell,m) + 0.5 \rfloor$.
18: **else**
19: The mediators set $S(\ell,m) = 0$.
20: **end if**
21: $S(m,\ell) \leftarrow S(\ell,m)$
22: **end for**

Ensure: **T** get $S(\ell,m)$.

Protocol 2 InnerProduct: Computing inner product between shared vectors.

Require: $\mathbf{v}_i$, $i \in [2]$, are two vectors in $\mathbb{Z}_p^N$. Every mediator T_d in $\mathbf{T} = \{T_1, \ldots, T_D\}$ holds D'-out-of-D vector shares in them, denoted $\mathbf{v}_{i,d}$, where $D' = \lfloor (D+1)/2 \rfloor$.

1: Each T_d, $d \in [D]$, computes $s_d \leftarrow \sum_{n \in [N]} \mathbf{v}_{1,d}(n) \cdot \mathbf{v}_{2,d}(n)$.
2: Any selection of $2D'-1$ mediators out of **T** use their s_d-shares in order to interpolate a polynomial $F(\cdot)$ of degree $2D'-2$ that agrees with those $2D'-1$ s_d-shares.
3: The inner product between the two input vectors is $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle \leftarrow F(0)$.

Ensure: **T** gets $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle$.

4.2 Computing predicted ratings

In view of Eq. (2) and the definitions preceding it, the predicted rating is given by

$$P(u_n, b_m) := \overline{R(b_m)} + \frac{u_{n,m} - v_{n,m}}{w_{n,m}}, \tag{7}$$

where

$$\overline{R(b_m)} = \frac{\sum_{n \in [N]} R(n,m)}{\sum_{n \in [N]} \xi(R(n,m))}, \tag{8}$$

$$u_{n,m} \coloneqq \sum_{\ell \in [M]} \mathbf{s}_m(\ell) \cdot R(n,\ell)\,, \tag{9}$$

$$v_{n,m} \coloneqq \sum_{\ell \in [M]} \mathbf{s}_m(\ell) \cdot \overline{R(b_\ell)} \cdot \xi(R(n,\ell))\,, \tag{10}$$

$$w_{n,m} \coloneqq \sum_{\ell \in [M]} \mathbf{s}_m(\ell) \cdot \xi(R(n,\ell))\,, \tag{11}$$

and $\mathbf{s}_m$, $m \in [M]$, are as defined in Section 3.1.

Before moving on to explaining how the mediators can compute each of the values in Eq. (7), we observe that the average ratings are rational numbers. However, the computation of $v_{n,m}$, Eq. (10), must be carried out in the secret sharing field $\mathbb{Z}_p$, since it involves entries of the matrix $\xi(R)$, in which the mediators hold shares in $\mathbb{Z}_p$. Therefore, we replace Eq. (10) with the following approximate equation,

$$v_{n,m} = \frac{1}{Q} \sum_{\ell \in [M]} c_\ell \cdot \xi(R(n,\ell)) \quad \text{where } c_\ell = \lfloor Q \cdot \mathbf{s}_m(\ell) \cdot \overline{R(b_\ell)} + 0.5 \rfloor\,. \tag{12}$$

Here, Q is a re-scaling factor like the one that we used in Step 17 of Protocol 1 in order to translate the read-valued similarity scores into integral ones.

4.2.1 Computing linear combinations of secrets. Before we describe the needed computations in the offline phase (Section 4.2.2) and in the online phase (Section 4.2.3), we observe that the sums in Eqs. (8), (9), (11) and (12) are all linear combinations of secrets. Namely, they all take the form $\sigma = \sum_j s_j r_j$, where s_j are known integer coefficients and r_j are private values that are shared among the mediators by a D'-out-of-D secret sharing scheme. The mediators can compute the sum σ, without recovering the private values r_j, as follows. Letting r_j^d denote T_d's share in r_j, $d \in [D]$, then by the linearity of secret sharing, the values $\{\sum_j s_j r_j^d : d \in [D]\}$ are proper D'-out-of-D shares in $\sigma = \sum_j s_j r_j$. Hence, any subset of D' mediators can use the latter shares in order to recover the desired sum σ.

