Federated Recommendation Systems

Liu Yang[2], Ben Tan[1], Vincent W. Zheng[1(✉)], Kai Chen[2], and Qiang Yang[1,2]

[1] WeBank, Shenzhen, China
{btan,vincentz}@webank.com
[2] Hong Kong University of Science and Technology, Hong Kong, China
{lyangau,kaichen,qyang}@cse.ust.hk

Abstract. Recommender systems are heavily data-driven. In general, the more data the recommender systems use, the better the recommendation results are. However, due to privacy and security constraints, directly sharing user data is undesired. Such decentralized silo issues commonly exist in recommender systems. There have been many pilot studies on protecting data privacy and security when utilizing data silos. But, most works still need the users' private data to leave the local data repository. Federated learning is an emerging technology, which tries to bridge the data silos and build machine learning models without compromising user privacy and data security. In this chapter, we introduce a new notion of federated recommender systems, which is an instantiation of federated learning on decentralized recommendation. We formally define the problem of the federated recommender systems. Then, we focus on categorizing and reviewing the current approaches from the perspective of the federated learning. Finally, we put forward several promising future research challenges and directions.

1 Introduction

The recommender system (RecSys) plays an essential role in real-world applications. It has become an indispensable tool for coping with information overload and is a significant business for a lot of internet companies around the world. In general, the more data RecSys use, the better the recommendation performance we can obtain. The RecSys need to know as much as possible from the user to provide a reasonable recommendation. They collect the private user data, such as the behavioral information, the contextual information, the domain knowledge, the item metadata, the purchase history, the recommendation feedback, the social data, and so on. In pursuit of better recommendations, some recommender systems integrate multiple data sources from other organizations. All these informative user data is centrally stored at the database of each organization to support different kinds of recommendation services.

Liu Yang and Ben Tan are both co-first authors with equal contribution. This work was done during Liu Yang's internship at WeBank.

Q. Yang et al. (Eds.): Federated Learning, LNAI 12500, pp. 225–239, 2020.
https://doi.org/10.1007/978-3-030-63076-8_16

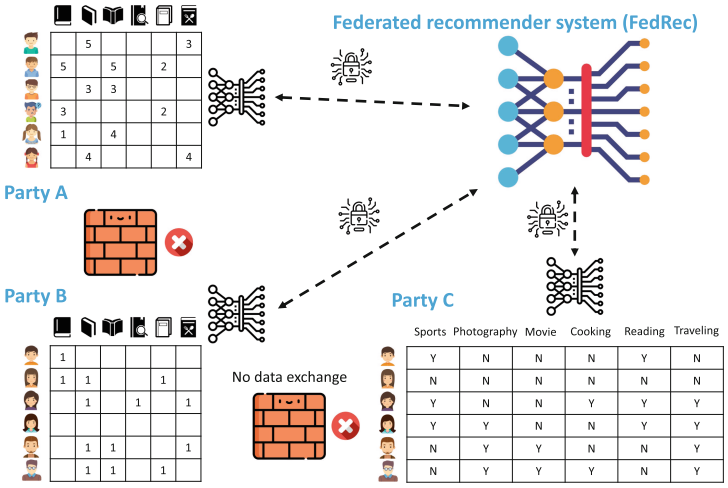


Fig. 1. The illustration of a Federated Recommender System. FedRec addresses the data silo issue and builds recommender systems without compromising privacy and security.

However, data centralization in RecSys could lead to serious privacy and security risks. For example, recommenders may unsolicitedly collect users' private data and share the data with third parties for profits. Besides, user privacy may also leak during data transmission. Moreover, in recent years, several acts protecting the privacy and security have come out, such as the General Data Protection Regulation (GDPR)[1]. The protection of privacy and security is an integral part of the RecSys. There have been pilot studies to protect user privacy and data security in the RecSys [6]. These approaches typically utilize obfuscation or cryptography techniques. Some of them add noises in different procedures of the recommendation. Others encrypt data before transmitting it to the recommender. However, most of them still need private data to leave their local data repository. How to enable recommendations across data silos securely and privately remains a challenging task.

Federated learning is an emerging technology for decentralized machine learning [13]. It protects parties' data privacy in the joint training of machine learning models. Parties could be mobile devices or organizations [26]. User private data is stored locally at each party. Only the intermediate results, *e.g.*, parameter updates, are used to communicate with other parties. Federated learning allows knowledge to be shared among multiple parties without compromising user privacy and data security. Compared with the conventional data-centralized machine learning approaches, federated learning reduces both the privacy risks and costs. This area has been paid more and more attention recently, in both academia and industry.

[1] GDPR is a regulation in EU law on data protection and privacy in the European Union and the European Economic Area. https://gdpr.eu/.

