FedeRank: User Controlled Feedback with Federated Recommender Systems

Vito Walter Anelli, Yashar Deldjoo, Tommaso Di Noia, Antonio Ferrara*, and Fedelucio Narducci

Politecnico di Bari, Bari, Italy
{firstname.lastname}@poliba.it

Abstract. Recommender systems have shown to be a successful representative of how data availability can ease our everyday digital life. However, data privacy is one of the most prominent concerns in the digital era. After several data breaches and privacy scandals, the users are now worried about sharing their data. In the last decade, Federated Learning has emerged as a new privacy-preserving distributed machine learning paradigm. It works by processing data on the user device without collecting data in a central repository. In this paper, we present FedeRank, a federated recommendation algorithm. The system learns a personal factorization model onto every device. The training of the global model is modeled as a synchronous process between the central server and the federated clients. FedeRank takes care of computing recommendations in a distributed fashion and allows users to control the portion and type of data they want to share. By comparing with state-of-the-art centralized algorithms, extensive experiments show the effectiveness of FedeRank in terms of recommendation accuracy, even with a small portion of shared user data. Further analysis of the recommendation lists' diversity and novelty guarantees the suitability of the algorithm in real production environments.

Keywords: Recommender Systems · Collaborative Filtering · Federated Learning · Learning to Rank

1 Introduction

Recommender Systems (RSs) are well-known information-filtering systems widely adopted for mitigating the information-overload problem. Indeed, the broad amount of items and services has caused a cognitive impairment that takes the name of over-choice, or choice overload. RSs have proved to be very useful in making possible personalized access to these catalogs of items. These systems are generally hosted on centralized servers and train their models by exploiting massive proprietary and sensitive data. However, public awareness related to data collection was spurred and increased. In recent years, an increasing number of countries have introduced regulations to protect user privacy and data security.

* Corresponding author

Representative examples are the GDPR in the European Union [12], the CCPA in California [7], and the Cybersecurity Law in China [38]. Such data protection policies prohibit free data circulation and force personal data to remain isolated and fragmented.

In this context, Google has recently proposed Federated Learning (FL) as a privacy-by-design technique which tackles data isolation while meeting the need for big data [20, 30]. FL trains a global machine-learning model by leveraging both users' data and personal devices' computing capabilities. Unlike previous approaches, it keeps data on the devices (e.g., laptops, mobile phones, tablets, and edge devices) without sharing them with a central server. Today, Federated Learning is considered the best candidate to face the data privacy, control, and property challenges posed by the data regulations.

Among the recommendation paradigms proposed in the literature, Collaborative Filtering (CF) demonstrated a very high accuracy [29, 41]. The strength of CF recommendation algorithms is that users who expressed similar tastes in the past tend to agree in the future as well. One of the most prominent CF approaches is the Latent Factor Model (LFM) [23]. LFMs uncover users and items latent representation, whose linear interaction can explain observed feedback.

In this paper, we introduce FedeRank, a novel factorization model that embraces the Federated Learning paradigm. A disruptive effect of employing FedeRank is that users participating in the federation process can decide if and how they are willing to disclose their private sensitive preferences. Indeed, FedeRank mainly leverages non-sensitive information (e.g., items the user has not experienced). Here, we show that even only a small amount of sensitive information (i.e., items the user has experienced) lets FedeRank reach a competitive accuracy. How incomplete data impacts the performance of the system is an entirely unexplored field. Analogously, it is still to establish the minimum amount of data necessary to build an accurate recommendation system [40]. At the same time, preserving privacy at the cost of a worse tailored recommendation may frustrate users and reduce the "acceptance of the recommender system" [32]. In this work, instead of focusing on how to protect personal information from privacy breaches (that is explored in other active research fields), we investigate how to guarantee the users the control and property of their data as determined by regulations. The work's contributions are manifold due to the number of open challenges that still exist with the new Federated Learning paradigm. To summarize, our contributions in this paper include:

- the development of the first, to the best of our knowledge, federated pair-wise recommendation system, and an analysis of the quality of recommendation with respect to local computation amount;
- an investigation on the best trade-off between between sharing personal data and recommendation utility;
- an analysis of the algorithmic bias on the final recommendation lists, based on the feedback deprivation level.

To this extent, we have carried out extensive experiments on three real-world datasets (*Amazon Digital Music*, *LibraryThing*, and *MovieLens 1M*) by considering two evaluation criteria: (a) the accuracy of recommendations measured by exploiting precision and recall, (b) beyond-accuracy measures to evaluate the novelty, and the diversity of recommendation lists. The experimental evaluation shows that FedeRank provides high-quality recommendations, even though it leaves users in control of their data.

2 Related Work

In the last decades, academia and industry have proposed several competitive recommendation algorithms. Among the Collaborative Filtering algorithms, the most representative examples are undoubtedly Nearest Neighbors systems, Latent Factor Models, and Neural Network-based recommendation systems. The Nearest Neighbors scheme has shown its competitiveness for decades. After them, factorization-based recommendation emerged thanks to the disruptive idea of Matrix Factorization (MF). Nevertheless, several generalized/specialized variants have been proposed, such as FM [33], SVD++ [21], PITF [36], FPMC [35]. Unfortunately, rating-prediction-oriented optimization, like SVD, has shown its limits in the recommendation research [31]. Consequently, a new class of *Learning to Rank* algorithms has been developed in the last decade, mainly ranging from point-wise [25] to pair-wise [34], through List-wise [37] approaches. Among pair-wise methods, BPR [34] is one of the most broadly adopted, thanks to its outstanding capabilities to correctly rank preserving an acceptable computational complexity. Finally, in the last years, methods that exploit the various architectures of deep neural networks have established themselves either in search and recommendation research.

