A Survey on Federated Recommendation Systems

Zehua Sun*, Yonghui Xu*, Yong Liu, Wei He, Yali Jiang, Fangzhao Wu, Lizhen Cui†

***Abstract*—Federated learning has recently been applied to recommendation systems to protect user privacy. In federated learning settings, recommendation systems can train recommendation models only collecting the intermediate parameters instead of the real user data, which greatly enhances the user privacy. Beside, federated recommendation systems enable to collaborate with other data platforms to improve recommended model performance while meeting the regulation and privacy constraints. However, federated recommendation systems faces many new challenges such as privacy, security, heterogeneity and communication costs. While significant research has been conducted in these areas, gaps in the surveying literature still exist. In this survey, we—(1) summarize some common privacy mechanisms used in federated recommendation systems and discuss the advantages and limitations of each mechanism; (2) review some robust aggregation strategies and several novel attacks against security; (3) summarize some approaches to address heterogeneity and communication costs problems; (4)introduce some open source platforms that can be used to build federated recommendation systems; (5) present some prospective research directions in the future. This survey can guide researchers and practitioners understand the research progress in these areas.**

***Index Terms*—Recommendation Systems, Federated Learning, Privacy, Security, Heterogeneity, Communication costs.**

I. Introduction

In recent years, recommendation systems have been widely used to model user interests so as to solve information overload problems in many real-world fields, e.g., e-commerce [1] [2], news [3] [4] and healthcare [5] [6]. To further improve the recommendation performance, such systems usually collect as much data as possible, including a lot of private information of users, such as user attributes, user behaviors, social relations, and context information.

Although these recommendation systems have achieved remarkable results in terms of accuracy, most of them require a central server to store collected user data, which exist potential privacy leakage risks because user data could be sold to a third party without user consent, or stolen by motivated attackers. In addition, due to privacy concerns and regulation restrictions, it becomes more difficult to integrate data from other platforms to improve recommendation performance. For example, regulations such as General Data Protection Regulation (GDPR) [7] set strict rules on collecting user data and sharing data between different platforms, which may lead to insufficient data for recommendation systems and further affects the recommendation performance.

Zehua Sun, Yonghui Xu, Wei He, Yali Jiang and Lizhen Cui are with Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR) & Software School, Shandong University.

Yong Liu are with Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University, Singapore.

Fangzhao Wu are with Microsoft Research Asia, China.

*Zehua Sun and Yonghui Xu are Co-First authors.

†Corresponding author: clz@sdu.edu.cn.

Federated learning is a privacy-preserving distributed learning scheme proposed by Google [8], which enables participants to collaboratively train a machine learning model by sharing intermediate parameters (e.g., model parameters, gradients) to the central server instead of their real data. Therefore, combining federated learning with recommendation systems becomes a promising solution for privacy-preserving recommendation systems. In this paper, we term it federated recommendation system (FedRS).

A. Challenges

While FedRS avoid direct exposure of real user data and provides a privacy-aware paradigm for model training, there are still some core challenges that need to be addressed.

Challenge 1: Privacy concerns. Privacy protection is often the major goal of FedRS. In FedRS, each participant jointly trains a global recommendation model by sharing intermediate parameters instead of their real data, which makes an important step towards privacy-preserving recommendation systems. However, a curious sever can still infer some sensitive information (e.g., user behavior, ratings) from the intermediate parameters [9] [10].

Challenge 2: Security attacks. In FedRS, participants may be malicious. They can attack the security of FedRS by poisoning the local training samples or the intermediate parameters uploaded. As a result, attackers can increase/decrease the exposure ratio of specific items [12] or degrade the overall performance of the recommendation model [11]. In addition, some attackers try to use well-designed constraints to approximate the patterns of benign participants, which further increases the difficulty of defense and detection [55].

Challenge 3: Heterogeneity. FedRS also faces the problem of system, statistical and privacy heterogeneity. System heterogeneity means that the storage, computing, and communication capabilities of clients usually vary greatly, clients with limited capability may become stragglers and affect training efficiency. Statistical heterogeneity means that data in different clients is often not independent and identically distributed (Non-IID), which significantly affects global model convergence and personalization of recommendation results. Privacy heterogeneity means that users and information usually have different privacy constraints, thus using the same privacy budgets for them will bring unnecessary loss of accuracy and efficiency.

Challenge 4: Communication costs. To achieve satisfactory recommendation performance, clients need to communicate with the central server for multiple rounds. However, the real-world recommendation systems are usually built on complex deep learning models and millions of intermediate parameters needs to be communicated [13]. Therefore, clients

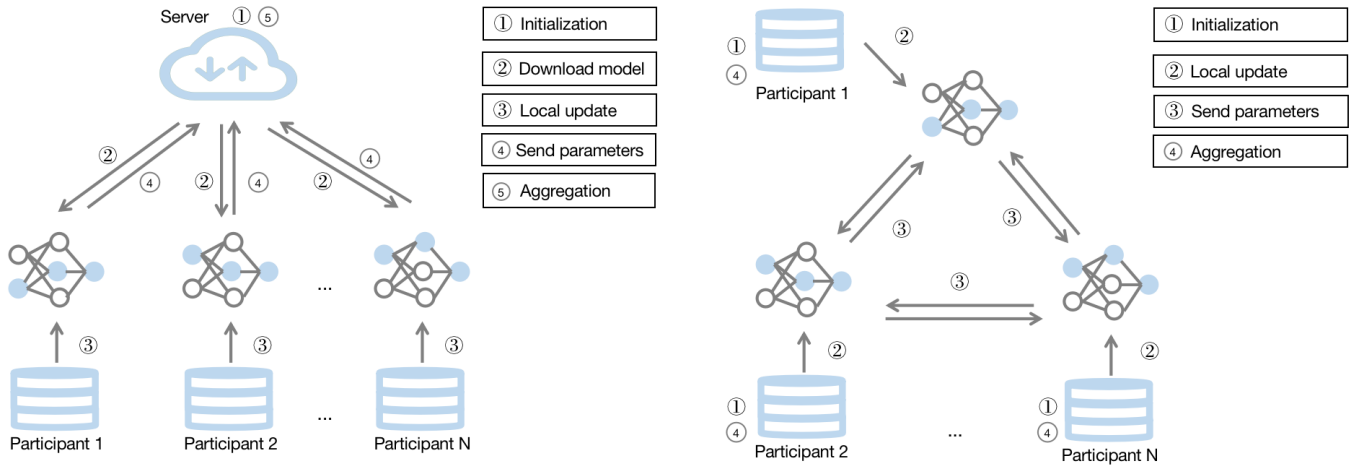


Fig. 1: Communication architecture of FedRS.

may be hard to afford severe communication costs, which limits the application of FedRS in large-scale industrial scenarios.

B. Related Surveys

There are many surveys that have focused on recommendation systems or federated learning. For example, Adomavicius *et al*. [14] provide a detailed categorization of recommendation methods and introduce various limitations of each method. Yang *et al*. [15] give the definition of federated learning and discuss its architectures and applications. And Li *et al*. [16] summarize the unique characteristics and challenges of federated learning. However the existing surveys usually treat recommendation systems and federated learning separately, and few work surveyed specific problems in FedRS [17]. Yang *et al*. [17] categorize FedRS from the aspect of the federated learning and discuss the algorithm-level and system-level challenges for FedRS. However, they do not provide comprehensive methods to address privacy, security, heterogeneity, and communication costs challenges.

C. Our Contribution

Compared with the previous surveys, this paper makes the following contributions: (1) We provide a comprehensive overview of FedRS from the perspectives of definition, communication architectures and categorization. (2) We summarize the state-of-the-art studies of FedRS in terms of privacy, security, heterogeneity and communication costs areas. (3) We introduce some open source platforms for FedRS, which can help engineers and researchers develop algorithms and deploy applications of FedRS. (4) We discuss the promising future directions for FedRS.

The rest of the paper is organized as follow: Section II discusses the overview of FedRS. Section III-Section VI summarize the state-of-the-art studies of FedRS from the aspects of privacy, security, heterogeneity and communication costs. Section VII introduces the existing open source platforms. Section VIII presents some prospective research directions. Finally, Section IX concludes this survey.

II. Overview of Federated Recommendation Systems

A. Definition

FedRS is a technology that provides recommendation services in a privacy preserving way. To protect user privacy, the participants in FedRS collaboratively train the recommendation model by exchanging intermediate parameters instead of sharing their own real data. In ideal case, the performance of recommendation model trained in FedRS should be closed to the performance of the recommendation model trained in the data centralized setting, which can be formalized as:

$$|V_{FED} - V_{SUM}| < \delta. \tag{1}$$

where V_{FED} is the recommendation model performance in FedRS , V_{SUM} is the recommendation model performance in traditional recommendation systems for centralized data storage, and δ is a small positive numbers.

B. Communication Architecture

In FedRS, the data of participants is stored locally, and the intermediate parameters are communicated between the server and participants. There are two major communication architectures used in the study of FedRS, including client-server architecture and peer-peer architecture.

