doi: 10.1093/comjnl/bxab025

A Federated Learning Approach for Privacy Protection in Context-Aware Recommender Systems

Waqar Ali[1,2], Rajesh Kumar[1], Zhiyi Deng[1], Yansong Wang[1] and Jie Shao[1,3,*]

[1]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*

[2]*Faculty of Information Technology, The University of Lahore, Lahore 54000, Pakistan*
[3]*Sichuan Artificial Intelligence Research Institute, Yibin 644000, China*
**Corresponding author: shaojie@uestc.edu.cn*

Privacy protection is one of the key concerns of users in recommender system-based consumer markets. Popular recommendation frameworks such as collaborative filtering (CF) suffer from several privacy issues. Federated learning has emerged as an optimistic approach for collaborative and privacy-preserved learning. Users in a federated learning environment train a local model on a self-maintained item log and collaboratively train a global model by exchanging model parameters instead of personalized preferences. In this research, we proposed a federated learning-based privacy-preserving CF model for context-aware recommender systems that work with a user-defined collaboration protocol to ensure users' privacy. Instead of crawling users' personal information into a central server, the whole data are divided into two disjoint parts, i.e. user data and sharable item information. The inbuilt power of federated architecture ensures the users' privacy concerns while providing considerably accurate recommendations. We evaluated the performance of the proposed algorithm with two publicly available datasets through both the prediction and ranking perspectives. Despite the federated cost and lack of open collaboration, the overall performance achieved through the proposed technique is comparable with popular recommendation models and satisfactory while providing significant privacy guarantees.

Keywords: federated learning; privacy protection; context-aware recommender systems; collaborative filtering; reliable recommendations

Received 22 November 2020; Revised 19 February 2021; Accepted 10 March 2021
Handling editor: Xiaofeng Zhu

1. INTRODUCTION

The exponential web growth during the last two decades has drastically changed the living style of billions of people around the globe. The influence is not limited to individual behavior, yet it spans to various activities such as e-commerce, education, e-governance and general social interactions. The increasing popularity of recommender systems as a leading tool to filter relevant information and avoid information overload issues has attracted a huge number of people. This results in gigantic amounts of data generation reflecting users' personal preferences and can be utilized in a collaborative mode for reliable recommendation services. The more personal information available to the server, the more accurate and relevant recommendations can be generated for a particular user in context-aware recommender systems. The data providers, on the other side, have to guarantee the privacy concerns of individuals due to strict regulatory requirements and users' satisfaction. Therefore, data providers anonymize users' identities and other privacy-sensitive data attributes. Still, in presence of statistical data inferring algorithms privacy risks are everywhere in our digital society. For example, during the Netflix Prize, the data publisher completely anonymized user identities for privacy protection. Later, in 2007 a group of researchers at The University of Texas came up with a de-anonymization model [1]

that successfully recovered the anonymized identities through a simple statistical technique. In particular, results of the de-anonymization algorithm show that the data themselves contain potential information for the breach of privacy to reidentify anonymized users. Privacy risks in recommender systems have brought considerable challenges for both users and service providers.

With abundant social and economic activities switched to online platforms, the significance of data protection and privacy regulations is essential and highly demanded worldwide. The use and sharing of personal information to third parties for recommendation services bring a lot of technical challenges for service providers. Consequently, several data protection frameworks have been proposed to anonymize users' preferences during the recommendation process. The goal is to remove or encrypt sensitive data attributes before sharing with third-party servers while ensuring the data utility for recommendation services. A vast amount of literature has been published on privacy protection for recommender systems during the last two decades [2–5]. Primarily, the traditional techniques can be classified into three broad categories, namely data anonymization [3, 4], cryptographic techniques [5, 6] and differential privacy [7, 8]. All but not a few have their own technical limitations and obviously a huge computational overhead as well.

Furthermore, the collaborative filtering (CF)-based recommender systems force an edge device to send user data to the cloud where a recommendation model utilizes this data for training. Despite the presence of data anonymization methods, differential privacy and encryption techniques, certain privacy breaches may occur while moving data from edge devices to a central server. Also, this cycle repeats to avoid model outdates and mitigate concept drift. From the privacy point of view, this is not desired by the end user to expose all his/her data to an unknown third-party system running on the cloud. Although the conventional privacy-preserving techniques played a vital role for privacy protection, on the other side, such techniques add an unbearable computational overhead on top of the recommendation process.