To make RSs work properly (easing the user decision-making process and boosting the business), they need to collect user information related to attributes, demands, and preferences [17], jeopardizing the user's privacy. In this scenario — and, more generally, in any scenario with a system learning from sensitive data — Federated Learning was introduced for meeting privacy shortcomings by horizontally distributing the model's training over user devices [30]. Beyond privacy, Federated Learning has posed several other challenges and opened new research directions [18]. In the last years, Federated learning has extended to a more comprehensive idea of privacy-preserving decentralized collaborative ML approaches [39]. These techniques included horizontal federations where different devices (and local datasets) share the same feature space. On the contrary, in vertical federations, devices share training samples that differ in feature space.

Some researchers focused the attention on the decentralized and distributed matrix-factorization approaches [11, 13]. However, in this work, we focus on federated learning principles theoretically and practically different from classical distributed approaches. Indeed, Federated Learning assumes the presence of a coordinating server and the use of private and self-produced data on each node. In general, distributed approaches do not guarantee these assumptions. Ammad-ud-din *et al.* [3] propose a federated implementation of collaborative filtering, whose security limits are analyzed in [10], which uses the SVD-MF method for implicit feedback [16]. Here, the training is a mixture of Alternating Least Squares (ALS) and Stochastic Gradient Descent (SGD) for preserving users' privacy. Nevertheless, incomprehensibly, almost no work addressed top-N recommendation exploiting the "Learning to rank" paradigm. In this sense, one rare example is the work by Kharitonov *et al.* [19], who recently proposed to combine evolution strategy optimization with a privatization procedure based on differential privacy. The previous work focuses neither on search or recommendation. Perhaps, like ours, it can be classified as a federated learning-to-rank algorithm. Finally, Yang *et al.* [40] identified some recent Federated Learning challenges and open research directions.

3 Approach

In this section, we introduce the fundamental concepts regarding the Collaborative Filtering recommendation using a Federated Learning scheme. Along with the problem definition, the notation we adopt is presented.

The recommendation problem over a set of users $\mathcal{U}$ and a set of items $\mathcal{I}$ is defined as the activity of finding for each user $u \in \mathcal{U}$ an item $i \in \mathcal{I}$ that maximizes a utility function $g : \mathcal{U} \times \mathcal{I} \to \mathbb{R}$. Let $\mathbf{X} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$ be the user-item matrix containing for each element x_{ui} an implicit feedback (e.g., purchases, visits, clicks, views, check-ins) of user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$. Therefore, $\mathbf{X}$ only contains binary values, $x_{ui} = 1$ and $x_{ui} = 0$ denoting whether user u has consumed or not item i, respectively.

The recommendation model is based on Factorization approach, originally introduced by Matrix Factorization [24], that became popular in the last decade thanks to its state-of-the-art recommendation accuracy [26]. This technique aims to build a model Θ in which each user u and each item i is represented by the embedding vectors $\mathbf{p}_u$ and $\mathbf{q}_i$, respectively, in the shared latent space $\mathbb{R}^F$. Let assume $\mathbf{X}$ can be factorized such that the dot product between $\mathbf{p}_u$ and $\mathbf{q}_i$ can explain any observed user-item interaction x_{ui}, and any non-observed interaction can be estimated as $\hat{x}_{ui}(\Theta) = b_i(\Theta) + \mathbf{p}_u^T(\Theta) \cdot \mathbf{q}_i(\Theta)$ where b_i is a term denoting the bias of the item i.

Among pair-wise approaches for learning-to-rank the items of a catalog, Bayesian Personalized Ranking [34] is the most broadly adopted, thanks to its capabilities to correctly rank with *acceptable* computational complexity. Given a training set defined by $\mathcal{K} = \{(u, i, j) \mid x_{ui} = 1 \land x_{uj} = 0\}$, BPR minimizes the ranking loss by exploiting the criterion $\max_{\Theta} G(\Theta)$, with $G(\Theta) = \sum_{(u,i,j) \in \mathcal{K}} \ln\ \sigma(\hat{x}_{uij}(\Theta)) - \lambda \|\Theta\|^2$, where $\hat{x}_{uij}(\Theta) = \hat{x}_{ui}(\Theta) - \hat{x}_{uj}(\Theta)$ is a real value modeling the relation between user u, item i and item j, $\sigma(\cdot)$ is the sigmoid function, and λ is a model-specific regularization parameter to prevent overfitting. Pair-wise optimization applies to a wide range of recommendation models, including factorization. Hereafter, we denote the model $\Theta = \langle \mathbf{P}, \mathbf{Q}, \mathbf{b} \rangle$, where $\mathbf{P} \in \mathbb{R}^{|\mathcal{U}| \times F}$ is a matrix whose u-th row corresponds to the vector $\mathbf{p}_u$, and $\mathbf{Q} \in \mathbb{R}^{|\mathcal{I}| \times F}$ is a matrix in which the i-th row corresponds to the vector $\mathbf{q}_i$. Finally, $\mathbf{b} \in \mathbb{R}^{|\mathcal{I}|}$ is a vector whose i-th element corresponds to the value b_i.

3.1 FedeRank

FedeRank redesigns the original factorization approach for a federated setting. Indeed, the initial factorization model and its variants use a single, centralized model, which does not guarantee users to control their data. FedeRank splits the pair-wise learning model Θ among a central server S and a federation of users $\mathcal{U}$. Federated learning aims to optimize a global loss function by using data distributed among a federation of users' devices. The rationale is that the server no longer collects private users' data. Rather, it aggregates the results of some steps of local optimizations performed by clients, preserving privacy, ownership, and locality of users' data [6]. Formally, let Θ be the machine learning model parameters, and $G(\Theta)$ be a loss function to minimize. In Federated learning, the users $\mathcal{U}$ of a federation collaborate to minimize G (under the coordination of a central server S) without sharing or exchanging their raw data. From an algorithmic point of view, S shares Θ with the federation of devices. Then, the optimization problem of minimizing G is locally solved. Since each user participates to the federation with her personal data and with her personal client device, we hereinafter will interchangeably use the terms "client", "user", and "device".