Client-Server Architecture. Client-server architecture is the most common communication architecture used in FedRS, as shown in Fig. 1(a), which relies on a trusted central server to perform initialization and model aggregation tasks. In each round, the server distributes the current global recommendation model to some selected clients. Then the selected clients use the received model and their own data for local training, and send the updated intermediate parameters (e.g., model parameters, gradients) to the server for global aggregation. The

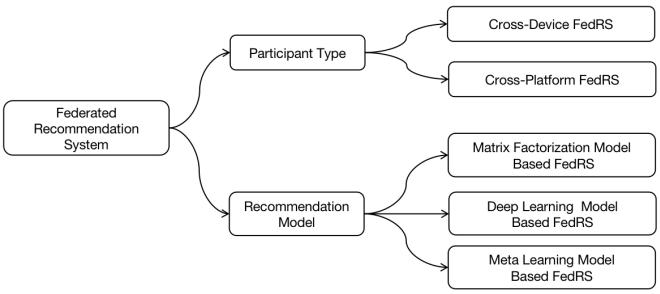


Fig. 2: Categorization of federated recommendation systems.

client-server architecture requires a central server to aggregate the intermediate parameters uploaded by the clients. Thus, once the server has a single point of failure, the entire training process will be seriously affected [18]. In addition, the curious server may infer the clients' privacy information through the intermediate parameters, leaving potential privacy concerns [10].

Peer-Peer Architecture. Considering the single point of failure problem for client-server architecture in FedRS, Hegeds $et\ al.$ [19] design a peer-peer communication architecture with no central server involved in the communication process, which is shown in Fig. 1(b). During each communication round, each participant broadcasts the updated intermediate parameters to some random online neighbors in the peer to peer network, and aggregates received parameters into its own global model. In this architecture, the single point of failure and privacy issues associated with a central server can be avoided. However, the aggregation process occurs on each client, which greatly increases the communication and computation overhead for clients [20].

C. Categorization

In FedRS, the participants are responsible for the local training process as the data owners. They can be different mobile devices or data platforms. Considering the unique properties for different types of participants, FedRS usually have different application scenarios and designs. Besides, there are also some differences between different recommendation models in the federation process. Thus, we summarize the current FedRS and categorize them from the perspectives of participant type and recommendation model. Fig. 2 shows the summary of the categorization of FedRS.

1) Participant Type: Based on the type of participants, FedRS can be categorized into cross-device FedRS and cross-platform FedRS.

Cross-device FedRS. In cross-device FedRS, different mobile devices are usually treated as participants [21] [22]. The typical application of cross-device FedRS is to build a personal recommendation model for users without collecting their local data. In this way, users can enjoy recommend service while protecting their private information. The number of participants in cross-device FedRS is relative large and each participant keeps a small amount of data. Considering the limited computation and communication abilities for mobile devices, cross-device FedRS cannot handle very complex training tasks. Besides, due to the power and the network status, the mobile devices may drop out of the training process. Thus, the major challenges for cross-device FedRS are how to improve the efficiency and deal with straggler problem of devices during training process.

Cross-platform FedRS. In cross-platform FedRS, different data platforms are usually treated as participants who want to collaborate to improve recommendation performance while meeting regulation and privacy constraints [23] [24] [25]. For example, In order to improve the recommendation performance, recommendation systems often integrate data from multiple platforms (e.g., e-commercial platforms , social platforms). However, due to the privacy and regulation concerns, the different data platforms are often unable to directly share their data with each other. In this scenario, cross-platform FedRS can be used to collaboratively train recommendation models between different data platforms without directly exchanging their users' data. Compared to cross-device FedRS, the number of participants in cross-platform FedRS is relatively small, and each participant owns relative large amount of data. An important challenge for cross-platform FedRS is how to design a fair incentive mechanism to measure contributions and benefits of different data platforms. Besides, it is hard to find a trusted server to manage training process in cross-platform FedRS, so a peer to peer communication architecture can be a good choice in this case.

2) Recommendation Model: According to the different recommendation models used in FedRS, FedRS can be categorized into matrix factorization model based FedRS, deep learning model based FedRS and meta learning model FedRS.

Matrix factorization model based FedRS. Matrix factorization [26] is the most common model used in FedRS, which formulates the user-item interaction or rating matrix $R \in \mathbb{R}^{N\times M}$ as a linear combination of user profile matrix $U \in \mathbb{R}^{N\times K}$ and item profile matrix $V \in \mathbb{R}^{M\times K}$:

$$R = UV^T. \tag{2}$$

then uses the learned model to recommendation new items to the user according to the predicted value. In matrix factorization model based FedRS, the user factor vectors are stored and updated locally on the clients, and only the item factor vectors [27] or the gradients of item factor vectors [21] [22] [10] [28] [29] are uploaded to the server for aggregation. Matrix factorization model based FedRS can simply and effectively capture user tastes with the interaction and rating information between users and items. However it still has many limitations such as sparsity (the number of ratings to be predicted is much smaller than the known ratings) and cold-start (new users and new items lacks ratings information) problems [14].

Deep learning model based FedRS. To learn more complex representations of users and items and improve the recommendation performance, deep learning technology has been widely used in recommendation systems. However, as privacy regulations get stricter, it becomes more difficult for recommendation systems to collect enough user data to build a high performance deep learning model. To make the full

use of user data while meeting privacy regulations, many effective deep learning model based FedRS have been proposed [30] [31] [32]. Considering different model structures, deep learning model based FedRS usually adopt different model update and intermediate parameter transmit processes. For examples, Perifanis $et\ al.$ [30] propose a federated neural collaborative filtering (FedNCF) framework based on NCF [33]. In FedNCF, the clients locally update the network weights as well as the user and item profiles, then upload the item profile and network weights after masking to the server for aggregation. Wu $et\ al.$ [31] propose a federated graph neural network (FedGNN) framework based on GNN. In FedGNN, the clients locally train GNN models and update the user/item embeddings from their local sub-graph, then send the perturbed gradients of GNN model and item embedding to the central server for aggregation. Besides, Huang $et\ al.$ [34] propose a federated multi-view recommendation framework based on Deep Structured Semantic Model (DSSM [35]). In FL-MV-DSSM, each view i locally trains the user and item sub-models based on their own user data and local shared item data, then send the perturbed gradients of both user and item sub-models to server for aggregation. Although deep learning model based FedRS achieve outstanding performance in terms of accuracy, the massive model parameters of deep learning models bring huge computation and communication overhead to the clients, which presents a serious challenge for real industrial recommendation scenarios.

Meta learning model based FedRS. The most of existing federated recommendation studies are built on the assumption that data distributed on each client is independent and identically (IID). However, learning a unified federated recommendation model often performs poorly when handling the Non-IID and highly personalized data on clients. Meta learning model can quickly adapt to new tasks while maintaining good generalization ability [36], which makes it particularly suitable for FedRS. In meta learning model based FedRS, the server aggregates the intermediate parameters uploaded by clients to learn a model parameter initialization, and the clients fine-tune the initialed model parameters in local training phase to fit to their local data [37] [38]. In this way, meta learning model based FedRS can adapt the clients' local data to provide more personalized recommendations. Although the performance of meta learning model based FedRS are generally better than learning a unified global model, the private information leakage can still occur during the learning process of model parameter initialization [37].

III. Privacy of Federated recommendation Systems

In the model training process of FedRS, the user data is stored locally and only the intermediate parameters are uploaded to a server, which can further protect user privacy while keeping recommendation performance. Nevertheless, several research works show that the central server can still infer some sensitive information based on intermediate parameters. For examples, a curious server can identify items the user has interacted with according to the non-zero gradients sent by the client [31]. Besides, the server can also infer the user ratings as long as obtaining the user uploaded gradients in two consecutive rounds [10]. To further protect privacy of FedRS, many studies have incorporated other privacy protection mechanisms into the FedRS, including pseudo items, homomorphic encryption, secret sharing and differential privacy. This section introduces the application of each privacy mechanism used in FedRS, and compare their advantages and limitations.

A. Pseudo Items

To prevent the server from inferring the set of items that users have interacted with based on non-zero gradients, some studies utilize pseudo items to protect user interaction behaviors in FedRS. The key idea of pseudo items is that the clients not only upload gradients of items that have been interacted with but also upload gradients of some sampled items that have not been with.

For example, Lin $et\ at.$ [22] propose a federated recommendation framework for explicit feedback scenario named FedRec, in which they design an effective hybrid filling strategy to generated virtual ratings of unrated items by following equation:

$$r_{ui}^{'} = \begin{cases} \frac{\sum_{k=1}^{m} y_{uk} r_{uk}}{\sum_{k=1}^{m} y_{uk}}, & t < T_{predict} \\ \hat{r}_{ui}, & t \geq T_{predict} \end{cases} \tag{3}$$

where t denotes the number of current training iteration, and $T_{predict}$ denotes the iteration number when chooses the average value or predict value as virtual rating value to a sampled item i. However, the hybrid filling strategy in FedRec introduces extra noise to the recommendation model, which inevitably affects the model performance. To tackle this problem, Feng $et\ at.$ [39] design a lossless version of FedRec named FedRec++. FedRec++ divides clients into ordinary clients and denoising clients. The denosing clients collect noisy gradients from ordinary clients and send the summation of the noisy gradients to server to eliminate the gradient noise.

