Efficient and Robust Regularized Federated Recommendation

Langming Liu
City University of Hong Kong
Hong Kong, China

Wanyu Wang
City University of Hong Kong
Hong Kong, China

Xiangyu Zhao
City University of Hong Kong
Hong Kong, China

Zijian Zhang*
Jilin University
Changchun, China

Chunxu Zhang
Jilin University
Changchun, China

Shanru Lin
City University of Hong Kong
Hong Kong, China

Yiqi Wang
Michigan State University
East Lansing, United States

Lixin Zou
Wuhan University
Wuhan, China

Zitao Liu
Jinan University
Guangzhou, China

Xuetao Wei
Southern University of Science and Technology
Shenzhen, China

Hongzhi Yin
The University of Queensland
Brisbane, Australia

Qing Li
The Hong Kong Polytechnic University
Hong Kong, China

Abstract

Recommender systems play a pivotal role across practical scenarios, showcasing remarkable capabilities in user preference modeling. However, the centralized learning paradigm predominantly used raises serious privacy concerns. The federated recommender system (FedRS) addresses this by updating models on clients, while a central server orchestrates training without accessing private data. Existing FedRS approaches, however, face unresolved challenges, including non-convex optimization, vulnerability, potential privacy leakage risk, and communication inefficiency. This paper addresses these challenges by reformulating the federated recommendation problem as a convex optimization issue, ensuring convergence to the global optimum. Based on this, we devise a novel method, RFRec, to tackle this optimization problem efficiently. In addition, we propose RFRecF, a highly efficient version that incorporates non-uniform stochastic gradient descent to improve communication efficiency. In user preference modeling, both methods learn local and global models, collaboratively learning users' common and personalized interests under the federated learning setting. Moreover, both methods significantly enhance communication efficiency, robustness, and privacy protection, with theoretical support. Comprehensive evaluations on four benchmark datasets demonstrate RFRec and RFRecF's superior performance compared to diverse baselines. The code is available to ease reproducibility[1].

*Zijian Zhang is corresponding author. zhangzj2114@mails.jlu.edu.cn

[1]https://github.com/Applied-Machine-Learning-Lab/RFRec

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679682

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

Recommendation, Federated Learning, Communication

ACM Reference Format:

Langming Liu, Wanyu Wang, Xiangyu Zhao, Zijian Zhang, Chunxu Zhang, Shanru Lin, Yiqi Wang, Lixin Zou, Zitao Liu, Xuetao Wei, Hongzhi Yin, and Qing Li. 2024. Efficient and Robust Regularized Federated Recommendation . In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3627673.3679682

1 Introduction

In recent years, recommender systems (RSs) [10, 12, 17, 21, 22, 25, 28, 40, 44, 48, 53] have become prevalent in various practical scenarios, such as e-commerce, online movie or music platforms, providing personalized recommendations for users [4, 19, 26, 30, 31, 33, 45]. Matrix Factorization (MF)-based models [10, 16, 17, 38], the most prevalent RS solutions, leverage a centralized learning paradigm that collects user data and trains the model in a central server. The centralized approach can address user preferences well and has achieved great success. However, there emerge increasing concerns from individual users about privacy issues during information storage and collection in central server [51, 52]. In response, the authorities are strengthening data privacy protection legally [52], such as issuing the General Data Protection Regulation (GDPR) [43]. Exploring a practical approach that can train the models without directly utilizing user data is urgent in the recommendation domain.

Federated Learning (FL) [14, 15, 36] has proven its efficacy in addressing the aforementioned data-isolated problem. In contrast to the centralized learning paradigm, FL learns the optimal model by continuously communicating the model parameters or gradients between the local client (user) and central server without aggregating client data to the server. Therefore, FL is promising to be the

solution to address the privacy issue in the recommendation. Recently, some efforts have been exerted to integrate the FL technique into recommender systems to address the privacy issue, named as federated recommender systems (FedRS) [1, 5, 7, 24, 27, 39, 46, 49]. Using the same optimization formulation as centralized RS, mainstream FedRS methods (e.g., MF-based FedRS) learn the user and item latent feature vectors and utilize their inner product to predict the rating matrix. In the FL setting, the user feature vectors are updated only on the clients. In contrast, the gradients of item feature vectors are aggregated to the server for item feature matrix updating. This approach can train the recommendation model without accessing and utilizing user data, alleviating privacy concerns.

However, we recognize that existing FedRS methods generally suffer from several deficiencies. **1) Non-convex Optimization**: most FedRS methods solve the RS optimization problem formulated as the MF error minimization directly [1, 5, 24], which is essentially non-convex. The non-convex optimization may converge to sub-optimal points, jeopardizing the recommendation performance. Moreover, constrained by the FL setting, the user and item feature vectors defined in the formulation must be updated in the client and server, respectively, which can lead to unstable model training. **2) Vulnerability**: the existing methods leverage the alternating update rule, where the client and server highly depend on each other to update [1, 47, 49]. Therefore, the update process may terminate when some clients cannot participate due to network or device issues. **3) Privacy Leakage Risk**: in the training process, the transmission of gradients may still lead to private information leaks [5]. Specifically, knowing two continuous gradients of a user during the communication process, the server can deduce the user ratings by the algebra operation of these two gradients. Though privacy protection methods can be equipped to prevent privacy leakage, such as leveraging homomorphic encryption and generating pseudo items [5, 29, 37], such methods bring considerable additional computational or communication costs. **4) Communication Inefficiency**: FedRS methods have a low convergence rate due to the alternating update rule, no matter alternating least squares (ALS) [8, 47] or alternating stochastic gradient descent (SGD) [3, 17]. Specifically, these methods converges to the optimal with ε-accuracy in at least $O(\log m + \log \frac{1}{\varepsilon})$ iterations [11, 18, 47], where m is the number of items. Hence, the number of communication rounds of existing methods is also $O(\log m + \log \frac{1}{\varepsilon})$, which is tremendous when m is large.

In this paper, we first propose a problem formulation for FedRS and reformulate its optimization as a convex problem, which guarantees that the model converges to global optimal. As a regularized empirical risk minimization (RERM) [34, 54], it consists of cumulative local loss (task term) and global loss (regularization term). Under the guidance of this formulation, we design two methods, namely RFRec and its faster version, RFRecF. In particular, we propose a **R**egularized **F**ederated learning method for **Rec**ommendation (**RFRec**) to solve this RERM problem in a local GD manner. It theoretically ensures strict convergence to optimal with a linear convergence rate, which has an edge on communication efficiency. In addition, to pursue further communication efficiency, we leverage a non-uniform SGD training manner to update the task term and regularization term stochastically (**RFRecF**). This optimization mechanism enables RFRecF to achieve fewer expected communication rounds per iteration, thereby reducing communication costs and striking a balance between performance and efficiency. The minimum communication iterations of both methods is $O(\log \frac{1}{\varepsilon})$, which is independent of item number m and achieves a remarkable reduction over existing methods. Our main contributions can be summarized as follows:

- We propose a convex optimization formulation of FedRS, formulated as a regularized empirical risk minimization (RERM) problem, which can guide FedRS methods design with theoretical support for convergence and communication efficiency.
- We put forward RFRec to solve the proposed RERM problem and the faster version, RFRecF, for further communication efficiency. We theoretically and experimentally prove their robustness and privacy preservation capability, and they own a guarantee of advancing communication efficiency.
- Extensive experiments are conducted on four public benchmark datasets, demonstrating the state-of-the-art performance of our proposed methods against advanced baselines. We also conduct in-depth analysis to verify the efficacy and efficiency.

2 Preliminary

2.1 Recommender Systems

Denote the number of users and items as n, m, respectively. In RSs, the prevalent MF-based methods [16, 17, 38] suppose the rating matrix $\hat{\boldsymbol{R}} \in \mathbb{R}^{n\times m}$ is the inner product of user feature matrix $\boldsymbol{U} \in \mathbb{R}^{d\times n}$ and item feature matrix $\boldsymbol{V} \in \mathbb{R}^{d\times m}$, formulated as $\hat{\boldsymbol{R}} = \boldsymbol{U}^T\boldsymbol{V}$. So the predicted rating of user i to item j is computed as $\hat{r}_{ij} = \boldsymbol{u}_i^T\boldsymbol{v}_j$, where $\boldsymbol{u}_i \in \mathbb{R}^d$ and $\boldsymbol{v}_j \in \mathbb{R}^d$ are user and item latent factor vectors, respectively. The optimization objective is as follows:

$$\begin{aligned} J &= \|\boldsymbol{R} - \boldsymbol{U}^T\boldsymbol{V}\|^2 + \lambda_u\|\boldsymbol{U}\|^2 + \lambda_v\|\boldsymbol{V}\|^2 \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m}(r_{ij} - \boldsymbol{u}_i^T\boldsymbol{v}_j)^2 + \lambda_u\sum_{i=1}^{n}\|\boldsymbol{u}_i\|^2 + \lambda_v\sum_{j=1}^{m}\|\boldsymbol{v}_j\|^2, \end{aligned} \tag{1}$$

where λ_u, λ_v are scaling parameters of penalties.