To set up the framework, we consider the central server S holding a model $\Theta_S = \langle \mathbf{Q}, \mathbf{b} \rangle$, where $\mathbf{Q} \in \mathbb{R}^{|\mathcal{I}| \times F}$ is a matrix in which i-th row represents the embedding $\mathbf{q}_i$ for item i in the catalog, while the element b_i of $\mathbf{b} \in \mathbb{R}^{|\mathcal{I}|}$ is the bias of item i. On the other hand, each user $u \in \mathcal{U}$ holds a local model $\Theta_u = \langle \mathbf{p}_u \rangle$, where $\mathbf{p}_u \in \mathbb{R}^F$ corresponds to the representation of user u in the latent space of dimensionality F. Each user holds a private interaction dataset $\mathbf{x}_u \in \mathbb{R}^{|\mathcal{I}|}$, which — compared to a centralized recommender system — corresponds to the $\mathbf{X}$'s u-th row. The user u leverages her private dataset $\mathbf{x}_u$ to build the local training set $\mathcal{K}_u = \{(u, i, j) \mid x_{ui} = 1 \land x_{uj} = 0\}$. Finally, the overall number of interactions in the system can be obtained by exploiting the local datasets. Let us define it as $X^+ = \sum_{u \in \mathcal{U}} |\{x_{ui} | x_{ui} = 1\}|$.

The training procedure iterates for E *epochs*, in each of which *rpe rounds of communication* between the server and the devices are performed. A round of communication envisages a **Distribution to Devices** → **Federated Optimization** → **Transmission to Server** → **Global Aggregation** sequence. The notation $\{\cdot\}_S^t$ denotes an object computed by the server S at round t, while $\{\cdot\}_u^t$ indicates an object computed by a specific client u at round t.

1. **Distribution to Devices.** Let $\{\mathcal{U}^-\}_S^t$ be a subset of $\mathcal{U}$ with cardinality m, containing m clients $u \in \mathcal{U}$. The set $\{\mathcal{U}^-\}_S^t$ can be either defined by S, or the result of a request for availability sent by S to clients in $\mathcal{U}$. Each client $u \in \{\mathcal{U}^-\}_S^t$ receives from S the latest snapshot of $\{\Theta_S\}_S^{t-1}$.
2. **Federated Optimization.** Each user $u \in \{\mathcal{U}^-\}_S^t$ generates the set $\{\mathcal{K}_u^-\}_u^t$ containing T random triples (u, i, j) from $\mathcal{K}_u$. It is worth underlining that Rendle [34] suggests, for a centralized scenario, to train the recommendation model by randomly choosing the training triples from $\mathcal{K}$, to avoid data is traversed item-wise or user-wise, since this may lead to slow convergence.

Conversely, in a federated approach, we require to train the model user-wise. Indeed, the learning is separately performed on each device (u), that only knows the data in $\mathcal{K}_u$. Thanks to the user-wise traversing, FedeRank can decide who controls (the designer or the user) the number of triples T in the training set $\{\mathcal{K}_u^-\}_u^t$, to tune the degree of local computation. With the local training set, the user u can compute her contribution to the overall optimization of Θ_S with the following update:

$$\{\Delta\Theta_S\}_u^t = \{\Delta\langle\mathbf{Q},\mathbf{b}\rangle\}_u^t := \sum_{(u,i,j)\in\{\mathcal{K}_u^-\}_u^t} \frac{\partial}{\partial\Theta_S} \ln\ \sigma(\hat{x}_{uij}(\{\Theta_S\}_S^{t-1};\{\Theta_u\}_u^{t-1})), \tag{1}$$

plus a regularization term. At the same time, the client u updates its local model Θ_u, and incorporates it in the current model by using:

$$\{\Delta\Theta_u\}_u^t = \{\Delta\langle\mathbf{p}_u\rangle\}_u^t := \sum_{(u,i,j)\in\{\mathcal{K}_u^-\}_u^t} \frac{\partial}{\partial\Theta_u} \ln\ \sigma(\hat{x}_{uij}(\{\Theta_S\}_S^{t-1};\{\Theta_u\}_u^{t-1})), \tag{2}$$

plus a regularization term. The partial derivatives in Eq. 1 and 2 are straightforward, and can be easily computed by following the scheme proposed by Rendle *et al.* [34].

At the end of the federated computation, each client can update its local model Θ_u — containing the user profile — by aggregating the computed update:

$$\{\Theta_u\}_u^t := \{\Theta_u\}_u^{t-1} + \alpha\{\Delta\Theta_u\}_u^t, \tag{3}$$

with α being the learning rate.