Federated learning [9] is recently explored as a promising machine learning paradigm to mutually gain generalizable knowledge from the data distributed over several edge points without physically transferring the users' rating preference data to the central server. Inherently, it is a collaborative learning framework where multiple collaborators train their local models at the same time and then send their model weights to a central server to be aggregated into a global model. The server aggregates the weights collected from all collaborators and sends back to the collaborating clients for their model updates. This process of weight aggregation and transmission repeats for a predefined number of epochs or for a loss function to meet a certain level of accuracy condition. Finally, the global model converges in collaboration with local models. Its inherent property, i.e. keeping personal data on the client side and sending only updated model weights to the server side, is highly suitable for privacy-preserving recommender systems.

In this research, we proposed a federated learning-based privacy-preserving CF framework for context-aware recommender systems named Fed-CARS. It works with a user-defined collaboration protocol (UDCP) to guarantee user privacy. Instead of crawling users' personal information into a central server, the whole data are divided into two disjoint parts, i.e. personal data and sharable public data. The goal is to exploit the inbuilt power of federated learning for training a central CF model with a set of user-wise local models in a distributed fashion. Definitely, if the users' personal data do not leave the edge device, the model is free from the privacy concerns and third-party reliability issues. Additionally, on top of the federated learning, we have defined a flexible collaboration protocol UDCP that allows the user to tune the degree of collaboration with the global model. To summarize, this work makes the following major contributions:

- To the best of our knowledge, this is a first attempt to exploit federated learning for context-aware recommender systems, which also maintains reasonable level of accuracy and ranking performance for recommendation while ensuring privacy by means of federated architecture.
- We propose a novel method Fed-CARS, utilizing collaborative intelligence by means of a federated learning framework and additional context information in a distributed fashion. The proposed model trains a set of local models with only users' self-maintained data, server-provided preference weights and contextual information.
- Another agility of the Fed-CARS model is a flexible UDCP, by which each user can locally control the degree of collaboration with the local and global recommendation models.

The rest of the article is organized as follows: Section 2 presents an overview of the existing work done for privacy protection and federated learning in simple and context-aware recommender systems. Section 3 illustrates the proposed Fed-CARS model and UDCP protocol by specifying the preliminary concepts and formal definitions. Section 4 presents the experimental setup and result. Finally, Section 5 concludes with possible future extensions of this research work.

2. RELATED WORK

Context-aware recommender systems have gained a significant amount of attention in recent years. It states that a users' choice not only depends on like-minded neighbors but also strongly connects to the situation or context in which a particular item is selected. Context-based recommendations have very strong inspiration from to-the-point item filtering that leads toward

quality recommendations. Several types of additional context information such as time, location, social interaction have been taken into account. As an early milestone, Adomavicius *et al*. [10] introduced the significance of context information and a multidimensional approach to incorporate the contextual information in the recommendation process. During the Netflix Prize, Koren [11] highlighted the long-ranging temporal dynamics for the Netflix dataset and suggested to integrate temporal features into the recommendation model. Recently, many researchers [12–18] have utilized the strength of deep learning for context-based recommendations. Apart from the significance of the context information in deep learning models, privacy breaches are the major downside from users' point of view. Privacy concerns in recommender systems have been highlighted in recent literature [5–7, 19, 20]. The users' privacy concerns are regarding the fact that if the recommender system has full access to all the contextual attributes then it might have a serious consequence if an attacker can get or infer this information collectively. Knijnenburg and Kobsa [21] discussed privacy breach possibilities for contextual attributes and proposed a unified approach as a general privacy protection framework.

Federated learning by Google [22, 23] came with a goal to train a high-quality central model by means of several local models, each of which resides on the client side. Inherently, it is a collaborative learning framework where multiple collaborators train their models at the same time and then send their model updates to a central server for aggregation into a global model. The privacy-preserving nature of federated learning attracted a lot of research attention in recent years. For example, the authors of [24–27] proposed federated learning-based algorithms for privacy protection in mobile edge computing. For the same stream, the authors of [28, 29] took advantage of secure and privacy preserved nature of federated learning for IoT applications. Zheng *et al*. [30] highlighted the significance of federated learning over differential privacy scheme for a general machine learning model. Furthermore, Yang *et al*. [9] presented a general overview of common federated learning algorithms, applications and challenges for implementing more specific machine learning models.

