Pattern Recognition 129 (2022) 108746

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Federating recommendations using differentially private prototypes

Mónica Ribero[a,*], Jette Henderson[b], Sinead Williamson[b,c,d], Haris Vikalo[a]

[a] Department of Electrical and Computer Engineering, University of Texas at Austin, TX, USA
[b] CognitiveScale, Austin, TX, USA
[c] Department of Statistics, University of Texas at Austin, TX, USA
[d] Department of Information, Risk and Operations Management, University of Texas at Austin, TX, USA

ARTICLE INFO

Article history:
Received 25 May 2021
Revised 5 April 2022
Accepted 24 April 2022
Available online 26 April 2022

Keywords:
Recommender systems
Differential Privacy
Federated Learning
Cross-Silo Federated Learning
Matrix Factorization

ABSTRACT

Machine learning methods exploit similarities in users' activity patterns to provide recommendations in applications across a wide range of fields including entertainment, dating, and commerce. However, in domains that demand protection of personally sensitive data, such as medicine or banking, how can we learn recommendation models without accessing the sensitive data and without inadvertently leaking private information? Many situations in the medical field prohibit centralizing the data from different hospitals and thus require learning from information kept in separate databases. We propose a new federated approach to learning global and local private models for recommendation without collecting raw data, user statistics, or information about personal preferences. Our method produces a set of locally learned prototypes that allow us to infer global behavioral patterns while providing differential privacy guarantees for users in any database of the system. By requiring only two rounds of communication, we both reduce the communication costs and avoid excessive privacy loss associated with typical federated learning iterative procedures. We test our framework on synthetic data, real federated medical data, and a federated version of Movielens ratings. We show that local adaptation of the global model allows the proposed method to outperform centralized matrix-factorization-based recommender system models, both in terms of the accuracy of matrix reconstruction and in terms of the relevance of recommendations, while maintaining provable privacy guarantees. We also show that our method is more robust and has smaller variance than individual models learned by independent entities.

1. Introduction

Machine learning models exploit similarities in users' interaction patterns to provide recommendations in applications across fields including entertainment (e.g., books, movies, and articles), dating, and commerce. Such recommendation models are typically trained using millions of data points on a single, central system, and are designed under the assumption that the central system has complete access to all the data. Further, they assume that accessing the model poses no privacy risk to individuals. In many settings, however, these assumptions do not hold. In particular, in domains such as healthcare, privacy requirements and regulations may preclude direct access to data. Moreover, models trained on such data can inadvertently leak sensitive information about patients and clients. In addition to privacy concerns, when data is gathered in a distributed manner, centralized algorithms may lead to excessive memory usage and generally require significant communication resources.

As a concrete example, consider applications of recommender systems in the healthcare domain. There, recommender systems have been used in a variety of tasks including decision support [14], clinical risk stratification [21] and automatic detection of omissions in medication lists [20]. Such systems are typically built using electronic health records (EHRs), which are subject to privacy constraints that limit the ability to share the data between hospitals. This restricts practical applications of recommender systems in healthcare settings as single hospitals typically do not have sufficient amounts of data to train insightful models. Even when training based on a single hospital's data is possible, the resulting models will not capture distributional differences between hospitals, thus limiting their applicability to other hospitals.

In this paper, we present a novel federated and differentially private recommendation framework for the cross-silo setting where each user's data is associated with one of many silos or entities that share the same features, e.g., hospitals, schools, or banks. The method assumes individuals are grouped into entities, at least

* Corresponding author.
E-mail address: mribero@utexas.edu (M. Ribero).

https://doi.org/10.1016/j.patcog.2022.108746
0031-3203/

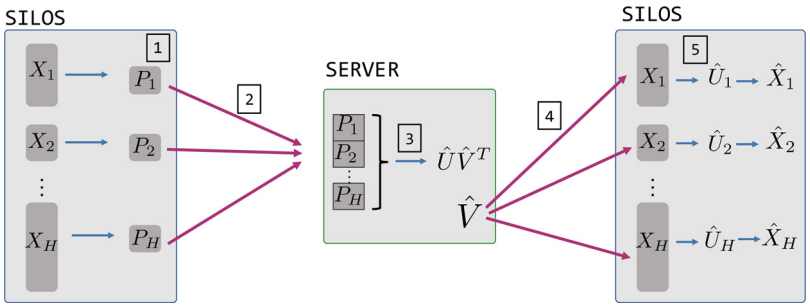


Fig. 1. Federated differentially private framework with reduced communication. Silos use their local data, $X_i \in \mathbb{R}^{m_i \times n}$, to produce k differentially private prototypes, $P_i \in \mathbb{R}^{k \times n}$ (1), that they then send to the server (2). The server aggegates the prototypes ($P \in \mathbb{R}^{H \cdot k \times n}$) and factorizes the resulting matrix to estimate $\hat{V} \in \mathbb{R}^{n \times \ell}$ (3). Then, the server broadcasts $\hat{V}$ to silos (4). Locally, each silo uses $\hat{V}$ to estimate $\hat{X}_i$ and make recommendations (5).

some of which are large enough to learn informative user prototypes; we do not require privacy within an entity. Each silo tries to predict a user preference on a set of items based on each user historical preferences. Recently, federated learning [43] was proposed as algorithmic framework for the settings where the data is distributed across many clients or silos, and due to practical constraints cannot be centralized. In federated learning, a shared server sends a global model to each client, who then update the model using their local data. The clients send information about the local models (for example, gradients) to the server. The server updates the shared model based on the received client information and broadcasts the updated model to the clients. This procedure is repeated until convergence. Federated learning has proved efficient in training deep neural networks for image classification [22,43] and text generation tasks [19,69].

While federated methods address practical computing and communication concerns, privacy of the users in a federated system is potentially vulnerable. Although such systems do not share data directly, the model updates sent to the server may contain sufficient information to uncover model features and raw data information [9,24,35,48], possibly leaking information about the users. These concerns motivate us to adopt differential privacy [16] as a framework for limiting exposure of users' data in federated systems. A differentially private mechanism is a randomized algorithm which allows us to bound the dependence of the output on a single data point. This, in turn, translates to bounds on the amount of additional information a malicious actor could infer about a single individual if that individual were included in the training set.

The differential mechanism presents itself as a natural solution to privacy concerns of users in federated systems, but combining the two paradigms faces some major challenges. The key ones emerge due to the differences in how the two frameworks function. On the one hand, federated learning algorithms are typically iterative and involve multiple querying of the individual entities to collect up-to-date information. On the other hand, in a differential privacy setting where the information obtained in each query must be privatized via injecting noise, the total amount of noise required to be added to a query scales linearly with the number of iterations (thus reducing utility of the system and the information content) [29,42].

We propose a method that allows for learning in federated settings. We avoid data collection and guarantee user-level differential privacy, without the communication and noise overhead introduced in typical iterative federated learning algorithms. We illustrate our system model in Fig. 1. Each silo is tasked with maintaining the privacy of the data entrusted to it against possible attacks by malicious entities. An untrusted server is available to learn centralized models and communicate (in both directions) with the individual silos. Our method learns per-silo recommender models by sharing information between silos in a federated manner, without compromising users' privacy or requiring excessive communication. Specifically, our method learns differentially private prototypes for each silo, and then uses those prototypes to learn global model parameters on a central server. These parameters are returned to silos which use them to learn local recommender models without any further communication (and, therefore, without any additional privacy risk).

To our knowledge, the proposed framework is the first scheme that introduces differential privacy mechanisms to federated recommendations. In Table 1 we compare our algorithm with previous and contemporary work. Unlike typical federated learning algorithms, our method requires only two global communication steps. Such a succinct communication reduces the amount of noise required to ensure differential privacy while also reducing communication overhead and minimizing the risk of communication interception. Yet despite providing differential privacy guarantees to participating silos, the framework allows each silo to benefit from data held by other silos through building its own private, uniquely adapted model. Specific contributions of the paper can be summarized as follows:

Private prototypes through efficient data summarization techniques. Inspired by differentially private $k-$means [4] we propose Algorithm 3, an efficient recovery algorithm to produce private data representations for each silo.

A federated, differentially private framework for recommendation We propose Algorithm 1, a federated recommendation framework, avoiding centralized data collection. Our algorithm allows for learning latent representation of products and services while bounding the privacy risk to the participating users in indepen-

Algorithm 1: Federated Recommender System (Federated RecSys).

Input: Per-entity ratings matrices $\{X_h \in \mathcal{B}(0, \Lambda)\}_{h=1}^{H}$; hyperparameters $k, \epsilon, \delta, \lambda$.
Result: Shared $n \times \ell$ item matrix $\hat{V}$, private $n_h \times \ell$ user matrices $\hat{U}_h$, private reconstructions $\hat{X}_h$.

for $h \in \{1, \dots, H\}$ **do**
  $P_h \leftarrow$ `private_prototypes`$(X_i, \epsilon, \delta, k)$
  Send P_h to server
end
Compile prototypes, $P = [P_1^T, P_2^T, \cdots, P_H^T]^T$
Estimate $\hat{V}$ from P following Eq 2
Broadcast $\hat{V}$ to all entities.
for $h \in \{1, \dots, H\}$ **do**
  Estimate $\hat{U}_h$ given $\hat{V}$ following Eq 2.
  Predict $\hat{X}_h = \hat{U}_h \hat{V}^T$
end

Table 1
Comparison of our prototype-based federated recommender system with related work. "Fed." indicates whether the model works without collecting federated datasets, "DP" indicates what level of privacy guarantee is achieved (see Section 2.4), "Comm." indicates the number of communication rounds (uplink to server and downlink to client), "Cross-silo" indicates if it is suitable for the cross-silo setting. "∼" denotes the system can be adapted for this setting but hasn't been tested on it. In the "Assumptions" column we list additional assumptions needed for the method to work. r denotes a bound on silos/client's number of local records. H denotes the number of silos, $H \ll m$. n refers to number of items, m is the total number of users. $k \ll m$ is the number of prototypes for each silo in our framework. p (typically $p \gg n$) is the model dimension (number of model parameters) for a specific approach. T denotes number communication rounds. ℓ denotes the number of latent factors in MF. All methods assume silos share the same features.

Method	Fed.	DP	Cross-silo	Comm.	Assumptions
Federated Tensor Factorization [12,33]	Yes	∼	Yes	$2Tpm$	None
Centralized rec. sys[44,68].	No	User level	No	nm	None
FedeRank[2].	Yes	∼	∼	(mk+nk)T	None
Rec sys with public data [66]	Yes	Alternative guarantee	∼	nm	Public data
Content based recommender [56]	Yes	User level	∼	$2Tpm$	Access to items' side information
MetaMF [39,49]	Yes	∼	∼	$2Tpm$	None
DP-Prototypes Fed-Rec-Sys	**Yes**	**User level**	**Yes**	$knH + \ell n$	At least k-users per silo

dent silos. This is accomplished by estimating the column space of an interaction matrix from the differentially private prototypes via matrix factorization.