3. **Transmission to Server.** In a purely distributed architecture, each user in $\mathcal{U}^-$ returns to S the computed update. Here, instead of sending $\{\Delta\Theta_S\}_u^t$, each user transmits a modified version $\{\Delta\Theta_S^\Phi\}_u^t$. To introduce this aspect of FedeRank, let us define $\mathcal{F} = \{i, \forall(u,i,j) \in \{\mathcal{K}_u^-\}_u^t\}$, and $\Phi = \langle \mathbf{Q}^\Phi, \mathbf{b}^\Phi \rangle$, with $\mathbf{Q}^\Phi \in \mathbb{R}^{|\mathcal{I}|\times F}$, and $\mathbf{b}^\Phi \in \mathbb{R}^{|\mathcal{I}|}$.
Each row $\mathbf{q}_i^\Phi$ of $\mathbf{Q}^\Phi$ and each element b_i^Φ of $\mathbf{b}^\Phi$ assume their value according to the probabilities:

$$\begin{gathered} P(\mathbf{q}_i^\Phi = \mathbf{1}, b_i^\Phi = 1 \mid i \in \mathcal{F}) = \pi, \quad P(\mathbf{q}_i^\Phi = \mathbf{0}, b_i^\Phi = 0 \mid i \in \mathcal{F}) = 1-\pi, \\ P(\mathbf{q}_i^\Phi = \mathbf{1}, b_i^\Phi = 1 \mid i \notin \mathcal{F}) = 1 \end{gathered} \tag{4}$$

Based on $\{\mathbf{Q}^\Phi\}_u^t$ and $\{\mathbf{b}^\Phi\}_u^t$, $\Delta\Theta_S^\Phi$ can be computed as it follows:

$$\{\Delta\Theta_S^\Phi\}_u^t = \{\Delta\Theta_S\}_u^t \odot \{\Phi\}_u^t := \left\langle \{\Delta\mathbf{Q}\}_u^t \odot \{\mathbf{Q}^\Phi\}_u^t, \{\Delta\mathbf{b}\}_u^t \odot \{\mathbf{b}^\Phi\}_u^t \right\rangle, \tag{5}$$

where the operator $\odot$ denotes the Hadamard product. This transformation is motivated by a possible privacy issue.

The update $\Delta\mathbf{Q}$ computed in Eq. 1 is a matrix whose rows are non-zero in correspondence of the items i and j of all the triples $(u, i, j) \in \mathcal{K}_u^-$ [34]. An analogous behavior can be observed for the elements of $\Delta\mathbf{b}$. Focusing on the non-zero elements, we observe that, for each triple $(u, i, j) \in \mathcal{K}_u^-$, the updates $\{\Delta\mathbf{q}_i\}_u^t$ and $\{\Delta\mathbf{q}_j\}_u^t$, as well as $\{\Delta b_i\}_u^t$ and $\{\Delta b_j\}_u^t$, show the same absolute value with opposite sign [34]. In fact, this makes completely distinguishable for the server the consumed and the non-consumed items of user u, allowing S to reconstruct $\mathcal{K}_u^-$, thus raising a privacy issue.

Since our primary goal is to put users in control of their data, we leave users the possibility to choose a fraction π of positive item updates to send. The remaining positive item updates (a fraction $1 - \pi$) are masked by setting them to zero, by means of the transformation in Eq. 5. On the other hand, the negative updates are always sent to S, since their corresponding rows are always multiplied by a $\mathbf{1}$ vector. Indeed, these updates are related to non-consumed items, which are indistinguishably negative or missing values and are assumed to be *non-sensitive* data.

4. **Global Aggregation.** Once S has received $\{\Delta\Theta_S^{\Phi}\}_u^t$ from all clients $u \in \mathcal{U}^-$, it aggregates the received updates in $\mathbf{Q}$ and $\mathbf{b}$ to build the new global model, with α being the learning rate:

$$\{\Theta_S\}_S^t := \{\Theta_S\}_S^{t-1} + \alpha \sum_{u \in \mathcal{U}^-} \{\Delta\Theta_S^{\Phi}\}_u^t. \tag{6}$$

4 Experiments

Datasets. We have investigated the performance of FedeRank considering three well-known datasets: *Amazon Digital Music* [28], *LibraryThing* [42], and *MovieLens 1M* [15]. The former includes the users' satisfaction feedback for a catalog of music tracks available with Amazon Digital Music service. It contains 1,835 users and 41,488 tracks, with 75,932 ratings ranging from 1 to 5. *LibraryThing* collects the users' ratings on a book catalog. It captures the interactions of 7,279 users on 37,232 books. It provides more than two million ratings with 749,401 unique ratings in a range from 1 to 10. The latter is *MovieLens 1M* dataset, which collects users' ratings in the movie domain: it contains 1,000,209 ratings ranging from 1 to 5, 6,040 users, and 3,706 items. We have filtered out users with less than 20 ratings (considering them as cold-users). Table 1 shows the characteristics of the resulting datasets adopted in the experiments.

Baseline Algorithms. We compared FedeRank with representative centralized algorithms to position its performance with respect to the state-of-the-art techniques: **VAE** [27], a non-linear probabilistic model taking advantage of Bayesian inference to estimate the model parameters; **User-kNN** and **Item-kNN** [22], two neighbor-based CF algorithms, that exploit cosine similarity to compute similarity between users or items; **BPR-MF** [34], the centralized vanilla BPR-MF implementation; and **FCF** [3], the only federated recommendation approach,

Table 1: Characteristics of the evaluation dataset used in the offline experiment: $|\mathcal{U}|$ is the number of users, $|\mathcal{I}|$ the number of items, X^+ the amount of positive feedback.