Due to secure and privacy preserved inherent nature, the federated learning has got remarkable attention in recommender systems. Malle *et al*. [31] proposed a graph-based federated recommendation architecture that collectively creates and updates a global knowledge graph. An early milestone was the federated CF recommendation model [32]. It has got considerable popularity with a very simple intuition that the item vectors placed on a central server and the user vectors should reside locally on each edge device. Afterward, authors like Chen *et al*. [33] and Jalalirad *et al*. [34] took advantage of meta-learning-based algorithms for federated recommendations. More recently, Tan *et al*. [35] proposed a federated recommender system for online services that trains a recommendation model on data from multiple parties without revealing the private information of each party. The proposed model contains four layers including data layer, an algorithm layer, a service layer and an interface layer. The authors also discussed various online applications such as content recommendation, product recommendation and online advertising through the proposed model. Muhammad *et al*. [36] presented another technique to accelerate distributed learning for the recommendation tasks. It has accomplished an early training by sampling from a diverse set of clients in each training round and applying an active aggregation method that propagates the updated model to the other clients. Although some of the existing methods [31, 32, 34–37] have taken the advantage of federated learning for improving recommendation reliability, the significance of context information has not been addressed. Differently, our contribution in this work is that we model context information in a flexible and adaptive manner. Additionally, our model highlights how the default architecture of federated learning is highly suitable for privacy protection of users in recommender systems.

3. PROPOSED METHOD

In this section, we introduce the proposed Fed-CARS algorithm and UDCP protocol to guarantee users' privacy while learning in a distributed environment. First, we introduce a general federated learning approach and its key properties that make it highly suitable for privacy protection in recommender systems. Then, a high-level conceptual framework for UDCP protocol is presented followed by a detailed discussion on the Fed-CARS model. An overview of the Fed-CARS model is presented in Figure 1.

3.1. Preliminaries

Given a set of users $U = \{U_1, U_2, U_3, ..., U_{\mathcal{N}}\}$ and a set of items $I = \{I_1, I_2, I_3, ..., I_{\mathcal{M}}\}$, a rating history for each user U_i on item I_j is in the form of $U_{ij} \in R$. R is a general input rating matrix containing overall rating preferences recorded for all users in the system. p_i and q_j are the contextual embeddings of user i and item j, respectively, to describe context information C_{pq}. The rating behavior of user U_i in R is denoted as R_{ui} such that $R_{ui} \subseteq R$. For each user U_i, we have maintained a set T_{ui} representing the timestamps for the behavioral history of U_i, expressed as $T_{ui} = \{t_{1ui}, t_{2ui}, t_{3ui}, ...\}$. Finally, C_{pui} represents information about multiple contextual attributes related to user U_i. The objective of our proposed Fed-CARS system is to predict reliable ratings given by user U_i to item I_j and a top-n recommendation list of items for user U_i by learning from rating preferences R_{ui}, timestamps T_{ui} and available contextual information C_{pui} while preserving user privacy. Formally, suppose the rating history R_{ui} with timestamp T_{ui} and the context information C_{ui} are given for the user U_i, Equation (1) presents the overall objective of predicting the top-n recommendation