Reducing communication and amplifying privacy by avoiding iterations. We enable federating recommendations under communication constraints by building in the requirement that the number of communication rounds between participating silos and the shared server is only two, thus reducing the communication compared to contemporary work. Following the notation in Table 1, the communication rate for related work is either linear in time T or in the total number of users in the system m. Our system design inherently circumvents the dependence in T. We avoid the dependency in m by transmitting prototypes ($k \cdot H \ll m$) instead of privatized datasets.

Extensive experimental verification of the above methodology in realistic federated learning settings. We demonstrate generalizable representations and strong predictive performance in benchmarking tests on synthetic and real-world data comparing the proposed framework with individual models and conventional federated schemes that lack privacy guarantees.

2. Background and related work

2.1. Recommender systems

The goal of recommender systems is to suggest new content to users. Recommender systems can be broadly classified in two categories: content filtering and collaborative filtering. Under the content-based paradigm, user and/or item profiles are constructed from demographic data or item information, respectively. For example, a user profile could include age while movies could be associated with genre, principal actors, etc. With this information, similarity between users or items can be computed and utilized for recommendation via, for example, clustering or nearest neighbors techniques [37]. In this context, [56] proposes to train an encoder through differentially private and federated version of SGD for recommendation of news articles based on features extracted from their titles. Collaborative filtering [18] relies on past user behaviour (ratings, views, purchases) to make recommendations, avoiding the need for additional data collection [23,36]. Our focus in this paper is on collaborative filtering, an active area of research [3,11,38,70], although the proposed methodology could readily be extended to incorporate additional content-based information [3,57]. Below we introduce notation and summarize relevant techniques.

Consider a set $\mathcal{U}$ of m users and a set $\mathcal{I}$ with n items, where each user has interacted with a subset of the items. We assume that the interactions for user i can be summarized via a partially observed feedback vector $\mathbf{x}_i \in \mathbb{R}^n$, and that all user-item interactions can be represented by a partially observed matrix $X \in \mathbb{R}^{m \times n}$. Entries x_{ij} can be in the form of explicit feedback, e.g. numerical ratings from 1 to 5, or implicit, such as binary values indicating that a user viewed or clicked on some content [26,28,72]. The goal is to predict items that a user would like but has not previously interacted with (i.e., to predict which of the missing values in $\mathbf{x}_i$ have high values).

2.2. Matrix factorization

Matrix factorization is a popular and effective collaborative filtering approach used in many different fields to find low dimensional representation of users and items [36,37,41,61].

A matrix factorization approach assumes that users and items can be characterized in a low dimensional space $\mathbb{R}^\ell$ for some $\ell \ll \min(m, n)$. Specifically, the partially observed matrix X can be approximated by $X \approx UV^T$, where $U \in \mathbb{R}^{m \times \ell}$ aggregates users' representations, and $V \in \mathbb{R}^{n \times \ell}$ collects items' representations. In this paper, we constrain the estimates of U and V to be non-negative (i.e., non-negative factorization). Such a constraint often results in more interpretable latent factors and improved performance [41,71]. In this setting, U and V can be estimated as

$$\hat{U}, \hat{V} = argmin_{U,V \geq 0} \|X - UV^T\|_2^2 + f(U, V) \tag{1}$$

where $f(U, V)$ is a regularizer. For the remainder of this paper, we assume $f(U, V) = \lambda(\|U\|^2 + \|V\|^2)$.

Since we only have access to a subset of the entries of X, (1) is solved by minimizing the error over the training set of ratings T,

$$argmin_{U,V \geq 0} \frac{1}{N} \sum_{x_{ij} \in T} \|x_{ij} - \mathbf{u}_i \mathbf{v}_j^T\|_2^2 + \lambda(\|U\|^2 + \|V\|^2), \tag{2}$$

where $\mathbf{u}_i$ denotes the ith row of U (i.e., the latent representation for the ith user) and $\mathbf{v}_j$ is the jth row of V (i.e., the latent representation for the jth item).

2.3. Federated learning

Federated learning [43] was introduced as a framework for learning models in a decentralized manner, and originally applied to learning with neural networks. The goal of federated learning is to infer a global model without collecting raw data from participating users. This is achieved by having the users (or entities representing multiple users) locally compute model updates based on their data and share these updates with a central server. The server then updates the global model and sends it back to the users.

While they avoid directly sharing users' data, most federated learning algorithms offer no formal guarantees that a malicious agent could not infer private information from the updates. For example, in a naïve application of the original federated learning

method [43] to a matrix-factorization-based recommender system, each entity shares parameters including a low-dimensional representation of each user, leading to a high risk of potential privacy breaches.

Some adaptations of federated learning have been proposed specifically for recommendation systems [2,10,27,39,49,50], where testing is performed locally, allowing for adaptation. However, the iterative nature of these procedures requires constant communication of parameters without any differential privacy guarantees, as defined below.

Alternative federation methods have been proposed for matrix factorization, where the information being shared is less easily mapped back to individual users. Authors in [33] consider federated tensor factorization for computational phenotyping. There, the objective function is broken into subproblems using the Alternating Direction Method of Multipliers [ADMM, 8] where the alternated optimization is utilized to distribute the optimization between different entities. User factors are learned locally, and then server updates the global factor matrix and sends it back to each entity. In a similar way, [12] perform federated matrix factorization by taking advantage of alternating least squares. They decouple the optimization process, globally updating items' factors and locally updating users' factors. These two approaches converge to the same solution as non-federated methods. However, since current variables need to be shared at each optimization stage, this technique requires large communication rates and users' synchronization. While either of the above factorization methods could be adapted to recommender systems, they also lack strict privacy guarantees and require extensive communication.

2.4. Privacy

A number of private recommender systems have been developed using a cryptographic approach [25,34,47,58]. Such methods use encryption to protect users by encoding personal information via cryptographic functions before it is communicated, e.g., a key protected classification model [58]. In the healthcare context, authors in [25] have applied cryptographic methods to providing physician recommendations. However, these methods require centralizing the dataset to perform calculations on the protected data, which may be infeasible when the total data size is large or communication bandwidth is limited, or where regulations prohibit sharing of individuals' data even under encryption.

Differential privacy [16] is a statistical notion of privacy that bounds the potential privacy loss an individual risks by allowing her data to be used in the algorithm.

Definition 2.1. A randomized algorithm $\mathcal{M}$ satisfies ϵ-differential privacy (ϵ-DP) if for any datasets A and B differing by only one record and any subset S of outcomes $S \in range(\mathcal{M})$,

$$Pr(\mathcal{M}(A) \in S) \leq e^{\epsilon} \cdot Pr(\mathcal{M}(B) \in S).$$

In other words, for any possible outcome, including any given individual record to a data set can change the probability of that outcome by at most a multiplicative constant which depends on ϵ. Differential privacy has been applied to recommender systems by adding noise to the average item ratings and the item-item covariance matrix [44]. However, this approach is designed for systems wherein a centralized server needs to collect all the data to derive users and items' means and covariances. Differential privacy is more difficult to impose in iterative algorithms, such as those commonly used in federated learning scenarios, since the iterative nature of these algorithms requires splitting privacy budget ϵ across iterations, thus bringing forth technical challenges [1,22,42,65].

In a recommender systems context, [44,68] rely on differential privacy results to obtain privacy guarantees, but they require access to the centralized raw data. This makes it unsuited for the data-distributed setting we consider.

An alternative to differential privacy is to assume the existence of public data coming from the same or a similar distribution to the private dataset [66]. considers matrix factorization methods to learn $X \approx UV^T$ (see Section 2.2) in the setting where we can learn the item representation matrix V from publicly available data. The public item matrix is then shared with private entities to locally estimate their latent factors matrix U. The applicability of this approach is hindered by potentially limited access to public data, which is the case in sensitive applications such as healthcare recommendations. Our approach provides an alternative method for learning a shared estimate of V from appropriately obscured private data.

2.5. Differentially private prototypes

Our design of private prototypes is motivated by the efficient differentially private k-means estimator for high-dimensional data introduced in [4]. This algorithm first relies on the Johnson-Lindenstrauss lemma to project the data into a lower dimensional space that still preserves much of the data's underlying structure. Then, the space is recursively subdivided, with each subregion and its corresponding centroid being considered a candidate centroid with probability that depends on the number of points in the region and the desired value of privacy ϵ (for details, see line 4 in Algorithms 2, and 5). The final k-means are selected from the candidate centroids by recursively swapping out candidates using the exponential mechanism [45], selecting a candidate with probability proportional to $\exp\left(\frac{\epsilon q(X,y)}{2\Delta q}\right)$, where $q(D,y)$ denotes the score of output y given dataset D, Δq is the sensitivity of the score function q defined by $\Delta q = \sup_{D\sim D',y} |q(D,y) - q(D',y)|$, and $D \sim D'$ denotes any two datasets differing by only one record. Here, the score for each potential collection is the clustering loss (see line 5 in Algorithms 2 and 6). The selected candidates are mapped back to the original space by taking a noisy mean of data points in the corresponding cluster, providing ϵ-DP.

Algorithm 2: private_prototypes(X, k, ϵ, δ). The overall algorithm is ϵ-DP. candidate and localswap are respectively Algorithms 2 and 3 of Balcan et al. [4].

Input: data $X \in \mathcal{B}(0,\Lambda) \subseteq \mathbb{R}^{n\times p}$, hyperparameters k, ϵ, δ
Result: cluster centers $z_1, z_2, \ldots, z_k \in \mathbb{R}^m$
1 Set latent dimension $p = 8\log n$, number of trials $T = 2\log\frac{1}{\delta}$
2 **for** $t = 1, \ldots, T$ **do**
3 Randomly project data from $\mathbb{R}^m \to \mathbb{R}^p$ via the Johnson-Lindenstrauss lemma: $Y = \frac{1}{\sqrt{p}} XG^T$, where $G \sim \mathcal{N}(0,1)^{p\times m}$
4 $C \leftarrow$ `candidate`$(Y, \frac{\epsilon}{6T}, \delta)$
5 $\{u_1, \ldots, u_k\} \leftarrow$ `localswap`$(Y, C, \frac{\epsilon}{6T}, \delta)$Partition Y into $S_j = \{i : j = argmin_l ||y_i - u_l||\}$, $j = 1, \ldots, k$.
6 Recover $z_j^{(t)} =$ `sparse_recovery`$(\{x_i\}_{i\in S_j}, j = 1, \ldots, k, \epsilon, \delta)$
7 **end**
8 Choose $z_1, \ldots, z_k$ by sampling Z from $Z^{(1)}, Z^{(2)}, \ldots, Z^{(T)}$ with probability proportional to $\exp\left(-\frac{\epsilon\mathcal{L}(Z^{(t)})}{24\Lambda^2}\right)$
9 **return** $z_1, \ldots, z_k$

The Balcan et al. [4] method is one of a number of differentially private algorithms for finding cluster representatives or prototypes. Blum et al. [6] introduced SuLQ k-means, where the server updating clusters' centers receives only noisy averages. Unlike the approach of Balcan et al. [4], this algorithm does not have guaran-

tees on the convergence of the loss function. Nissim et al. [51] and Wang et al. [63] use a similar framework but calibrate the noise by local sensitivity, which is difficult to estimate without assumptions on the dataset [73]. Private coresets have been used to construct differentially private k-means and k-medians estimators [17], but this approach does not scale to large data sets.