2.2 Federated Learning

The prevalent optimization objective of FL [20] is an ERM problem

$$\min_x \frac{1}{n}\sum_{i=1}^{n} f_i(x), \tag{2}$$

where n is the number of clients, x encodes the parameters of the global model (e.g., weight and bias of NN), and f_i represents the aggregate loss of the i-th client.

2.3 Federated Recommender Systems

The optimization objective of FedRS is the same as RS. However, in the setting of FL, the rating vector $\boldsymbol{R}_i \in \mathbb{R}^{1\times d}$ is only available on user i, which obstacles the update of item vectors. The solution is to separate updates into two parts: clients and servers.

The user vectors $\boldsymbol{u}_i$ are updated locally by GD:

$$\nabla_{u_i} J = -2\sum_{j=1}^{m}(r_{ij} - \boldsymbol{u}_i^T\boldsymbol{v}_j)\boldsymbol{v}_j + 2\lambda_u\boldsymbol{u}_i \tag{3}$$

The item vector $\boldsymbol{v}_j$ is updated in central server by aggregating gradient $h(i,j)$ calculated in clients:

$$\begin{aligned} &\text{client: } h(i,j) = (r_{ij} - \boldsymbol{u}_i^T \boldsymbol{v}_j)\boldsymbol{u}_i \\ &\text{server: } \nabla_{v_j} J = -2\sum_{i=1}^{n} h(i,j) + 2\lambda_v \boldsymbol{v}_j \end{aligned} \tag{4}$$

The updated item vectors will be distributed to clients for the next update step. FedRS is implicitly meant to train several local models (i.e., $\boldsymbol{u}_i$) and a global model (i.e., $\boldsymbol{V}$). We observe a gap between the optimization objectives of FedRS and general FL, making it inconvenient to design effective algorithms and conduct analyses.

3 Methodology

We reformulate the FedRS problem with an RERM formulation. Then, we propose RFRec and RFRecF to solve the problem, as illustrated in Figure 1, providing robustness and strong privacy protection. The convergence and communication results are provided.

3.1 Problem Formulation of FedRS

To interpret the recommendation task under the FL setting, we formulate FedRS as a RERM problem, learning the local and global models simultaneously. We define the local models as $x = (x_1, x_2, \cdots, x_n)$, where tuple $x_i = (\boldsymbol{u}_i, \boldsymbol{V}_{(i)})$ denotes the local model of client (user) i, and $\boldsymbol{u}_i \in \mathbb{R}^d$ is the user feature vector. Notice that $\boldsymbol{V}_{(i)} \in \mathbb{R}^{d\times m}$ is the local item matrix, which differs from $v_j \in \mathbb{R}^d$. Denote $\boldsymbol{R}_i \in \mathbb{R}^{1\times m}$ as user i's rating vector. The global model is $\bar{\boldsymbol{V}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{V}_{(i)}$. Here, we formularize the problem of FedRS as follows

$$\min_{x_1,\cdots,x_n} \{F(x) := f(x) + \lambda\psi(x)\}, \tag{5}$$

$$f(x) := \sum_{i=1}^{n} f_i(x_i) = \sum_{i=1}^{n} \{\|\boldsymbol{R}_i - \boldsymbol{u}_i^T \boldsymbol{V}_{(i)}\|^2 + \lambda_u \|\boldsymbol{u}_i\|^2\},$$

$$\psi(x) := \sum_{i=1}^{n} \psi_i(x_i) = \frac{1}{2}\sum_{i=1}^{n} \|\boldsymbol{V}_{(i)} - \bar{\boldsymbol{V}}\|^2,$$

where f and ψ are the cumulative local loss and the regularization term. f_i and ψ_i are the losses of client i. Notice that $\|\cdot\|$ refers to the L2-norm. Minimizing f obtains optimal local model $x_i = (\boldsymbol{u}_i, \boldsymbol{V}_{(i)})$ for each client. The regularization term ψ is added to penalize the difference between $\boldsymbol{V}_{(i)}$ and $\bar{\boldsymbol{V}}$. Therefore, minimizing ψ obtains the global item matrix $\bar{\boldsymbol{V}}$. In addition, compared to the original optimization formulation (1), we omit the penalty term $\lambda_v\|\boldsymbol{V}_{(i)}\|^2$, since the regularization term ψ can undertake the same role as it and this penalty term may jeopardize the training process.

By tuning λ to balance the learning of local and global models, we can learn the optimal recommendation model by solving problem (5). In addition, the regularization term can provide strong convexity, referred to in Lemma 3.3. Moreover, the formulation (5) can guide the designing of efficient algorithms. We denote $x(\lambda) := (x_1(\lambda), \cdots, x_n(\lambda))$ as the optimal model of problem (5). Then, we aim at devising relative training methods to obtain $x(\lambda)$.

3.2 Methods

We propose two methods to solve optimization problem (5), RFRec and RFRecF. Instead of communicating gradients like most FedRS methods, the proposed methods communicate models for federated learning, alleviating the risk of privacy leaks. In addition, they achieve higher communication efficiency than FedRS methods.

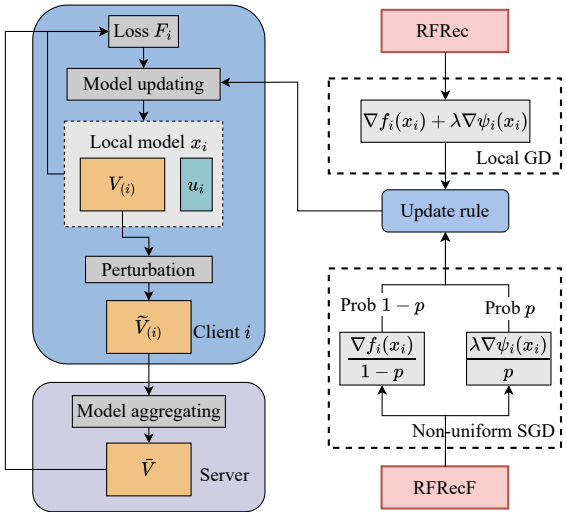


Figure 1: Overview of RFRec and RFRecF.

3.2.1 ***RFRec.*** To achieve the optimal solution of problem (5) stably and efficiently, we propose RFRec, which conducts local GD in clients to update models. First, We add the regularization term $\psi_i(x_i)$ to main target to get the local optimization target $F_i(x_i) = f_i(x_i) + \lambda\psi_i(x_i)$. Then, the local model $x_i = (\boldsymbol{u}_i, \boldsymbol{V}_{(i)})$ can be updated on client i by gradient descent update rule as follows

$$x_i^{k+1} = x_i^k - \alpha\nabla F_i(x_i), \tag{6}$$

where α is the learning rate, λ is the penalty parameter, and x^k denotes the model at step k. However, clients i cannot compute $\psi_i(x_i)$ without knowing the average model $\bar{\boldsymbol{V}}$. So, the clients upload $\boldsymbol{V}_{(i)}$ to the server for aggregation (i.e., calculating $\bar{\boldsymbol{V}}$). Afterward, $\bar{\boldsymbol{V}}$ is distributed to clients for the local model update. The final outputs of RFRec are the local models $\{\boldsymbol{u}_i\}_{i=1}^n$ and the global model $\bar{\boldsymbol{V}}$, which means $\boldsymbol{u}_i$ is only available on user i, and $\bar{\boldsymbol{V}}$ is shared parameter. The pseudo-code is specified in Algorithm 1.

The significant advantages of RFRec are guaranteeing convergence to optimal and possessing a nice communication efficiency, specified in Section 3.5 and 3.6. However, the communication cost is still high as RFRec needs two communication rounds per iteration.

3.2.2 ***RFRecF.*** To further improve the communication efficiency based on RFRec, we apply a non-uniform SGD [3, 9] method in the FL training process and name it RFRecF (fast version). We define the stochastic gradient of F as follows:

$$G(x) := \begin{cases} \dfrac{\nabla f(x)}{1-p} & \text{with probability} \quad 1-p, \\ \dfrac{\lambda\nabla\psi(x)}{p} & \text{with probability} \quad p, \end{cases} \tag{7}$$

where $G(x)$ is an unbiased estimator of $\nabla F(x)$. Notice that the SGD here is not the standard approach that randomly samples data or mini-batch data for updates [3]. Instead, we add randomness by stochastically using the gradient of the main loss f or regularization term ψ. Then we similarly define the local stochastic gradient $G_i(x_i)$