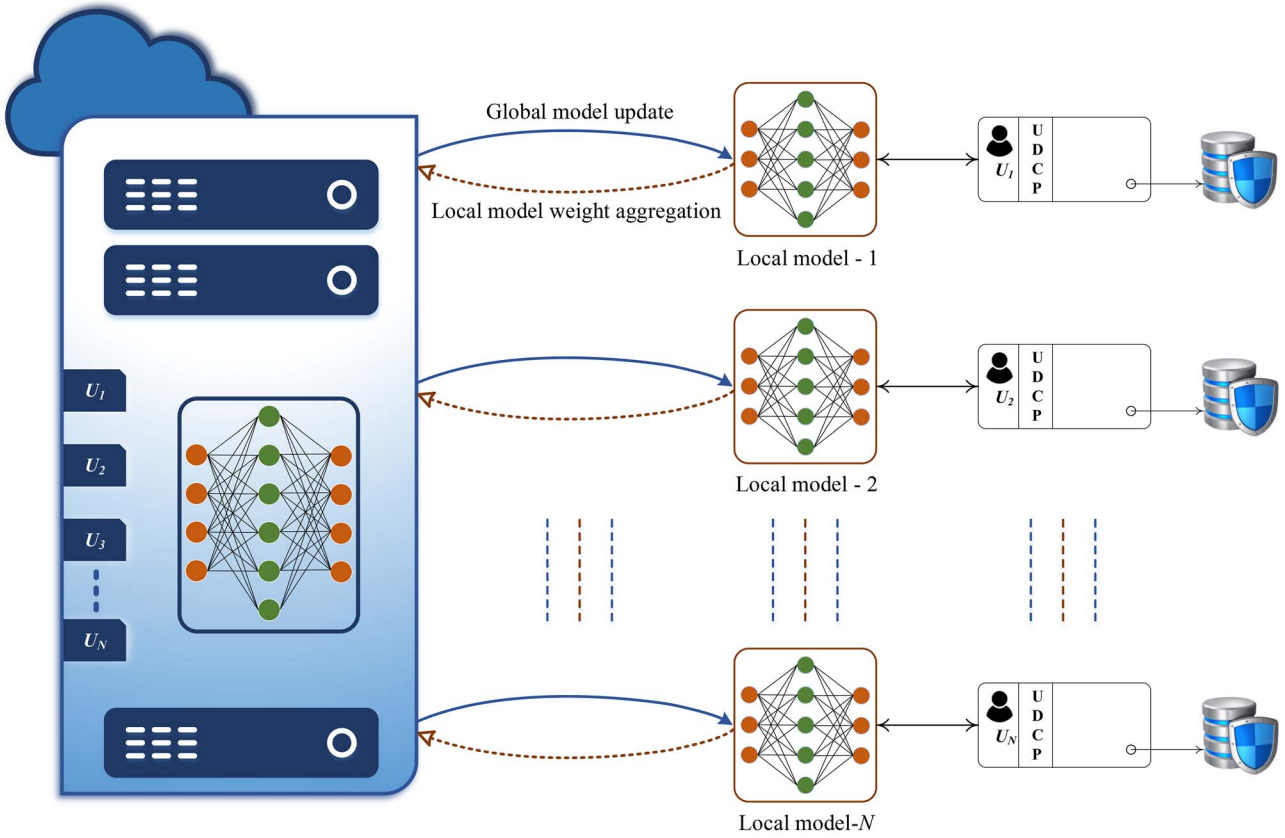


FIGURE 1. Overview of the proposed Fed-CARS model with UDCP protocol, each user has a local recommender that learns through the users' rating history and available contextual information.

list for an unknown rating.

$$C_{pui} = \{Y = 1 | R_{ui}, T_{ui}, C_{ui}\}. \tag{1}$$

3.2. Federated architecture

An overview of the proposed Fed-CARS model and associated UDCP is illustrated in Figure 1, The dotted line from client-side to server-side shows the client-side weight submission at federated server, whereas the solid arrow from server-side to client-side indicates the submission of federated weights after aggregation process. UDCP will act as a data filtration bridge between users' self-maintained data and local recommender. The proposed solution works as a federated learning system, where each client connected to a central server has its own recommendation model, which is trained through weights provided by federated-server and locally stored data. The server preserves a global recommendation model that aggregates the weights received from each client and controls the distributed learning process. Each client trains the local recommendation model and sends the weights to the server. The server has a special aggregation function that aggregates these weights and sends back to all collaborating clients for their model updates. This process of weight aggregation and transmission repeats for an input K number of epochs. Finally, the global model converges in collaboration with local models. Formally, consider a non-convex optimization for a neural network represented by a parameterized function f_θ with parameters $\theta \in \mathbb{R}^d$. For each user $U_i \in \mathcal{N}$ let $\mathcal{D}_i$ presents the total number of items $\{(x_i^1, y_i^1), ..., (x_i^j, y_i^j), ..., (x_i^{|\mathcal{D}|}, y_i^{|\mathcal{D}|})\}$ rated by user U_i. $|D_i|$ is the length of U_i items and $(x_i^j, y_i^j) \in \mathcal{X} \times \mathcal{Y}$ is a sample item that we locally evaluate with x_i^j being the input and y_i^j being the output of local recommender. The loss function such as $l(\theta, (x_i^j, y_i^j)) : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ for a user U_i is formally defined in Equation (2):

$$L(\boldsymbol{\theta}, \mathcal{D}_i) \triangleq \frac{1}{|\mathcal{D}_i|} \sum_{(x_i^j, y_i^j) \in \mathcal{D}_i} l\left(\boldsymbol{\theta}, (x_i^j, y_i^j)\right). \tag{2}$$

For simplicity, we can write the loss for user U_i as $L_i(\theta)$. Furthermore, the overall loss of all $\mathcal{N}$ users denoted by $L(\theta)$ is an aggregation of $\mathcal{W}_i \times L_i(\theta)$ and is formally presented in