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{I}\rvert$	X^+	$\frac{X^+}{\lvert\mathcal{U}\rvert}$	$\frac{X^+}{\lvert\mathcal{I}\rvert}$	$\frac{X^+}{\lvert\mathcal{I}\rvert\cdot\lvert\mathcal{U}\rvert}\%$
Amazon DM	1,835	41,488	75,932	41.38	1.83	0.000997%
LibraryThing	7,279	37,232	749,401	102.95	20.13	0.002765%
MovieLens 1M	6,040	3,706	1,000,209	165.60	269.89	0.044684%

to date, based on MF[1]. We have evaluated FedeRank considering $|\mathcal{U}^-| = 1$. That is, in each round of communication we involve only a single client to avoid noisy results. We thereby guarantee the sequential training, needed to compare against centralized pir-wise techniques. We have investigated with two different FedeRank settings. In the **first setting**, we have set $T = 1$, i.e., each client extracts solely one triple (u, i, j) from its dataset when asked for training the model; with this special condition, we test if FedeRank is effectively comparable to BPR. Moreover, to make the comparison completely fair, we extract triples as proposed by Rendle *et al.* [34]. The **second setting** follows a real Federated scenario where the client local computation is not limited to a single triple. Specifically, the number T of triples extracted by each client is set to $\frac{X^+}{|\mathcal{U}|}$.

Reproducibility and Evaluation Metrics. To train FedeRank, we have adopted a realistic temporal hold-out 80-20 splitting for the training set and test set [14]. We have further split the training set adopting a temporal hold-out strategy on a user basis to pick the last 20% of interactions as a validation set. Hence, we have explored a grid in the range $\{0.005, \ldots, 0.5\}$. Then, to ensure a fair comparison, we have used the same learning rate to train FedeRank. We have set up the remaining parameters as follows: the user- and positive item-regularization parameter is set to 1/20 of the learning rate; conversely, the negative item-regularization parameter is set to 1/200 of the learning rate as suggested in *mymedialite*[2] implementation as well as in [4]. Moreover, for each setting, we have selected the best model in the first 20 epochs. Finally, the number of latent factors is equal to 20. This value reflects a trade-off between latent factors' expressiveness and memory space limits (given by a realistic Federated Learning environment). We have measured the recommendation accuracy by exploiting: Precision ($P@N$) (the proportion of relevant items in the recommendation list), and Recall ($R@N$), that measures the relevant suggested items. Regarding diversity, we have adopted Item Coverage (IC) and Gini Index (G). The former provides the overall number of diverse recommended items, and it highlights the degree of personalization expressed by the model [1]. The latter measures how unequally an RS provides users with different items [9], being higher values corresponding to more tailored lists.

[1] Since no source code is available, we reimplemented it in the reader's interest.

[2] http://www.mymedialite.net/

Table 2: Recommendation performance of FedeRank on *Amazon Digital Music*, *LibraryThing* and *MovieLens 1M*, with respect to the other baselines. For each configuration of T, we show the best FedeRank setting of π based on P@10.

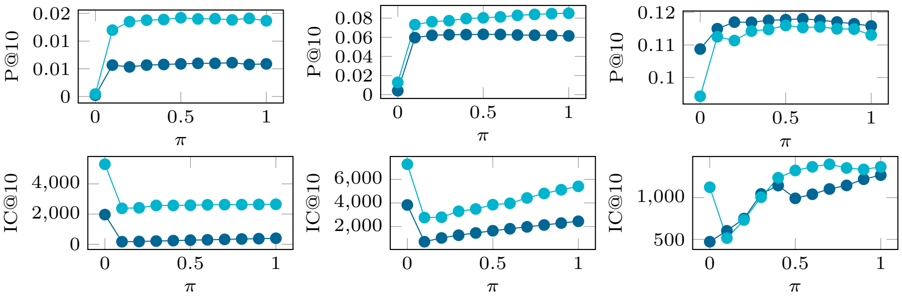
	Amazon Digital Music				**LibraryThing**				**MovieLens 1M**			
	P@10	**R@10**	**IC@10**	**G@10**	**P@10**	**R@10**	**IC@10**	**G@10**	**P@10**	**R@10**	**IC@10**	**G@10**
Random	0.00005	0.00005	14186	0.28069	0.00054	0.00028	31918	0.60964	0.00871	0.00283	3666	0.85426
Most Popular	0.00469	0.00603	24	0.00023	0.05013	0.03044	36	0.00031	0.10224	0.03924	118	0.00569
User-kNN	0.01940	0.02757	4809	0.04115	0.14193	0.10115	3833	0.01485	0.12613	0.06701	737	0.04636
Item-kNN	0.02147	0.03171	4516	0.03801	0.20214	0.14778	12737	0.09979	0.08873	0.05475	2134	0.19292
VAE	0.01580	0.02289	3919	0.04179	0.10834	0.07711	7800	0.04638	0.11735	0.06192	1476	0.09259
BPR-MF	0.00921	0.01298	739	0.00415	0.07009	0.04303	3082	0.01359	**0.11911**	0.05817	**1444**	**0.08508**
FCF	0.00839	0.01222	**2655**	0.01861	**0.10760**	0.04392	829	0.01305	0.10760	0.04392	829	0.01305
FedeRank												
$T = 1$	0.00610	0.00889	349	0.00136	0.06309	0.03738	1650	0.00512	0.11805	**0.05902**	1041	0.06608
$T = X^+/\|\mathcal{U}\|$	**0.01422**	**0.02060**	2586	**0.02153**	0.08512	**0.05627**	**5404**	**0.02784**	0.11599	0.05571	1326	0.02513