In this chapter, we introduce a new notion of Federated Recommender System (FedRec), as shown in Fig. 1. Compared to the conventional RecSys, FedRec primarily protects user privacy and data security through decentralizing private user data locally at each party. According to the data structure of recommendation tasks, we conclude the FedRec categorization. Moreover, we illustrate with typical real-world scenarios for each categorization and explain the existing solutions according to each scenario. When building real-world FedRec, people could encounter different challenges. On the one hand, the prevalent RecSys is so complicated and continuously improved with the state-of-the-art machine learning algorithms. On the other hand, there exist many open questions as new challenges that recommendations bring to federated learning. For these challenges, we categorize them at two levels, *i.e.*, algorithm-level and system-level, and discuss the solutions in the existing works.

Overall, our contributions are threefold: 1) We propose the notion of FedRec and provide a categorization method according to the data structure of the RecSys; 2) We make a first survey on the existing works about FedRec in terms of each category; 3) We give a discussion about the challenges that exist in the FedRec.

2 Federated Recommender System

To protect privacy in RecSys, we introduce the new notion of the Federated Recommender System (FedRec). FedRec adopts the data decentralization architecture. Parties keep their private data locally and train recommendation models collaboratively in a secure and privacy-preserving way. Each party could be a RecSys or data provider. A RecSys party basically contains the rating information, the user profiles, and the item attributes. A data provider party owns more user profiles or item attributes. In the following parts of this section, firstly, we define the FedRec. Secondly, we conclude the categories of FedRec in terms of its data structure. In each category, we give the problem definition, describe typical real-world scenarios, and discuss corresponding related works.

2.1 Definition of Federated Recommender System

Define N parties, K of whom are recommender systems, *i.e.*, $\mathcal{G}_{k\in\{1,...,K\}} = \{\mathcal{U}_k, \mathcal{I}_k, \boldsymbol{R}_k, \boldsymbol{X}_k, \boldsymbol{X}_k^{'}\}$. $\mathcal{U}_k = \{u_k^1, u_k^2, ..., u_k^{n_k}\}$ and $\mathcal{I}_k = \{i_k^1, i_k^2, ..., i_k^{m_k}\}$ stand for the user set and item set respectively. $\boldsymbol{R}_k \in \mathbb{R}^{n_k \times m_k}$ is the rating matrix. $\boldsymbol{X}_k \in \mathbb{R}^{n_k \times d_k}$ and $\boldsymbol{X}_k' \in \mathbb{R}^{m_k \times d_k^{'}}$ represent the user profiles and item attributes respectively. The other H parties are data providers containing user profiles, *i.e.*, $\mathcal{D}_{h\in\{1,...,H\}} = \{\mathcal{U}_h, \boldsymbol{X}_h\}$, or item attributes, *i.e.*, $\mathcal{D}_h = \{\mathcal{I}_h, \boldsymbol{X}_h^{'}\}$.

Definition 1. *FedRec aims to collaboratively train recommendation model(s) among multiple parties without direct access to the private data of each other:*

$$\arg\min_{\tilde{\boldsymbol{\theta}}_k} \sum_{k=1}^{K} L(\boldsymbol{R}_k, f^{fed}_{\tilde{\boldsymbol{\theta}}_k}(\mathcal{U}_k, \mathcal{I}_k | \mathcal{G}_k, z(\mathcal{G}_{k' \in \{1,\ldots,K\}\setminus\{k\}}), z(\mathcal{D}_{h\in\{1,\ldots,H\}}))), \tag{1}$$

where $L(\cdot,\cdot)$ is a loss function, $f^{fed}_{\tilde{\boldsymbol{\theta}}_k}(\cdot,\cdot)$ is the prediction model for the kth FedRec, and $z(\cdot)$ stands for the data processing technique that exchanges intermediate results between parties instead of the raw data.

We expect that the performance of FedRec is better than the performance of each RecSys training with its own data, while very close to the performance of simply aggregating all parties' data together without considering data privacy and security:

$$|V(f^{fed}_{\tilde{\boldsymbol{\theta}}_k}) - V(f_{\boldsymbol{\theta}_k})| > \delta \quad and \quad |V(f^{sum}_{\tilde{\boldsymbol{\theta}}_k}) - V(f^{fed}_{\tilde{\boldsymbol{\theta}}_k})| \leq \epsilon, \tag{2}$$

where $\delta \in \mathbb{R}^+$, $\epsilon \in \mathbb{R}^*$, and $V(\cdot)$ is the evaluation function utilized by RecSys. The prediction model $f_{\boldsymbol{\theta}_k}$ is obtained via separately training the model with the recommender's own data:

$$\arg\min_{\boldsymbol{\theta}_k} L(\boldsymbol{R}_k, f_{\boldsymbol{\theta}_k}(\mathcal{U}_k, \mathcal{I}_k | \mathcal{G}_k)). \tag{3}$$