Algorithm 1 RFRec

Input: n clients and server, Ratings $\boldsymbol{R}$
Output: local user vectors $\{\boldsymbol{u}_i\}_{i=1}^n$ and global item matrix $\bar{\boldsymbol{V}}$
Initialization: $x = \{x_i\}_{i=1}^n = \{\boldsymbol{u}_i, \boldsymbol{V}_{(i)}\}_{i=1}^n$, $\bar{\boldsymbol{V}} \sim \mathcal{N}(0, 10^{-4})$, parameters λ, α
for $k = 0, 1, \cdots, K$ **do**
  Local client update
  for $i = 1, \cdots, n$ in parallel **do**
    $x_i \leftarrow x_i - \alpha(\nabla f_i(x_i) + \lambda \nabla \psi_i(x_i))$
    Client i generates and sends perturbed $\widetilde{\boldsymbol{V}}_{(i)}$ to server
  end for
  Central server aggregate: $\bar{\boldsymbol{V}} \leftarrow \frac{1}{n}\sum_{i=1}^n \widetilde{\boldsymbol{V}}_{(i)}$
  Distributed $\bar{\boldsymbol{V}}$ to clients
end for

as an unbiased estimator of $\nabla F_i(x_i)$. The updating step of RFRecF on client i is formulated as

$$x_i^{k+1} = x_i^k - \alpha G_i(x_i^k). \tag{8}$$

The clients conduct the stochastic update throughout the process, while the central server only undertakes the model aggregation. Specifically, we generate a Bernoulli random variable $\zeta = 1$ with probability p and 0 with probability $1-p$ each time. If $\zeta = 0$, the client conducts a local stochastic update, and else the server calculates the average model $\bar{\boldsymbol{V}}$. In addition, the local stochastic update is divided into two parts: **main update** (i.e., GD of $\nabla f_i(x_i)$) with probability $1-p$ and **moving to average** (i.e., GD of $\nabla \psi_i(x_i)$) with probability p. Note that the communication is only needed when the two consecutive ζ are different (i.e., $\zeta_{k-1} = 0, \zeta_k = 1$ or $\zeta_{k-1} = 1, \zeta_k = 0$), which provide high communication efficiency. The pseudo-code of RFRecF is specified in Algorithm 2.

RFRecF can provide even higher communication efficiency than RFRec, and the theoretical support is in Section 3.6. Note that both proposed methods conduct complete model updates at clients, while the server only aggregates models. This approach is critical to enhance **Robustness**, and we will discuss it in the next section.

3.3 In-depth Analysis of Robustness

The existing methods are vulnerable when facing low participation of clients (e.g., some devices lose connection) since client and server updates highly depend on each other (Section 2.3). To address the issue, we strip off update tasks from the server, which means the server only undertakes aggregation. There are two advantages: **1) For server**: the model aggregation of $\boldsymbol{V}_{(i)}$ will not be influenced when a portion of devices cannot participate in aggregation. By the law of large numbers, the sample average converges almost surely to the expected value, so we can use the average of remaining devices $\frac{1}{n_t}\sum_{i=1}^{n_t} \boldsymbol{V}_{(i)}$ to approximate $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{V}_{(i)}$, where $n_t < n$ represents the number of participated devices at round t. **2) For clients**: each client can continue local updates even without connection to the server for a while since the average model $\bar{\boldsymbol{V}}$ is stable.

3.4 Enhanced Privacy Protection

Initially, there was a vast difference between local model $\boldsymbol{V}_{(i)}$, where the information leak is more likely to happen when each client uploads $\boldsymbol{V}_{(i)}$ to server. To address this potential issue, we apply a clipping mechanism and leverage local differential privacy (LDP) [2, 6] (e.g., using the Laplace mechanism) on $\boldsymbol{V}_{(i)}^k$ as follows:

$$\widetilde{\boldsymbol{V}}_{(i)}^k = \text{Clip}(\boldsymbol{V}_{(i)}^k, \delta) + \text{Laplace}(0, s) \tag{9}$$

The protecting mechanism ensure ϵ-DP ($\epsilon = \frac{2\delta}{s}$), where ϵ is privacy budget, smaller the better. Then, the perturbed $\widetilde{\boldsymbol{V}}_{(i)}^k$ is sent to the server for averaging. The perturbation can be mostly offset when the server takes the average of $\widetilde{\boldsymbol{V}}_{(i)}^k$, which has a slight influence on convergence. In Section 3.5, we will show that RFRec and RFRecF's convergence results still hold even after adding this perturbation.

Algorithm 2 RFRecF

Input: n clients and server, Ratings $\boldsymbol{R}$
Output: local user vectors $\{\boldsymbol{u}_i\}_{i=1}^n$ and global item matrix $\bar{\boldsymbol{V}}$
Initialization: $x = \{x_i\}_{i=1}^n = \{\boldsymbol{u}_i, \boldsymbol{V}_{(i)}\}_{i=1}^n$, $\bar{\boldsymbol{V}} \sim \mathcal{N}(0, 10^{-4})$, parameters λ, α, p
for $k = 0, 1, \cdots, K$ **do**
  random variable $\zeta_k = 1$ with prob p and 0 with prob $1-p$
  if $\zeta_k = 0$ **then**
    local client update
    for $i = 1, \cdots, n$ in parallel **do**
      if $\zeta_{k-1} = 0$ or $k = 0$ **then**
        main update: $x_i \leftarrow x_i - \frac{\alpha}{1-p}\nabla f_i(x_i)$
      else
        moving to average: $\boldsymbol{V}_{(i)} \leftarrow \boldsymbol{V}_{(i)} - \frac{\alpha}{p}\lambda(\boldsymbol{V}_{(i)} - \bar{\boldsymbol{V}})$
      end if
    end for
  else
    central server aggregate: $\bar{\boldsymbol{V}} \leftarrow \frac{1}{n}\sum_{i=1}^n \widetilde{\boldsymbol{V}}_{(i)}$
  end if
  Communication round
  if $\zeta_{k-1} = 0, \zeta_k = 1$ **then**
    for $i = 1, \cdots, n$ in parallel **do**
      client i generates and sends perturbed $\widetilde{\boldsymbol{V}}_{(i)}$ to server
    end for
  else if $\zeta_{k-1} = 1, \zeta_k = 0$ **then**
    server send $\bar{\boldsymbol{V}}$ to clients
  end if
end for

3.5 Convergence Results

In this section, we give the convergence results of the proposed methods, demonstrating their effectiveness theoretically and laying the foundation for deducing communication results, and the important proofs are attached in **Appendix** A. We first reasonably assume the model parameters are uniformly bounded.

Assumption 1. *Suppose the model parameters are uniformly bounded such that* $\|\boldsymbol{u}_i\| \le M_u, \|\boldsymbol{V}_{(i)}\| \le M_v, \|\boldsymbol{R}_i\| \le M_r$.

The theoretical results in this paper are all based on Assumption 1, and we omit the description of assumption for simplicity. Then, we define convexity and smoothness as follows.

Definition 3.1. A function $g : \mathbb{R}^d \to \mathbb{R}$ is $L-$smooth with $L > 0$ if

$$g(x) \le g(y) + \nabla g(y)^T (x-y) + \frac{L}{2}\|x-y\|^2, \quad \forall x, y \in \mathbb{R}^d,$$

and it is μ-strongly convex with $\mu > 0$ if

$$g(x) \ge g(y) + \nabla g(y)^T (x-y) + \frac{\mu}{2}\|x-y\|^2, \quad \forall x, y \in \mathbb{R}^d.$$

Lemma 3.2. *Each f_i ($i = 1, \cdots, n$) is L-smooth.*

Lemma 3.3. *Let $\lambda > \frac{2}{\lambda_u} M_r^2 + 6M_u^2$, we have F is μ-strongly convex and L_F-smooth with $L_F = L + \lambda$.*

The strong convexity and smoothness of function are beneficial for model convergence, as the former can speed up the convergence rate while the latter can stabilize the training process. We first give the convergence results of the proposed methods (i.e., RFRec and RFRecF) without perturbation as follows.

Theorem 3.4. ***(RFRec)*** *When $0 \le \alpha \le \frac{1}{L+\lambda}$, we have*

$$\mathbb{E}[\|x^k - x(\lambda)\|^2] \le (1-\alpha\mu)^k \|x^0 - x(\lambda)\|^2, \quad \forall k \in \mathbb{N}, \tag{10}$$

Theorem 3.5. ***(RFRecF)*** *When $0 < \alpha \le \frac{1}{2\mathcal{L}}$, we have*

$$\mathbb{E}[\|x^k - x(\lambda)\|^2] \le (1-\alpha\mu)^k \|x^0 - x(\lambda)\|^2 + \frac{2\alpha\sigma^2}{\mu}, \quad \forall k \in \mathbb{N}, \tag{11}$$

where $\mathcal{L} := \max\{\frac{L}{1-p}, \frac{\lambda}{p}\}$, and

$$\sigma^2 := \sum_{i=1}^{n} \left(\frac{1}{1-p} \|\nabla f_i(x_i(\lambda))\|^2 + \frac{\lambda^2}{p} \|\boldsymbol{V}_{(i)}(\lambda) - \bar{\boldsymbol{V}}(\lambda)\|^2 \right).$$

Generally, the variance term σ^2 will be eliminated with iterations. Specifically, When $k \to 0$, we have $\boldsymbol{V}_{(i)} \to \bar{\boldsymbol{V}}$ for all i, and $\nabla f(x(\lambda)) \to 0$, therefore $\sigma^2 \to 0$. So, the results reflect that both RFRec and RFRecF have **linear convergence rates**, and the iteration times are both $\frac{1}{\alpha\mu} \log \frac{1}{\varepsilon}$ to obtain $O(\varepsilon)$-accuracy. For RFRec, $\alpha \le \frac{1}{L+\lambda}$; for RFRecF, we have $\alpha \le \frac{1}{2\mathcal{L}}$ and $\mathcal{L} = \max\{\frac{L}{1-p}, \frac{\lambda}{p}\} \ge L + \lambda$ (equivalent when $p = \frac{\lambda}{L+\lambda}$). Then we conclude the results of optimal iteration times as follows.