Although pseudo items can effectively protect user interaction behaviors in FedRS, it does not modify the gradients of rated items. The curious server can still infer user ratings on the gradients uploaded by users [10].

B. Homomorphic Encryption

To further protect the user ratings in FedRS, many studies attempt to encrypt intermediate parameters before uploading them to the server. Homomorphic encryption mechanism allows mathematical operation on encrypted data [40], so it is well suited for the intermediate parameters upload and aggregation processes in FedRS.

For example, Chai $et\ at.$ [10] propose a secure federated matrix factorization framework named FedMF, in which clients use Paillier homomorphic encryption mechanism [41] to encrypt the gradients of item embedding matrix before uploading them to the server, and the server aggregates gradients on the cipher-text. Due to the characteristics of homomorphic encryption, FedMF can achieve the same recommendation accuracy as the traditional matrix factorization.

TABLE I: Comparison between different privacy mechanism.

Privacy Mechanisms	Ref	Main Protect Object	Accuracy Loss	Communication/Computation Costs
Pseudo Items	[22] [39] [31] [44] [45]	Interaction Behaviors	✓	Low Costs
Homomorphic Encryption	[10] [31] [42]	Ratings High-order Graph Social Features	✗	High Computation Costs
Secret Sharing	[29] [45]	Ratings	✗	High Communication Costs
Local Differential Privacy	[27] [31] [44]	Ratings	✓	Low Costs

However, FedMF causes serious computation overheads since all computation operations are performed on the ciphertext and most of system's time is spent on server updates. Besides, FedMF assumes that all participants are honest and will not leak the secret key to the server, which is hard to guarantee in reality.

Besides, many studies also utilize homomorphic encryption mechanism to integrate private information from other participants to improve recommendation accuracy [31] [42]. For examples, Wu $et\ al.$ [31] use homomorphic encryption mechanism to find the anonymous neighbors of users to expanse local user-item graph. And Perifanis $et\ al.$ [42] use Cheon-Kim-Kim-Song (CKKS) fully homomorphic encryption mechanism [43] to incorporate learned parameters between user's friends after the global model is generated.

Homomorphic encryption mechanism based FedRS can effectively protect user ratings while maintaining recommendation accuracy. Besides, it can prevent privacy leaks when integrating information from other participants. However, homomorphic encryption brings huge computation costs during operation process. And it is also a serious challenge to keep the secret key not be obtained by the server or other malicious participants.

C. Secret Sharing

As another encryption mechanism used in FedRS, secret sharing mechanism breaks intermediate parameters up into multiple pieces, and distributes the pieces among participants, so that only when all pieces are collected can reconstruct the intermediate parameters.

For example, Ying [29] proposes a secret sharing based federated matrix factorization framework named ShareMF. The participants divide the item matrix gradients g^{plain} into several random numbers that meet:

$$g^{plain} = g^{sub1} + g^{sub2} + ... + g^{subt}. \tag{4}$$

Each participant keeps one of the random numbers and send the rest to $t-1$ sampled participants, then uploads the sum of received and kept numbers as hybrid gradients to the server for aggregation. ShareMF protects the user ratings and interaction behaviors from being inferred by the server, but the rated items can still be leaked to other participants who received the split numbers. To tackle this problem, Lin $et\ al.$ [45] combine secret sharing and pseudo items mechanisms to provide stronger privacy guarantee.

Secret sharing mechanism based FedRS can protect user ratings while maintaining recommendation accuracy, and have lower computation costs compared to homomorphic encryption based FedRS. But the exchange process of pieces between participants greatly increases the communication costs.

D. Local Differential Privacy

Considering the huge computation or communication costs caused by encryption based mechanisms, many studies try to use perturbation based mechanisms to adapt to large-scale FedRS for the industrial scenarios. Local differential privacy (LDP) mechanism allows to statistical computations while guaranteeing each individual participant's privacynoise [46] [47], which can be used to perturb the intermediate parameters in FedRS.

For example, Dolui $et\ al.$ [27] propose a federated matrix factorization framework, which applies differential privacy on item embedding matrix before sending it to server for weighted average. However, the server can still infer which items the user has rated just by comparing the changes in item embedding matrix.

In order to achieve a more comprehensive privacy protection during model training process, Wu $et\ al.$ [31] combines pseudo items and LDP mechanisms to protect both user interaction behaviors and ratings in FedGNN. Firstly, to protect user interaction behaviors in FedGNN, the clients randomly sample N items that they have not interacted with, then generate the virtual gradients of item embeddings by using a same Gaussian distribution as the real embedding gradients. Secondly, to protect user ratings in FedGNN, the clients apply a LDP module to clip the gradients according to their L2-norm with a threshold δ and perturb the gradients by adding zero-mean Laplacian noise. The LDP module of FedGNN can be formulated as follow:

$$g_i = clip(g_i, \delta) + Laplace(0, \lambda). \tag{5}$$

where λ is the Laplacian noise strength. However, the gradient magnitude of different parameters varies during training process, thus it is usually not appropriate to perturb gradients at different magnitudes with a constant noise strength. So Liu

et al. [44] propose to add dynamic noise according to the gradients, which can be formulated as follow:

$$g_i = clip(g_i, \delta) + Laplace(0, \lambda \cdot mean(g_i)). \quad (6)$$

Local differential privacy mechanism doesn't bring heavy computation and communication overhead to FedRS, but the additional noise inevitably affects the performance of the recommendation model. Thus, in the actual application scenario, we must consider the trade-off between the privacy and recommendation accuracy.

E. Comparison

To protect stronger privacy guarantee, many privacy mechanisms (i.e., pseudo items, homomorphic encryption, differential privacy privacy and secret sharing) have been widely used in FedRS, and the comparison between these mechanisms is shown in Table I. Firstly, the main protect objects of these mechanisms are different: pseudo items mechanism is to protect user interaction behaviors, and the rest mechanisms is to protect user ratings. Besides, homomorphic encryption can aslo integrate data from other paritcipants in a privacy-preserving way. Secondly, homomorphic encryption and secret sharing are both encryption-based mechanisms, and they can protect privacy while keeping accuracy. However, the high computation cost of homomorphic encryption limits the application in large-scale industrial scenarios. Although the secret sharing mechanism reduces the computation costs, the communication costs increase greatly. Pseudo items and differential privacy mechanisms protect privacy by adding random noise, which has low computation costs and don't bring additional communication costs. But the addition of random noise will inevitably affect model performance to a certain extent.

IV. Security of Federated recommendation Systems

Apart from privacy leakage problems, traditional recommendation systems for centralized data storage are also vulnerable to poisoning attacks (shilling attacks) [48] [49] [50] [51] [52] [53]. Attackers can poison recommendation systems and make recommendations as their desires by injecting well-crafted data into the training dataset. But most of these poisoning attacks assume that the attackers have full prior knowledge of entire training datasets. Such assumption may be not valid for FedRS since the data in FedRS is distributed and stored locally for each participant. Thus, FedRS provides a stronger security guarantee than traditional recommendation systems. However, the latest studies indicate that attackers can still conduct poisoning attacks on FedRS with limited prior knowledge [12] [11] [54] [55]. In this section, we summarize some novel poisoning attacks against FedRS and provide some defense methods.

A. Poisoning Attacks

According to the goal of attacks, the poisoning attacks against FedRS can be categorized into targeted attacks and untargeted attacks as shown in Table II.

1) Target Poisoning Attacks: The goal of target attacks on FedRS is to increase or decrease the exposure chance of specific items, which are usually driven by financial profit. For example, Zhang *et al.* [12] propose a poisoning attack for item promotion (PipAttack) against FedRS by utilizing popularity bias. To boost the rank score of target items, PipAttack use popularity bias to align target items with popular items in the embedding space. Besides, to avoid damaging recommendation accuracy and be detected, PipAttack designs a distance constraint to keep modified gradients uploaded by malicious clients closed to normal ones.

In order to further reduce the degradation of recommendation accuracy caused by targeted poisoning attacks, and the proportion of malicious clients needed to ensure the attack effectiveness, Rong [54] propose a model poisoning attack against FedRS (FedRecAttack), which makes use of a small proportion of public interactions to approximate the user feature matrix, then uses it to generate poisoned gradients.

Both PipAttack and FedRecAttack rely on some prior knowledge. For example, PipAttack assumes the attacker is available for popularity information, and FedRecAttack assumes the attacker can get public interactions. So the attack effectiveness is greatly reduced in the absence of prior knowledge, which makes both attacks not generic in all FedRS. To make attackers conduct effective poisoning attacks to FedRS without the prior knowledge, Rong *et al.* [55] design two methods (i.e., random approximation and hard user mining) for malicious clients to generate poisoned gradients. In particular, random approximation (A-ra) uses Gaussian distribution to approximate normal users' embedding vectors, and hard user mining (A-hum) uses gradient descent to optimize users' embedding vectors obtained by A-ra to mine hard users. In this way, A-hum can still effectively attack FedRS with extremely small proportion of malicious users.