4.2.2 Offline computations. Here we identify all of the values in Eqs. (7)–(12) that do not depend on n and, consequently, can be computed by the mediators already in the offline phase:

(1) Compute the numerator and denominator in Eq. (8) from the shares in R's and $\xi(R)$'s entries, as explained in Section 4.2.1, and then divide them in order to obtain $\overline{R(b_m)}$ for all $m \in [M]$.
(2) Compute the vectors $\mathbf{s}_m$, $m \in [M]$, from the similarity score matrix S, according to their definition in Section 3.1.
(3) Given the average item ratings and the vectors $\mathbf{s}_m$, the mediators will proceed to compute the integer coefficients c_ℓ, $\ell \in [M]$, as defined in Eq. (12).

4.2.3 Online computations. Assume that a vendor V_k had submitted a query in the form $(n,m) \in I_k \times J_k$. Namely, a query that asks for a predicted rating that a user u_n, whom V_k serves ($n \in I_k$), would give to an item b_m that V_k offers ($m \in J_k$). Upon receiving such a query, the mediators will compute $u_{n,m}$, $w_{n,m}$ and $v_{n,m}$ from the shares that they got in R's and $\xi(R)$'s entries, and the known coefficients in the linear combinations in Eqs. (9)+(11)+(12), respectively, as described in Section 4.2.1. Then, they will compute the predicted rating, $P(u_n, b_m)$, by plugging into Eq. (7) the values $u_{n,m}$, $w_{n,m}$ and $v_{n,m}$ which they had just computed, as well as the item average rating $\overline{R(b_m)}$, which they had already computed in the offline phase. Finally, they will send the computed predicted rating to V_k.

Throughout the above described process, the private user-item ratings remain protected against the mediators, under our assumption of honest majority.

4.3 Computing the most recommended items

When a vendor V_k, $k \in [K]$, wants to recommend to one of his users, u_n, $n \in I_k$, items that will most likely appeal to her, V_k submits a query to the mediators so that they can jointly and privately find the h items from $\{b_m : m \in J_k\}$ that maximize the score $\hat{s}(m)$, Eq. (3), and that u_n still had not rated. Define for every $m \in [M]$ the vector $\mathbf{s}'_m$ as follows: $\mathbf{s}'_m(\ell) = S(m,\ell)$ if $\ell \in N_q(m)$ while $\mathbf{s}'_m(\ell) = 0$ otherwise. Then, by Eq. (3),

$$\hat{s}(m) = \sum_{\ell \in [M]} \mathbf{s}'_m(\ell) \cdot \xi(R(n,\ell)) \,. \tag{13}$$

The mediators will compute the vectors $\mathbf{s}'_m$, $m \in [M]$, once in the offline phase. The vectors $\mathbf{s}'_m$ may have negative entries, since the set of q nearest neighbors of b_m could include items b_ℓ for which $S(m,\ell) < 0$. We note that $S(m,\ell) \in [-Q, Q]$. Indeed, the original cosine-score, Eq. (1), is confined to the interval $[-1, 1]$, as implied by the Cauchy-Schwarz inequality. Hence, after the rescaling by Q (see Step 17 in Protocol 1), the entries of the matrix S are confined to the interval $[-Q, Q]$. Therefore, as the number of nonzero addends in the sum in Eq. (13) is at most q, we infer that $\tilde{s}(m) := (qQ + 1) + \hat{s}(m) \geq 1$. As the mapping from $\hat{s}$ to $\tilde{s}$ is monotone, we can look for the h items that maximize $\tilde{s}$.

We proceed to present Protocol 3 which privately computes the subset of h items b_m, $m \in J_k$, with highest $\tilde{s}(m)$ scores. The protocol starts by V_k submitting a query to the mediators (Step 1). That query includes an index $n \in I_k$ of some user u_n that V_k serves. Then (Step 2), the mediators jointly generate a random and secret permutation π over J_k – the set of indices of items that V_k offers.

Next, the mediators perform the computations in Steps 3-7. Each mediator T_d sets a vector of shares $\mathbf{x}_d$ which is initialized to hold in its mth entry, $m \in J_k$, the value $(qQ + 1) + \sum_{\ell \in [M]} \mathbf{s}'_m(\ell)\xi(R)^d(n,\ell)$ (Step 4). In view of Eq. (13) and our discussion in Section 4.2.1, the set $\{\mathbf{x}_d(m) : d \in [D]\}$ is a set of D'-out-of-D shares in $\tilde{s}(m) = (qQ + 1) + \hat{s}(m)$. Next (Step 5), T_d multiplies each entry in $\mathbf{x}_d$ with the corresponding D'-out-of-D share in $1 - \xi(R)(n,m)$. The resulting set $\{\mathbf{x}_d(m) : d \in [D]\}$ is now a set of $(2D' - 1)$-out-of-D shares in $\tilde{s}(m) \cdot (1 - \xi(R)(n,m))$ (as it is the same computation of producing shares in a product of two shared secrets, as we did in Protocol 2). Then (Step 6), T_d sends to V_k the vector $\mathbf{y}_d$ which is the result of permuting $\mathbf{x}_d$ entries using the secret permutation π.