Corollary 3.6. *The optimal number of iterations of RFRec is $\frac{L+\lambda}{\mu} \log \frac{1}{\varepsilon}$ by selecting $\alpha = \frac{1}{L+\lambda}$. The optimal number of iterations of RFRecF is $\frac{2(L+\lambda)}{\mu} \log \frac{1}{\varepsilon}$ by selecting $p = \frac{\lambda}{L+\lambda}$ and $\alpha = \frac{1}{2(L+\lambda)}$.*

RFRec obtains the same convergence rate as GD, $O(\log \frac{1}{\varepsilon})$. In contrast, the convergence rate of RFRecF seems worse than RFRec since each iteration only conducts one SGD step. Nevertheless, RFRecF can obtain a better communication efficiency than RFRec and will be shown in Section 3.6 and Section 4.3.

Then, we provide the convergence result of the proposed methods with perturbation as follows.

Corollary 3.7. ***(Perturbation)*** *The optimal iteration time of RFRec and RFRecF is $O(\log \frac{1}{\varepsilon})$ to obtain $(O(\varepsilon) + \frac{2\alpha s^2}{\mu})$-accuracy when adding the perturbation with noise strength s.*

The above result indicates that the proposed methods still guarantee convergence. However, an additional error $\frac{2\alpha s^2}{\mu}$ emerges in the error neighbor, related to perturbation. Recall that the perturbation ensures $\frac{2\delta}{s}$-DP, which means a trade-off exists between model convergence and privacy.

3.6 Analysis of Communication Rounds

In this section, we deduce the communication results of RFRec and RFRecF using the convergence results. Then, we discuss the parameter selection strategy to obtain optimal communication efficiency.

3.6.1 ***Optimal communication rounds.*** We first define the communication round [35] to represent communication cost.

Definition 3.8. A communication round is a round where clients send information (e.g., model parameters and gradients) to the server, or the server sends information to clients.

The communication rounds of RFRec are twice the iterations. Since at each iteration, the clients upload perturbed model $\widetilde{\boldsymbol{V}}_{(i)}$ to the server, and the server distributes average model $\bar{\boldsymbol{V}}$ to the clients.

We consider the expected number of communication rounds of RFRecF because of the applied SGD manner. According to Algorithm 2, communication is only needed at the current iteration when the current and previous iterations are at different places (e.g., client→server, server→client). Thus, the probability of a communication round at each iteration is $2p(1-p)$, and the expected number of communication rounds of RFRecF is as follows.

Lemma 3.9. *The expected numbers of communication rounds in T iterations of RFRecF and RFRecF are $2T$ and $2p(1-p)T$, respectively.*

According to derivation of Corollary 3.6, we can obtain optimal T of RFRecF by selecting $\alpha = \frac{1}{2\mathcal{L}}$. It is trivial to obtain the expected number of communication rounds of RFRecF is $\frac{4}{\mu} \max\{Lp, \lambda(1-p)\} \log \frac{1}{\varepsilon}$, where the optimal value of p is $\frac{\lambda}{L+\lambda}$, the same as in Corollary 3.6. We conclude the communication results as follows.

Corollary 3.10. *The optimal number of communication rounds of RFRec is $2\frac{L+\lambda}{\mu} \log \frac{1}{\varepsilon}$ by selecting $\alpha = \frac{1}{L+\lambda}$. The optimal expected number of communication rounds of RFRecF is $\frac{4\lambda}{L+\lambda} \frac{L}{\mu} \log \frac{1}{\varepsilon}$ by selecting $p = \frac{\lambda}{L+\lambda}$ and $\alpha = \frac{1}{2(L+\lambda)}$.*

From Corollary 3.10, RFRec obtains high communication efficiency, $O(\log \frac{1}{\varepsilon})$, surpassing most FedRS methods with $O(\log m + \log \frac{1}{\varepsilon})$ [11, 18]. Surprisingly, RFRecF obtains a better communication efficiency than RFRec regardless of the value λ. Since $(L+\lambda)^2 \ge 4L\lambda$, RFRecF only needs at most half communication rounds of RFRec to obtain $O(\varepsilon)$-accuracy, theoretically. However, in reality, RFRecF cannot achieve such high efficiency, since σ^2 will not converge to 0 rapidly, which may retard the convergence of the model.

3.6.2 ***Parameter selection.*** We aim to select parameters for the proposed methods to achieve optimal communication efficiency. According to the theoretical results in Corollary 3.10, reducing the value of λ can dwindle the communication rounds of both proposed methods. Nevertheless, λ cannot be too small. Directly, Lemma 3.3 and convergence will not hold if λ is too small. When λ decreases, the influence of penalty term $\psi(x)$ gets slight, and local model $\boldsymbol{V}_{(i)}$ moves slowly towards $\bar{\boldsymbol{V}}$, which obstacle the convergence of $\bar{\boldsymbol{V}}$ to the optimal global model. Therefore, a moderate penalty parameter λ is needed to balance the trade-off between recommendation performance and communication efficiency.

If we know the lipschitz constant L, one intuitive way is selecting $\lambda = L$, then computing $\alpha = \frac{1}{2L}$ for RFRec and $\alpha = \frac{1}{4L}$ for RFRecF.

Table 1: Statistics of the datasets.

Datasets	# Users	# Items	# Interactions
ML-100k	943	1, 682	100, 000
ML-1M	6, 041	3, 706	1, 000, 209
KuaiRec	7, 176	10, 728	12, 530, 806
Jester	73, 421	100	4, 136, 360

This approach balances the importance of local and global loss (i.e., f, ψ), which is beneficial for training stability. However, L is usually unknown in practice. In this case, we need to select λ, α by grid search. Fortunately, we can highly narrow the search range by using the relationship between the optimal choice of λ and α. Supposing we select $\lambda = L$, the optimal choices are $\alpha = \frac{1}{2\lambda}$ for RFRec and $\alpha = \frac{1}{4\lambda}$ for RFRecF so that we can design the search ranges of λ, α, correspondingly. For example, the search range of λ is $\{5, 10, 20, 40\}$, while the range of α is $\{0.1, 0.05, 0.025, 0.0125\}$. In particular, for RFRecF, we let $p = 0.5$ ($p = \frac{\lambda}{L+\lambda}$), which means the equal weights of the main update and moving to average.

4 Experiments

In this section, we aim to answer the following research questions:

- **RQ1**: How does the recommendation performance of proposed RFRec and RFRecF compared to state-of-the-art FedRS models?
- **RQ2**: Does RFRec and RFRecF obtain better communication efficiency compared to FedRS models?
- **RQ3**: How's the contribution of each component?
- **RQ4**: How robust are the proposed methods?
- **RQ5**: How RFRec and RFRecF influenced by parameter?
- **RQ6**: How to balance the privacy protection and performance?

4.1 Experimental Settings

4.1.1 ***Datasets and Evaluation Metrics.*** To evaluate the effectiveness of the proposed RFRec and RFRecF, we conduct experiments on four benchmark datasets: (1) **ML-100k**: It contains 100 thousands of user interactions (i.e., ratings) on movies. (2) **ML-1m**: It contains 1 million user interactions on movies. (3) **KuaiRec**: It contains 12, 530, 806 user interactions (i.e., watch ratio) on video. (4) **Jester**: It contains about 4, 136, 360 user ratings on jokes. To prevent the influence of extreme values on the experiment, we filter out the data with a watch ratio greater than 20 in the KuaiRec dataset, losing less than 1‰ data. The statistics of the datasets are shown in Table 1. We consider *MAE* and *RMSE* as the evaluation metrics, and each is widely used in the recommendations.

4.1.2 ***Baselines.*** To demonstrate the effectiveness of our model, we compare RFRec with two centralized RS methods (PMF, SVD++) and state-of-the-art FedRS methods.

- **PMF** [38]: One of the most popular MF-based recommendation methods, modeling the latent features of users and items.
- **SVD++** [16]: A popular variant of SVD considers user and item biases to enhance recommendation performance.
- **FCF** [1]: The first MF-based federated recommendation method.
- **FedRec** [24]: A federated recommendation method that generates virtual ratings for privacy-preserving.
- **FedFast** [39]: A federated method using clustering and sampling to achieve high communication efficiency.
- **SemiDFEGL** [42]: A method leverages federated ego graph.
- **FedNCF** [41]: An NCF-based method integrating FedAvg [35].