The recommender $f^{sum}_{\tilde{\boldsymbol{\theta}}_k}$ is obtained via training the recommendation model with all parties' data simply consolidated together:

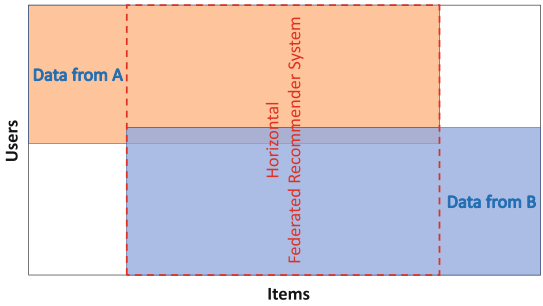
$$\arg\min_{\tilde{\boldsymbol{\theta}}_k} \sum_{k=1}^{K} L(\boldsymbol{R}_k, f^{sum}_{\tilde{\boldsymbol{\theta}}_k}(\mathcal{U}_k, \mathcal{I}_k | \mathcal{G}_{k' \in \{1,\ldots,K\}}, \mathcal{D}_{h\in\{1,\ldots,H\}})). \tag{4}$$

2.2 Categorization of Federated Recommender System

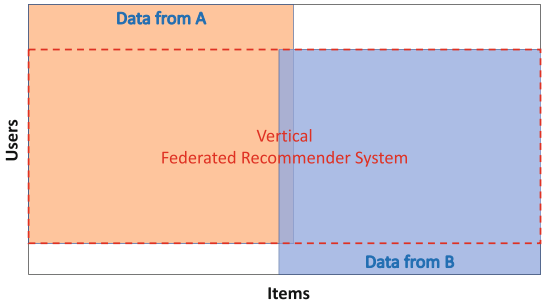
We categorize the typical scenarios of FedRec according to the data structure of the RecSys. RecSys mainly consists of two types of entities, *i.e.*, users and items. Shared users or items naturally connect the parties of FedRec. As shown in Fig. 2(a), 2(b) and 2(c), we divide FedRec into **Horizontal FedRec**, **Vertical FedRec** and **Transfer FedRec** according to the sharing situation of users and items. In this subsection, we describe the details of each category and provide typical scenarios for illustration. Related works about FedRec are discussed under the corresponding categories.

Horizontal Federated Recommender System. As shown in Fig. 2(a), the horizontal FedRec is introduced where items are shared, but users are different between parties. Under this setting, the parties could be in the form of individual users or sets of users.

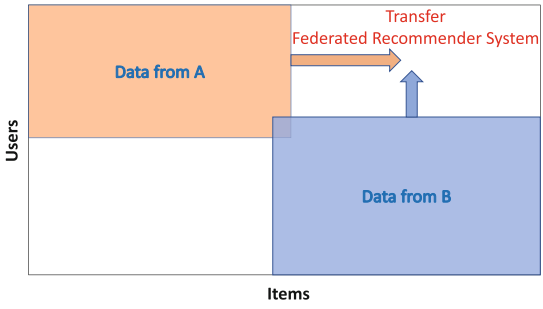
Definition 2. *Given N parties and each party contains a set of users or an individual user, i.e., $\mathcal{G}_{i\in\{1,\ldots,N\}} = \{\mathcal{U}_i, \mathcal{I}_i, \boldsymbol{R}_i, \boldsymbol{X}_i, \boldsymbol{X}_i^{'}\}$, $\mathcal{U}_i \neq \mathcal{U}_j, \mathcal{I}_i = \mathcal{I}_j, \forall \mathcal{G}_i, \mathcal{G}_j, i \neq j$, horizontal FedRec aims to train a recommender model by integrating users' historical behaviors on shared items from different parties, without revealing user's privacy:*



(a) Horizontal FedRec. Items are shared, but users are different between parties.



(b) Vertical FedRec. Users are shared, but items are different between parties.



(c) Transfer FedRec. Neither users nor items are shared between parties.

Fig. 2. The categorization of federated recommender systems.

$$\arg\min_{\tilde{\theta}} \sum_{k=1}^{N} L(\boldsymbol{R}_k, f_{\tilde{\theta}}^{fed}(\mathcal{U}_k, \mathcal{I}_k | z(\mathcal{G}_{k' \in \{1,\ldots,K\} \setminus \{k\}}))). \tag{5}$$

Typical Scenario of Horizontal FedRec. As shown in Fig. 3, users enjoy a personalized movie recommendation service provided by a movie recommender. But they do not want their private data to be collected. Inside the recommender, to preserve the data privacy of each user, we prefer to have the training data distributed on the local devices. Each user device is regarded as a party containing the rating information between one specific user and all items. Those devices can build a RecSys together to achieve both personalization and privacy requirements.