2) Untarget Poisoning Attacks: The goal of untarget attacks on FedRS is to degrade the overall performance of recommendation model, which are usually conducted by competing companies. For example, Wu *et al.* [11] propose an untargeted poisoning attack to FedRS named FedAttack, which uses globally hard sampling technique [62] to subvert model training process. More specifically, after inferring user's interest from local user profiles, the malicious clients select candidate items that best match the user's interest as negative samples, and select candidate items that least match the user's interest as positive samples. FedAttack only modifies training samples, and the malicious clients are also similar to normal clients with different interests, thus FedAttack can effectively damage the performance of FedRS even under defense.

B. Defense Methods

To reduce the influence of poisoning attacks on FedRS, many defense methods have been proposed in the literature, which can be classified into robust aggregation and anomaly detection.

1) Robust Aggregation: The goal of robust aggregation is to guarantee global model convergence when up to 50% of participants are malicious [63], which selects statistically

TABLE II: Representative works on the security of FedRS. RA refers to robust aggregation and AD refers to anomaly detection.

Works	Ref	Attack Type		Poison Object		Defense Type		Goal
		Target	Untarget	Model	Data	RA	AD	
PipAttack	[12]	✓		✓				Increase/decrease popularity of target items.
FedRecAttack	[54]	✓		✓				Increase/decrease popularity of target items.
A-ra/A-hum	[55]	✓		✓				Increase/decrease popularity of target items.
FedAttack	[11]		✓		✓			Degrade the overall performance of FedRS.
Median	[56]					✓		Guarantee global model convergence.
Trimmed-Mean	[56]					✓		Guarantee global model convergence.
(Multi-)Krum	[57]					✓		Guarantee global model convergence.
Bulyan	[58]					✓		Guarantee global model convergence.
Norm-Bounding	[59]					✓		Guarantee global model convergence.
A-FRS	[60]					✓		Guarantee global model convergence.
FSAD	[61]						✓	Identify and filter poisoned parameters.

more robust values rather than the mean values of uploaded intermediate parameters for aggregation.

Median [56] selects the median value of each updated model parameter independently as aggregated global model parameter, which can represent the centre of the distribution better. Specifically, the server ranks each $i-th$ parameter of n local model update, and uses the median value as $i-th$ parameter of global model.

Trimmed-Mean [56] removes the maximum and minimum values of each updated model parameter independently, and then takes the mean value as aggregated global model parameter. Specifically, the server ranks each $i-th$ parameter of n local model update, remove β smallest and β largest values, and uses the mean value of remained $n-2\beta$ as $i-th$ parameter of global model. In this way, Trimmed-Mean can effectively reduce the impact of outliers.

Krum and Multi-Krum [57]. Krum selects a local model that is the closest to the others as global model. Multi-Krum selects multiple local models by using Krum, then aggregates them into a global model. In this way, even if the selected parameter vectors are uploaded by malicious clients, their impact is still limited because they are similar to other local parameters uploaded by normal clients.

Bulyan [58] is a combination of Krum and Trimmed-Mean, which iteratively selects m local model parameter vectors through Krum, and then performs Trimmed-Mean on these m parameter vectors for aggregation. With high dimensional and highly non-convex loss function, Bulyan can still converge to effectual models.

Norm-Bounding [59] clips the received local parameters to a fixed threshold, then aggregates them to update the global model. Norm-Bounding can limit the contribution of each local model updates so as to mitigate the affect of poisoned parameters on the aggregated model.

A-FRS [60] utilizes gradient-based Krum instead of model parameter-based Krum to filter malicious clients in momentum-based FedRS. A-FRS theoretically guarantees that if the selected gradient is closed to the normal gradient, the momentum and model parameters will also be close to the normal momentum and model parameters.

Although these robust aggregation strategies provide convergence guarantees to some extent, most of them (i.e., Bulyan, Krum, Median and Trimmed-mean) greatly degrade the performance of FedRS. Besides, some noval attacks(i.e., PipAttack, FedAttack) [12] [11] utilize well-designed constraints to approximate the patterns of normal users and circumvent defenses, which further increases the difficulty of defense.

2) Anomaly Detection: The purpose of anomaly detection strategy is to identify the poisoned model parameters uploaded by malicious clients and filter them during the global model aggregation process. For example, Jiang $et\ al.$ [61] propose an anomaly detection strategy named federated shilling attack detector (FSAD) to detect poisoned gradients in federated collaborative filtering scenarios. FSAD extracts 4 novel features according to the gradients uploaded by clients, then uses the gradient-based features to train a semi-supervised bayes classifier so as to identify and filter the poisoned gradients. However, in FedRS, the interests of different users vary widely, thus the parameters they uploaded are usually quite different, which increases the difficulty of anomaly detection [54].

V. Heterogeneity of Federated Recommendation Systems

Compared with traditional recommendation systems, FedRS face more severe challenges in terms of heterogeneity, which are mainly reflected in system heterogeneity, statistical heterogeneity and model heterogeneity, as shown in Fig. 3.

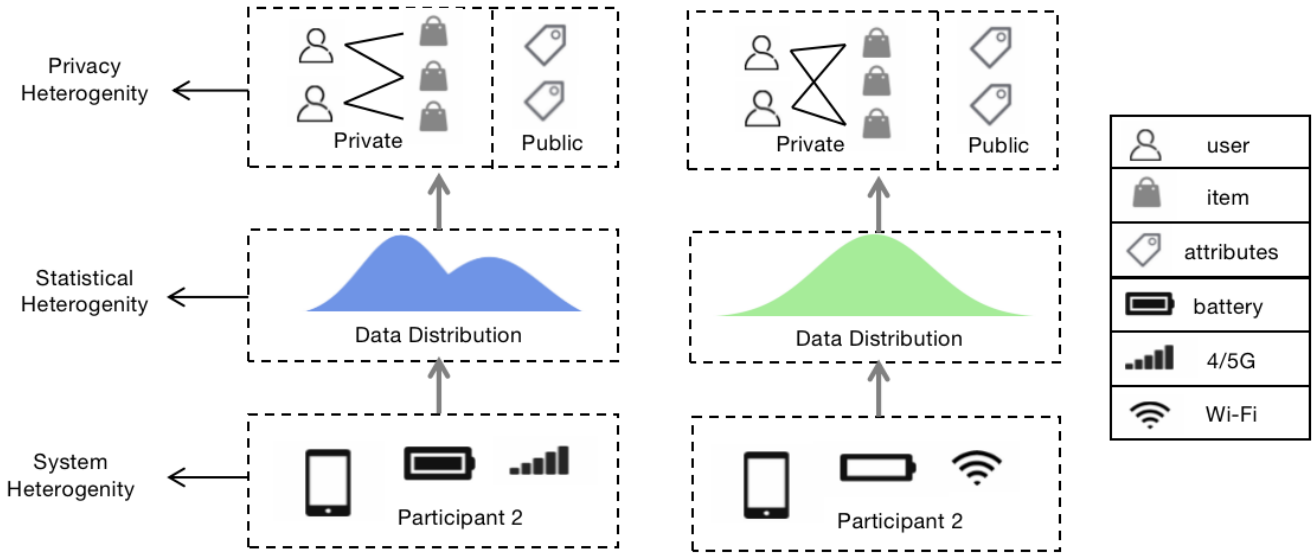


Fig. 3: Heterogeneity of federated recommendation systems.

System heterogeneity refers to client devices have significantly different storage, computation, and communication capabilities. Devices with limited capabilities greatly affects training efficiency, and further reduces the accuracy of the global recommendation model. [64]; Statistical heterogeneity refers to the data collected by different clients is usually not independent and identically distributed (non-IID). As a result, simply training a single global model is difficult to generalize to all clients, which affects the personalization of recommendations [65]; Privacy heterogeneity means that the privacy constraints of different users and information vary greatly, so simply treating them with the same privacy budgets will carry unnecessary costs [66]. This section introduces some effective approaches to address the heterogeneity of FedRS.

A. System Heterogeneity

In FedRS, the hardware configuration, network bandwidth and battery capacity of participating clients varies greatly, which results in diverse computing capability, communication speed, and storage capability [16]. During the training process, the clients with limited capacity could become stragglers, and even drop out of current training due to network failure, low battery and other problems [18]. The system heterogeneity significantly delays the training process of FedRS, further reducing the recommendation accuracy of the global model. To make the training process compatible with different hardware structures and tolerate the straggling and exit issues of clients, the most common methods are asynchronous communication [67] [18] and clients selection [68].

Asynchronous communication. Considering the synchronous communication based federated learning must wait for straggler devices during aggregation process, many asynchronous communication strategies are presented to improve training efficiency. For examples, FedSA [67] proposes a semi-asynchronous communication method, where the server aggregates the local models based on their arrival order of each round. FedAsync [18] uses a weighted average strategy to aggregate the local models based on staleness, which assigns less weight to delayed feedback in update process.

Clients selection. Client selection approach selects clients for updates based on resource constraints so that the server can aggregate as many local updates as possible at the same time. For example, in FedCS [68], the server sends a resource request to each client so as to get their resource information, then estimates the required time of model distribution, updating and uploading processes based on the resource information. According to the estimated time, the server determine which clients can participant in training process.