3. A Differentially private federated recommender system

We propose `Federated RecSys`, a model for learning a recommender system in a federated setting where different silos possess independent private datasets with different numbers of records. We assume the data is distributed across H silos such that each silo possesses data for at least k users. The partially observed user-item interaction matrix associated with the hth entity is denoted by X_h.

We assume that the training data is sensitive and should not be shared outside the silo to which it belongs. While each silo will need to communicate information to a non-private server, we wish to ensure this communication guarantees differential privacy and does not leak sensitive information.

In order for differential privacy and federated recommender systems to work in concert, our framework must accomplish two objectives: (1) make recommendations privately by injecting noise in a principled way, and (2) reduce the number of communications to minimize the amount of injected noise. The solutions to these requirements are interrelated. We first describe a method that reduces the number of communication steps to two and then proceed to describe how to solve the privacy challenge.

3.1. A one-shot private system

Most federated learning methods require multiple rounds of communication between entities and a central server, which poses a problem for differential privacy requirements. Specifically, we can think of each round of communication from silos to the server as a query sent to the individual silos, which has potential to leak information. If we query an ϵ-DP mechanism K times, then the sequence of queries is only $K\epsilon$-DP [46]. In practice, this means that the more communication we require, the more noise must be added to maintain the same level of differential privacy.

Our differentially private federated recommender system, introduced in Algorithm 1, minimizes the amount of noise a differential privacy technique will introduce by limiting the number of communication calls between the entities. Recall that our objective of matrix factorization-based recommendations involves estimating $\hat{X} = \hat{U}\hat{V}^T$. Given $\hat{V}$, each silo can privately estimate $\hat{X}_h = \hat{U}_h\hat{V}^T$ without releasing any information about X_h. Building upon this idea, we constrain the communication to only two rounds, back and forth. Unlike [66], we do not assume access to a public data set. Instead, we construct a shared item representation $\hat{V}$ based on privatized prototypes P_h collected from each silo (step 5). These prototypes are designed to: (a) contain similar information as X_h, thus allowing construction of an accurate item representation; (b) be of low dimension relative to X_h, hence minimizing communication load; and (c) maintain differential privacy with respect to the individual users. We elaborate on building prototypes in Section 3.2.

Once we have generated prototypes for each silo (locally, steps 2–4), we send them to a centralized server where we learn a shared item representation $\hat{V}$ through traditional matrix factorization (step 7) (see Section 2). This shared matrix is then communicated back to the individual entities that use $\hat{V}$ to learn their own users' profile matrices and make local predictions (step 9–10).

In contrast to iterative methods (see references in Section 2.3), the proposed approach requires only two rounds of communication: one from the entities to the server and one from the server to the entities. In addition to reducing communication costs and removing the need for synchronization, this strategy allows us to conserve the privacy budget. With only one communication step requiring privatization, we are able to minimize the noise that must be added to guarantee a desired level of privacy.

3.2. Learning prototypes

In this section we present Algorithm 2, an efficient and private algorithm to learn prototypes that are representative of each siloed private local data set. We first discuss their necessity (i.e., the lack of public data) and communication advantages. Then, we lay out a theoretical intuition and justification for our methods. Finally, we present the algorithm details.

Reducing communication. One way to ensure user-level privacy and avoid communicating a sanitized data set is by assuming existence of a public data that can be learned from. However, this is not a realistic assumption in our use cases; in settings like healthcare and banking, data sets are never publicly available due to data sensitivity and value. We consider methods that find differentially private *prototypes* with the aim of obtaining fewer samples that still capture much of the variation present in the individual silo data, instead of differentially private dataset synthesis methods (see Bowen and Liu [7] for a survey). These methods tend to be ill-suited for high-dimensional settings and would involve sending a large amount of data to the server.

Since we will use these prototypes to capture low-rank structure, provided each silo sends the number of prototypes larger than the rank, it is possible for such prototypes to contain the information required to recover singular vectors of X_h yet still be smaller than X_h, thus reducing the amount of information that needs to be communicated.

Distributional closeness. When selecting the prototype mechanism, we use the following two observations. The first observation implies $k-$means is analytically related to NMF, our recommender system framework. The second observation is that the solution to $k-$means defines the closest distribution with finite support to the private data underlying distribution, which theoretically justifies our selection of a $k-$means type algorithm over alternatives discussed in Section 2.

Observation 1. Non-negative matrix factorization (NMF) and spectral clustering have been shown to be equivalent [13]. $k-$ means is a special case of spectral clustering where the similarities are measured using the standard inner-product linear kernel matrix.

Observation 2. (Theorem 3 in Pollard [53]) Let $m_1, \ldots, m_k$ be the optimum of the k-means objective on a dataset $X = \{x_i\}_{i=1}^n$ distributed according to some distribution P on Ω, and let $\mathcal{M}_k$ be the set of discrete distributions on Ω with support size at most k. Then, the discrete distribution implied by $m_1, \ldots, m_k$ is the closest discrete distribution to P in $\mathcal{M}_k$ with respect to the 2-Wasserstein metric.

Since we are learning the item matrix $\hat{V}$ via NMF, Observation 1 suggests that one should capture the centroids of clusters in X_h to preserve spectral information. Observation 2 implies that the prototypes obtained via k-means are close, in a distributional sense, to the underlying distribution. Following these facts, we consider prototype generation methods based on private k-means. Since the learned prototypes are created to capture the same latent representation that would be captured by NMF, we expect the estimated item matrix $\hat{V}$ to be close to the true V.

Private prototypes algorithm. Our private prototypes generation algorithm is formalized as Algorithm 2. At its base is the differentially private candidates framework of Balcan et al. [4], used to find cluster assignments for each (potentially high-dimensional) private

record on a silo (lines 3–5). However, we significantly alter this scheme by augmenting it with a novel recovery algorithm that preserves accuracy and privacy. The new algorithm increases overall efficiency by exploiting sparsity of the data and deploying the Gumbel trick, often used to efficiently sample from discrete distributions [5,15,52]. In particular, after obtaining cluster assignments for each data point (line 5 in Algorithm 2), instead of taking noisy means or sequentially applying the exponential mechanism to recover the non-zero entries of the centroid, we introduce a procedure that draws noise from a Gumbel distribution with probability density function given by $p(y;b) = \frac{1}{b}\exp\left(-(y/b + e^{-y/b})\right)$, adds it to the centroid mean, and takes the top-s entries; here s denotes the number of non-zero entries in the dataset. Note that the Gumbel trick protects from revealing which entries have large values but does not protect the value itself; therefore, we also add Laplace noise to the original value entry in case it has been selected to be revealed (line 4 in Algorithm 3). We formalize this procedure as Algorithm 3.

Algorithm 3: sparse_recovery Sparse recovery in high dimension.

Input: Data belonging to same cluster $\{x_i\}_{i=1}^m \subseteq \mathbb{R}^n$ with $\|x_i\|_\infty \leq \Lambda$, $\|x_i\|_0 \leq s$; hyperparameters ϵ, δ

Result: Centroid $v \in \mathbb{R}^n$

1 Compute $\mu = \frac{1}{n}\sum_{i=1}^n x_i$

2 $\tilde{\mu} \leftarrow \mu + Y$ with $Y \sim \text{Gumbel}(\frac{\epsilon}{2s\Lambda}|\mu|)$

3 $I \leftarrow \{i : \tilde{\mu}_i \text{ is in top } s\}$

4 $v \leftarrow \begin{cases} \mu_i + Lap\left(\frac{2\Lambda s}{\epsilon n}\right) & \text{if } i \in I \\ 0 & \text{otherwise} \end{cases}$

5 **return** v

3.3. Private federated recommender system

Algorithm 1 summarizes our entire differentially private federated recommender system. We prove its privacy guarantee in Theorem 3.1.

Theorem 3.1. *Algorithm 1 is ϵ-Differentially Private.*

Proof. The server interacts with the private datasets X_h only once when it is collecting the private prototypes. Let q be a utility or score function and Δq its sensitivity (see Section 2.5). Durfee and Rogers [15] prove that adding noise $Y \sim \text{Gumbel}(\frac{2\Delta q}{\epsilon})$ to the utility function q and selecting the top k values from the noisy utility is equivalent to applying the exponential mechanism k times; therefore, transmission of a single prototype is ϵ-DP. In our algorithm, q is determined by each entry's magnitude. The parallel composition theorem [46] establishes that the overall privacy budget is given by the maximum of the individual budgets, implying that the overall algorithm is ϵ-DP. □

3.4. Time complexity

It follows from Corollary 1 in [4] that the runtime of Algorithm 2 is polynomial in m, n and ℓ. Although non-negative matrix factorization is NP-hard [62], our method can take advantage of local SGD which, under appropriate regularization, converges to a local minimum in $O(\frac{1}{\sqrt{T}})$. Hence, our framework runs in polynomial time, making it suitable for the cross-silo setting where low complexity is desirable despite considerably stronger computational resources than in the cross-device scenario.

Note that while iterative-based methods also involve NMF, these methods evade the computation of prototypes. However, this comes at the cost of splitting the privacy budget across communication rounds and hyperparameter tunning grids, leading to an increased overall computational cost (due to multiple trials) and poor performance as demonstrated in Section 4.

4. Experiments

We start this section by describing the datasets and evaluation metrics used to assess our methods. To demonstrate the ability of `Federated Recsys` to provide high-quality recommendations in realistic settings, we apply the system to real-world datasets in Section 4.5. We then study prototype learning in Section B.3 and compare it to regular $k-$means.