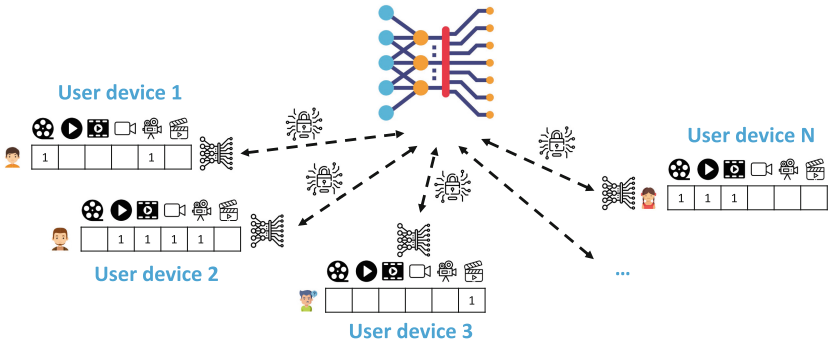


Fig. 3. The typical scenario of Horizontal FedRec. Each party is the device of an individual user. They share the same items but have different users.

Several current works focus on this scenario. [4] proposed a Federated Collaborative Filter (FCF) algorithm based on matrix factorization. In traditional RecSys, the matrix factorization algorithms work by decomposing the user-item rating matrix into the product of two lower matrices, *i.e.*, the user latent factors matrix and the item latent factors matrix. In the FedRec setting, FCF introduces a central server to maintain the shared item latent factors, while the user latent factors are stored locally on each device. In each iteration of training, the server distributes the item latent factors to each party. Then, parties update their user latent factor by local rating data and send the item latent factor updates back to the server for aggregation. During the training process, only the model updates are transmitted. No users' private data is collected. To avoid interaction with a third-party central server, [8] provided a fully-decentralized matrix factorization approach without central server. Parties communicate directly with each other to update the model. Besides, [5] proposed another decentralized method of matrix factorization. Local models are exchanged in the neighborhood, not with an arbitrary party. This approach further improves the performance of the algorithm. Moreover, [2] proposed a federated meta-learning framework for the recommendation. It regards the recommendation for each user as one separate task and

designs a meta-learner to generate each task parameters. This framework utilizes a support set to generate the recommendation model on each party and computes the loss gradient on a query set. In addition, [11] offered another federated meta-learning algorithm for recommendation. It needs no separate support and query sets. The latter one performs relatively well within considerably fewer episodes in the experiments. Furthermore, [16] proposed a distributed factorization machine algorithm, which is known as DiFacto. It addresses the efficiency problem when scaling to large amounts of data and large numbers of users.

All the works mentioned above do not adopt other security methods. They own a privacy advantage compared to the data-centralized approaches. However, privacy risks still exist when transferring plain-text model parameters. A few works further utilize the obfuscation methods based on the data-centralized architecture. The obfuscation methods contain the anonymization, the randomization, and the differential privacy techniques. Among them, the differential privacy (DP) technique is a popular method. It incorporates random noise to anonymize data and protect privacy. It also offers a provable privacy guarantee and low computation costs. [19] proposed the private social recommendation (PrivSR) algorithm by utilizing the DP technique. This approach is based on a matrix factorization method with the friends-impacting regularizer. Since an inference attack can be conducted from the contribution of one particular user, the DP noise is added into the objective function to perturb the individual's involvement. [14] proposed the federated online learning to the rank algorithm by using users' online feedback. It trains the ranking model on local devices in a way that respects the users' privacy and utilizes the DP technique to protect model privacy on the server. DP noise is injected into the communicated values before transmitted to the server, which is different from the PrivSR. However, DP also introduces additional noise. These works involve a trade-off between performance and privacy.

To avoid performance loss, the other works make use of the cryptography techniques instead of the obfuscation methods. The cryptography methods contain homomorphic encryption (HE), secure multi-party computation (SMC) protocols, *etc.* They guarantee good security protection without the loss of accuracy. HE techniques have been widely utilized because it allows computing over encrypted data without access to the secret key. [1] proposed the secure federated matrix factorization algorithm (FedMF) with HE schemes. Each user encrypts the item latent factor updates with HE before transmitting. Besides, the item latent factor is aggregated and maintained by the central server under the encrypted form. No information of latent factors and updates will be leaked to the introduced server. [15] provided an efficient privacy-preserving item-based collaborative filtering algorithm. An SMC protocol is designed to compute the summation of private values of each party without revealing them. Then with this protocol, the PrivateCosine and PrivatePearson algorithm are implemented to calculate the item correlations. Final recommendations are generated using the correlations without revealing privacy.