4.1.3 ***Implementation Details.*** Following previous studies [16, 38, 49], we use Gaussian distribution to initialize trainable parameters. We optimize baselines utilizing Adam [13]. From the suggestions of previous works [1, 24, 38], we set the size of latent features as $d = 20$ and the maximum iteration number as $K = 100$. For all baseline models, we set penalty parameters as $\lambda_u = \lambda_v = 0.1$. For our proposed model, we select the learning rate $\alpha = 0.05$ and $\alpha = 0.025$ for RFRec and RFRecF, respectively. In addition, we set the penalty parameter $\lambda = 10$ and threshold $p = 0.5$. Moreover, we provide parameter analysis in Section 4.6. We implement our proposed methods[2] in Python 3.10.11 and Pytorch 2.0.1+cu118.

4.2 Overall Performance Comparison (RQ1)

To demonstrate the effectiveness of our proposed methods, RFRec and RFRecF, for improving recommendation performance in the FL setting, we compare our model with centralized baselines (PMF, SVD++) and federated baselines (FCF, FedRec, FedFast, SemiDFEGL, FedNCF) in four datasets. We report the main results of recommendation performance in Table 2, where we can observe that:

- In general, RFRec and RFRecF perform significantly better than other federated baselines on all datasets, demonstrating the superiority of the proposed methods. We attribute the improvements to that the proposed new formulation (5) of FedRS can well balance the learning of local and global model. In addition, the proposed methods, RFRec and RFRecF, both provide stable training process, which guarantee convergence to the optimal model.
- Significantly, the recommendation performances of the proposed methods are highly close to centralized methods, demonstrating the effectiveness of our methods. We attribute the excellent performance to the fact that our proposed optimization problem (5) can well approximate the original recommendation problem (1) by selecting suitable penalty parameter λ.
- Comparing the recommendation performance of our methods, RFRec outperforms RFRecF in most cases, as the GD manner is more stable than the SGD manner, which reflects a trade-off between recommendation accuracy and efficiency (SGD converges faster).

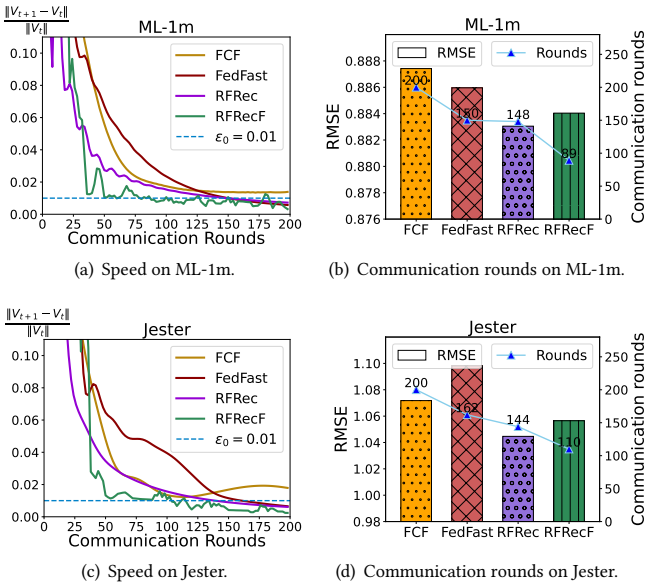
4.3 Communication Rounds (RQ2)

In this section, we evaluate the communication efficiency of the proposed methods. We use FCF and FedFast as baselines to compare the communication costs, as they have a relatively good recommendation performance among all baselines. As the communication medium for the baselines and our methods have the same size (i.e., gradient and item matrix are both $\mathbb{R}^{d\times m}$), we utilize the communication rounds as the metric. The communication will terminate when the model converges. The target of FedRS is to train an optimal item matrix $\boldsymbol{V}$, so we set the stop criterion of communication as $\frac{\|\boldsymbol{V}_{t+1}-\boldsymbol{V}_t\|}{\|\boldsymbol{V}_t\|} \leq \varepsilon_0$ and the maximum communication number as 200. We report the communication results in Figure 2.

[2] https://github.com/Applied-Machine-Learning-Lab/RFRec

Table 2: Overall recommendation performance comparison. All improvements are statistically significant (i.e., two-sided t-test with $p < 0.05$) over federated baselines. In each row, the best result of federated methods is bold.

Datasets	Metrics	Centralized		Federated					Ours	
		PMF	SVD++	FCF	FedRec	FedFast	SemiDFEGL	FedNCF	**RFRec**	**RFRecF**
ML-100k	MAE	0.7341 ± 0.0014	0.7289 ± 0.0011	0.7602 ± 0.0026	0.7549 ± 0.0041	0.7545 ± 0.0040	0.7583 ± 0.0019	0.7562 ± 0.0011	**0.7237 ± 0.0015**	0.7317 ± 0.0064
	RMSE	0.9318 ± 0.0026	0.9273 ± 0.0025	0.9614 ± 0.0028	0.9832 ± 0.0046	0.9403 ± 0.0051	0.9645 ± 0.0025	0.9612 ± 0.0024	**0.9325 ± 0.0023**	0.9385 ± 0.0077
ML-1m	MAE	0.6878 ± 0.0029	0.6924 ± 0.0009	0.7014 ± 0.0018	0.7289 ± 0.0023	0.7056 ± 0.0085	0.7130 ± 0.0038	0.7090 ± 0.0047	0.6937 ± 0.0034	**0.6906 ± 0.0033**
	RMSE	0.8762 ± 0.0031	0.8863 ± 0.0015	0.8874 ± 0.0022	0.9161 ± 0.0025	0.8860 ± 0.0095	0.8937 ± 0.0049	0.8966 ± 0.0050	**0.8831 ± 0.0038**	0.8840 ± 0.0042
KuaiRec	MAE	0.3814 ± 0.0012	0.3760 ± 0.0010	0.4861 ± 0.0031	0.5949 ± 0.0015	0.4678 ± 0.0032	0.4798 ± 0.0027	0.3934 ± 0.0052	**0.3684 ± 0.0017**	0.3781 ± 0.0022
	RMSE	0.7275 ± 0.0015	0.7145 ± 0.0015	0.7569 ± 0.0029	0.7969 ± 0.0030	0.7387 ± 0.0038	0.7511 ± 0.0031	0.7474 ± 0.0056	**0.7163 ± 0.0025**	0.7280 ± 0.0049
Jester	MAE	0.8062 ± 0.0012	0.8006 ± 0.0008	0.8175 ± 0.0021	0.8473 ± 0.0011	0.8321 ± 0.0066	0.8275 ± 0.0029	0.8330 ± 0.0030	0.8118 ± 0.0017	**0.8086 ± 0.0008**
	RMSE	1.0440 ± 0.0032	1.0430 ± 0.0018	1.0718 ± 0.0064	1.0755 ± 0.0012	1.0982 ± 0.0127	1.0726 ± 0.0038	1.0712 ± 0.0028	**1.0447 ± 0.0021**	1.0565 ± 0.0035



(a) Speed on ML-1m. (b) Communication rounds on ML-1m.

(c) Speed on Jester. (d) Communication rounds on Jester.

Figure 2: Communication results.

- The communication efficiency of RFRec and RFRecF consistently surpasses federated baselines on two datasets, demonstrating the effectiveness of the proposed method. Specifically, the communication efficiency of FedFast outperforms FCF since it uses the clustering and sampling approach to speed up the training process. Although RFRec only slightly exceeds FedFast in communication efficiency, it provides much better recommendation performance. In particular, RFRecF significantly leads among all methods regarding communication efficiency. Because RFRec leverages a flexible SGD manner to reduce communication rounds.
- In fact, the number of communication rounds of FCF is larger than the maximum number (200), as it does not achieve the stop criterion. Since the optimal number of communication rounds is $O(\log m + \log \frac{1}{\epsilon})$, which is gigantic when item number m is large.
- Again, the results reflect the trade-off between efficiency and accuracy. While RFRec has a more stable convergence process and better recommendation performance, RFRecF converges faster, providing higher communication efficiency.

4.4 Ablation Study (RQ3)

To test the contributions of two critical parts, we conduct the ablation study at two variants of RFRec and RFRecF over four datasets,

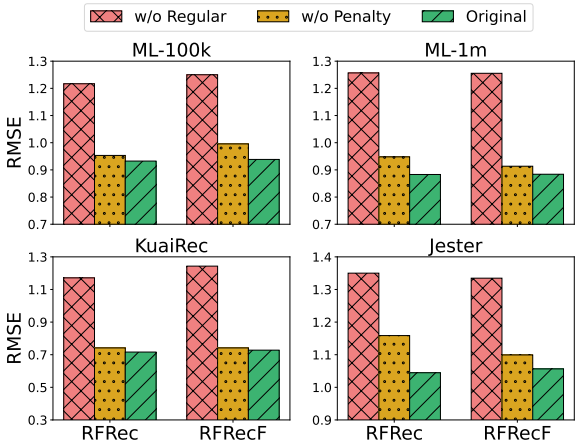


Figure 3: Impact of different components.

Table 3: Robustness of recommendation performance.