4.1 Performance of Federated Learning to Rank

We begin our experimental evaluation by investigating the efficacy of FedeRank, and we assess whether its performance is comparable to baseline algorithms. Table 2 depicts the results in terms of accuracy and diversity. For FedeRank, we reported distinct results related to the two federated experimental settings. The Table is visually split into two parts. The algorithms in the bottom part (BPR-MF, FCF, and FedeRank) are the factorization-based models. The upper part provides the positioning of FedeRank to the other state-of-the-art approaches. Starting with the factorization-based methods, we can note that BPR-MF outperforms FedeRank for $T = 1$, but it remains at about 67% and 88% of the centralized algorithm for *Amazon Digital Music* and *LibraryThing*, respectively. However, the realistic Federated setting is with $T = X^+/|\mathcal{U}|$. Here, FedeRank consistently improves the recommendation performance with respect to BPR-MF and FCF, over the three datasets. Actually, for *Amazon Digital Music* and *LibraryThing* FedeRank improves accuracy metrics of about 50% and 25% with respect to BPR-MF. The achievement can be explained as an advantage brought by the increased local computation. It is worth noticing that this results partially contradict Rendle *et al.*[34] since they hypothesize that traversing user-wise the training triples would worsen the recommendation performance. The accuracy improvements we observed are not visible in *MovieLens 1M*, where we witness results comparable or worse than BPR-MF, probably due to the overfitting caused by the very high ratio between ratings and items. FedeRank with increased computation still results robust with respect to the IC metric, since, in general, it outperforms or remains comparable to FCF and BPR-MF.

4.2 Analysis of Positive Feedback Transmission Ratio

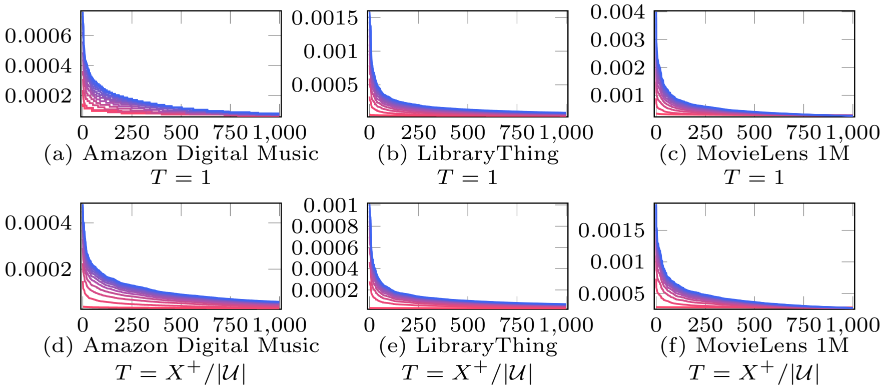
We have extensively analyzed the behavior of FedeRank when tuning π for sending progressive fractions of positive feedback in $[0.0, \ldots, 1.0]$ with step 0.1. We believe that the most important dimensions for this analysis are accuracy (Precision), and aggregate diversity (Item Coverage). Figure 1 reports the results for



(a) Amazon Digital Music (b) LibraryThing (c) MovieLens 1M

Fig. 1: F1 performance at different values of π in the range $[0.1, 1]$. Dark blue is $T = 1$, light blue is $T = X^+/|\mathcal{U}|$.

the two experimented settings. Even here, *Amazon Digital Music* and *LibraryThing* show similar trends. The accuracy of the recommendation progressively increases reaching the maximum with fractions 0.8 and 0.5, respectively, for $T = 1$, and with fractions 0.9 and 1.0 for $T = X^+/|\mathcal{U}|$. First, this suggests that, at the beginning of the training, some positive feedback is needed for establishing the value of an item. Notwithstanding, even with $\pi = 0.1$ (i.e., sharing just 10% of private information), we witness a jump in recommendation accuracy (one order of magnitude), reaching up to 92% of the best accuracy. We should also observe another significant behavior. With a fraction of 0.0, we observe a high value of *IC*, with poor recommendation accuracy. It suggests that the system could not capture population preferences, and it behaves similarly to Random. However, even with a small fraction of positive feedback like 0.1, we observe a significant decrease in diversity metrics. The system learns which items are popular and starts suggesting them. Moreover, if we observe large fractions, we may notice that diversity increases as we feed the system with more information. For *MovieLens 1M*, it is worth noticing that FedeRank shows accuracy performance extremely close to the best value by sharing only 10% of positive interactions. This behavior may be due to several reasons. Firstly, *MovieLens 1M* is a relatively dense dataset in the recommendation scenario (it has a sparsity of 0.955). Secondly, it shows a very high user-item ratio [2] (i.e., 1.63) compared to *Amazon Digital Music* (0.04), and *LibraryThing* (0.20), and it shows high values for the average number of ratings per user (132.87), and ratings per item $(216, 56)$. All these clues suggest that the system learns how to rank items even without the need for the totality of ratings. If we focus on diversity metrics, *IC*, and *Gini*, we may notice that diversity is progressively increasing from fraction 0.1 to 1.0. It suggests that the system recommends a small number of popular items with a fraction of 0.1, while it provides more diversified recommendation lists considering larger portions of positive user feedback. At this stage of the analysis, we can draw an interesting consideration: in general, the highest accuracy values do not correspond to the fraction of 1.0. This observation connects to a broad debate regarding the amount of information needed for a recommendation task. Specifically, the experiments show that, initially, the recommender struggles to



(a) Amazon Digital Music $T = 1$

(b) LibraryThing $T = 1$

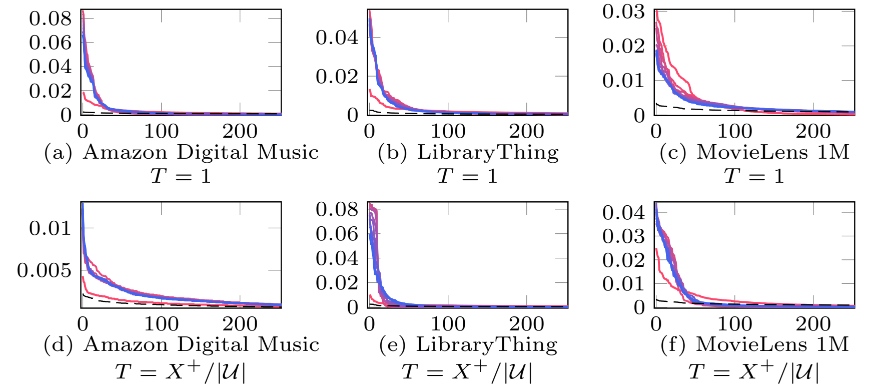
(c) MovieLens 1M $T = 1$

(d) Amazon Digital Music $T = X^+/|\mathcal{U}|$

(e) LibraryThing $T = X^+/|\mathcal{U}|$

(f) MovieLens 1M $T = X^+/|\mathcal{U}|$

Fig. 2: Normalized number of item updates during the training: the 1,000 most updated items for different values of π (from $\pi = 0.0$ in red to $\pi = 1.0$ in blue).