Vertical Federated Recommender System. The vertical FedRec is shown in Fig. 2(b). Two parties shared the same user set, but different item set or feature spaces. Under this setting, the parties could be different recommenders or data providers.

Definition 3. *Given two parties, one of whom is a RecSys, i.e.,* $\mathcal{G}_A = \{\mathcal{U}_A, \mathcal{I}_A, \boldsymbol{R}_A, \boldsymbol{X}_A, \boldsymbol{X}_A^{'}\}$*, the other one is a data provider or the other recommender. Taking a data provider as an example, we have* $\mathcal{D}_B = \{\mathcal{U}_B, \boldsymbol{X}_B\}$*, and* $\mathcal{U}_A = \mathcal{U}_B = \mathcal{U}$*. The vertical FedRec aims to train a recommender model by exploiting the side information of users from the data provider or other recommenders. The training process is completed in a secure and privacy-preserving manner:*

$$\arg\min_{\tilde{\boldsymbol{\theta}}} L(\boldsymbol{R}_A, f_{\tilde{\boldsymbol{\theta}}}^{fed}(\mathcal{U}, \mathcal{I}_A, z(\boldsymbol{X}_B)|z(\mathcal{D}_B))). \tag{6}$$

Typical Scenario of Vertical FedRec. As illustrated in Fig. 4, the participants contain a RecSys, and a data provider. For instance, one party is a book RecSys and the other party is a data provider who can offer rich user profiles. They have a large set of users in common. The vertical FedRec helps to build a better book recommendation service without data privacy leakage.

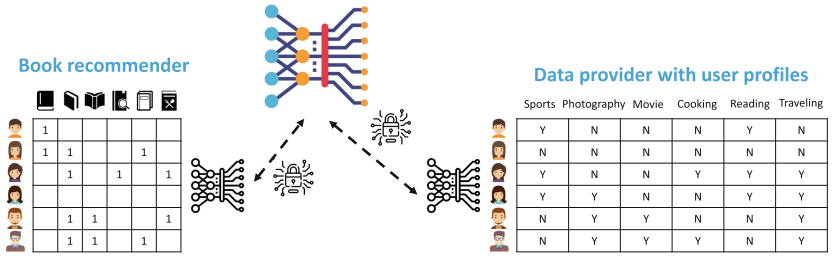


Fig. 4. The typical scenario of Vertical FedRec. One party is a book recommender, while the other one is a data provider with user profiles. They share the same users but have different items.

Several existing works have been designed for such a feature distributed learning problem where party A and B hold different feature sets. [10] proposed an asynchronous stochastic gradient descent algorithm. Each party could use an arbitrary model to map its local features to a local prediction. Then local predictions from different parties are aggregated into a final output using linear and nonlinear transformations. The training procedure of each party is allowed to be at various iterations up to a bounded delay. This approach does not share any raw data and local models. Therefore, it has fewer privacy risks. Besides, for a higher level of privacy, it can easily incorporate the DP technique. Similar to horizontal FedRec, there are also works that further utilize cryptography techniques. [3] presented a secure gradient-tree boosting algorithm. This algorithm

adopts HE methods to provide lossless performance as well as preserving privacy. And [7] proposed a secure linear regression algorithm. MPC protocols are designed using garbled circuits to obtain a highly scalable solution.

Parties of vertical FedRec could also be two recommenders with different item sets. For instance, a movie RecSys and a book RecSys have a large user overlapping but different items to recommend. It is assumed that users share a similar taste in movies with books. With FedRec, the two parties want to train better recommendation algorithms together in a secure and privacy-preserving way. [21] proposed a secure, distributed item-based CF method. It jointly improves the effect of several RecSys, which offer different subsets of items to the same underlying population of users. Both the predicted ratings of items and their predicted rankings could be computed without compromising privacy nor predictions' accuracy.

Transfer Federated Recommender System. As Shown in Fig. 2(c), in the transfer federated recommender system, neither users nor items are shared between parties. In most cases, the parties are different recommender systems.

Definition 4. *Given two parties, who are different recommender systems, i.e.,* $\mathcal{G}_S = \{\mathcal{U}_S, \mathcal{I}_S, \boldsymbol{R}_S, \boldsymbol{X}_S, \boldsymbol{X}_S^{'}\}$ *as the source-domain party,* $\mathcal{G}_T = \{\mathcal{U}_T, \mathcal{I}_T, \boldsymbol{R}_T, \boldsymbol{X}_T, \boldsymbol{X}_T^{'}\}$ *as the target-domain party, and* $\mathcal{U}_S \neq \mathcal{U}_T, \mathcal{I}_S \neq \mathcal{I}_T$*. Generally,* $\boldsymbol{R}_S$ *contains much more rating information than* $\boldsymbol{R}_T$*. Transfer FedRec aims to train a recommender model by transferring knowledge from the source-domain party to the target-domain party, without revealing user privacy:*