Methods	Metrics	0%	20%	50%	80%	90%
RFRec	MAE	0.6937	0.6958	0.7083	0.7090	0.7117
	RMSE	0.8831	0.8870	0.8956	0.8966	0.9001
RFRecF	MAE	0.6906	0.6974	0.7017	0.7038	0.7064
	RMSE	0.8840	0.8891	0.9005	0.9048	0.9057

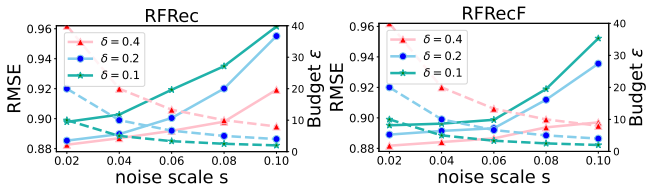
including (1) *w/o Regular*: without the regularization term ψ, and (2) *w/o Penalty*: without the penalty term $\|\boldsymbol{U}\|^2$. We record the RMSE to compare the recommendation performances, as demonstrated in Figure 3. The regularization term is the most critical part, which contributes the most to performance since it guarantees the convergence of each local model $\boldsymbol{V}_{(i)}$ to global model $\bar{\boldsymbol{V}}$. Furthermore, the penalty term can improve performance in all cases by constraining the scale of model parameters to alleviate overfitting.

4.5 Robustness Test (RQ4)

We experiment on ML-1m to demonstrate the robustness of our methods, with a rate of the dropped devices from 0% to 90% to simulate the lousy case (e.g., losing connection). We report the results in Table 3. Our methods are robust enough to provide a good recommendation performance even in extreme cases where the participation rate is meager, from 10% to 50%. The results demonstrate the effectiveness of our proposed methods in real scenarios, such

Table 4: Parameter sensitivity of RMSE on ML-1m.

Methods	Parameter	$\lambda = 5$	$\lambda = 10$	$\lambda = 20$	$\lambda = 40$
RFRec	$\alpha = 0.1000$	0.8912	0.8948	0.8963	0.9036
	$\alpha = 0.0500$	0.8957	**0.8846**	0.8869	0.8887
	$\alpha = 0.0250$	0.9057	0.8928	0.8897	0.8983
	$\alpha = 0.0125$	0.8895	0.8870	0.8891	0.8920
RFRecF	$\alpha = 0.1000$	0.9169	0.9035	0.9066	0.9139
	$\alpha = 0.0500$	0.9254	0.9108	0.8923	0.9070
	$\alpha = 0.0250$	0.9101	**0.8830**	0.8856	0.8908
	$\alpha = 0.0125$	1.0491	0.9588	0.9102	0.8992



(a) Privacy budget and RMSE of RFRec. (b) Privacy budget and RMSE of RFRecF.

Figure 4: Privacy budget and RMSE.

as industrial scenarios, where slight performance degradation due to losing connections of devices is acceptable.

4.6 Hyper-Parameter Analysis (RQ5)

In this section, we tune the important hyperparameters λ, α, p to find the best choice. Specifically, we conduct search on λ and α in range $\{0.1, 0.05, 0.025, 0.0125\}$ and $\{5, 10, 20, 40\}$, according to Section 3.6.2. The experiment results are reported in Table 4, where a general observation is that the performance in the margin of the search range is worse than the center. In particular, the optimal choice of λ is 10 for two methods. While the optimal choices of α are 0.05 and 0.025 for RFRec and RFRecF, respectively. The optimal choice conforms to the theoretical deduction in Section 3.6.2. Then, we tune value p for RFRecF after obtaining the best λ, α. We find the overall trend of RMSE is a U curve, with optimal value $p = 0.5$.

4.7 Privacy Protection Test (RQ6)

To show the trade-off between privacy-preserving and performance mentioned at Corollary 3.7 and determine the best parameters in Equation (9), we experiment on ML-1m dataset by tuning clipping threshold δ and noise scale s. The privacy budget is computed by $\epsilon = \frac{2\delta}{s}$, the smaller the better. We report the main result in Figure 4. Generally, there exists a trade-off between privacy protection and accuracy. For example, when we amplify s, RMSE gets larger, which means worse recommendation performance, and the privacy budget ϵ becomes smaller, meaning better privacy strength. In contrast, δ has a contrary effect. We choose parameter pair $\delta = 0.2, s = 0.04$ for RFRec, and parameter pair $\delta = 0.2, s = 0.06$ for RFRec, providing enough privacy protection and nice performance.

5 Related works

The mainstream of FedRs [1, 23, 24, 39] is based on Matrix factorization (MF), which assumes that the rating is the inner product of user and item feature vectors. To preserve user privacy, FedRS models divide the update into server and client. However, the communication of gradients cannot protect user information strictly. In FedMF [5], it has been proved that the central server can deduce a user's ratings by leveraging two consecutive gradients of this user. Motivated by this severe hidden threat to user privacy, some recent works [5, 29, 32, 37] make efforts to enhance privacy protection through homomorphic encryption, pseudo-item generation, and LDP. Other methods [27, 42, 49, 50] try to improve the capability of FedRS in some special domains. For example, FedGNN [49] integrated the graph neural network (GNN) into the framework of FL to improve the representation learning and simultaneously communicate the perturbed gradient to alleviate privacy leaks. Besides, to equip itself with the ability to handle users' heterogeneous (Non-IID) data, MetaMF [27] leverages the meta-learning technique to allow personalization in local clients. Detecting the incompatibility of federated aggregation methods and NCF model update, FedNCF [41] propose a framework that decomposes the aggregation in the MF step and model averaging step.

The mainstream of FedRS methods cannot guarantee stable convergence since solving the non-convex problem and leveraging the alternating update approach. Our methods guarantee the convergence to the optimal and provide high communication efficiency.

6 Conclusion

In this paper, we reformulate the optimization problem of FedRS as an RERM problem to adapt to the FL setting. To effectively and efficiently solve the proposed optimization problem, we offer two methods, RFRec and RFRecF, leveraging local GD and non-uniform SGD manners, respectively, to learn the optimal model parameters. We provide the theoretical analysis of convergence and communication, showcasing our proposed methods' stabilization and high communication efficiency. In addition, we demonstrate the privacy protection mechanism of the proposed methods. Extensive experiments demonstrate the superiority and effectiveness of our proposed methods over other state-of-the-art FedRS methods.

A Appendix

Proof. (Lemma 3.2) It is trivial to proof $\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x-y\|$. Then, we can use the property that L-Lipschitz continuity of gradients is equivalent to L-smoothness to prove the smoothness. □

Proof. (Lemma 3.3) Recall that $F_i(x) = \|R_i - u^T V\|^2 + \lambda_u \|u\|^2 + \frac{\lambda}{2}\|V_{(i)} - \bar{V}\|^2$, we can derive the first order derivative as

$$\nabla F_i = \begin{bmatrix} -2V(R_i - u^T V)^T \\ -2u(R_i - u^T V) \end{bmatrix} + \begin{bmatrix} 2\lambda_u u \\ (1 - \frac{1}{n})\lambda V \end{bmatrix}, \tag{12}$$

Then we have

$$\nabla^2 F_i = 2\begin{bmatrix} VV^T + \lambda_u I_d & 2V \otimes u^T - R_i \otimes I_d \\ 2V^T \otimes u - R_i^T \otimes I_d & I_m \otimes uu^T + \frac{1}{2}(1 - \frac{1}{n})\lambda I_{md} \end{bmatrix}. \tag{13}$$

The Hessian matrix can be decomposed as follows:

$$\begin{aligned} \nabla^2 F_i &= 2\begin{bmatrix} VV^T & 2V \otimes u^T \\ 2V^T \otimes u & I_m \otimes uu^T \end{bmatrix} + 2\begin{bmatrix} \lambda_u I_d & -R_i \otimes I_d \\ -R_i^T \otimes I_d & \frac{1}{2}(1 - \frac{1}{n})\lambda I_{md} \end{bmatrix} \\ &= 2A^T A + \frac{2}{\lambda_u} B^T B + \frac{2}{\lambda_u}\begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{2}(1 - \frac{1}{n})\lambda_u \lambda I_{md} - C \end{bmatrix}, \end{aligned} \tag{14}$$

where $A = \begin{bmatrix} V^T & 2I_m \otimes u^T \end{bmatrix}, B = \begin{bmatrix} \lambda_u I_d & -R_i \otimes I_d \end{bmatrix}, C = R_i^T R_i \otimes I_d + 3\lambda_u I_m \otimes uu^T$. Notice that C is the sum of two symmetric matrices. According to Weyl's inequality and the eigenvalues of Kronecker product are the pair-wise products of eigenvalues of matrices, we have $\lambda_{\max}(C) \leq \lambda_{\max}(R_i^T R_i) + 3\lambda_u \lambda_{\max}(uu^T)$. Since $\text{rank}(R_i^T R_i) = 1$ and $\text{tr}(R_i^T R_i) = \text{tr}(R_i R_i^T) = \|R_i\|^2$, we have $\lambda_{\max}(R_i^T R_i) = \|R_i\|^2 \leq M_r^2$. Similarly, we have $\lambda_{\max}(uu^T) = \|u\|^2 \leq M_u^2$. By letting $\lambda > \frac{2}{\lambda_u} M_r^2 + 6M_u^2$, we have F_i is μ-strongly convex ($M^T M$ is positive definite). Thus, F is μ-strongly convex.