(a) Amazon Digital Music $T = 1$

(b) LibraryThing $T = 1$

(c) MovieLens 1M $T = 1$

(d) Amazon Digital Music $T = X^+/|\mathcal{U}|$

(e) LibraryThing $T = X^+/|\mathcal{U}|$

(f) MovieLens 1M $T = X^+/|\mathcal{U}|$

Fig. 3: Normalized number of recommendations for each item (colored curves from $\pi = 0.0$ in red to $\pi = 1.0$ in blue) vs. normalized amount of positive feedback per item (black dashed curve). The 250 most popular items are shown.

suggest relevant items without positive feedback (fraction 0.0). However, with a small injection of feedback, the system starts to work well. Nonetheless, in *Amazon Digital Music* and *LibraryThing*, if we increase the fraction, we witness an increase concerning accuracy only until a certain value of π. Although this consideration, we observe an increase in diversity metrics when we continue to increase the value of π. Since it has a small or even detrimental impact on accuracy, those items might be unpopular items erroneously suggested to users.

4.3 Study on FedeRank algorithmic bias

In this section, we study how incomplete transmission of user feedback affects the popularity of items in the final recommendations and during the learning process. It is essential to discover whether the exploitation of a Federated Learning approach influences the algorithmic bias, determining popular items to be over-represented [5, 8]. To conduct this study, we have re-trained FedeRank with all the previously considered π. For each experiment, we analyzed the data flow between the clients and the server. Afterward, we have extracted the number of updates for each item. Figure 2 illustrates the occurrences for the 1,000 most updated items. In the Figure, the curves denote the different values of π. Analogously, we considered for each experiment the final top-10 recommendation list of each user. Following the same strategy, we analyzed the occurrences of the items in the recommendation. Then, we ordered items from the most to the least recommended, and we plotted the occurrences of the first 250 in Figure 3. To compare the different datasets, we have normalized the values considering the overall dataset occurrences. Figure 2 shows that data disclosure — i.e., the value of π — highly influences the information exchanged during the training process. Additionally, the update frequency curve exhibits a constant behavior for all the datasets, when $\pi = 0.0$. This trend suggests that items are randomly updated without taking into account any information about item popularity. This behavior explains the high *IC* entirely observed in Figure 1 for $\pi = 0.0$. Moreover, the curve for $\pi = 0.1$ shows that the exchanged data is enough to provide the system with information about item popularity. The curves suggest that the information on item popularity is being injected into the system. By increasing the value of π, the trend becomes more evident. Due to the original rating distribution, the system initially exchanges more information about the very popular items. To analyze the algorithmic bias, we can observe Figure 3. Remarkably, item popularity in recommendation lists does not vary as we may expect based on the previous analysis. The setting $\pi = 0.0$ is an exception, as extensively explained before. Since in *Amazon Digital Music* and *LibraryThing* the updates sent by the clients are randomly selected between the negative items, FedeRank acts like a Random recommender. Thus, the system cannot catch popularity information, and, as the plots make clear, it struggles to make the right items popular. Finally, we can focus on the curves for $\pi > 0$. It is particularly noteworthy that the different π curves behave similarly, and they propose the same proportion of popular items. The curves' trends suggest that the recommendation model completely absorbs the initial variation in exchanged item distribution, unveiling another unknown aspect of factorization models.

5 Conclusion and Future Work

In this paper, we have tackled the problem of putting users in control of their private data for a recommendation scenario. Witnessing the growing concern about privacy, users might want to exploit their sensitive data and share only a small fraction of their feedback. In such a context, classic CF approaches are no more feasible. To overcome these problems, we have proposed FedeRank, a novel recommendation framework that respects the Federated Learning paradigm. With FedeRank, private user feedback remains on user devices unless they decide to share it. On the other hand, FedeRank ensures high-quality recommendations despite the constrained setting. We have extensively studied the performance of FedeRank by comparing it with other state-of-the-art methods. We have then analyzed the impact of progressive deprivation of user feedback, and we studied the effects on the diversity of the recommendation results. Finally, we have investigated if the federated algorithm imposes an algorithmic bias to the generated recommendation lists. The study paves the way for further research directions. On the one hand, the results' analysis suggests that centralized recommender systems are not performing at their best. Indeed, feeding recommender systems with all the available feedback, without any filtering, may lead to a performance worsening. On the other hand, the competitive results of FedeRank suggest that the Federated Learning-based algorithms show a recommendation quality that makes them suitable to be adopted on a massive scale.