Equation (3):

$$L(\theta) \triangleq \sum_{i \in \mathcal{N}} \mathcal{W}_i L_i(\theta) \quad \text{where} \quad \mathcal{W}_i = \frac{|\mathcal{D}_i|}{\sum_{i \in \mathcal{N}} |\mathcal{D}_i|}, \tag{3}$$

where for each user U_i, $\mathcal{W}_i$ is the proportion of total locally rated items by user U_i as compared with the overall item ratings rated by other users. The value of $\mathcal{W}_i$ depends on the size of local dataset. Additionally, the local dataset $\mathcal{D}_i$ of a target user is divided into two disjoint sets, i.e. the training set $\mathcal{D}_i^{train}$ and the testing set $\mathcal{D}_i^{test}$, where $\mathcal{D}_i > \mathcal{D}_i^{train} > \mathcal{D}_i^{test}$ for all $i \in \mathcal{N}$. Given the model parameter θ, learning rate α and the training set $\mathcal{D}_i^{train}$ with contextual embedding provided by UDCP, the client-side recommender first updates θ using one-step gradient descent such that:

$$\Omega_i(\theta) = \theta - \alpha \nabla_\theta L(\theta, \mathcal{D}_i^{train}). \tag{4}$$

In short, the overall objective of the proposed Fed-CARS model deduced from Equation (4) with α being the learning rate, and loss $L(\Omega_i, \mathcal{D}_i^{test})$ for updated model parameter Ω_i based on $\mathcal{D}_i^{test}$ is formally described in Equation (5):

$$\min_{\boldsymbol{\theta}} \quad \sum_{i \in \mathcal{N}} \mathcal{W}_i L(\Omega_i(\theta), \mathcal{D}_i^{\text{test}}). \tag{5}$$

It is comprehensive now to solve the learning objective described in Equation (5) in distributed fashion. As illustrated in Figure 1, each user updates model parameter θ locally based on its local self-protected dataset $\mathcal{D}_i$ and broadcasts the updated weights to the server for global aggregation. Inspired from the existing meta-learning-based recommendation models [33, 34, 37–39], a meta-learning framework for model learning at client side called model-agnostic meta-learning (MAML) [40] can be adopted. Unlike traditional recommender systems, the entire user–item interactions are not available, but each user has its own data instead. The objective of bringing meta-learning at client-side recommender is to utilize its few-shot learning power, i.e. the ability to converge with limited data samples [34, 37]. Additionally, as outlined in Algorithm 1, the client-side models initially start training with locally available data, UPCP-provided contextual embedding and server-transmitted random set of weights, i.e. $\boldsymbol{\theta}^0$, and we call this as local model training. The local recommendation model at each user U_i for all $i \in \mathcal{N}$ first updates $\boldsymbol{\theta}_i^t$ (t indicates global epoch count) using $\mathcal{D}_i^{train}$ as specified in Equation (4) and then utilizing updated $\boldsymbol{\theta}_i^t$ and Ω_i^t to find the next iteration parameters as formally described in Equation (6):

$$\boldsymbol{\theta}_i^{t+1} = \boldsymbol{\theta}_i^t - \beta \nabla_\theta L(\Omega_i^t, \mathcal{D}_i^{test}). \tag{6}$$

Here, β is used to specify the local learning rate to achieve next iteration $\boldsymbol{\theta}_i^{t+1}$ parameters through the current $\mathcal{D}_i^{test}$ test dataset. At the server side, these weights are aggregated through a global aggregation function. The next iteration parameters $\boldsymbol{\theta}_i^{t+1}$ received from $\mathcal{N}$ users are aggregated as stated in Equation (7) to obtain an updated form of $\boldsymbol{\theta}_i^{t+1}$. After aggregation, the updated parameters $\boldsymbol{\theta}_i^{t+1}$ are sent back to the client models. The entire process is formally outlined in Algorithm 1. The client-side recommendation models use the updated $\boldsymbol{\theta}_i^{t+1}$ weights to initialize the local model again for the next iteration. In our model, this process will be repeated for a predefined input K number of times.

$$\boldsymbol{\theta}^{t+1} = \frac{1}{\mathcal{N}} \sum_{i \in \mathcal{N}} \mathcal{W}_i \boldsymbol{\theta}_i^{t+1}. \tag{7}$$

Algorithm 1 Context-aware federated learning

Input: $\mathcal{N}$, K, α, β, $\mathcal{W}_i$ for each $i \in \mathcal{N}$
Output: Optimized model parameter $\boldsymbol{\theta}^t$ after initially specified K number of iterations

```
foreach t = 1, 2, 3, ..., K do
    if t = 1 then
        Set θ_i^t ← Item popularity wise random weights
        Send θ_i^t to all local models
    else
        Compute global model parameter θ_i^{t+1}
        using Eq. (7) with all parameters received from
        N set of client-side recommendation models.
        Set θ_i^t ← global aggregated θ^{t+1} from Eq. (7)
        Send θ_i^t to all local models
    end
    foreach each user i ∈ N do
        Compute the updated model parameters Ω_i^t
        with SGD using D_i^train, C_pui, W_i such as:
        Ω_i^t = θ_i^t − α∇_θ L(θ, D_i^train)
        Compute θ_i^{t+1} using Eq. (6) with D_i^test
        Send θ_i^{t+1} back to the federated server
    end
end
```