B. Statistical Heterogeneity

Most of the existing federated recommendation studies are built on the assumption that data in each participant is independent and identically distributed (IID). However, the data distribution of each client usually varies greatly, hence training a consistent global model is difficult to generalized to all clients under non-IID data and inevitably neglects the personalization of clients [66]. To address the statistical heterogeneity problem of FedRS, many effective strategies have been proposed, which are mainly based on meta learning [38] [69] and clustering [70] [71].

Meta learning. As known as "learning to learn", meta learning technology aims to quickly adapt the global model learned by other tasks to a new task by using only a few number of samples [36]. The rapid adaptation and good generalization abilities makes it particularly well-suited for building personalized federated recommendation models. For examples, FedMeta [38] uses Model-Agnostic Meta-Learning (MAML) [73] algorithm to learn a well-initialized model that can be quickly adapted to clients, and effectively improve the personalization and convergence of FedRS. However, FedMeta

needs to compute the second-order gradients, which greatly increases computation costs. Besides, the data split process also brings a huger challenge for clients with limited samples. Based on FedMeta, Wang $et\ al.$ [69] propose a new meta learning algorithm called Reptile which applies the approximate first-order derivatives for the meta-learning updates, which greatly reduces the computation overloads of clients. Moreover, Reptile doesn't need a data split process, which makes it also suitable for clients with limited samples.

Clustering. The core idea of clustering is training personalized models jointly with the same group of homogeneous clients. For examples, Jie $et\ al.$ [70] uses historical parameter clustering technology to realize personalized federated recommendation, in which the server aggregates local parameters to generate global model parameters and clusters the local parameters to generate clustering parameters for different client groups. Then the clients combine the clustering parameters with the global parameters to learn personalized models. Luo $et\ al.$ [71] propose a personalized federated recommendation framework named PerFedRec, which constructs a collaborative graph and integrates attribute information so as to jointly learn the user representations by federated GNN. Based on the learned user representations, clients are clustered into different groups. And each cluster learns a cluster-level recommendation model. At last, each client can obtain a personalized model by merging the global recommendation model, the cluster-level recommendation model, and the fine-tuned local recommendation model. Although clustering based approaches can alleviate statistical heterogeneity, the clustering and combination process greatly increase the computation costs.

C. Privacy Heterogeneity

In reality, the privacy restrictions of different participants and information vary greatly, thereby using the same high level of privacy budget for all participants and information is unnecessary, which even increases the computation/communication costs and degrades the model performance.

Heterogeneous user privacy. In order to adapt the privacy needs for different users, Anelli $et\ al.$ [72] present a user controlled federated recommendation framework named FedeRank. FedeRank introduces a probability factor $\pi \in [0, 1]$ to control the proportion of interacted item updates and masks the remain interacted item update by setting them to zero. In this way, FedeRank allows users decide the proportion of data they want to share by themselves, which addresses the heterogeneity of user privacy.

Heterogeneous information privacy. In order to adapt the privacy needs of different information components, HPFL [66] designs a differentiated component aggregation strategy. To obtain the global public information components, the server directly weighted aggregates the local public components with same properties. And to obtain the global privacy information components, the user and item representations are kept locally, and the server only aggregates the local drafts without the need to align the presentations. With the differentiated component aggregation strategy, HPFL can safely aggregate components with heterogeneous privacy constraints in user modeling scenarios.

VI. Communication Costs of Federated recommendation Systems

To achieve satisfactory recommendation performance, FedRS requires multiple communications between server and clients. However, the real-world recommendation systems are usually conducted by complexity deep learning models with large model size [74], and millions of parameters needs to be updated and communicated [13], which brings severe communication overload to resource limited clients and further affects the application of FedRS in large-scale industrial scenarios. This section summarizes some optimization methods to reduce communication costs of FedRS, which can be classified into importance-based updating [75] [20] [76] [77], model compression [78] [79], active sampling [80] and one shot learning [81].

A. Importance-based Model Updating

Importance-based model updating selects importance parts of the global model instead of the whole model to update and communicate, which can effectively reduce the communicated parameter size in each round.

For examples, Qin $et\ al.$ [75] propose a federated framework named PPRSF, which uses 4-layers hierarchical structure for reducing communication costs, including the recall layer, ranking layer, re-ranking layer and service layer. In the recall layer, the server roughly sorts the large inventory by using public user data, and recalls relatively small number of items for each client. In this way, the clients only need to update and communicate the candidate item embeddings, which greatly reduces the communication costs between server and clients, and the computation costs in the local model training and inference phases. However, the recall layer of PPRSF need to get some public information of users, which raises certain difficulty and privacy concerns.

Yi $et\ al.$ [20] propose an efficient federated news recommendation framework called Efficient-FedRec, which breaks the news recommendation model into a small user model and a big news model. Each client only requests the user model and a few news representations involved in their local click history for local training, which greatly reduces the communication and computation overhead. To further protect specific user click history against the server, they transmit the union news representations set involved in a group of user click history by using a secure aggregation protocol [82].

Besides, Khan $et\ al.$ [76] propose a multi-arm bandit method (FCF-BTS) to select part of the global model that contains a smaller payload to all clients. The rewards of selection process is guided by Bayesian Thompson Sampling (BTS) [83] approach with Gaussian priors. Experiments show that FCF-BTS can reduce 90% model payload for highly sparse datasets. Besides, the selection process occurs in the server side, thus avoiding additional computation costs on the clients. But FCF-BTS causes 4% - 8% loss in recommendation accuracy.

To achieve a better balance between recommendation accuracy and efficiency, Ai $et\ al.$ [77] propose an all-MLP

network that uses a Fourier sub-layer to replace the self-attention sub-layer in a Transformer encoder so as to filter noise data components unrelated to the user's real interests, and adapts an adaptive model pruning technique to discard the noise model components that doesn't contribute to model performance. Experiments show that all-MLP network can significantly reduce communication and computation costs, and accelerates the model convergence.

Importance-based model updating strategies can greatly reduce communication and computation costs at the same time, but only selecting the important parts for updating inevitably reduces the recommendation performance.

B. Model Compression

Model Compression is a well-known technology in distributed learning [84], which compresses the communicated parameters per round to be more compact.

For examples, Konen $et\ al.$ [78] propose two methods (i.e., structured updates and sketched updates) to decrease the uplink communication costs under federated learning settings. Structured updates method directly learns updates from a pre-specified structure parameterized using fewer variables. Sketched updates method compresses the full local update using a lossy compression way before sending it to server. These two strategies can reduce the communication costs by 2 orders of magnitude.

To reduce the uplink communication costs in deep learning based FedRS, JointRec [79] combines low-rank matrix factorization [85] and 8-bit probabilistic quantization [86] methods to compress weight update. Supposing the weight update matrix of client n is $H_n^{a\times b}$, $a \leq b$, low-rank matrix factorization decomposes $H_n^{a\times b}$ into two matrices: $H_n^{a\times b} = U_n^{a\times k} V_n^{k\times b}$, where $k = b/N$ and N is a positive number that influences the compression performance. And 8-bit probabilistic quantization method transforms the position of matrix value into 8-bit value before send it to server. Experiments demonstrate that JointRec can realize $12.83\times$ larger compression ratio while maintaining recommendation performance.

Model compression methods achieve significant results in reducing the uplink communication costs. However, the reduction of communication cost sacrifices the computation resources of the clients, so it's necessary to consider the trade-off between computation and communication costs when using model compression.

C. Client Sampling

In traditional federated learning frameworks [8], the server randomly selects clients to participate in the training process and simply aggregates the local models by average, which requires a large number of communications to realize satisfactory accuracy. Client sampling utilizes efficient sampling strategies so as to improve the training efficiency and reduce the communication rounds.

For example, Muhammad $et\ al.$ [80] propose an effective sampling strategy named FedFast to speed up the training efficiency of federated recommendation models while keeping more accuracy. FadFast consists of two efficient components: ActvSAMP and ActvAGG. ActvSAMP uses K-means algorithm to cluster users based on their profile, and samples clients in equal proportions from each cluster. And ActvAGG propagates local updates to the other clients in the same cluster. In this way, the learning process for these similar users is greatly accelerated and overall efficiency of the FedRS is consequently improved. Experiments show that FedFast reduces communication rounds by 94% compared to FedAvg [8]. However, FedFast is faced with the cold start problem because it requires a number of users and items for training. Besides, FedFast needs to retrain the model to support new users and items.

D. One Shot Federated Learning

The goal of one shot federated learning mechanism is to reduce communication rounds of FedRS [87] [88], which limits its communication to a single round to aggregate knowledge of local models. For example, Eren $et\ al.$ [81] implement an one-shot federated learning framework for cross-platform FedRS named FedSPLIT. FedSPLIT aggregates model through knowledge distillation [89], which can generate client specific recommendation results with just a single pair of communication rounds between the server and clients after a small initial communication. Experiments show that FedSPLIT realizes similar root-mean-square error (RMSE) compared with multi-round communication scenarios, but it is not applicable to the scenario where the participant is a individual user.

VII. Open Source Platforms

This section introduces five open source platforms that can be used to build FedRS: Federated AI Technology Enabler (Fate)[1], Tensorflow Federated (TFF)[2], Pysyft[3], PaddleFL[4], and FederatedScope[5]. The comparison among some existing open source platforms is shown in Table III.