References

1. G. Adomavicius and Y. Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Trans. Knowl. Data Eng.*, 24(5):896–911, 2012.
2. G. Adomavicius and J. Zhang. Impact of data characteristics on recommender systems performance. *ACM Trans. Management Inf. Syst.*, 3(1):3:1–3:17, 2012.
3. M. Ammad-ud-din, E. Ivannikova, S. A. Khan, W. Oyomno, Q. Fu, K. E. Tan, and A. Flanagan. Federated collaborative filtering for privacy-preserving personalized recommendation system. *CoRR*, abs/1901.09888, 2019.
4. V. W. Anelli, T. D. Noia, E. D. Sciascio, C. Pomo, and A. Ragone. On the discriminative power of hyper-parameters in cross-validation and how to choose them. In *Proc. 13th ACM Conf. on Recommender Systems*, pages 447–451. ACM, 2019.
5. R. Baeza-Yates. Bias in search and recommender systems. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*, page 2, 2020.
6. K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecný, S. Mazzocchi, H. B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander. Towards federated learning at scale: System design. *CoRR*, abs/1902.01046, 2019.
7. California State Legislature. The california consumer privacy act of 2018, 2018.
8. R. Cañamares and P. Castells. Should I follow the crowd?: A probabilistic analysis of the effectiveness of popularity in recommender systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 415–424, 2018.
9. P. Castells, N. J. Hurley, and S. Vargas. Novelty and diversity in recommender systems. In F. Ricci, L. Rokach, and B. Shapira, editors, *Recommender Systems Handbook*, pages 881–918. Springer, 2015.
10. D. Chai, L. Wang, K. Chen, and Q. Yang. Secure federated matrix factorization. *IEEE Intelligent Systems*, (01):1–1, aug 5555.
11. E. Duriakova, E. Z. Tragos, B. Smyth, N. Hurley, F. J. Peña, P. Symeonidis, J. Geraci, and A. Lawlor. Pdmfrec: a decentralised matrix factorisation with tunable user-centric privacy. In *Proc. of the 13th ACM Conf. on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019.*, pages 457–461, 2019.
12. European Commission. 2018 reform of eu data protection rules, 2018.

13. R. Fierimonte, S. Scardapane, A. Uncini, and M. Panella. Fully decentralized semi-supervised learning via privacy-preserving matrix completion. *IEEE Trans. Neural Networks Learn. Syst.*, 28(11):2699–2711, 2017.
14. A. Gunawardana and G. Shani. Evaluating recommender systems. In F. Ricci, L. Rokach, and B. Shapira, editors, *Recommender Systems Handbook*, pages 265–308. Springer, 2015.
15. F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.
16. Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. of the 8th IEEE Int. Conf. on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pages 263–272. IEEE Computer Society, 2008.
17. A. J. P. Jeckmans, M. Beye, Z. Erkin, P. H. Hartel, R. L. Lagendijk, and Q. Tang. Privacy in recommender systems. In N. Ramzan, R. van Zwol, J. Lee, K. Clüver, and X. Hua, editors, *Social Media Retrieval*, Computer Communications and Networks, pages 263–281. Springer, 2013.
18. P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. A. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. D'Oliveira, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao. Advances and open problems in federated learning. 2019.
19. E. Kharitonov. Federated online learning to rank with evolution strategies. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 249–257, 2019.
20. J. Konecný, H. B. McMahan, D. Ramage, and P. Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *CoRR*, abs/1610.02527, 2016.
21. Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pages 426–434, 2008.
22. Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(1):1–24, 2010.
23. Y. Koren and R. M. Bell. Advances in collaborative filtering. In F. Ricci, L. Rokach, and B. Shapira, editors, *Recommender Systems Handbook*, pages 77–118. Springer, 2015.
24. Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.
25. Y. Koren and J. Sill. Ordrec: an ordinal model for predicting personalized item rating distributions. In B. Mobasher, R. D. Burke, D. Jannach, and G. Adomavicius, editors, *Proc. of the 2011 ACM Conf. on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, pages 117–124. ACM, 2011.
26. D. kumar Bokde, S. Girase, and D. Mukhopadhyay. Role of matrix factorization model in collaborative filtering algorithm: A survey. *CoRR*, abs/1503.07475, 2015.
27. D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference*, pages 689–698, 2018.

28. J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proc. of the 38th Int. ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, pages 43–52, 2015.
29. B. McFee, L. Barrington, and G. R. G. Lanckriet. Learning content similarity for music recommendation. *IEEE Trans. Audio, Speech & Language Processing*, 20(8):2207–2218, 2012.
30. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.
31. S. M. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems*, pages 1097–1101, 2006.
32. K. Muhammad, Q. Wang, D. O'Reilly-Morgan, E. Z. Tragos, B. Smyth, N. Hurley, J. Geraci, and A. Lawlor. Fedfast: Going beyond average for faster training of federated recommender systems. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1234–1242, 2020.
33. S. Rendle. Factorization machines. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pages 995–1000, 2010.
34. S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In J. A. Bilmes and A. Y. Ng, editors, *UAI 2009, Proc. of the Twenty-Fifth Conf. on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 452–461. AUAI Press, 2009.
35. S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 811–820, 2010.
36. S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010*, pages 81–90, 2010.
37. Y. Shi, M. Larson, and A. Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 269–272, 2010.
38. Standing Committee of the National People's Congress of Popular Republic of China. China internet security law, 2017.
39. Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM TIST*, 10(2):12:1–12:19, 2019.
40. Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu. *Federated learning.* Morgan & Claypool Publishers, 2019.
41. J. Yuan, W. Shalaby, M. Korayem, D. Lin, K. AlJadda, and J. Luo. Solving cold-start problem in large-scale recommendation engines: A deep learning approach. In *2016 IEEE Int. Conf. on Big Data, BigData 2016, Washington DC, USA, December 5-8, 2016*, pages 1901–1910. IEEE Computer Society, 2016.
42. T. Zhao, J. McAuley, and I. King. Improving latent factor models via personalized feature projection for one class recommendation. In *Proc. of the 24th ACM Int. on Conf. on information and knowledge management*, pages 821–830, 2015.