Then, we can leverage the smoothness of f_i and analyze the Hessian of ψ to prove the smoothness. Observe that

$$\nabla^2 f(x) = \text{diag}(\nabla^2 f_1(x_1), \nabla^2 f_2(x_2), \cdots, \nabla^2 f_n(x_n)).$$

Then by Lemma 3.2, we have $\nabla^2 f_i(x_i) \preceq L I_{(m+1)d}$, which implies $\nabla^2 f(x) \preceq L I_{(m+1)nd}$ and thereby f is L-smooth.

For function ψ, we have $\nabla_V \psi(x) = (V_{(1)} - \bar{V}, \cdots, V_{(n)} - \bar{V})^{\mathrm{T}}$. By denoting e as the all one vector in $\mathbb{R}^n$, we have

$$\nabla_V^2 \psi(x) = (I_n - \frac{1}{n} ee^T) \otimes I_{md},$$

where $I_n - \frac{1}{n} ee^T$ is a circulant matrix with maximum eigenvalue 1. Hence $\nabla^2 \psi(x) \preceq I_{(m+1)nd}$, and ψ is 1-smooth. It follows that F is μ-strongly convex and L_F-smooth with $L_F = L + \lambda$. □

Proof. (Theorem 3.4) Denote $x^* = x(\lambda)$ and the residual $r^k = x^k - x^*$. By the update rule in Algorithm 1, we have $r^{k+1} = r^k - \alpha \nabla F(x^k)$. Taking inner products for both sides, we obtain

$$\begin{aligned} \|r^{k+1}\|^2 &= \|r^k\|^2 - 2\alpha \langle r^k, \nabla F(x^k) \rangle + \alpha^2 \|\nabla F(x^k)\|^2 \\ &\leq (1 - \alpha\mu)\|r^k\|^2 - 2\alpha(F(x^k) - F(x^*)) + \alpha^2 \|\nabla F(x^k)\|^2 \\ &\leq (1 - \alpha\mu)\|r^k\|^2 + 2\alpha(\alpha L - 1)(F(x^k) - F(x^*)) \\ &\leq (1 - \alpha\mu)\|r^k\|^2 \quad (0 \leq \alpha \leq \frac{1}{L + \lambda}), \end{aligned}$$

where we use μ-strongly convexity and L-smoothness of F. □

Before the main proof of Theorem 3.5, we give the result of expected smoothness of G as follows.

Lemma A.1. *(Expected Smoothness) For every $x \in \mathbb{R}^d$, the stochastic gradient G of F satisfies*

$$\begin{aligned} \mathbb{E}[\|G(x) - G(x(\lambda))\|^2] &\leq 2\mathcal{L}(F(x) - F(x(\lambda))), \\ \mathbb{E}[\|G(x)\|^2] &\leq 4\mathcal{L}(F(x) - F(x(\lambda))) + 2\sigma^2. \end{aligned}$$

Proof. By the definition of the stochastic gradient G, we have

$$\mathbb{E}[\|G(x) - G(x(\lambda))\|^2] \leq \frac{2L_f}{1-p} D_f(x, x(\lambda)) + \frac{2\lambda^2 L_\psi}{p} D_\psi(x, x(\lambda))$$

where D_f, D_ψ are bregman distances. Since $D_f + \lambda D_\psi = D_F$ and $\nabla F(x(\lambda)) = 0$, we can continue as

$$\begin{aligned} \mathbb{E}[\|G(x) - G(x(\lambda))\|^2] &\leq 2 \max\{\frac{L}{1-p}, \frac{\lambda}{p}\} D_F(x, x(\lambda)) \\ &= 2\mathcal{L}(F(x) - F(x(\lambda))). \end{aligned}$$

This proves the first bound. For the second estimate, we have

$$\begin{aligned} \mathbb{E}[\|G(x)\|^2] &\leq 2\mathbb{E}[\|G(x) - G(x(\lambda))\|^2] + 2\mathbb{E}[\|G(x(\lambda))\|^2] \\ &\leq 4\mathcal{L}(F(x) - F(x(\lambda))) + 2\sigma^2, \end{aligned}$$

where it is trivial to show $\sigma^2 = \mathbb{E}[\|G(x(\lambda))\|^2]$. □

Proof. (Theorem 3.5) According to Algorithm 2, we have $r^{k+1} = r^k - \alpha G(x^k)$. Then similar to the proof of RFRec we have

$$\|r^{k+1}\|^2 = \|r^k\|^2 - 2\alpha \langle r^k, G(x^k) \rangle + \alpha^2 \|G(x^k)\|^2$$

By taking expectation on the both sides, we obtain

$$\begin{aligned} \mathbb{E}[\|r^{k+1}\|^2] &= \|r^k\|^2 - 2\alpha \langle r^k, \nabla F(x^k) \rangle + \alpha^2 \mathbb{E}[\|G(x^k)\|^2] \\ &\leq (1 - \alpha\mu)\|r^k\|^2 - 2\alpha(F(x^k) - F(x^*)) + \alpha^2 \mathbb{E}[\|G(x^k)\|^2] \\ &\leq (1 - \alpha\mu)\|r^k\|^2 + 2\alpha(2\alpha\mathcal{L} - 1)(F(x^k) - F(x^*)) + 2\alpha^2\sigma^2 \\ &\leq (1 - \alpha\mu)\|r^k\|^2 + 2\alpha^2\sigma^2 \quad (0 \leq \alpha \leq \frac{1}{2\mathcal{L}}), \end{aligned}$$

where the second inequality applies the result in Lemma A.1. Then, we apply this bound recursively to obtain final result. □

Proof. (Corollary 3.7) Denote the perturbation of each client i as δ_i. The perturbation will only influence the averaging step, formulated as $\bar{V}_\delta = \frac{1}{n} \sum_{i=1}^n (V_{(i)} + \delta_i) = \bar{V} + \bar{\delta}$, where $\bar{\delta} = \frac{1}{n} \sum_{i=1}^n \delta_i$ with mean 0 and variance $2s^2/n$. We know

$$\nabla \psi_\delta(x) = (V_{(1)} - \bar{V} - \bar{\delta}, \cdots, V_{(n)} - \bar{V} - \bar{\delta})^{\mathrm{T}}$$

We can calculate the momentums of $\nabla \psi_\delta(x)$:

$$\begin{aligned} \mathbb{E}[\nabla \psi_\delta(x)] &= (V_{(1)} - \bar{V}, \cdots, V_{(n)} - \bar{V})^{\mathrm{T}} = \nabla \psi(x), \\ \mathbb{E}[\|\nabla \psi_\delta(x)\|^2] &= \|\nabla \psi(x)\|^2 + n\mathbb{E}[\bar{\delta}^2] = \|\nabla \psi(x)\|^2 + 2s^2. \end{aligned}$$

Since $f(x)$ is not perturbed, we have

$$\mathbb{E}[\|\nabla F_\delta(x)\|^2] = \mathbb{E}[\|\nabla f(x) + \lambda \nabla \psi_\delta(x)\|^2] = \|\nabla F(x)\|^2 + 2\lambda^2 s^2$$

Then we have

$$\begin{aligned} \mathbb{E}[\|r^{k+1}\|]^2 &= \|r^k\|^2 - 2\alpha \langle r^k, \nabla F(x^k) \rangle + \alpha^2 \|\nabla F(x^k)\|^2 + 2\lambda^2 s^2 \\ &\leq (1 - \alpha\mu)\|r^k\|^2 + 2\alpha^2\lambda^2 s^2 \quad (0 \leq \alpha \leq \frac{1}{L + \lambda}), \end{aligned}$$

and we can use it recursively to obtain desired bound.

$$\mathbb{E}[\|r^k\|^2] \leq (1 - \alpha\mu)^k \|r^0\|^2 + \frac{2\alpha\lambda^2 s^2}{\mu}.$$

Similarly, we can obtain the convergence result of RFRecF. □

Acknowledgments

This research was partially supported by Research Impact Fund (No.R1015-23), APRC - CityU New Research Initiatives (No.9610565, Start-up Grant for New Faculty of CityU), CityU - HKIDS Early Career Research Grant (No.9360163), Hong Kong ITC Innovation and Technology Fund Midstream Research Programme for Universities Project (No.ITS/034/22MS), Hong Kong Environmental and Conservation Fund (No. 88/2022), and SIRG - CityU Strategic Interdisciplinary Research Grant (No.7020046), Huawei (Huawei Innovation Research Program), Tencent (CCF-Tencent Open Fund, Tencent Rhino-Bird Focused Research Program), Ant Group (CCF-Ant Research Fund, Ant Group Research Fund), Alibaba (CCF-Alimama Tech Kangaroo Fund (No. 2024002)), CCF-BaiChuan-Ebtech Foundation Model Fund, Kuaishou, and Key Laboratory of Smart Education of Guangdong Higher Education Institutes, Jinan University (2022LSYS003).

References

[1] Muhammad Ammad-Ud-Din, Elena Ivannikova, Suleiman A Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and Adrian Flanagan. 2019. Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888* (2019).