A. Federated AI Technology Enabler

Federated AI Technology Enabler (Fate) [90] is the first open source platform for federated learning around the world, which aims to enable companies and organizations to collaborate on data while keeping data privacy and security. Fate supports various machine learning algorithms under federated learning settings, including logistic regression, XGBOOST, deep learning and transfer learning. Besides, Fate integrates homomorphic encryption, differential privacy and secret sharing mechanisms to protect privacy against the curious server.

The structure of FATE consists of seven major modules: FederatedML, EggRoll, FATE-FLow, FATE-Board, FATE-Serving, KubeFATE and FATE-cloud. FederatedML implements privacy-preserving federated machine learning algorithms; EggRoll manages the distributed computation framework; FATE-FLow coordinates the execution of the algorithm

[1]https://github.com/FederatedAI/FATE
[2]https://github.com/tensorflow/federated
[3]https://github.com/OpenMined/PySyft
[4]https://github.com/PaddlePaddle/PaddleFL
[5]https://github.com/alibaba/FederatedScope

components; FATE-Board provides visualization for building and evaluating models; FATE-Serving provides online inference for the users; KubeFATE helps deploy Fate platform by using cloud native technologies; FATE-cloud provide cross-cloud deployment and management services.

Fate provides a federated recommendation module (FederatedRec) to solve the recommendation problems of rate prediction and item ranking tasks. FederatedRec implements many common recommendation algorithms under federated learning settings, including factorization machine, matrix factorization, SVD, SVD++ and generalized matrix factorization.

B. Tensorflow Federated

Tensorflow Federated (TFF) [91] is a lightweight system developed by Google, which provides the building blocks to enables developers to implement own federated models based on TensorFlow. Besides, developers can plug any existing Keras model into TFF with just a few lines of code. To enhancing privacy guarantees for federated learning, TFF integrates differential privacy mechanism.

The interfaces of TFF are organized in two layers API (i.e., Federated Learning API and Federated Core API). Federated Learning API implements high-level interfaces for developers to make training and evaluation process of federated learning. Federated Core API provides lower-level interfaces to express novel federated learning algorithms by using TensorFlow and distributed communication operators.

Based on TFF, Singhal *et al*. [92] implements a model-agnostic framework for fast partial local federated learning, which is suitable for large-scale collaborative filtering recommendation scenarios.

C. Pysyft

PySyft [93] is developed by Open-Mined, which also provides the building blocks for developers to implement own federated recommendation algorithms. Compared with TFF, PySyft can work with both Tensorflow and Pytorch. For privacy protection, PySyft can flexibly and simply integrate homomorphic encryption, differential privacy and secret sharing mechanisms so as to defend against the honest-but-curious server and participants. However, Pysyft doesn't disclose the detailed interface design or system architecture.

D. PaddleFL

PaddleFL [94] is an open source federated learning platform developed by Baidu, which integrates both differential privacy and secret sharing mechanisms to provide privacy guarantees. PaddleFL contains two major components: Data Parallel and Federated Learning with MPC (PFM). Data Parallel is responsible for defining, distributing and training a federated learning task. PFM implements secure multi-party computation to ensure training and inference security.

PaddleFL provides many federated recommendation algorithms that can be used directly. For example, PaddleFL implements a classical session-based recommendation model Gru4rec [95] under the federated learning settings and provide simulated experiments on real world dataset. But the simulated experiment suppose all datasets in different organizations are homogeneous, which is only satisfied under ideal case. In addition, PaddleFL also provides a strategy to train a Click-Through-Rate(CTR) model by using FedAvg [8] algorithm.

E. FederatedScope

FederatedScope [96], developed by Alibaba, is a flexible federated learning platform for heterogeneity. FederatedScope employs an event-driven architecture to support asynchronous training, and coordinate participants with personalized behaviors and multiple goals into federated learning scenarios. FederatedScope can easily support different machine learning libraries such as Tensorflow and Pytorch. Besides, FederatedScope enables various kinds of plug-in components and operations thar can be used for efficient further development. For privacy protection plug-ins, FederatedScope integrates homomorphic encryption, differential privacy and secret sharing mechanisms to enhance privacy guarantees. In the federated recommendation scenario, FederatedScope has built in matrix factorization models, datasets (Netflix and MovieLen) and trainer under different federated learning settings.

VIII. Future Directions

This section presents and discusses many prospective research directions in the future. Although some directions have been covered in above sections, we believe they are necessary for FedRS, and need to be further researched.

Decentralized FedRS. Most of current FedRS are based on client-server communication architecture, which faces single-point-of-failure and privacy issues caused by the central server [97]. While much work has been devoted to decentralized federated learning [98] [99], few decentralized FedRS have been studied. A feasible solution is to replace client-server communication architecture with peer-peer communication architecture to achieve fully decentralized federated recommendation. Hegeds *et al*. [19] propose a fully decentralized matrix factorization framework based on gossip learning [100], where each participant sends their copy of the global recommendation model to random online neighbors in the peer to peer network.

Incentive mechanisms in FedRS. FedRS collaborate with multiple participants to train a global recommendation model, and the recommendation performance of global model is highly dependent on the quantity and quality of data provided by the participants. Therefore, it is significant to design an appropriate incentive mechanism to inspire participants to contribute their own data and participate in collaborative training, especially in the cross-organization federated recommendation scenarios. The incentive mechanisms must be able to measure the clients' contribution to the global model fairly and efficiently.

Privacy of serving phase. Although many studies have combined different privacy mechanisms to protect user privacy in the training phase of FedRS, the privacy protection for the serving phase is still underexplored. To prevent user recommendation results from leaking, most of the current

TABLE III: The comparison among some existing open source platforms.

Platforms		Fate	TFF	Pysyft	PaddleFL	FederatedScope
Publisher		WeBank	Google	OpenMined	Baidu	Alibaba
Audience	Academia	✓	✓	✓	✓	✓
	Industry	✓				✓
Models	Neural Network	✓	✓	✓	✓	✓
	Tree Model	✓				✓
	Linear Model	✓	✓	✓	✓	✓
Privacy	Homomorphic encryption	✓		✓		✓
	Differential Privacy	✓	✓	✓	✓	✓
	Secret Sharing	✓		✓		✓
Libraries	Tensorflow		✓	✓		✓
	Pytorch			✓		✓

studies assume local serving, where the server sends the entire set of candidate items to clients, and clients generate recommendation results locally [10] [21]. However, such design brings enormous communication, computation and memory costs for clients since there are usually millions of items in real-world recommendation systems. Another feasible solution is online serving, where clients send encrypted or noised user embedding to the server to recall top-N candidate items, then clients generate personalized recommendation results based one these candidate items [101]. Nevertheless, there is a risk of privacy leakage associated with online serving, because recalled items are known to the server.

Cold start problem in FedRS. The cold start problem means that recommendation systems cannot generate satisfactory recommendation results for new users with little history interactions. In federated settings, the user data is stored locally, so it is more difficult to integrate other auxiliary information (e.g., social relationships) to alleviate the cold start problem. Therefore, it is a challenging and prospective research direction to address the cold start problem while ensuring user privacy.

Secure FedRS. In the real world, the participants in the FedRS are likely to be untrustworthy. Therefore, participants may upload poisoned intermediate parameters to affect recommendation results or destroy recommendation performance. Although some robust aggregation strategies [57] and detection methods [61] have been proposed to defense poisoning attacks in federated learning settings, most of them does not work well in FedRS. One one hand, some strategies such as Krum, Median and Trimmed-mean degrade the recommendation performance to a certain extend. One the other hand, some novel attacks [11] use well-designed constraints to mimic the patterns of normal users, extremely increasing the difficulty to be detected and defensed. Currently, there is still no effective defense methods against these poisoning attacks while maintaining recommendation accuracy.

IX. Conclusion

A lot of effort has been devoted to federated recommendation systems. A comprehensive survey is significant and meaningful. This survey summarizes the latest studies from aspects of the privacy, security, heterogeneity and communication costs. Based on these aspects, we also make a detailed comparison among the existing designs and solutions. Moreover, we present many prospective research directions to promote development in this field. FedRS will be a promising field with huge potential opportunities, which requires more efforts to develop.

Acknowledgments

This research is partially supported by the National Key R&D Program of China No.2021YFF0900800, the NSFC No.91846205, the Shandong Provincial Key Research and Development Program (Major Scientific and Technological Innovation Project) (No.2021CXGC010108), the Shandong Provincial Natural Science Foundation (No.ZR202111180007), the Fundamental Research Funds of Shandong University, and the Special Fund for Science and Technology of Guangdong Province under Grant (2021S0053).

References

[1] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," in *Proceedings of the 2nd ACM Conference on Electronic Commerce*, pp. 158–167, 2000.

[2] J. B. Schafer, J. A. Konstan, and J. Riedl, "E-commerce recommendation applications," *Data mining and knowledge discovery*, vol. 5, no. 1, pp. 115–153, 2001.

[3] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li, "Drn: A deep reinforcement learning framework for news recommendation," in *Proceedings of the 2018 world wide web conference*, pp. 167–176, 2018.