After collecting the responses from all mediators, V_k selects a subset of $2D' - 1$ vectors from $\{\mathbf{y}_d : d \in [D]\}$ in order to interpolate their entries for each $m \in J_k$ and disclose the underlying shared secret, denoted z_m (Step 8). In view of the above discussion, $z_m = \tilde{s}(\pi^{-1}(m)) \cdot (1 - \xi(R)(n, \pi^{-1}(m)))$. Namely, z_m equals the $\tilde{s}$-score of the item $b_{\pi^{-1}(m)}$, if that item had not been rated so far by u_n, while z_m equals zero otherwise. Since $\tilde{s}(m) \geq 1$ for all m, then the indices $i_1, \ldots, i_h$ of the largest values in $\{z_m : m \in J_k\}$ identify the indices of the h items that were not rated so far by u_n and have largest $\tilde{s}$-scores. V_k sends those indices to the mediators who proceed to apply on them the inverse permutation π^{-1} and send them back to V_k (Steps 9-10). Consequently, V_k now has the indices of the top h items, yet unrated by u_n, which maximize the $\tilde{s}$- and $\hat{s}$-scores.

4.3.1 Privacy. The only information that the mediators obtain in the course of Protocol 3 is the indices of the top h items to recommend to u_n. In case that such information may be deemed sensitive, it is possible to apply an additional layer of security as follows. The vendors agree upfront on a random and secret orderings of both the user set $U = \{u_1, \ldots, u_N\}$, and the global set of items, $B = \{b_1, \ldots, b_M\}$. If those random orderings are kept secret from the mediators, then Protocol 3 reveals no information to the mediators.

As for the vendor V_k, the protocol does leak to him more information than just the desired output, being the sought-after h indices. V_k learns also the set of $\tilde{s}$-scores of all items that were still unrated by u_n. However, owing to

Protocol 3 Computing for u_n the top h yet unrated items offered by V_k.

1: V_k submits to $\mathbf{T}$ a query $n \in I_k$.
2: $\mathbf{T}$ jointly generate a secret and random permutation π over J_k.
3: **for all** $d \in [D]$ **do**
4: T_d defines an M_k-dimensional vector $\mathbf{x}_d$ and sets $\mathbf{x}_d(m) \leftarrow (qQ+1) + \sum_{\ell \in [M]} \mathbf{s}'_m(\ell)\xi(R)^d(n,\ell)$, $\forall m \in J_k$.
5: T_d computes $\mathbf{x}_d(m) \leftarrow \mathbf{x}_d(m) \cdot (1 - \xi(R)^d(n,m))$, $\forall m \in J_k$.
6: T_d sends to V_k the permuted vector $\mathbf{y}_d \leftarrow \pi(\mathbf{x}_d)$.
7: **end for**
8: For every $m \in J_k$, V_k uses $2D'-1$ shares out of the received shares $\{\mathbf{y}_d(m) : d \in [D]\}$ in order to recover the underlying shared secret z_m, $m \in J_k$.
9: V_k identifies the h values of z_m, $m \in J_k$, which are largest, and sends their indices, denoted $\{i_1, \ldots, i_h\}$, to $\mathbf{T}$.
10: The mediators send back to V_k the set $\{\pi^{-1}(i_1), \ldots, \pi^{-1}(i_h)\}$.
Ensure: V_K gets the indices in $J_k \setminus J(n)$ of the top h items to be recommended to u_n.

the random permutation π which is kept secret from him, V_k is unable to associate any of those scores to any specific item (beyond the fact that the top h scores relate to the top h items). Such excess information is benign since it does not disclose private user-item ratings owned by other vendors. (We note that the mediators could perform the entire computation of the top h items on their own, using secure MPC protocols for multiplying [7] and comparing [16] shared values. However, the very modest gain in privacy would be accompanied by a significant toll on runtime.)