To implement standard k-means we used the Python library `scikit-learn`. For private prototypes, we modified and implemented in Python the publicly available MATLAB code from [4][1]. We provide an open-source implementation of our code and experiments.[2]

4.1. Datasets

We test the proposed scheme on three different datasets. The first one is a synthetic dataset intended to simulate discrete processes such as ratings or counting event occurrences. The relevant matrices are generated as $U \sim \text{N}(0,1) \in \mathbb{R}^{m\times\ell}$, $V \sim \text{N}(0,1) \in \mathbb{R}^{n\times\ell}$, and $X \sim \text{Pois}(\exp(UV^T))$. We set $n = 100,000$, $m = 500$, $\ell = 100$, and distribute the data uniformly across 10 different entities.

The second dataset is from the eICU Collaborative Research Database [54], which contains data collected from critical care units throughout the continental United States in 2014 and 2015. Providing a realistic cross-silo federated recommendations setting, this dataset has been used in numerous benchmarking tests of methods for healthcare data analysis and prediction [55,60,64]. Since different visits can have diverse causes and diagnoses, we count each patient visit as a separate observation. We use the `laboratories` and `medicines` tables from the database to create a 2-way table where each row represents a patient and each column either a lab or medicine. Matrix X is composed using data from over 190k patients, 457 laboratories and medications, and 205 hospitals. Each entry x_{ij} represents how many times a patient took a test or a medication. The goal of this task is to recommend treatments.

Finally, we consider the Movielens 1M dataset containing 1,000,209 anonymous ratings from 6040 MovieLens users on approximately 3900 movies. We use the first digit of each user's ZIP code to set up a natural federation of the data, resulting in 10 different entities. The goal of this task is to recommend movies and is a classical benchmark in the recommendation field [30,39,67].

4.2. Evaluation metrics

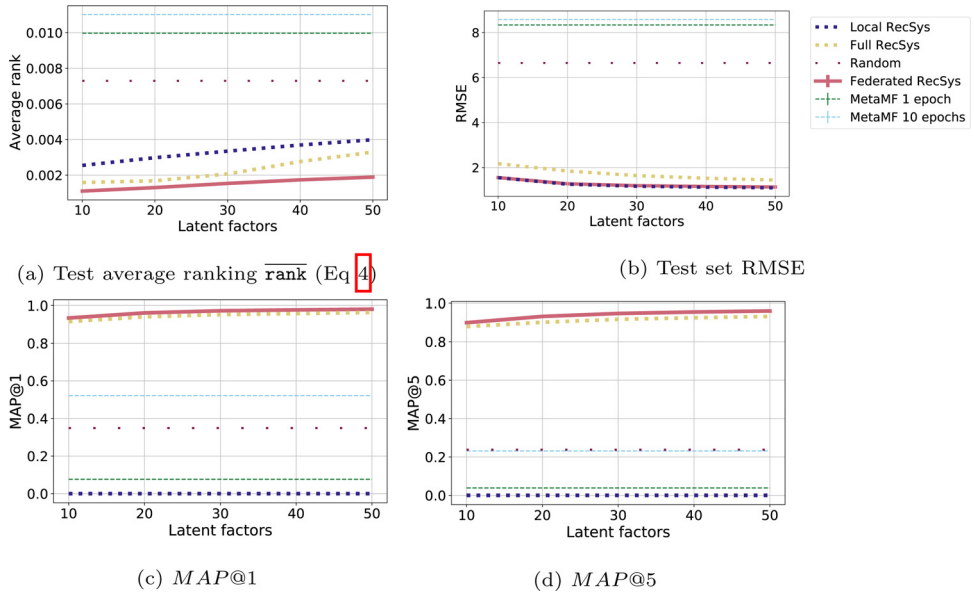
To evaluate the quality of the private prototypes, we use the k-means objective, defined as the average Euclidean distance from original dataset to the closest mean. Formally, the loss $\mathcal{L}(z_1,\dots,z_k;\mathcal{D})$ for dataset $\mathcal{D} = \{x_1,\dots,x_m\} \in \mathbb{R}^n$ and means $z_1,\dots z_k \in \mathbb{R}^n$ is defined as

$$\mathcal{L}(z_1,\dots,z_k;\mathcal{D}) = \sum_{i=1}^m \min_j \|x_i - z_j\|^2. \tag{3}$$

To assess the convergence of matrix factorization and perform parameter tuning, we use the Root Mean Squared Error (RMSE) between the real X and the reconstructed $\hat{X} = \hat{U}\hat{V}^T$. In the case of

[1] https://github.com/mouwenlong/dp-clustering-icml17

[2] https://github.com/mriberodiaz/federatedRecsys



(a) Test average ranking $\overline{\texttt{rank}}$ (Eq 4) (b) Test set RMSE

(c) $MAP@1$ (d) $MAP@5$

Fig. 2. Comparison of several methods for eICU dataset. The results are averaged over 5 trials. In all cases, the standard deviation is bounded by $5 \cdot 10^{-3}$ and not pictured due to scale.

the synthetic data, RMSE is a suitable measure to examine the fit quality since we have access to the ground truth.

However, the RMSE is unable to capture the quality of recommendations. To compare the real and predicted rankings for those datasets, we use Mean Average Ranking [26] ($\overline{\texttt{rank}}$) and Mean Average Precision ($MAP@k$) [31,40,59].

Concretely, let $\texttt{rank}_{ui}$ be the percentile of the predicted ranking of item i for user u, where 0 means very highly ranked and above all other items. We calculate $\overline{\texttt{rank}}$ on a test set $\mathcal{T}$ defined as

$$\overline{\texttt{rank}} = \frac{\sum_{(u,i):x_{ui}\in\mathcal{T}} x_{ui}\texttt{rank}_{ui}}{\sum_{(u,i):x_{ui}\in\mathcal{T}} x_{ui}}. \tag{4}$$

This measure compares the similarity between the real and predicted ranks. Intuitively, for a random ranking the expected $\texttt{rank}_{ui}$ is 0.5, so $\overline{\texttt{rank}} \geq 0.5$ means a ranking no better than random. Conversely, lower values indicate highly ranked recommendations matching the users' patterns.

Precision at k is defined as the proportion of correct recommendations within the top-k prediction scores. Define $p(u,k) = \frac{1}{k}\sum_{i=1}^{k} s_{u,it(i)}$ where $s_{u,it(i)}$ is the true rank of user u for item $it(i)$ ranked by a model at position i. Average Precision at k for user u ($AP@k(u)$) measures the relevance of scores of the top-k recommendations for each user and its order by computing

$$AP@k(u) = \frac{\sum_{i=1}^{k} p(u,i)\cdot \text{rel}(i)}{\text{number relevant items}}, \tag{5}$$

where $rel(i)$ is an indicator function with value 1 when item at rank k is relevant and 0 otherwise. Finally, $MAP@k$ is defined as the mean of individual average precisions over users,

$$MAP@k = \frac{\sum_{u\in U} AP@k(u)}{|\mathcal{U}|}. \tag{6}$$

4.3. Comparisons between methods

To evaluate the proposed framework, `Federated RecSys` is tested on real-world data from the eICU and the Movielens 1M datasets. We construct a test set $\mathcal{T}$ by randomly selecting 20% of the users, and for each selected user, randomly selecting five entries.

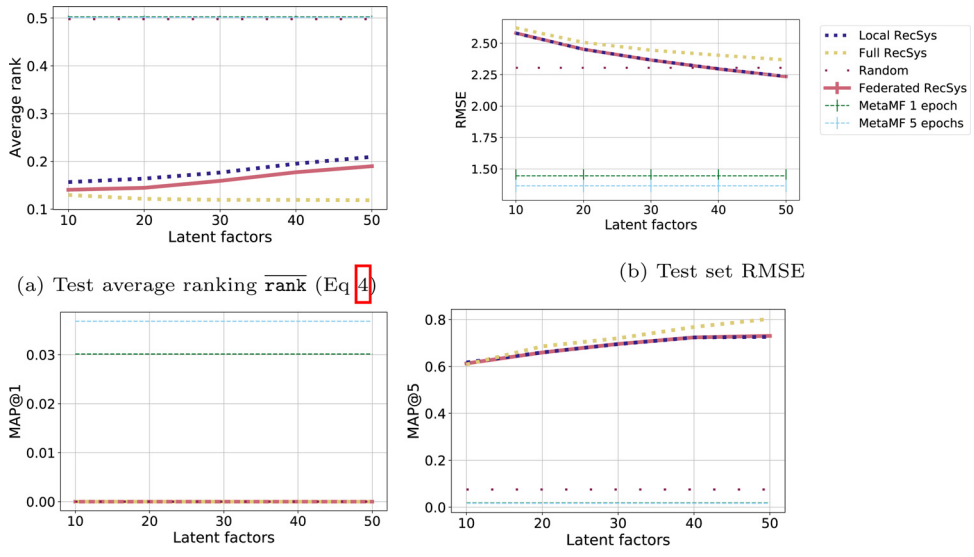
We reiterate that, to our knowledge, there exists no prior recommendation method which both provides privacy guarantees and accommodates federated nature of datasets. In the absence of such methods, we compare our proposed private federated recommender system with: (1) `Full RecSys`: a non-private centralized matrix factorization method which provides a lower bound on the best result one could achieve using matrix factorization while disregarding privacy and federation constraints; (2) `Local RecSys`: individual local matrix factorization for each silo, which is the simplest solution to providing privacy in federated settings; (3) `Random`: random recommendations; and (4) `MetaMF 1 epoch` and `MetaMF 10 epochs`: A differentially private version of Meta-MF [49] through DP-SGD [1]. Details on how we privatize Meta-MF are provided in the appendix. We report results averaged over 5 trials.

4.4. Hyperparameter selection

Hyperparameter optimization breaks differential privacy unless the privacy budget is split among different hyperparamter configurations. Instead of selecting the latent dimensions, we present results for ℓ varying between 10 and 50. We see that performance is relatively consistent across different values. In theory, we could learn hyperparameter values in a differentially private manner, using the exponential mechanism. However, since our method does not seem particularly sensitive to these parameters, we feel this would not be a judicious use of privacy budget. Instead, we suggest practitioners select values for the number of latent factors ahead of time, based on their domain knowledge of the latent dimensionality of the data.

For the private prototypes, we fix $k = 10$ for the hospitals' data and $k = 50$ for Movielens, and in both cases set $\epsilon = 1$. Given that Meta-MF is an iterative procedure, long training requires increasing noise magnitude. Below we report the performance of `Meta-MF1ep`, trained for 1 epoch with small noise magnitude, and `Meta-MF10ep`, trained for 10 epochs with larger noise magnitude.

In the appendix, we explore various settings of the regularization parameter λ. Again, we find that performance is not particularly sensitive to this parameter. In this section, we use $\lambda = 0.1$ throughout.