[4] J. Liu, P. Dolan, and E. R. Pedersen, "Personalized news recommendation based on click behavior," in *Proceedings of the 15th international conference on Intelligent user interfaces*, pp. 31–40, 2010.

[5] W. Yue, Z. Wang, J. Zhang, and X. Liu, "An overview of recommendation techniques and their applications in healthcare," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 4, pp. 701–717, 2021.

[6] J. Kim, D. Lee, and K.-Y. Chung, "Item recommendation based on context-aware model for personalized u-healthcare service," *Multimedia Tools and Applications*, vol. 71, no. 2, pp. 855–872, 2014.

[7] J. P. Albrecht, "How the gdpr will change the world," *Eur. Data Prot. L. Rev.*, vol. 2, p. 287, 2016.

[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[9] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[10] D. Chai, L. Wang, K. Chen, and Q. Yang, "Secure federated matrix factorization," *Intelligent Systems, IEEE*, vol. PP, no. 99, pp. 1–1, 2020.

[11] C. Wu, F. Wu, T. Qi, Y. Huang, and X. Xie, "Fedattack: Effective and covert poisoning attack on federated recommendation via hard sampling," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, (New York, NY, USA), p. 4164–4172, Association for Computing Machinery, 2022.

[12] S. Zhang, H. Yin, T. Chen, Z. Huang, Q. V. H. Nguyen, and L. Cui, "Pipattack: Poisoning federated recommender systems for manipulating item promotion," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 1415–1423, 2022.

[13] C.-L. Liao and S.-J. Lee, "A clustering based approach to improving the efficiency of collaborative filtering recommendation," *Electronic Commerce Research and Applications*, vol. 18, pp. 1–9, 2016.

[14] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[15] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, 2019.

[16] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[17] L. Yang, B. Tan, V. W. Zheng, K. Chen, and Q. Yang, "Federated recommendation systems," in *Federated Learning*, pp. 225–239, Springer, 2020.

[18] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[19] I. Hegedűs, G. Danner, and M. Jelasity, "Decentralized recommendation based on matrix factorization: a comparison of gossip and federated learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 317–332, Springer, 2019.

[20] J. Yi, F. Wu, C. Wu, R. Liu, G. Sun, and X. Xie, "Efficient-fedrec: Efficient federated learning framework for privacy-preserving news recommendation," *arXiv preprint arXiv:2109.05446*, 2021.

[21] 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," *arXiv preprint arXiv:1901.09888*, 2019.

[22] G. Lin, F. Liang, W. Pan, and Z. Ming, "Fedrec: Federated recommendation with explicit feedback," *Intelligent Systems, IEEE*, vol. PP, no. 99, pp. 1–1, 2020.

[23] C. Chen, L. Li, B. Wu, C. Hong, L. Wang, and J. Zhou, "Secure social recommendation based on secret sharing," *arXiv preprint arXiv:2002.02088*, 2020.

[24] WuChuhan, WuFangzhao, LyuLingjuan, HuangYongfeng, and XieXing, "Fedctr: Federated native ad ctr prediction with cross platform user behavior data," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2021.

[25] S. Kalloori and S. Klingler, "Horizontal cross-silo federated recommender systems," in *Fifteenth ACM Conference on Recommender Systems*, pp. 680–684, 2021.

[26] Koren, Yehuda, Bell, Robert, Volinsky, and Chris, "Matrix factorization techniques for recommender systems.," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[27] K. Dolui, I. Cuba Gyllensten, D. Lowet, S. Michiels, H. Hallez, and D. Hughes, "Towards privacy-preserving mobile applications with federated learning: The case of matrix factorization (poster)," in *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 624–625, 2019.

[28] J. Hua, C. Xia, and S. Zhong, "Differentially private matrix factorization," in *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, p. 1763–1770, AAAI Press, 2015.

[29] S. Ying, "Shared mf: A privacy-preserving recommendation system," *arXiv preprint arXiv:2008.07759*, 2020.

[30] V. Perifanis and P. S. Efraimidis, "Federated neural collaborative filtering," *Knowledge-Based Systems*, vol. 242, p. 108441, 2022.

[31] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "Fedgnn: Federated graph neural network for privacy-preserving recommendation," *arXiv preprint arXiv:2102.04925*, 2021.

[32] M. Imran, H. Yin, T. Chen, N. Q. V. Hung, A. Zhou, and K. Zheng, "Refrs: Resource-efficient federated recommender system for dynamic and diversified user preferences," *ACM Trans. Inf. Syst.*, aug 2022. Just Accepted.

[33] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

[34] M. Huang, H. Li, B. Bai, C. Wang, K. Bai, and F. Wang, "A federated multi-view deep learning framework for privacy-preserving recommendations," *arXiv preprint arXiv:2008.10808*, 2020.

[35] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 2333–2338, 2013.

[36] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," *arXiv preprint arXiv:1904.04232*, 2019.

[37] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[38] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," *arXiv preprint arXiv:1802.07876*, 2018.

[39] F. Liang, W. Pan, and Z. Ming, "Fedrec++: Lossless federated recommendation with explicit feedback," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, pp. 4224–4231, 2021.

[40] A. Abbas, A. Hidayet, U. A. Selcuk, and C. Mauro, "A survey on homomorphic encryption schemes: Theory and implementation," *Acm Computing Surveys*, vol. 51, no. 4, pp. 1–35, 2017.

[41] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International conference on the theory and applications of cryptographic techniques*, pp. 223–238, Springer, 1999.

[42] V. Perifanis, G. Drosatos, G. Stamatelatos, and P. S. Efraimidis, "Fedpoirec: Privacy preserving federated poi recommendation with social influence," *arXiv preprint arXiv:2112.11134*, 2021.

[43] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *International conference on the theory and application of cryptology and information security*, pp. 409–437, Springer, 2017.

[44] Z. Liu, L. Yang, Z. Fan, H. Peng, and P. S. Yu, "Federated social recommendation with graph neural network," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 4, pp. 1–24, 2022.

[45] Z. Lin, W. Pan, and Z. Ming, "Fr-fmss: federated recommendation via fake marks and secret sharing," in *Fifteenth ACM Conference on Recommender Systems*, pp. 668–673, 2021.

[46] C. Dwork, "Calibrating noise to sensitivity in private data analysis," *Lecture Notes in Computer Science*, vol. 3876, no. 8, pp. 265–284, 2012.

[47] M. Hardt and K. Talwar, "On the geometry of differential privacy," in *Proceedings of the forty-second ACM symposium on Theory of computing*, pp. 705–714, 2010.

[48] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik, "Data poisoning attacks on factorization-based collaborative filtering," *Advances in neural information processing systems*, vol. 29, 2016.

[49] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

[50] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proceedings of the 34th annual computer security applications conference*, pp. 381–392, 2018.

[51] H. Huang, J. Mu, N. Z. Gong, Q. Li, B. Liu, and M. Xu, "Data poisoning attacks to deep learning based recommender systems," *arXiv preprint arXiv:2101.02644*, 2021.

[52] H. Zhang, Y. Li, B. Ding, and J. Gao, "Practical data poisoning attack against next-item recommendation," in *Proceedings of The Web Conference 2020*, pp. 2458–2464, 2020.

[53] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[54] D. Rong, S. Ye, R. Zhao, H. N. Yuen, J. Chen, and Q. He, "Fedrecattack: Model poisoning attack to federated recommendation," *arXiv preprint arXiv:2204.01499*, 2022.

[55] D. Rong, Q. He, and J. Chen, "Poisoning deep learning based recommender model in federated learning scenarios," *arXiv preprint arXiv:2204.13594*, 2022.

[56] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*, pp. 5650–5659, PMLR, 2018.

[57] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[58] R. Guerraoui, S. Rouault, *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*, pp. 3521–3530, PMLR, 2018.

[59] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?," *arXiv preprint arXiv:1911.07963*, 2019.

[60] C. Chen, J. Zhang, A. K. Tung, M. Kankanhalli, and G. Chen, "Robust federated recommendation system," *arXiv preprint arXiv:2006.08259*, 2020.

[61] Y. Jiang, Y. Zhou, D. Wu, C. Li, and Y. Wang, "On the detection of shilling attacks in federated collaborative filtering," in *2020 International Symposium on Reliable Distributed Systems (SRDS)*, pp. 185–194, 2020.

[62] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus, "Hard negative mixing for contrastive learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21798–21809, 2020.

[63] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and S. Y. Philip, "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[64] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[65] Z. Jie, S. Chen, J. Lai, M. Arif, and Z. He, "Personalized federated recommendation system with historical parameter clustering," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–11.

[66] J. Wu, Q. Liu, Z. Huang, Y. Ning, H. Wang, E. Chen, J. Yi, and B. Zhou, "Hierarchical personalized federated learning for user modeling," in *Proceedings of the Web Conference 2021*, pp. 957–968, 2021.

[67] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "Fedsa: A semi-asynchronous federated learning mechanism in heterogeneous edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3654–3672, 2021.

[68] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE international conference on communications (ICC)*, pp. 1–7, IEEE, 2019.