$$\arg\min_{\tilde{\boldsymbol{\theta}}} \sum_{k \in \{S,T\}}^{N} \lambda_k L(\boldsymbol{R}_k, f_{\tilde{\boldsymbol{\theta}}_k}^{fed}(\mathcal{U}_k, \mathcal{I}_k | z(\mathcal{G}_{k' \in \{S,T\} \setminus \{k\}}))), \tag{7}$$

where λ_k *is the weight for balancing the performance of two parties.*

Typical Scenario of Transfer FedRec. As shown in Fig. 5, a popular book recommender system in region A wants to help another new movie recommender system in region B to collaboratively learn a movie recommendation model. In this case, both users and items of the two parties are different.

Since both users and items are different between parties, it's challenging to construct a federated recommender system directly. However, federated transfer learning [20] offers a feasible scheme. A limited set of co-occurrence samples is used as a "bridge" to transfer knowledge from the source domain to the target domain. At first, parties update their neural networks using local data. Then, they together optimize the loss on the co-occurrence samples. The secret sharing technique is adopted to design a secure and efficient algorithm. Similarly, this algorithm can be applied in the transfer FedRec scenario via co-occurrence users or items.

As we have reviewed, horizontal FedRec managing RecSys across individuals or user sets is important and attracts lots of research attention. Vertical FedRec and transfer FedRec building RecSys among organizations are typical tasks in recommendation businesses. Yet, vertical and transfer FedRec are still underexplored areas with a lot of opportunities.

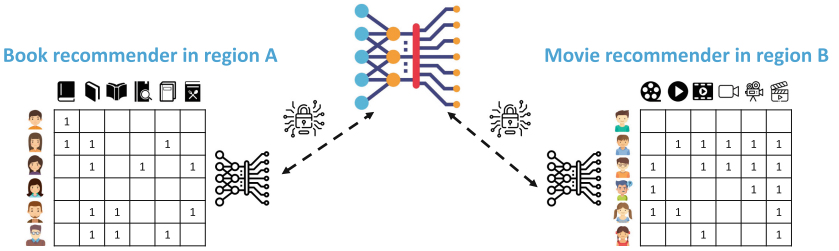


Fig. 5. The typical scenario of Transfer FedRec. One party is a book recommender, while the other one is a movie recommender in the different region. They share neither users nor items.

3 Challenges and Future Directions

In this section, we discuss the possible challenges when constructing FedRec. An industrial FedRec is more than the recommendation algorithms. It should also contain a comprehensive design of the system. Therefore, our discussion about the challenges is divided into the algorithm level and the system level. At the algorithm level, we discuss the possible difficulties of designing different federated recommender algorithms using popular models in the current recommendation area. Meanwhile, at the system level, we list several critical challenges of designing FedRec in terms of the characteristics of RecSys. Besides, we discuss current solutions for all the problems mentioned.

3.1 Algorithm-Level Challenges

Federated Deep Model for Recommendation. Deep recommendation models could cause severe problems when utilizing non-linear activation functions. Complex functions, *e.g.*, tanh and relu activation functions, are not well supported by HE. This limitation seriously affects the deep models' application in FedRec. For solving this problem, [9] utilized low degree polynomials as the approximation of activation functions. There exists a trade-off between the model performance and the degree of polynomial approximation. This work provides the polynomial approximations with the lowest degrees as possible for three common activation functions, *i.e.*, ReLU, Sigmoid, and Tanh.

Federated Graph Model for Recommendation. Protecting the privacy of structure information in the graph is the main difficulty of federalizing the graph-based models. The Graph-based models for recommendations utilize the relation information between users and items to enrich their representations. The relation information is more complicated than the feature information. Different secure methods are adopted to protect the privacy of the graph in the present works. For instance, [22] utilized a graph sampling method to improve both the efficiency and privacy of the privacy-preserving association rules mining approaches.

Users decide locally and privately, whether to become part of the sample. They are in control of their data and maintain sensitive item sets. Users with common interests are represented by the user groups. Neither the recommender nor other users know about the specific item sets of one particular user.

Federated Reinforcement Learning Model for Recommendation. The challenge of federalizing reinforcement learning models is to delicately design the state, action, and reward to catch the instant user interest and decide what to share among parties. Although reinforcement learning has an vital role in RecSys, its application in FedRec is still underexplored. Yet, there have been several works about federated reinforcement learning applied in other areas. [18] provided the lifelong federated reinforcement learning architecture for robots to perform lifelong learning of navigation in cloud robotic systems. A knowledge fusion algorithm and transfer learning approach are designed to fuse the robots' prior knowledge and make robots quickly adapt to the new environments.