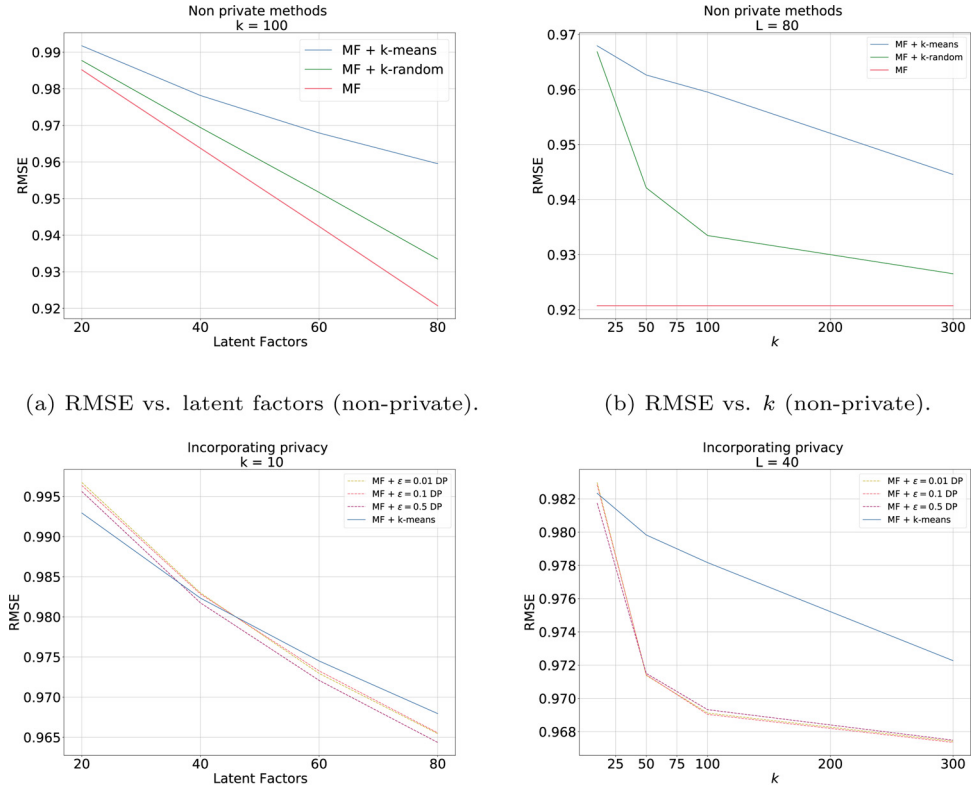
(a) Test average ranking $\overline{\texttt{rank}}$ (Eq 4)

(b) Test set RMSE

(c) $MAP@1$

(d) $MAP@5$

Fig. 3. Comparison of several methods on Movielens1M dataset.The results are averaged over 5 trials; the standard deviation for RecSys is bounded by $8 \cdot 10^{-4}$, for MetaMF by $5 \cdot 10^{-2}$.



(a) RMSE vs. latent factors (non-private).

(b) RMSE vs. k (non-private).

(c) RMSE vs. latent factors (private).

(d) RMSE vs. k for private methods.

Fig. 4. Varying parameters on synthetic data.

4.5. Evaluating the federated recommender system

eICU dataset

Fig. 2 a shows the average ranking quality for all methods, showing the ability of a model to sort items in the correct place. Our federated model (Federated RecSys) obtains the best ranking performance. As intended, the Federated RecSys allows each hospital to improve its predictions over local recommendations by obtaining relevant information from other hospitals without compromising its patients' information. We note that random predictions and Meta-MF are unable to capture the ordering structure learned by our system. The quality of recommendations can also be as-

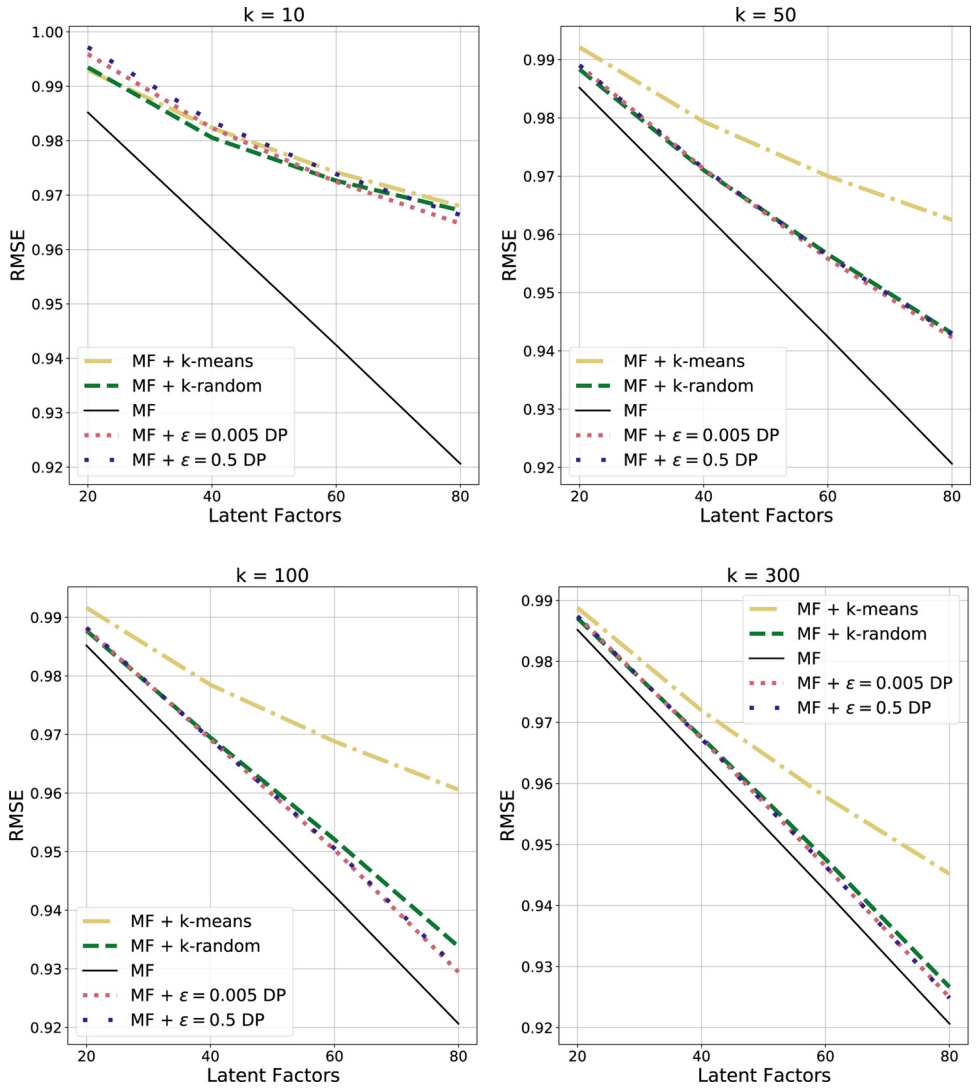


Fig. 5. Comparison of different prototype methods on synthetic data. As k and ℓ increase, k-random exemplars and private k-means maintain competitive performance, providing thus a private but still accurate and robust method.

sessed by their relevance, an effect captured in Fig. 2c and d where we observe that our system typically recommends relevant treatments for patients (i.e., treatments that where on the list of potential relevant treatments). We note that although local models are able to capture the ordering of items (small $\overline{\texttt{rank}}$), this method fails at delivering recommendations on the list of recommended items.

Fig. 2 b shows the average reconstruction error over test data, where again matrix factorization techniques outperform other models. We observe that factorization in larger dimensions improves test RMSE, but does not result in better ratings. On the one hand, RMSE shows the ability of the model to predict the value of a rating but does not provide much insight in the order of that rating, i.e., relative comparison to other items. On the other hand, the average $\overline{\texttt{rank}}$ does not necessarily decrease with the number of latent factors but provides insight in the rank of the elements. For example, always predicting the average rating would result in a good RMSE; however, this strategy does not provide better recommendations than the random model. For example, MetaMF predicts an average rating of 4 on the eICU data with a standard deviation of only 1, achieving good test RMSE, but has a poor $\overline{\texttt{rank}}$ due to its inability to sort items.

Movielens dataset. As expected, the centralized (`Full`) approach, which forms results by processing larger and more heterogeneous datasets, achieves larger test RMSE than the local models (`Local` and `Federated`). This is likely due to the fact that siloed data may have lower intrinsic dimension than the aggregated data. For example, if two silos have datasets lying in (nearly) orthogonal one-dimensional subspaces, local factorization in one dimension could have better performance than aggregating both silos datasets and still factorizing in one dimension; instead, centralized scheme should be factorizing in two dimensions since there are at least two significant latent components. Unlike the eICU dataset where patient data can vary more drastically across hospitals, federating by zip code might produce a more uniform distribution between entities. This uniformity across silos on Movielens also explains why `Local` performs closer to our method than on the eICU dataset. Specifically, in Movielens, each zip code contains enough information to learn a reasonable embedding, which is not the case for hospital data. The benefit from collaboration is more evident when we look at the ranking in Fig. 3a (which is the actual task of interest). Recall that an average ranking above 0.5 means the ranking is no better than random. Conversely, lower values indicate highly ranked recommendations matching the users'