5 EXPERIMENTS

5.1 overview

Our protocols issue the very same similarity scores, predicted ratings and predicted rankings as described in Section 3.1. Hence, we focus herein on runtime experiments in order to show the dramatic advantage that our protocols offer, in comparison to [22], in terms of runtimes. We stress that apart from runtime, our protocols offer other significant advantages of qualitative nature, as discussed in the Introduction.

▹ Experimental setting. All experiments were run on a virtual machine in the Google Could Platform with the c2-standard-60 machine (60 vCPUs, 240 GB memory). The algorithms were implemented in C++.

▹ Datasets. We used four publicly available datasets: MovieLens 100K, MovieLens 1M, MovieLens 10M, and MovieLens 20M. Those datasets are available online as CSV files at https://grouplens.org/datasets/movielens/X, where the suffix 'X' is '100K'/'1M'/'10M'/'20M'. Table 1 reports the main characteristics of all datasets: number of users N, number of items M, number of ratings $numR$, density (%) $D := \frac{numR}{NM} \times 100$, and the rating scale.

dataset	N	M	$numR$	density	scale
MovieLens 100K	943	1682	10^5	6.30%	[1,5]
MovieLens 1M	6040	3706	10^6	4.47%	[1,5]
MovieLens 10M	71567	10677	10^7	1.30%	[1,5]
MovieLens 20M	138493	26744	$2 \cdot 10^7$	0.54%	[1,5]

Table 1. Dataset characteristics

5.2 The offline phase

5.2.1 Runtimes for the vendors. The runtimes for the vendors are shown in Table 2. We assumed that there exists only one vendor who holds the entire user-item matrix. In a distributed setting, where the user-item matrix is distributed

between several vendors (see Section 3.2), one may derive the runtimes for a vendor V_k by multiplying the runtimes shown in Table 2 by the proportion of the matrix that is held by V_k, namely, by the fraction $\frac{N_k M_k}{NM}$. So for example, if the dataset MovieLens 20M is split evenly between $K = 5$ vendors, then each of the vendors will spend around 70 seconds in the case $D = 3$ and up to 4.13 minutes in the case $D = 9$. Such overhead is negligible as it occurs very infrequently (say, once every few days) and can be carried out in idle time.

We proceed to compare those results to the corresponding runtimes of the Single-Mediator Protocols (SMP) of [22]. We focus here on their protocol for the vertical scenario. In that scenario, each vendor V_k who possesses M_k columns out of the M columns in the matrix, has to perform $2NM_k$ encryptions. The right side of Table 2 shows the corresponding runtimes, when the homomorphic cipher is Paillier [17] (as used in [22]) with a modulus of 512 or 1024 bits. As with the runtimes for our distributed mediation protocols, we assumed that there is a single vendor, namely, the shown runtimes are the times to compute $2NM$ encryptions. In a genuine vertical split, those runtimes should be multiplied by the fraction of columns owned by that vendor (namely, by M_k/M). So for example, when there are K vendors and each of them possesses $M_k \sim M/K$ columns, the shown runtimes should be divided by K. As can be seen, the Single-Mediator protocol is not scalable. For example, in a balanced vertical split with $K = 5$, the runtime for each vendor with MovieLens 10M is roughly 24 hours, when using 512-bit encryptions, and more than a week when using 1024-cryptography. The runtimes for MovieLens 20M are roughly five times worse. However, as we already saw, our protocol's runtime on the MovieLens 20M dataset is only 4.13 minutes per vendor in the $K = 5$ vertical split scenario and $D = 9$ mediators.

dataset	D=3	D=5	D=7	D=9	SMP_{512}	SMP_{1024}
MovieLens 100K	0.095	0.224	0.296	0.415	0.898×10^3	6.62×10^3
MovieLens 1M	1.881	3.526	4.567	7.259	1.267×10^4	9.343×10^4
MovieLens 10M	72.925	110.062	184.564	220.324	4.327×10^5	31.894×10^5
MovieLens 20M	349.326	647.116	904.904	1240.474	2.097×10^6	15.459×10^6

Table 2. Runtimes (seconds) for a vendor in the offline phase. Left: our distributed mediated protocol for different values of D; right: runtimes for the Single-Mediator Protocols (SMP) of [22] with 512- and 1024-bit encryptions, in the vertical distribution scenario. All runtimes in the table need to be multiplied by the fraction of the user-item matrix entries that are owned by the vendor.