3.3. Context integration via UDCP

To make our model more flexible and adaptive, an additional layer is constructed between client-side recommendation model and users' preserved data. The UDCP, as illustrated in Figure 2, is a provision of flexible users' choice to integrate contextual information and self-maintained rating history with local model. The input for UDCP is users' self-maintained rating history, users-context and item-context information, whereas the output of this module acts as one of the inputs of local recommendation model. For mid-level processing, we construct a flexible architecture that allows each user to selectively incorporate contextual dimensions and rating history with the global model. This will not affect the overall recommendation performance in federated environment, since,

if a set of n users did not share the complete contextual information, the global model will get contextualized use of items from the remaining set of $(\mathcal{N} - n)$ users. In this way, with UDCP each user has the liberty to define his/her own feature sharing rules for the local recommendation model. Also, it is important to consider that user preferences are always dynamic and change with time. Therefore, the choice of data sharing with local model must be flexible to each user with liberty either to incorporate the contextual information or not. The heterogeneity of contextual embedding with local recommendation model will somehow affect the local recommender performance, but it will be automatically adjusted through subsequent global epochs and weight aggregation process. Consider an example for more practical illustration. Assume there are three types of users in a recommendation environment x, y and z with full, partial and zero-context provision respectively for local recommendation model. The set of users x will take the maximum advantage of all recorded contextual dimensions and achieve to supply a fine-grained weights for the global recommendation model at server side. Users type y will take partial advantage, whereas the set of users z will not be able to catch this advantage in the first epoch. However, during the second and subsequent global epochs the set x will compensate the partial and missing contextual attributes for the sets of users y and z in term of global model weight aggregation process. This is how all users in the system preserve the right to control the level of their data sharing even with the local recommendation model and can take advantage of other users' shared information.

Additionally, the contextual relationship between the deviating interest of the user from his/her rating history is described through the fact that earlier ratings have a lower impact on the preferences, whereas the recently considered items are more influential and better indicate the inclination of the user interests. For this purpose, a dedicated timestamp needs to be recorded for each user–item interaction in training and test datasets. The datasets in our experiments (Ml-100K and Ml-latest-small) have already marked timestamp for each user–item interaction. For item-context, we consider only commonly recorded item attributes, i.e. item-genre, item-tags and item-release year. After finishing this pre-filtering process, UDCP can figure out the contextual neighbors by utilizing item information publicly available on the server side and a specific users' private information. For example, consider a user U_i, whose contextual neighbors are those who exhibit similar behaviors to his/her recently marked top-n items, e.g. having a similar item-genre, identical item-tags or watched closely released movies with the top-n items in consideration. Formally, consider that I_1, I_2 and I_3 are the top-3 item-context neighbors for the user U_i because $C(U_i, I_1, C_p)$, $C(U_i, I_2, C_p)$ and $C(U_i, I_3, C_p)$ occupy the highest contextual relevance with the top-n items recently consumed by U_i, where C_p denotes the users' context-similarity score. For measuring C_p, a simple entropy-based solution is adopted, which is formally described

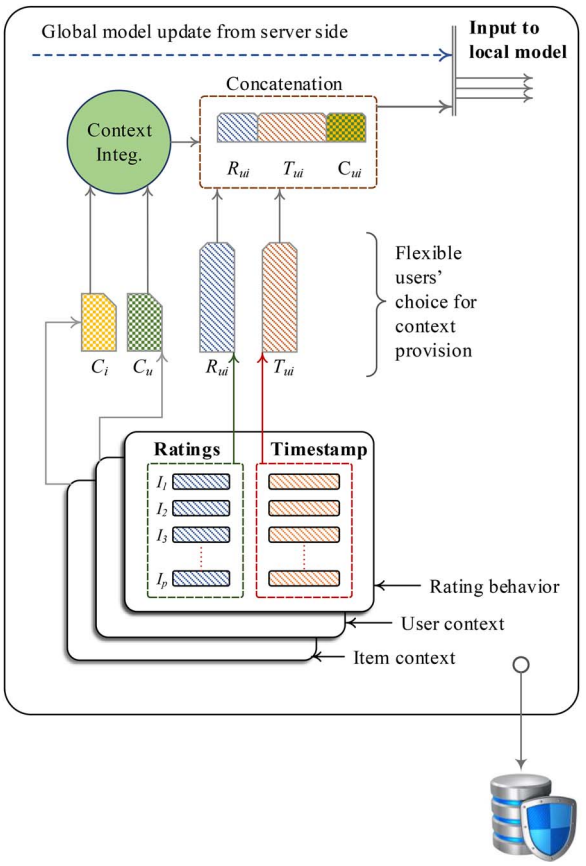


FIGURE 2. Overview of the UDCP with flexible users' choice either to integrate full contextual information, partial or even non.