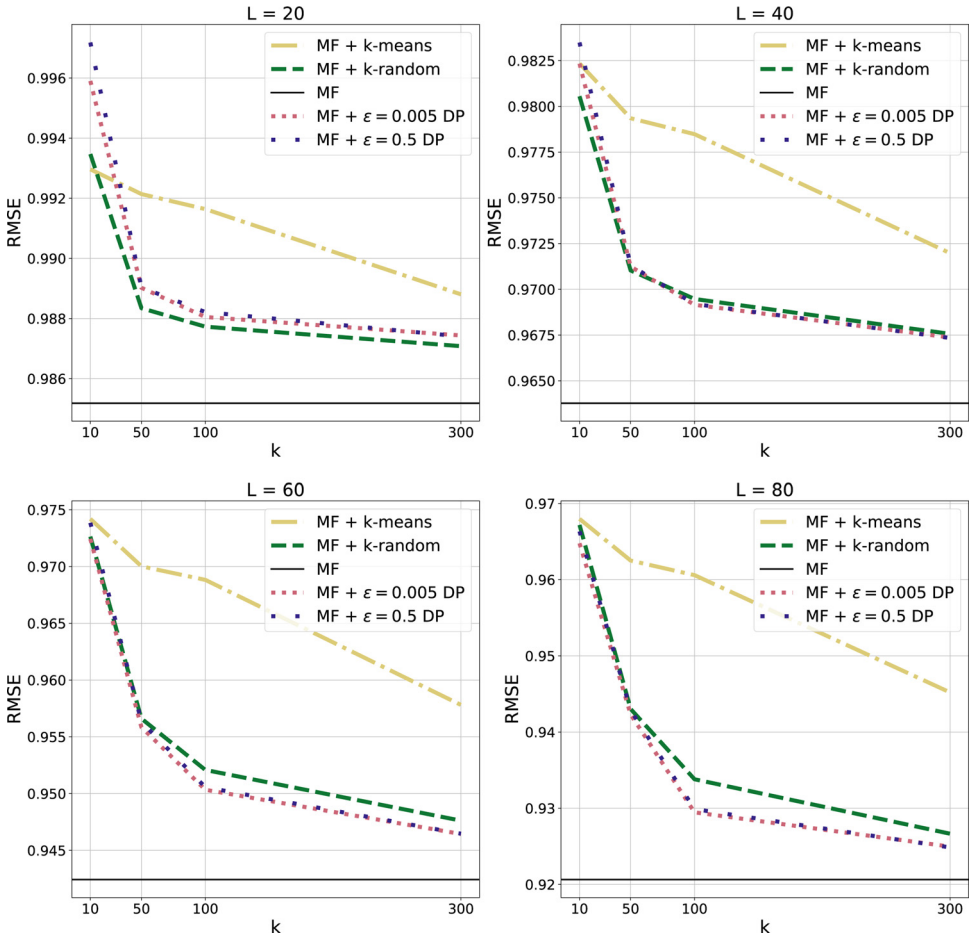


Fig. 6. Comparison of various methods for different values of ℓ on synthetic data. Private methods have superior performance for large ℓ.

patterns. With a small privacy budget, our method is able to share insights among entities without sacrificing their privacy, and delivers better recommendations. We observe that factorization based methods improve over random recommendations. Further, as observed in Fig. 3, although MetaMF is able to achieve lower RMSE in this setting, the private model is unable to learn proper ordering of elements, ultimately delivering poor recommendations, leaving `Federated RecSys` as the best private model.

Statistical significance of the results. The results for private methods that incorporate randomness are averaged over 5 trials. We find that in the case of eICU dataset, standard deviation is bounded by $5 \cdot 10^{-3}$ for all values of ℓ and all methods, and thus not pictured due to scale. For Movielens, `Federated` has standard deviation bounded by $8 \cdot 10^{-4}$ while the MetaMF's is bounded by $5 \cdot 10^{-2}$, confirming superior performance of our proposed method.

4.6. Evaluating the impact of federation and privacy on synthetic data

Recall that our algorithm differs from the standard matrix factorization schemes in two key aspects: first, it learns the item matrix $\hat{V}$ using *prototypes*, rather than the actual data; second, given $\hat{V}$, it learns the users' sub-matrices $\hat{U}_h$ independently, rather than jointly. Moreover, instead of learning the prototypes using exact k-means, we ensure differential privacy using an ϵ-DP algorithm. In this section, we explore the effect of these algorithmic features. In particular, we compare our framework with the following algorithms:

- **Matrix factorization**: Apply Eq. (2) until convergence on $X = [X_1^T, \ldots, X_H^T]^T$.
- **MF + k-means**: Apply Eq. (2) to factorize a matrix of exemplars $P \approx \hat{U}\hat{V}^T$, where P collects the k-means from each matrix $X_1, \ldots, X_H$. Use the estimate $\hat{V}$ to learn individual matrices $\hat{U}_h$ from X_h.
- **MF + k-random**: Identical to **MF + k-means**, but instead of using the cluster means, use k random samples from $X_1, \ldots, X_H$.
- **MF + ϵ-private prototypes**: Identical to **MF + k-means**, but instead of using true cluster means, use the generated ϵ-DP prototypes.

We first evaluate how k-means performs in a non-private setting. Fig. 4a and b show the RMSE when k and ℓ are fixed, respectively. In both figures, we see unsurprisingly that MF has the lowest RMSE, with k-random exemplars from the original dataset performing second best. For larger values of k in Fig. 4b, k-means performance deteriorates compared to k-random. By examining the centroids, we conclude that this is most likely due to k-means overfitting to outliers for large values of k while k-random performance improves as its number of exemplars approaches the full X. We note that our synthetic data does not contain any clusters, so this is the worst-case scenario for the k-means setting. Even though k-means does not perform as well as the other two methods, we observe that the difference in reconstructive performance

between the three methods is fairly small. However, none of the above methods guarantee privacy.

Next, we compare the performance of private k-means and non-private k-means. We do not include k-random exemplars in the comparison of private methods since sharing k random exemplars would disclose the information for these k users. In Fig. 4c, we consider the relatively small value $k = 10$ and investigate the effect of ϵ as the number of latent factors changes. As expected, larger values of ϵ (i.e., less private settings) yield better results. Here we observe little difference in the performance between the private and non-private algorithms. However, in Fig. 4d we see that for large k, the private methods perform significantly better than the non-private k-means, mirroring the results in Fig. 4b. We hypothesise that the noise introduced in the private and random scenarios acts as a regularizer, helping avoid overfitting. We note that, since the sensitivity of the random exemplar mechanism is equal to the range of the data, directly privatizing random exemplars would add excessive noise.

In both Fig. 4c and d, we find that decreasing ϵ (and therefore increasing privacy) does not have a significant negative effect on the reconstruction quality. In Fig. 4d, for larger values k, MF + private k-means performs equally well, even for the smallest value of ϵ, as the noise is averaged over a large number of samples. Here, we can guarantee 0.01-DP instead of 0.5-DP with a minimal drop in RMSE.

4.7. Varying parameters

In this section, we study how the performance of a recommender system is impacted by the variations in parameter values: the number of entities, number of latent factors ℓ, and number of prototypes k. First, recall that the true latent dimension of the synthetic dataset is 100, thus we expect RMSE to decrease for $\ell \in [20 - 80]$, as observed in Fig. 5. In Section 4.6 we showed results for fixed values of the number of prototypes k and the number of latent features ℓ. Below we show additional plots for different values of those parameters.

In Fig. 5 we observe that as the number of samples increases, random k-exemplars outperforms k-means for all values of ℓ. Note that private k-means performs well over a wide range of k. As k increases, private k-means converge to the same value for various values of ϵ (in our experiments we testd for $\epsilon \in \{0.005, 0.05, 0.5\}$; we omit $\epsilon = 0.05$ as it performs really closed to surrounding values). Fig. 6 compares all methods for different values of k. The difference in RMSE is clearer for small values of k. For large values of k, the performance of k-random and k-private approaches that of matrix factorization.

5. Conclusion

We propose a novel, differentially-private framework for recommender systems in federated settings. To enable collaboration and learning common patterns without compromising users' privacy, the proposed framework requires minimal communication between participating entities and thus avoids scaling up the privacy-inducing noise over time. Meanwhile, iterative procedures do not trivially extend to guarantee differential privacy and may require extensive hyperparameter tuning which considerably reduces the privacy budget and consequently harms the performance. We demonstrated high accuracy of the proposed framework on two realistically federated datasets. We observe that the performance of our method increases with k, and that the framework exhibits robustness with respect to the privacy hyperparameter ϵ – thus suggesting suitability for the high privacy regimes. Our work enables low-cost communication methods that take advantage of local data to learn highly accurate yet private statistics, prototypes, and/or synthetic datasets in the cross-silo setting.

A limitation of the present work is in assuming that each silo contains data from multiple users. In particular, the framework is not suitable for settings where each entity represents a single user. An interesting future direction is to explore how the notion of local differential privacy [32] could be applied in such settings, allowing us to work directly with single-user data. A second limitation is that the presented method has only been tested in the scenarios where silos share the same features. We leave to future work adaptations of our method to vertically partitioned data (i.e., scenarios where silos do not share features).

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Private k-means subroutines

Algorithm 4: private_partition (X, ϵ, δ, Q) [4].

Input: data $X \in \mathcal{B}(0, \Lambda) \subseteq \mathbb{R}^{n \times p}$, parameters ϵ, δ, initial cube Q s.t. $\{x_i\}_{i=1}^n \subseteq Q$
Result: Private Grid $C \subseteq \mathbb{R}^p$

Initialize depth a = 0, active set of cubes $\mathcal{A} = \{Q\}$, and set $C = \emptyset$
while $a \leq n$ **and** $\mathcal{A} \neq$ **do**
  $a = a + 1$
  $C = C \cup \left(\cup_{Q_i \in \mathcal{A}} \text{center}(Q_i)\right)$
  for $Q_i \in \mathcal{A}$ **do**
    Remove Q_i from $\mathcal{A}$
    Partition Q_i evenly in each dimension and obtain 2^p cubes $\{Q_i^{(l)}\}_{l=1}^{2^p}$
    for $l \in \{1, 2, \ldots, 2^p\}$ **do**
      Add $Q_i^{(l)}$ to $\mathcal{A}$ with probability $f\left(|Q_i^{(l)} \cap X|\right)$ where
      $f(m) = \begin{cases} \frac{1}{2} \exp -\epsilon'(\gamma - m)) & m \leq \gamma \\ 1 - \frac{1}{2} \exp \epsilon'(\gamma - m)), & \text{otherwise} \end{cases}$
      $\epsilon' = \frac{\epsilon}{2 \log n}$ and $\gamma = \frac{20}{\epsilon'} \log \frac{n}{\delta}$
    end
  end
end
return C

Algorithm 5: candidate (X, ϵ, δ) [4].

Input: data $X \in \mathcal{B}(0, \Lambda) \subseteq \mathbb{R}^{n \times p}$, parameters ϵ, δ
Result: Candidate center set $C \subseteq \mathbb{R}^{\cdot \times p}$

Initialize $C = \emptyset$
for $t = 1, 2, \ldots T = 25k \log \frac{n}{\delta}$ **do**
  Sample shift vector $v \sim \mathcal{U}([-\Lambda, \Lambda]^p)$
  Let $Q_v = [-\Lambda, \Lambda]^p + v$
  $C = C \cup \texttt{private\_partition}\left(X, \frac{\epsilon}{T}, \frac{\delta}{T}, Q_v\right)$
end
return C

Algorithm 6: localswap (X, C, ϵ, δ) [4].