[2] Pathum Chamikara Mahawaga Arachchige, Peter Bertok, Ibrahim Khalil, Dongxi Liu, Seyit Camtepe, and Mohammed Atiquzzaman. 2019. Local differential privacy for deep learning. *IEEE Internet of Things Journal* (2019), 5827–5842.

[3] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proc. of COMPSTAT*. 177–186.

[4] Oscar Celma. 2010. Music recommendation. In *Music recommendation and discovery: The long tail, long fail, and long play in the digital music space*. 43–85.

[5] Di Chai, Leye Wang, Kai Chen, and Qiang Yang. 2020. Secure federated matrix factorization. *IEEE Intelligent Systems* (2020), 11–20.

[6] Woo-Seok Choi, Matthew Tomei, Jose Rodrigo Sanchez Vicarte, Pavan Kumar Hanumolu, and Rakesh Kumar. 2018. Guaranteeing local differential privacy on ultra-low-power systems. In *Proc. of ISCA*. 561–574.

[7] Koustabh Dolui, Illapha Cuba Gyllensten, Dietwig Lowet, Sam Michiels, Hans Hallez, and Danny Hughes. 2019. Towards privacy-preserving mobile applications with federated learning: The case of matrix factorization (poster). In *Proc. of MobiSys*. 624–625.

[8] Justin P Haldar and Diego Hernando. 2009. Rank-constrained solutions to linear matrix equations using powerfactorization. *IEEE Signal Processing Letters* (2009), 584–587.

[9] Filip Hanzely and Peter Richtárik. 2020. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516* (2020).

[10] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proc. of ICDM*. 263–272.

[11] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. 2013. Low-rank matrix completion using alternating minimization. In *Proc. of STOC*. 665–674.

[12] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2010. *Recommender systems: an introduction*. Cambridge University Press.

[13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[14] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. 2016. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527* (2016).

[15] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).

[16] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc. of KDD*. 426–434.

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* (2009), 30–37.

[18] Kiryung Lee and Dominik Stöger. 2023. Randomly initialized alternating least squares: Fast convergence for matrix sensing. *SIAM Journal on Mathematics of Data Science* (2023), 774–799.

[19] Muyang Li, Zijian Zhang, Xiangyu Zhao, Wanyu Wang, Minghao Zhao, Runze Wu, and Ruocheng Guo. 2023. Automlp: Automated mlp for sequential recommendations. In *Proc. of WWW*. 1190–1198.

[20] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine* (2020), 50–60.

[21] Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. 2023. E4srec: An elegant effective efficient extensible solution of large language models for sequential recommendation. *arXiv preprint arXiv:2312.02443* (2023).

[22] Xinhang Li, Zhaopeng Qiu, Xiangyu Zhao, Zihao Wang, Yong Zhang, Chunxiao Xing, and Xian Wu. 2022. Gromov-wasserstein guided representation learning for cross-domain recommendation. In *Proc. of CIKM*. 1199–1208.

[23] Feng Liang, Weike Pan, and Zhong Ming. 2021. Fedrec++: Lossless federated recommendation with explicit feedback. In *Proc. of AAAI*. 4224–4231.

[24] Guanyu Lin, Feng Liang, Weike Pan, and Zhong Ming. 2020. Fedrec: Federated recommendation with explicit feedback. *IEEE Intelligent Systems* (2020), 21–30.

[25] Weilin Lin, Xiangyu Zhao, Yejing Wang, Tong Xu, and Xian Wu. 2022. AdaFS: Adaptive feature selection in deep recommender system. In *Proc. of KDD*. 3309–3317.

[26] Weilin Lin, Xiangyu Zhao, Yejing Wang, Yuanshao Zhu, and Wanyu Wang. 2023. Autodenoise: Automatic data instance denoising for recommendations. In *Proc. of WWW*. 1003–1011.

[27] Yujie Lin, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Dongxiao Yu, Jun Ma, Maarten de Rijke, and Xiuzhen Cheng. 2020. Meta matrix factorization for federated rating predictions. In *Proc. of SIGIR*. 981–990.

[28] Langming Liu, Liu Cai, Chi Zhang, Xiangyu Zhao, Jingtong Gao, Wanyu Wang, Yifu Lv, Wenqi Fan, Yiqi Wang, Ming He, et al. 2023. Linrec: Linear attention mechanism for long-term sequential recommender systems. In *Proc. of SIGIR*. 289–299.

[29] Ruixuan Liu, Yang Cao, Yanlin Wang, Lingjuan Lyu, Yun Chen, and Hong Chen. 2023. PrivateRec: Differentially Private Model Training and Online Serving for Federated News Recommendation. In *Proc. of KDD*. 4539–4548.

[30] Shuchang Liu, Qingpeng Cai, Bowen Sun, Yuhao Wang, Ji Jiang, Dong Zheng, Peng Jiang, Kun Gai, Xiangyu Zhao, and Yongfeng Zhang. 2023. Exploration and regularization of the latent action space in recommendation. In *Proc. of WWW*. 833–844.

[31] Ziru Liu, Jiejie Tian, Qingpeng Cai, Xiangyu Zhao, Jingtong Gao, Shuchang Liu, Dayou Chen, Tonghao He, Dong Zheng, Peng Jiang, et al. 2023. Multi-task recommendations with reinforcement learning. In *Proc. of WWW*. 1273–1282.

[32] Zhiwei Liu, Liangwei Yang, Ziwei Fan, Hao Peng, and Philip S Yu. 2022. Federated social recommendation with graph neural network. *ACM Transactions on Intelligent Systems and Technology (TIST)* (2022), 1–24.

[33] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. 2015. Recommender system application developments: a survey. *Decision support systems* (2015), 12–32.

[34] Ulysse Marteau-Ferey, Dmitrii Ostrovskii, Francis Bach, and Alessandro Rudi. 2019. Beyond least-squares: Fast rates for regularized empirical risk minimization through self-concordance. In *Proc. of COLT*. 2294–2340.

[35] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proc. of AISTATS*. 1273–1282.

[36] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. 2016. Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629* (2016), 2.

[37] Lorenzo Minto, Moritz Haller, Benjamin Livshits, and Hamed Haddadi. 2021. Stronger privacy for federated collaborative filtering with implicit feedback. In *Proc. of RecSys*. 342–350.

[38] Andriy Mnih and Russ R Salakhutdinov. 2007. Probabilistic matrix factorization. *Proc. of NeurIPS* (2007).

[39] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragos, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. 2020. Fedfast: Going beyond average for faster training of federated recommender systems. In *Proc. of KDD*. 1234–1242.

[40] Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G Azzolini, et al. 2019. Deep learning recommendation model for personalization and recommendation systems. *arXiv preprint arXiv:1906.00091* (2019).

[41] Vasileios Perifanis and Pavlos S Efraimidis. 2022. Federated neural collaborative filtering. *Knowledge-Based Systems* (2022), 108441.

[42] Liang Qu, Ningzhi Tang, Ruiqi Zheng, Quoc Viet Hung Nguyen, Zi Huang, Yuhui Shi, and Hongzhi Yin. 2023. Semi-decentralized federated ego graph learning for recommendation. In *Proc. of WWW*. 339–348.

[43] General Data Protection Regulation. 2018. General data protection regulation (GDPR). *Intersoft Consulting, Accessed in October* (2018).

[44] Paul Resnick and Hal R Varian. 1997. Recommender systems. *Commun. ACM* (1997), 56–58.

[45] J Ben Schafer, Joseph Konstan, and John Riedl. 1999. Recommender systems in e-commerce. In *Proc. of EC*. 158–166.

[46] Zehua Sun, Yonghui Xu, Yong Liu, Wei He, Lanju Kong, Fangzhao Wu, Yali Jiang, and Lizhen Cui. 2022. A survey on federated recommendation systems. *arXiv preprint arXiv:2301.00767* (2022).

[47] Gábor Takács and Domonkos Tikk. 2012. Alternating least squares for personalized ranking. In *Proc. of RecSys*. 83–90.

[48] Yuhao Wang, Ha Tsz Lam, Yi Wong, Ziru Liu, Xiangyu Zhao, Yichao Wang, Bo Chen, Huifeng Guo, and Ruiming Tang. 2023. Multi-task deep recommender systems: A survey. *arXiv preprint arXiv:2302.03525* (2023).

[49] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. 2021. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925* (2021).

[50] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Tao Qi, Yongfeng Huang, and Xing Xie. 2022. A federated graph neural network framework for privacy-preserving personalization. *Nature Communications* (2022), 3091.

[51] Liu Yang, Ben Tan, Vincent W Zheng, Kai Chen, and Qiang Yang. 2020. Federated recommendation systems. *Federated Learning: Privacy and Incentive* (2020), 225–239.

[52] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* (2019), 1–19.

[53] Chi Zhang, Rui Chen, Xiangyu Zhao, Qilong Han, and Li Li. 2023. Denoising and prompt-tuning for multi-behavior recommendation. In *Proc. of WWW*. 1355–1363.

[54] Yuchen Zhang and Lin Xiao. 2017. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *Journal of Machine Learning Research* (2017), 1–42.