in Equation (8):

$$C_p(U_i, I_j) = -\frac{1}{\log_2 D} \sum_{i=1}^{D} \sum_{j=1}^{M} \frac{v_{dm}}{n} \log_2 \frac{v_{dm}}{n}. \tag{8}$$

Here, D is the number of contextual dimensions and M is the number of possible contextual values in each dimension. The notation v_{dm} represents the occurrences of item I_j with contextual value m and dimension d, where n is the total number of observations of a user U_i. A simple normalization factor $1/(log_2 D)$ is used to ensure that the contextual score remains between zero and one.

4. EXPERIMENTAL EVALUATION

In this section, we describe the settings and results of our experiments that exhibit the effectiveness of proposed Fed-CARS model. For performance evaluation, we consider the ranking and rating prediction measures. Since it is time consuming to evaluate every user–item interaction, we fetched out the latest interactions for each user in the given dataset and

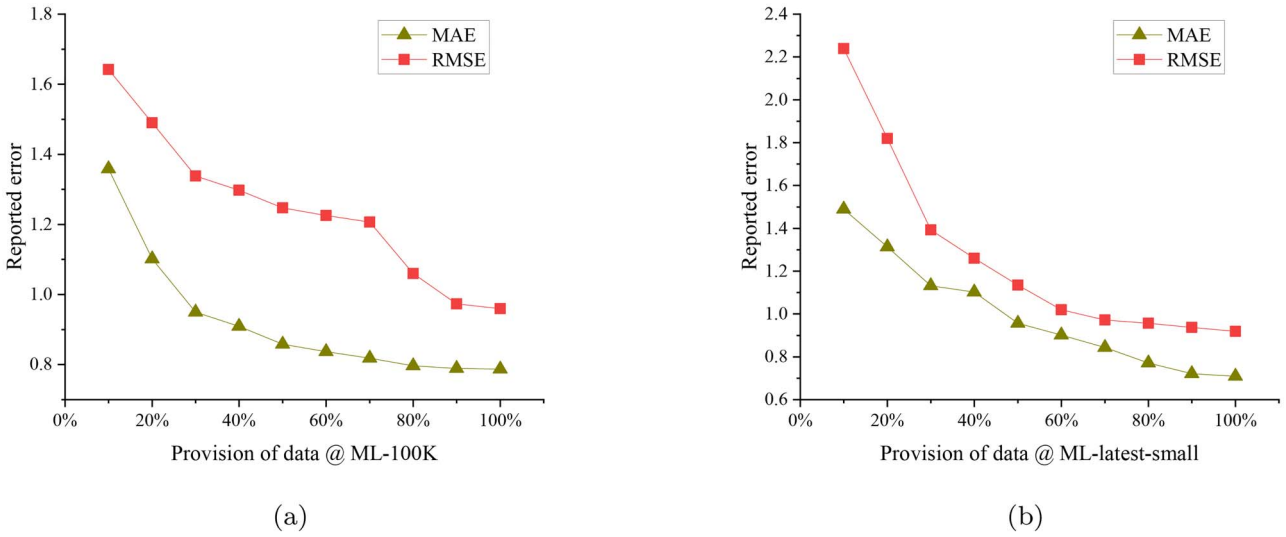


FIGURE 3. A timeline-wise reported error highlighting the significance of the Fed-CARS model working effectively with limited number of users.

randomly took 20 samples for evaluation. For utilizing contextual information, we have taken user-based context feeding and item-based context feeding for the proposed Fed-CARS model. First, using timestamp information we extracted time intervals between subsequent ratings as user-based contextual attributes. To effectively handle sparsity issues, the time intervals for each user are discretized to 24-hour time bins. Secondly, based on two input conditions, i.e. weekend or working day and daytime or nighttime, we extracted a number of input contexts for each item representing item-based context feeding. We train the Fed-CARS model separately to observe the significance of each context independently.