Input: data $X \in \mathcal{B}(0, \Lambda) \subseteq \mathbb{R}^{n \times p}$, parameters ϵ, δ, Candidate set $C \subseteq \mathbb{R}^{\cdot \times p}$

Result: Clustering centers $Z = [z_1, z_2, \ldots, _k] \subseteq C$

Uniformly sample k centers *i.i.d.* from C and form $Z^{(0)}$

$T \leftarrow \frac{n}{\delta}$

for $t = 1, 2, ., T$ **do**

  Choose $x \in Z^{(t-1)}, y \in C \setminus Z^{(t-1)}$ with probability proportional to $\exp -\epsilon \frac{\mathcal{L}(Z') - \mathcal{L}(Z^{(t-1)})}{8\Lambda^2(T+1)}$

  where $Z' = Z^{(t-1)} - \{x\} + \{y\}$

  $Z^{(t)} \leftarrow Z^{(t-1)} - \{x\} + \{y\}$

end

Choose $t \in \{1, 2, \ldots, T\}$ with probability in proportion to $\exp \frac{\epsilon \mathcal{L}(Z^{(t)})}{8(T+1)\Lambda^2}$

return $Z^{(t)}$

Appendix B. Further experiments on parameters variations.

B1. Regularization in recommender systems.

We do not perform hyperparameter tuning over latent factors since it degrades privacy. We rather show performance across different values in Section 4. For the regularization value λ when performing NMF we report in the main body results for $\lambda = 0.1$, since we did not observed significant variability, as observed in Fig. B.7.

B2. Differentially private meta-MF

Meta-MF [39] proposes to learn non-linear embeddings for users and items with multi-layer perceptrons, by minimizing the RMSE. to this end clients share gradients respect of model parameters. Since gradients are calculated on raw data, the model is not differentially private. For a fair comparison in our paper, we combined MetaMF with the Gaussian Mechanism to privatize gradients in the following way: after computing gradients, clients locally clip the gradient vector to have a maximum ℓ_2 norm of L, and then add zero-mean gaussian noise with standard deviation $\sigma = \frac{\sqrt{T}L\log(1\delta)\sqrt{T}}{\epsilon}$ to the gradient. After T rounds of training the procedure is (ϵ, δ)-Differentially Private. Notice our approach provides *pure* differential privacy, a more rigurous guarantee compared to the approximate differential privacy guarantee provided by the gaussian mechanism.

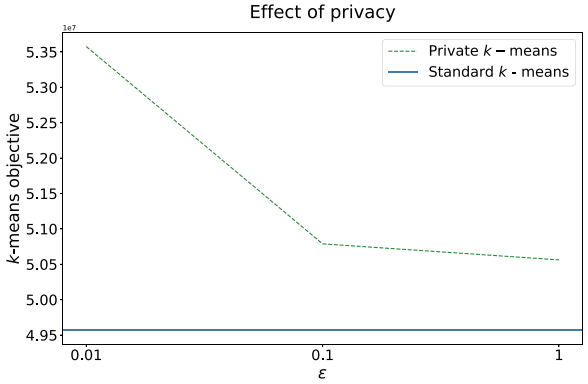


Fig. B2. k-means objective (Eq. (3)) vs. level of privacy. As ϵ decreases, private k-means approaches the objective of non-private k-means.

We do not perform hyperparameter tuning since it degrades privacy, but rather use the same hyperparameters used in the original paper. However, there is a trade-off between training time and noise, so we try two models, one trained for 1 epoch (lower noise magnitude pero round), and another one for 10 epochs but with higher noise magnitude.

B3. Private k-means vs k-means on poisson distributed data

For this experiment we use the previously described synthetic dataset, and study the average behaviour of private k-means. We observe in Fig. B.8 that, as ϵ increases, the level of privacy decreases. The reduction in privacy results in the k-means objective (see Eq. (3)) decreasing until it starts to approach the objective achieved by standard, non-private k-means. Additionally, large k does not necessarily result in better performance. Fig. B.9 shows that for large numbers of centers k, the private k-means algorithm repeats centers instead of overfitting, stalling the objective minimization.

B4. Varying number of entities.

Fig. B.10 shows the RMSE on the synthetic test dataset described in Fig 4.1. We observe that as the number of entities increases, the convergence improves. This is expected since the number of observations used to approximate the items' matrix also grows.

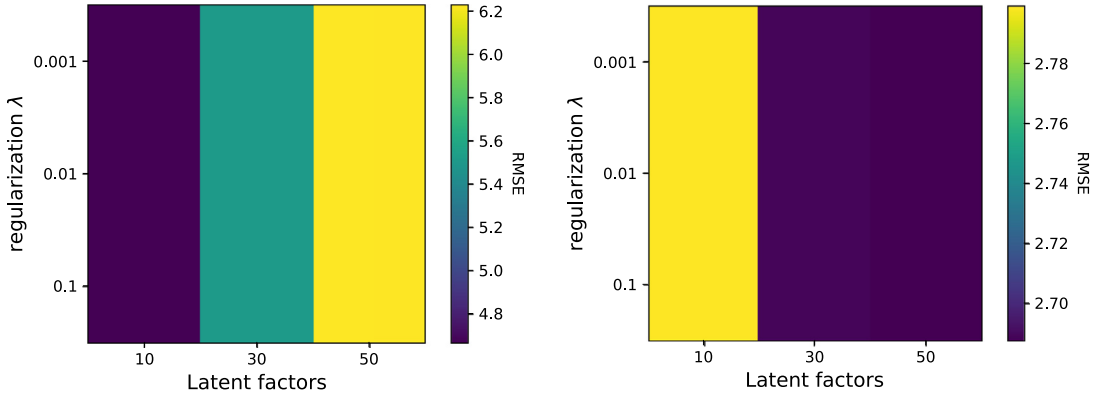


Fig. B1. Regularization value for NMF on the eICU dataset (left) and Movielens (right). For every value of latent factors regularization does not affect overall performance.

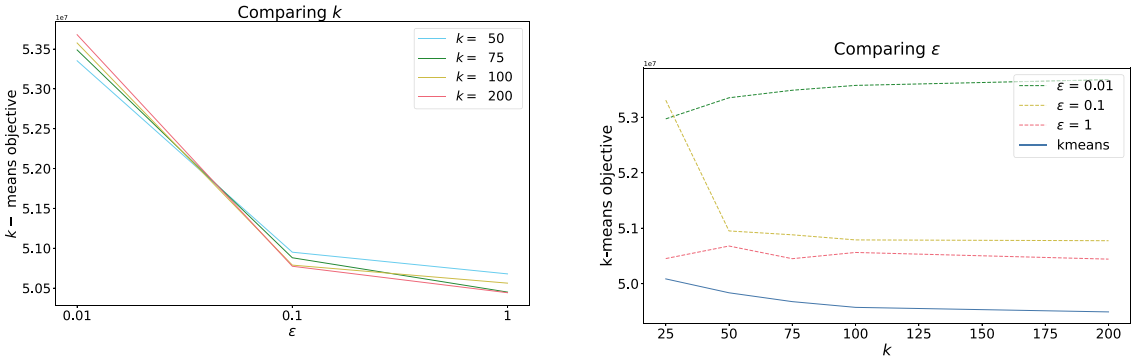


Fig. B3. Private k-means on synthetic data. Larger values of ϵ, i.e. less privacy, decrease the loss value. A large k does not necessarily result in better performance. Unlike non-private k-means, for larger values of k, the private k-means algorithm repeats centers instead of overfitting.

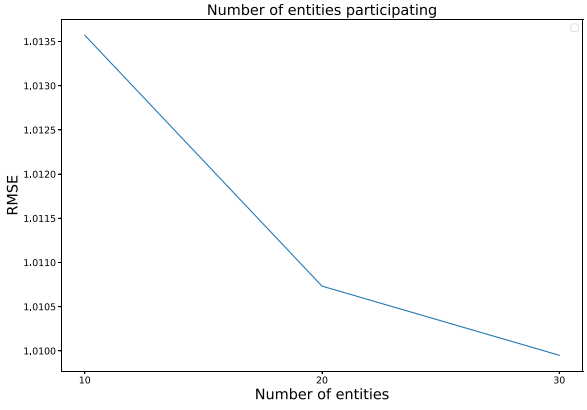


Fig. B4. Convergence of matrix factorization for different number of entities.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2022.108746.

References

[1] M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in: Proceedings of the SIGSAC, ACM, 2016, pp. 308–318.

[2] V.W. Anelli, Y. Deldjooo, T. Di Noia, A. Ferrara, F. Narducci, Federank: user controlled feedback with federated recommender systems, arXiv preprint arXiv:2012.11328 (2020).

[3] P. Bai, Y. Ge, F. Liu, H. Lu, Joint interaction with context operation for collaborative filtering, Pattern Recognit. 88 (2019) 729–738, doi:10.1016/j.patcog.2018.12.003.

[4] M.-F. Balcan, T. Dick, Y. Liang, W. Mou, H. Zhang, Differentially private clustering in high-dimensional Euclidean spaces, in: ICML, 2017, pp. 322–331.

[5] M. Balog, N. Tripuraneni, Z. Ghahramani, A. Weller, Lost relatives of the Gumbel trick, in: ICML, JMLR. org, 2017, pp. 371–379.

[6] A. Blum, C. Dwork, F. McSherry, K. Nissim, Practical privacy: the SuLQ framework, in: PODS, 2005, pp. 128–138.

[7] C.M. Bowen, F. Liu, Comparative study of differentially private data synthesis methods, arXiv preprint arXiv:1602.01063 (2016).

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., Distributed optimization and statistical learning via the alternating direction method of multipliers, FTML 3 (1) (2011) 1–122.

[9] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, D. Song, The secret sharer: Evaluating and testing unintended memorization in neural networks, in: {USENIX} Security Symposium, 2019, pp. 267–284.

[10] F. Chen, Z. Dong, Z. Li, X. He, Federated meta-learning for recommendation, arXiv preprint arXiv:1802.07876 (2018).

[11] Q. Dai, X.M. Wu, L. Fan, Q. Li, H. Liu, X. Zhang, D. Wang, G. Lin, K. Yang, Personalized knowledge-aware recommendation with collaborative and attentive graph convolutional networks, Pattern Recognit. 128 (2022) 108628, doi:10.1016/j.patcog.2022.108628.

[12] M. Ammad-ud din, E. Ivannikova, S.A. Khan, W. Oyomno, Q. Fu, K.E. Tan, A. Flanagan, Federated collaborative filtering for privacy-preserving personalized recommendation system, arXiv preprint arXiv:1901.09888 (2019).

[13] C. Ding, X. He, H.D. Simon, On the equivalence of nonnegative matrix factorization and spectral clustering, in: ICDM, 2005, pp. 606–610.

[14] L. Duan, W.N. Street, E. Xu, Healthcare information systems: data mining methods in the creation of a clinical recommender system, Enterp. Inf. Syst. 5 (2) (2011) 169–181.