3.2 System-Level Challenges

Design of Recall and Ranking in FedRec. The main challenge in the system level is to design privacy-preserving recall and ranking procedures with real-time feedback. RecSys sequentially adopts these two procedures to obtain the final recommendations. Conventionally, RecSys centrally collects the users' private data, and these two steps are designed to carry out on the central server. However, concerning user privacy, FedRec should modify the original design.

We discuss two extreme cases. The first case is server-side recall and participant side ranking. Firstly, each party sends the encrypted "noisy" model parameters to the server. Then recall procedure is carried out on the server-side. The resulted top-N items are then sent back to each party. Then, the ranking procedure is carried out at each party. There is a chance of privacy leakage because the server knows the exact results of recall. Several works have tried to address this problem. For example, [12] utilizes the private stream searching technique to obtain the result delivery without exposing its contents. The second case is participant-side recall and ranking. The server sends all item attributes and content to each party. Then, the whole recall and ranking procedures are carried out on the participant side. This design contains no leak of user privacy but will result in copious communication costs. Besides, it requires lots of computation resources and local storage for each party. However, with the fast development of 5G technology[2] in recent years, the communication cost problem could be alleviated to some extent.

Communication Cost in FedRec. Communication cost is one of the major problems that affect the performance of federated learning. Because of the

[2] 5G is the fifth generation wireless technology for digital cellular networks.

high-dimensional features and real-time requirement of RecSys, the communication cost problem is much serious in FedRec. Pilot works have tried to compress the high-dimensional features. Communication-mitigated federated learning (CMFL) [23] assumes that some local optimizations are not helpful to the global convergence, therefore reducing the total bits transferred in each update via data compression. CMFL identifies irrelevant updates made by each party and precludes them from updating. In more detail, it provides clients with feedback information regarding the global tendency of model updating. Each client checks if its update aligns with this global tendency and is relevant enough to model improvement.

Flexibility and Scalability in FedRec. As the number of parties keeps increasing, the challenge is to design better model-parallel and model-updating scheduling schema to guarantee convergence of the FedRec models. Many of the federated learning systems adopt a synchronous client-server architecture [17,25], which is inflexible and unscalable. In the RecSys, millions of users consume the recommendation services. Too many parties checking in at the same time can congest the network on the central server. It is hard to guarantee that all parties could participate in the whole process of federated training. As a result, the performance of the federated model severely suffers. Various solutions have been designed to address this challenge. Based on the client-server architecture, [25] proposed a new asynchronous federated optimization algorithm. The central server immediately updates the global model whenever receiving a local model from one arbitrary party. And the communication between parties and the central server is non-blocking. Abandoning the client-server architecture, [8] proposed the gossip learning algorithm, which can be regarded as a variant of federated learning with a fully decentralized architecture. Parties directly communicate with each other for collaborative training.

Non-IID Data in FedRec. The "long tail" phenomenon is common in RecSys and makes the non-IID data problem inevitable in FedRec. The performance of federated learning severely degrades due to the highly skewed non-IID. As the distance between the data distribution at each party becomes more significant, the accuracy of the model decreases accordingly. To alleviate the non-IID problem, a data-sharing strategy has been proposed by reducing the distance [27]. This approach shares a global data set of a uniform distribution over all classes among parties. In the initialization stage, a warm-up model, trained on the globally shared data, is distributed to each party instead of a random model. Then, the shared data and private data are used together to train the local model at each party.

Malicious Participants Cooperation in FedRec. In reality, the parties in the RecSys have a high probability of being untrustworthy [6]. These parties do not follow the frequently used assumption that both the participants and the

central server are honest-but-curious. They may behave incorrectly in gradient collecting or parameter updating, while the servers may be malicious as well. Therefore, the honest parties could have a privacy leak in these scenarios. Among the existing solutions, [24] proposed the DeepChain as one possible solution, which combines the Blockchain[3] and federated learning. Based on the Blockchain technique, DeepChain provides a value-driven incentive mechanism to force the participants to behave correctly, which preserves the privacy of local gradients and guarantees the auditability of the training process. Smart contracts, *i.e.*, the trading contract and the processing contract, are utilized to guide the secure training process.

4 Conclusion

In this chapter, we investigate the user privacy and data security in RecSys. The risk of security and privacy is mainly raised by the central collection and storage of users' private data. Considering the growing privacy concern and related acts like GDPR, we introduce the new notion of the federated recommender system (FedRec). With FedRec, multiple parties could collaboratively train better recommendation models with users' private data maintained locally at each party. We categorize FedRec according to the data structure of RecSys. Many existing works focus on the horizontal FedRec scenarios, while the vertical and transfer FedRec have been given less attention. Besides, many current prevailing recommendation algorithms have not been applied in FedRec, either. Therefore, FedRec is a promising direction with huge potential opportunities. In our future work, we will concentrate on implementing an open-source FedRec library with rich recommendation algorithms and overcoming the system-level challenges as they arise.