[69] Q. Wang, H. Yin, T. Chen, J. Yu, A. Zhou, and X. Zhang, "Fast-adapting and privacy-preserving federated recommender system," *The VLDB Journal*, vol. 31, no. 5, pp. 877–896, 2022.

[70] Z. Jie, S. Chen, J. Lai, M. Arif, and Z. He, "Personalized federated recommendation system with historical parameter clustering," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–11, 2022.

[71] S. Luo, Y. Xiao, and L. Song, "Personalized federated recommendation via joint representation learning, user clustering, and model adaptation," in *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, (New York, NY, USA), p. 4289–4293, Association for Computing Machinery, 2022.

[72] V. W. Anelli, Y. Deldjoo, T. D. Noia, A. Ferrara, and F. Narducci, "Federank: User controlled feedback with federated recommender systems," in *European Conference on Information Retrieval*, pp. 32–47, Springer, 2021.

[73] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*, pp. 1126–1135, PMLR, 2017.

[74] B. Acun, M. Murphy, X. Wang, J. Nie, C.-J. Wu, and K. Hazelwood, "Understanding training efficiency of deep learning recommendation models at scale," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 802–814, IEEE, 2021.

[75] J. Qin, B. Liu, and J. Qian, "A novel privacy-preserved recommender system framework based on federated learning," in *2021 The 4th International Conference on Software Engineering and Information Management*, pp. 82–88, 2021.

[76] F. K. Khan, A. Flanagan, K. E. Tan, Z. Alamgir, and M. Ammad-Ud-Din, "A payload optimization method for federated recommender systems," in *Fifteenth ACM Conference on Recommender Systems*, pp. 432–442, 2021.

[77] Z. Ai, G. Wu, B. Li, Y. Wang, and C. Chen, "Fourier enhanced mlp with adaptive model pruning for efficient federated recommendation," in *International Conference on Knowledge Science, Engineering and Management*, pp. 356–368, Springer, 2022.

[78] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[79] Sijing, Duan, Deyu, Zhang, Yanbo, Wang, Lingxiang, Li, Yaoxue, and Zhang, "Jointrec: A deep-learning-based joint cloud video recommendation framework for mobile iot," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2019.

[80] K. Muhammad, Q. Wang, D. O'Reilly-Morgan, E. Tragos, B. Smyth, N. Hurley, J. Geraci, and A. Lawlor, "Fedfast: Going beyond average for faster training of federated recommender systems," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1234–1242, 2020.

[81] M. E. Eren, L. E. Richards, M. Bhattarai, R. Yus, C. Nicholas, and B. S. Alexandrov, "Fedsplit: One-shot federated recommendation system based on non-negative joint matrix factorization and knowledge distillation," *arXiv preprint arXiv:2205.02359*, 2022.

[82] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, 2017.

[83] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3-4, pp. 285–294, 1933.

[84] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, "Atomo: Communication-efficient learning via atomic sparsification," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[85] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *arXiv preprint arXiv:1412.6115*, 2014.

[86] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," *arXiv preprint arXiv:1712.01887*, 2017.

[87] N. Guha, A. Talwalkar, and V. Smith, "One-shot federated learning," *arXiv preprint arXiv:1902.11175*, 2019.

[88] A. Kasturi, A. R. Ellore, and C. Hota, "Fusion learning: A one shot federated learning," in *International Conference on Computational Science*, pp. 424–436, Springer, 2020.

[89] Q. Li, B. He, and D. Song, "Practical one-shot federated learning for cross-silo setting," *arXiv preprint arXiv:2010.01017*, 2020.

[90] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "Fate: An industrial grade platform for collaborative learning with data protection.," *J. Mach. Learn. Res.*, vol. 22, no. 226, pp. 1–6, 2021.

[91] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan, *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[92] K. Singhal, H. Sidahmed, Z. Garrett, S. Wu, J. Rush, and S. Prakash, "Federated reconstruction: Partially local federated learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11220–11232, 2021.

[93] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose, *et al.*, "Pysyft: A library for easy federated learning," in *Federated Learning Systems*, pp. 111–139, Springer, 2021.

[94] "Paddlefl." https://github.com/PaddlePaddle/PaddleFL.

[95] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[96] Y. Xie, Z. Wang, D. Chen, D. Gao, L. Yao, W. Kuang, Y. Li, B. Ding, and J. Zhou, "Federatedscope: A comprehensive and flexible federated learning platform via message passing," *arXiv preprint arXiv:2204.05011*, 2022.

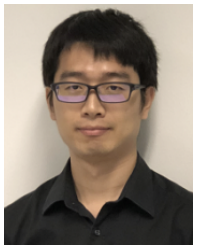
[97] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, J. Jin, H. Yu, and K. S. Ng, "Towards fair and privacy-preserving federated deep models," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2524–2541, 2020.

[98] L. Lyu, Y. Li, K. Nandakumar, J. Yu, and X. Ma, "How to democratise and protect ai: Fair and differentially private decentralised deep learning," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[99] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "Braintorrent: A peer-to-peer environment for decentralized federated learning," *arXiv preprint arXiv:1905.06731*, 2019.
[100] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip learning with linear models on fully distributed data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, 2013.
[101] T. Qi, F. Wu, C. Wu, Y. Huang, and X. Xie, "Uni-FedRec: A unified privacy-preserving news recommendation framework for model training and online serving," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, (Punta Cana, Dominican Republic), pp. 1438–1448, Association for Computational Linguistics, Nov. 2021.

Zehua Sun is currently pursuing his master's degree in the School of Software of Shandong University. He received his bachelor's degree in software engineering from the School of Software of Shandong University in 2017. His research interests include federated learning, recommendation systems and data mining.

Yonghui Xu is a full professor at Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR), Shandong University. Before that, he was a research fellow in the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), Nanyang Technological University, Singapore. He received his Ph.D. from the School of Computer Science and Engineering at South China University of Technology in 2017 and BS from the Department of Mathematics and Information Science Engineering at Henan University of China in 2011. His research areas include various topics in Trustworthy AI, knowledge graphs, expert systems and their applications in e-commerce and healthcare. He has been invited as reviewer of top journals and leading international conferences, such as, TKDE, TNNLS, IEEE Transactions on Cybernetics, Knowledge-Based System, TKDD, IJCAI and AAAI.



Yong Liu is a Senior Research Scientist at Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University (NTU). He was a Data Scientist at NTUC Enterprise, and a Research Scientist at Institute for Infocomm Research (I2R), A*STAR, Singapore. He received his Ph.D. degree in Computer Engineering from NTU in 2016 and B.S. degree in Electronic Science and Technology from University of Science and Technology of China (USTC) in 2008. His research interests include recommendation systems, natural language processing, and knowledge graph. He has been invited as a PC member of major conferences such as KDD, SIGIR, ACL, IJCAI, AAAI, and reviewer for IEEE/ACM transactions.

Wei He is a associate professor at Shandong university. He received bachelor and master degrees from computer science department of shandong university in 1994 and 1999 respectively, and received Ph.d. from engineering of shandong university in 2009. He won the progress first prize in science and technology of shandong province and the progress second prize in science and technology of shandong province, and excellent achievement in computer application. He has published more than 20 papers in the computer journal, journal of software of domestic and international journals conference. More papers were recorded by SCI, EI.

Yali Jiang is currently a Lecturer in the School of Software, Shandong University. She received her B.Sc., M.Sc. and Ph.D. degrees from Shandong University in 1999, 2002 and 2011, respectively. She is engaged in information security and cryptography research, her main research areas are public key security authentication system and lattice based cryptographic algorithm design and analysis, including cloud computing security, big data privacy protection, IoT security, etc. She has participated in the National 863 Program, Shandong Provincial Excellent Young and Middle-aged Research Award Fund, Shandong Provincial Natural Science Foundation and joint research projects of enterprises.

Fangzhao Wu is a Principal Researcher at Microsoft Research Asia, President of AAAI2022 and senior member of China Computer Society. He received the Ph.D. and B.S. degrees both from Electronic Engineering Department of Tsinghua University in 2017 and 2012 respectively. He published more than 100 academic papers and was cited nearly 3000 times He has won NLPCC2019 Excellent Paper Award, WSDM 2019 Outstanding PC and AAAI 2021 Best SPC. His research mainly focuses on responsible AI, privacy protection, natural language processing, and recommender systems. The research results have been applied in Microsoft News, Bing Ads and other Microsoft products.

LiZhen Cui (IET Fellow, IEEE Senior Member) is the Dean at School of Software, Shandong University. He is the Co-Director of Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR) and Research Center of Software & Data Engineering, Shandong University. He is the Associate Director of National Engineering Laboratory for E-Commerce Technologies. He is a Professor with the School of Software and the Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR), Shandong University, and also a Visiting Professor with Nanyang Technological University, Singapore. He was a Visiting Scholar with Georgia Tech, Atlanta, GA, USA. He received his bachelor's, M.Sc., and Ph.D. degrees from Shandong University, Jinan, China, in 1999, 2002 and 2005, respectively. He has authored or coauthored over 200 articles in journals and refereed conference proceedings. His research interests include big data management and analysis and AI theory and application.