5.2.2 Runtimes for the mediators. The runtimes for the mediators are shown in Table 3, on the left side of each column. The percentage shown on the right side of each column relates to the time spent in Step 1 of Protocol 2. For example, in the 20M dataset, when $D = 9$, nearly 54 minutes are spent in Step 1 of Protocol 2, while less than half a minute is spent on all other offline computations.

dataset	D=3		D=5		D=7		D=9	
MovieLens 100K	0.06	86.6%	0.048	79.1%	0.036	72.9%	0.03	67.7%
MovieLens 1M	0.48	67.2%	0.3	54.7%	0.24	46.4%	0.24	39.6%
MovieLens 10M	179	96.9%	178	96.9%	177	96.9%	177	96.9%
MovieLens 20M	3270	99.1%	3257	99.1%	3253	99.1%	3252	99.1%

Table 3. Runtimes (seconds) for a mediator in the offline phase

As can be seen, in the larger datasets, the runtimes for the mediators are much larger than those for the vendors. However, the lion's share of those runtimes is spent on computing the scalar products in Step 1 of Protocol 2. That computation is carried out even in a non-private implementation of that CF method. Hence, the cost of privacy that our protocols entail is quite insignificant.

Table 3 does not show the runtime for the mediator in the single-mediator offline protocol of [22]; that is because in that protocol, all of the computations in the offline phase are done by the vendors. However, when considering the runtimes for both the vendors and mediators, as shown in Tables 2 and 3, it is evident that the improvement offered by distributing the mediation and, consequently, of replacing expensive homomorphic encryptions with lightweight secret sharing computations, is overwhelming. Another advantage of our protocols is that the bulk of the computational overhead is transferred from the vendors to the mediators.

5.3 The online phase

The runtime for predicting ratings, for all testing configurations, was under 1 msec.[2] Computing the most recommended items (Protocol 3) is more time-consuming, as it requires computing M_k inner products of M-dimensional vectors. Table 4 shows the runtimes of that computation for each of the datasets, under the assumption that $M_k = M$, for both our protocols and the single-mediator protocols of [22]. When $M_k < M$, all shown runtimes should be multiplied by M_k/M. We note that also here, the advantage of distributing the mediation is manifested by a dramatic improvement in response times to recommendation queries.

dataset	Our protocol	SMP_{512} (V)		SMP_{512} (H)		SMP_{1024} (V)		SMP_{1024} (H)	
MovieLens 100K	<0.001	12.850	0.938	12.850	0.938	73.604	7.018	73.604	7.018
MovieLens 1M	<0.001	28.313	2.068	28.313	2.068	162.174	15.465	162.174	15.465
MovieLens 10M	0.044	81.572	5.959	81.572	5.959	467.225	44.555	467.225	44.555
MovieLens 20M	0.310	204.324	14.928	204.324	14.928	1170.317	111.602	1170.317	111.602

Table 4. Runtimes (seconds) for computing the most recommended items. Left: our distributed mediated protocol. Right: runtimes for the Single-Mediator Protocols of [22] with 512- and 1024-bit encryptions, in the vertical (V) and horizontal (H) distribution scenarios, for the mediator (left value within each column) and for the vendor (right value). All runtimes in the table need to be multiplied by M_k/M – the fraction of items offered by the corresponding vendor.

6 CONCLUSIONS

We presented herein secure multi-party protocols for performing item-based PPCF over distributed datasets for a general distribution scenario. Our protocols utilize mediation, in similarity to the protocols of [22], who showed the significant advantages of mediation in the context of PPCF. While their protocols used a single mediator, our protocols assume several independent mediators. That assumption enabled us to design protocols that are based on lightweight secret sharing computations rather than costly homomorphic encryption. Distributed mediation maintains all of the advantages of mediation that were identified in [22], and more: the distributed-mediator protocols offer stronger security and overwhelmingly shorter runtimes, in comparison to the single-mediator protocols. In addition, we considered a general distribution scenario that encompasses all previously considered distribution scenarios and extends them to a more realistic scenario, in which users have a choice between several competing vendors.

In summary, distributed mediation offers many attractive advantages in the context of PPCF: scalability; enhanced privacy; freeing the vendors from the need to communicate with each other, or the need to be online constantly; and enabling an economically-realistic collaboration model between the vendors. Hence, we intend to examine the applicability of distributed mediation to other CF algorithms, such as matrix factorization-based algorithms, e.g. [19], and compare their performance to that of existing privacy-preserving implementations of such algorithms, e.g. [14, 15].