[15] D. Durfee, R.M. Rogers, Practical differentially private top-k selection with pay-what-you-get composition, in: NeurIPS, 2019, pp. 3527–3537.

[16] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: TCC, Springer, 2006, pp. 265–284.

[17] D. Feldman, C. Xiang, R. Zhu, D. Rus, Coresets for differentially private k-means clustering and applications to privacy in mobile sensor networks, in: ISPN, 2017, pp. 3–16.

[18] D. Goldberg, D. Nichols, B.M. Oki, D. Terry, Using collaborative filtering to weave an information tapestry, Commun. ACM 35 (12) (1992) 61–71.

[19] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, D. Ramage, Federated learning for mobile keyboard prediction, arXiv preprint arXiv:1811.03604 (2018).

[20] S. Hasan, G.T. Duncan, D.B. Neill, R. Padman, Towards a collaborative filtering approach to medication reconciliation, in: Proceedings of the AMIA Annual Symposium, vol. 2008, American Medical Informatics Association, 2008, p. 288.

[21] S. Hassan, Z. Syed, From Netflix to heart attacks: Collaborative filtering in medical datasets, in: IHI, 2010, pp. 128–134.

[22] M.A. Heikkilä, A. Koskela, K. Shimizu, S. Kaski, A. Honkela, Differentially private cross-silo federated learning, arXiv preprint arXiv:2007.05553 (2020).

[23] J.L. Herlocker, J.A. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, 1999, pp. 230–237.

[24] B. Hitaj, G. Ateniese, F. Perez-Cruz, Deep models under the gan: Information leakage from collaborative deep learning, in: CCS, 2017, pp. 603–618.

[25] T.R. Hoens, M. Blanton, A. Steele, N.V. Chawla, Reliable medical recommendation systems with patient privacy, TIST 4 (4) (2013) 67.

[26] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: Proceedings of the ICDM, IEEE, 2008, pp. 263–272.

[27] A. Jalalirad, M. Scavuzzo, C. Capota, M. Sprague, A simple and efficient federated recommender system, in: Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, 2019, pp. 53–58, doi:10.1145/3365109.3368788.

[28] G. Jawaheer, M. Szomszor, P. Kostkova, Comparison of implicit and explicit feedback from an online music recommendation service, in: HetRec, 2010, pp. 47–51.

[29] P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., Advances and open problems in federated learning, arXiv preprint arXiv:1912.04977 (2019).

[30] R. Kannan, M. Ishteva, H. Park, Bounded matrix factorization for recommender system, Knowl. Inf. Syst. 39 (3) (2014) 491–511.

[31] A. Karatzoglou, L. Baltrunas, Y. Shi, Learning to rank for recommender systems, in: Proceedings of the 7th ACM Conference on Recommender Systems, 2013, pp. 493–494.

[32] S.P. Kasiviswanathan, H.K. Lee, K. Nissim, S. Raskhodnikova, A. Smith, What can we learn privately? SIAM J. Comput. 40 (3) (2011) 793–826.

[33] Y. Kim, J. Sun, H. Yu, X. Jiang, Federated tensor factorization for computational phenotyping, in: SIGKDD, 2017, pp. 887–895.

[34] A. Kobsa, J. Schreck, Privacy through pseudonymity in user-adaptive systems, TOIT 3 (2) (2003) 149–183.

[35] P.W. Koh, P. Liang, Understanding black-box predictions via influence functions, in: ICML, 2017, pp. 1885–1894.

[36] Y. Koren, Factorization meets the neighborhood: A multifaceted collaborative filtering model, in: SIGKDD, 2008, pp. 426–434.

[37] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer (Long Beach Calif) (8) (2009) 30–37.

[38] D.-S. Lee, G.-Y. Kim, H.-I. Choi, A web-based collaborative filtering system, Pattern Recognit 36 (2) (2003) 519–526, doi:10.1016/S0031-3203(02)00025-0. Biometrics.

[39] Y. Lin, P. Ren, Z. Chen, Z. Ren, D. Yu, J. Ma, M.d. Rijke, X. Cheng, Meta matrix factorization for federated rating predictions, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 981–990.
[40] C.D. Manning, P. Raghavan, H. Schütze, Evaluation in Information Retrieval, Cambridge University Press, 2008, pp. 139–161.
[41] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: RecSys, 2013, pp. 165–172.
[42] H.B. McMahan, G. Andrew, U. Erlingsson, S. Chien, I. Mironov, N. Papernot, P. Kairouz, A general approach to adding differential privacy to iterative training procedures, arXiv preprint arXiv:1812.06210 (2018).
[43] H.B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, AISTATS, 2017.
[44] F. McSherry, I. Mironov, Differentially private recommender systems: Building privacy into the netflix prize contenders, in: SIGKDD, 2009, pp. 627–636.
[45] F. McSherry, K. Talwar, Mechanism design via differential privacy, in: Proceedings of the FOCS, IEEE, 2007, pp. 94–103.
[46] F.D. McSherry, Privacy integrated queries: an extensible platform for privacy-preserving data analysis, in: Proceedings of the SIGMOD, ACM, 2009, pp. 19–30.
[47] B.N. Miller, J.A. Konstan, J. Riedl, Pocketlens: toward a personal recommender system, TOIS 22 (3) (2004) 437–476.
[48] S. Milli, L. Schmidt, A.D. Dragan, M. Hardt, Model reconstruction from model explanations, FAT*, 2019.
[49] P. Müllner, D. Kowald, E. Lex, Robustness of meta matrix factorization against strict privacy constraints, arXiv preprint arXiv:2101.06927 (2021).
[50] A. Nichol, J. Achiam, J. Schulman, On first-order meta-learning algorithms, arXiv preprint arXiv:1803.02999 (2018).
[51] K. Nissim, S. Raskhodnikova, A. Smith, Smooth sensitivity and sampling in private data analysis, in: STOC, 2007, pp. 75–84.
[52] G. Papandreou, A.L. Yuille, Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models, in: Proceedings of the ICCV, IEEE, 2011, pp. 193–200.
[53] D. Pollard, Quantization and the method of k-means, IEEE Trans. Inf. Theory 28 (2) (1982) 199–205.
[54] T.J. Pollard, A.E. Johnson, J.D. Raffa, L.A. Celi, R.G. Mark, O. Badawi, The eicu collaborative research database, a freely available multi-center database for critical care research, Sci. Data 5 (2018).
[55] B. Prenkaj, P. Velardi, D. Distante, S. Faralli, A reproducibility study of deep and surface machine learning methods for human-related trajectory prediction, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2169–2172.
[56] T. Qi, F. Wu, C. Wu, Y. Huang, X. Xie, Privacy-preserving news recommendation model learning, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, 2020, pp. 1423–1432.
[57] S. Rendle, Factorization machines, in: ICDM, 2010, pp. 995–1000.
[58] M.B. Sariyildiz, R.G. Cinbis, E. Ayday, Key protected classification for collaborative learning, Pattern Recognit. 104 (2020) 107327, doi:10.1016/j.patcog.2020.107327.
[59] E. Shakirova, Collaborative filtering for music recommender system, in: 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), 2017, pp. 548–550, doi:10.1109/ElConRus.2017.7910613.
[60] S. Sheikhalishahi, V. Balaraman, V. Osmani, Benchmarking machine learning models on multi-centre eicu critical care dataset, PLoS ONE 15 (7) (2020) e0235424.
[61] N. Srebro, R.R. Salakhutdinov, Collaborative filtering in a non-uniform world: Learning with the weighted trace norm, in: NeurIPS, 2010, pp. 2056–2064.
[62] S.A. Vavasis, On the complexity of nonnegative matrix factorization, SIAM J. Optim. 20 (3) (2010) 1364–1377.
[63] Y. Wang, Y.-X. Wang, A. Singh, Differentially private subspace clustering, in: NeurIPS, 2015, pp. 1000–1008.
[64] B. Wernly, B. Mamandipoor, P. Baldia, C. Jung, V. Osmani, Machine learning predicts mortality in septic patients using only routinely available abg variables: a multi-centre evaluation, Int. J. Med. Inform. 145 (2021) 104312.
[65] X. Wu, F. Li, A. Kumar, K. Chaudhuri, S. Jha, J. Naughton, Bolt-on differential privacy for scalable stochastic gradient descent-based analytics, in: Proceedings of the SIGMOD, ACM, 2017, pp. 1307–1322.
[66] Y. Xin, T. Jaakkola, Controlling privacy in recommender systems, in: NeurIPS, 2014, pp. 2618–2626.
[67] H.J. Xue, X. Dai, J. Zhang, S. Huang, J. Chen, Deep matrix factorization models for recommender systems, in: Proceedings of the IJCAI, vol. 17, Melbourne, Australia, 2017, pp. 3203–3209.
[68] J. Yang, X. Li, Z. Sun, J. Zhang, A differential privacy framework for collaborative filtering, Math. Probl. Eng. 2019 (2019).
[69] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, F. Beaufays, Applied federated learning: improving google keyboard query suggestions, arXiv preprint arXiv:1812.02903 (2018).
[70] X. Yu, F. Jiang, J. Du, D. Gong, A cross-domain collaborative filtering algorithm with expanding user and item features via the latent factor space of auxiliary domains, Pattern Recognit 94 (2019) 96–109, doi:10.1016/j.patcog.2019.05.030.
[71] S. Zhang, W. Wang, J. Ford, F. Makedon, Learning from incomplete ratings using non-negative matrix factorization, in: ICDM, 2006, pp. 549–553.
[72] Q. Zhao, F.M. Harper, G. Adomavicius, J.A. Konstan, Explicit or implicit feedback? engagement or satisfaction?: A field experiment on machine-learning-based recommender systems, SAC, 2018.
[73] T. Zhu, G. Li, W. Zhou, P.S. Yu, Differentially private data publishing and analysis: a survey, TKDE 29 (8) (2017) 1619–1638.

Mónica Ribero obtained her Bachelor's degree in Mathematics and is currently a Ph.D. student in Electrical and Computer Engineering at The University of Texas at Austin where she works on federated learning under privacy and communication constraints.

Jette Henderson earned her Ph.D. from The University of Texas at Austin in 2018 formulating methods of extracting insights from clinical data using tensor factorization. She currently performs research in algorithmic fairness and accountability at CognitiveScale.

Sinead Williamson earned her Ph.D. from the University of Cambridge in 2012, and is currently an Assistant Professor in the Department of Statistics and Data Science at the University of Texas at Austin. Her research focuses on Bayesian statistics and machine learning.

Haris Vikalo received a Ph.D. degree in electrical engineering from Stanford University in 2003. He is a Professor in the Department of Electrical and Computer Engineering at the University of Texas at Austin. His research interests include machine learning, signal processing, bioinformatics, and communications.