References

1. Chai, D., Wang, L., Chen, K., Yang, Q.: Secure federated matrix factorization. arXiv preprint arXiv:1906.05108 (2019)
2. Chen, F., Dong, Z., Li, Z., He, X.: Federated meta-learning for recommendation. arXiv preprint arXiv:1802.07876 (2018)
3. Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., Yang, Q.: SecureBoost: a lossless federated learning framework. arXiv preprint arXiv:1901.08755 (2019)
4. Ammad-ud din, M., et al.: Federated collaborative filtering for privacy-preserving personalized recommendation system. arXiv preprint arXiv:1901.09888 (2019)
5. Duriakova, E., et al.: PDMFRec: a decentralised matrix factorisation with tunable user-centric privacy. In: Proceedings of the 13th ACM Conference on Recommender Systems (2019)

[3] A blockchain is a growing list of records, called blocks, that are linked using cryptography.

6. Friedman, A., Knijnenburg, B.P., Vanhecke, K., Martens, L., Berkovsky, S.: Privacy aspects of recommender systems. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 649–688. Springer, Boston, MA (2015). https://doi.org/10.1007/978-1-4899-7637-6_19
7. Gascón, A., et al.: Secure linear regression on vertically partitioned datasets. IACR Cryptology ePrint Archive (2016)
8. Hegedűs, I., Danner, G., Jelasity, M.: Decentralized recommendation based on matrix factorization: a comparison of gossip and federated learning. In: Cellier, P., Driessens, K. (eds.) ECML PKDD 2019. CCIS, vol. 1167, pp. 317–332. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-43823-4_27
9. Hesamifard, E., Takabi, H., Ghasemi, M.: CryptoDL: deep neural networks over encrypted data. arXiv preprint arXiv:1711.05189 (2017)
10. Hu, Y., Niu, D., Yang, J., Zhou, S.: FDML: a collaborative machine learning framework for distributed features. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2019)
11. Jalalirad, A., Scavuzzo, M., Capota, C., Sprague, M.: A simple and efficient federated recommender system. In: Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (2019)
12. Jiang, J., Gui, X., Shi, Z., Yuan, X., Wang, C.: Towards secure and practical targeted mobile advertising. In: Proceedings of the 11th International Conference on Mobile Ad-hoc and Sensor Networks (2015)
13. Kairouz, P., et al.: Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977 (2019)
14. Kharitonov, E.: Federated online learning to rank with evolution strategies. In: Proceedings of the 20th ACM International Conference on Web Search and Data Mining (2019)
15. Li, D., et al.: An algorithm for efficient privacy-preserving item-based collaborative filtering. Future Gener. Comput. Syst. **55**, 311–320 (2016)
16. Li, M., Liu, Z., Smola, A.J., Wang, Y.X.: DiFacto: distributed factorization machines. In: Proceedings of the 9th ACM International Conference on Web Search and Data Mining (2016)
17. Li, Q., Wen, Z., He, B.: Federated learning systems: vision, hype and reality for data privacy and protection. arXiv preprint arXiv:1907.09693 (2019)
18. Liu, B., Wang, L., Liu, M., Xu, C.: Lifelong federated reinforcement learning: a learning architecture for navigation in cloud robotic systems. IEEE Robot. Autom. Lett. **4**, 4555–4562 (2019)
19. Meng, X., et al.: Personalized privacy-preserving social recommendation. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence (2018)
20. Sharma, S., Chaoping, X., Liu, Y., Kang, Y.: Secure and efficient federated transfer learning. arXiv preprint arXiv:1910.13271 (2019)
21. Shmueli, E., Tassa, T.: Secure multi-party protocols for item-based collaborative filtering. In: Proceedings of the 11st ACM Conference on Recommender Systems (2017)
22. Wainakh, A., Grube, T., Daubert, J., Mühlhäuser, M.: Efficient privacy-preserving recommendations based on social graphs. In: Proceedings of the 13th ACM Conference on Recommender Systems (2019)
23. Wang, L., Wang, W., Li, B.: CMFL: mitigating communication overhead for federated learning. In: Proceedings of the 39th IEEE International Conference on Distributed Computing Systems (2019)

24. Weng, J.S., Weng, J., Li, M., Zhang, Y., Luo, W.: DeepChain: auditable and privacy-preserving deep learning with blockchain-based incentive. IACR Cryptology ePrint Archive (2018)
25. Xie, C., Koyejo, S., Gupta, I.: Asynchronous federated optimization. arXiv preprint arXiv:1903.03934 (2019)
26. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. **10**, 1–19 (2019)
27. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-IID data. arXiv preprint arXiv:1806.00582 (2018)