[2] In all of our experiments in the online phase, we used $q = 80$ (see Section 3.1), as in [22].

REFERENCES

[1] Waseem Ahmad and Ashfaq A. Khokhar. An architecture for privacy preserving collaborative filtering on web portals. In *The Third International Symposium on Information Assurance and Security, IAS*, pages 273–278, 2007.

[2] Joël Alwen, Abhi Shelat, and Ivan Visconti. Collusion free protocols in the mediated model. In *CRYPTO*, pages 497–514, 2008.

[3] Anirban Basu, Jaideep Vaidya, and Hiroaki Kikuchi. Perturbation based privacy preserving slope one predictors for collaborative filtering. In *IFIPTM*, pages 17–35, 2012.

[4] Anirban Basu, Jaideep Vaidya, Hiroaki Kikuchi, and Theo Dimitrakos. Privacy-preserving collaborative filtering for the cloud. In *CloudCom*, pages 223–230, 2011.

[5] Anirban Basu, Jaideep Vaidya, Hiroaki Kikuchi, and Theo Dimitrakos. Privacy-preserving collaborative filtering on the cloud and practical implementation experiences. In *IEEE CLOUD*, pages 406–413, 2013.

[6] Iván Cantador, Ignacio Fernández-Tobías, Shlomo Berkovsky, and Paolo Cremonesi. Cross-domain recommender systems. In *Recommender Systems Handbook*, pages 919–959. 2015.

[7] Koji Chida, Daniel Genkin, Koki Hamada, Dai Ikarashi, Ryo Kikuchi, Yehuda Lindell, and Ariel Nof. Fast large-scale honest-majority MPC for malicious adversaries. In *CRYPTO*, pages 34–64, 2018.

[8] Richard Chow, Manas A. Pathak, and Cong Wang. A practical system for privacy-preserving collaborative filtering. In *ICDM Workshops*, pages 547–554, 2012.

[9] Michael D. Ekstrand, John Riedl, and Joseph A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):175–243, 2011.

[10] David Goldberg, David A. Nichols, Brian M. Oki, and Douglas B. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992.

[11] Hiroaki Kikuchi, Hiroyasu Kizawa, and Minako Tada. Privacy-preserving collaborative filtering schemes. In *ARES*, pages 911–916, 2009.

[12] Daniel Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *SDM*, pages 471–475, 2005.

[13] Burak Memis and Ibrahim Yakut. Privacy-preserving two-party collaborative filtering on overlapped ratings. *KSII Trans. Internet Inf. Syst.*, 8:2948–2966, 2014.

[14] Kartik Nayak, Xiao Shaun Wang, Stratis Ioannidis, Udi Weinsberg, Nina Taft, and Elaine Shi. Graphsc: Parallel secure computation made easy. In *IEEE Symposium on Security and Privacy*, pages 377–394, 2015.

[15] Valeria Nikolaenko, Stratis Ioannidis, Udi Weinsberg, Marc Joye, Nina Taft, and Dan Boneh. Privacy preserving matrix factorization. In *CCS*, pages 801–812, 2013.

[16] Takashi Nishide and Kazuo Ohta. Multiparty computation for interval, equality, and comparison without bit-decomposition protocol. In *PKC*, pages 343–360, 2007.

[17] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt*, pages 223–238, 1999.

[18] Huseyin Polat and Wenliang Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *ICDM*, pages 625–628, 2003.

[19] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[20] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[21] Erez Shmueli and Tamir Tassa. Secure multi-party protocols for item-based collaborative filtering. In *RecSys*, pages 89–97, 2017.

[22] Erez Shmueli and Tamir Tassa. Mediated secure multi-party protocols for collaborative filtering. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(2):1–25, 2020.

[23] Daiji Tanaka, Toshiya Oda, Katsuhiro Honda, and Akira Notsu. Privacy preserving fuzzy co-clustering with distributed cooccurrence matrices. In *SCIS&ISIS*, pages 700–705, 2014.

[24] Udi Weinsberg, Smriti Bhagat, Stratis Ioannidis, and Nina Taft. Blurme: inferring and obfuscating user gender based on ratings. In *RecSys*, pages 195–202, 2012.

[25] Ibrahim Yakut and Huseyin Polat. Arbitrarily distributed data based recommendations with privacy. *Data & Knowledge Engineering*, 72:239–256, 2012.

[26] Andrew C. Yao. Protocols for Secure Computation. In *FOCS*, pages 160–164, 1982.