4.1. Datasets

We evaluate the proposed approach with two publicly available datasets, Ml-100K and Ml-latest-small. The MovieLens datasets were constructed as part of the GroupLens[1] research project at The University of Minnesota in 1997 and have been updated several times until now. A brief overview for each dataset is as follows:

- **Ml-latest-small:** This dataset contains ratings from 670 users over 9742 selected movie items on the rating scale 0.5–5 with 0.5 rating interval. The updated version has 100836 ratings marked between 29 March 1996 and 24 September 2018. Additionally, it also contains 3683 text tags associated with the number of users–item pairs.
- **Ml-100K:** This dataset is most widely used in academic literature. It contains 100 000 ratings provided by 943 users over 1682 different movies. Each user at least rated 20 movies at the rating scale of 1–5. Additionally, user information such as age, gender, profession and item information such as genre, release date and IMDb URL are provided.

4.2. Performance comparison

To evaluate the prediction accuracy and ranking performance of the proposed Fed-CARS method and selected baseline techniques, we used an open-source software CaseRecommender [41][2] for evaluation. Although accuracy and ranking performance is not our primary concern, Table 1 demonstrates that the proposed method has significantly better performed over the best practices while preserving the privacy of individual data. The baseline techniques selected for comparison are as follows:

- Bayesian personalized ranking (BPR) is an algorithm designed for implicit feedback, adopting a pairwise ranking loss to optimize the latent factor models.
- Most popular recommender (POP) is very common recommendation technique in e-commerce and medium scale recommendation engines. It predicts a user's ranking based on a popularity rank of user and items.
- User-KNN is a well-known CF technique based on user similarity to find nearest neighbor users whose liked items are recommended to the subject user.
- Item-KNN is a well-known CF technique based on item similarity to find nearest neighbor items to recommend to the subject user.

[1] https://grouplens.org/datasets/movielens/

[2] https://github.com/caserec/CaseRecommender

TABLE 1. Experimental results of Fed-CARS on the Ml-100K and Ml-latest-small datasets.

Dataset	Method	MAE	RMSE	nDCG
Ml-100K	BPR/NMF	**0.7586**	0.9691	0.7114
	POP	0.8264	1.0317	0.5800
	User-KNN	0.7966	0.9830	0.7810
	Item-KNN	0.8227	1.0794	0.7536
	PaCo			0.5665
	Fed-CARS	0.7869	**0.9593**	**0.8492**
Ml-latest-small	BPR/NMF	**0.6946**	**0.9124**	0.4399
	POP	0.7601	0.9814	0.3690
	User-KNN	0.7161	0.9642	0.5508
	Item-KNN	0.7936	1.0723	0.5847
	PaCo			0.3632
	Fed-CARS	0.7088	0.9187	**0.6459**

- PaCo [42] is a co-clustering-based recommendation model that is highly scalable and resilient to noise. It is an extension of k-means and agglomerative hierarchical clustering approaches.

4.3. Evaluation metrics

The performance of our proposed algorithm is evaluated through both the prediction accuracy and ranking metrics. For prediction accuracy, we employed well-known measures mean absolute error (MAE) and root mean squared error (RMSE), wh for evaluating the ranking performance we used normalized discounted cumulative gain (nDCG). As discussed in [32, 34, 36, 43–45], these are the widely used measures of performance evaluation in recommender systems. Formal descriptions of MAE, RMSE and nDCG are given in Equations (1), (2) and (3), respectively.

$$MAE = \frac{1}{N}\sum_{ui\in N}(PR_{ui} - AR_{ui}). \tag{9}$$

$$RMSE = \sqrt{\frac{\sum_{ui\in N}(PR_{ui} - AR_{ui})^2}{|N|}}. \tag{10}$$

MAE estimates the average absolute deviation between the predicted ratings PR and actual ratings AR for each user–item pair. Here, PR_{ui} represents the predicted ratings of subject user u over the subject item i, where AR_{ui} represents the actual ratings. RMSE reflects the degree of deviation between the estimated ratings and actual ratings. It penalizes large deviation more heavily by squaring the errors before summing them. The formulation in Equation (2) states that for every user–item pair, we need to square and add all the differences of predicted ratings from actual ratings. Finally, we can get the RMSE score by dividing the total number of user–item pairs and taking square root of the value. The lower the value of both measures reflects a higher degree of prediction accuracy of recommender algorithm.

In information retrieval and recommender systems, nDCG is a measure of ranking performance. It is the ratio of the participants' DCG score over the ideal rankings' DCG score. The nDCG value at N places is an inverse log obtained on all positions of a ranking list that hold a relevant item. We truncated the ranked list at 10 for estimating nDCG in our experiments. Formally, nDCG is defined in Equation (3):

$$\text{nDCG} = \frac{1}{\text{IDCG}}\sum_{i=1}^{N}\frac{2^{m_i}-1}{\log_2(i+1)}. \tag{11}$$

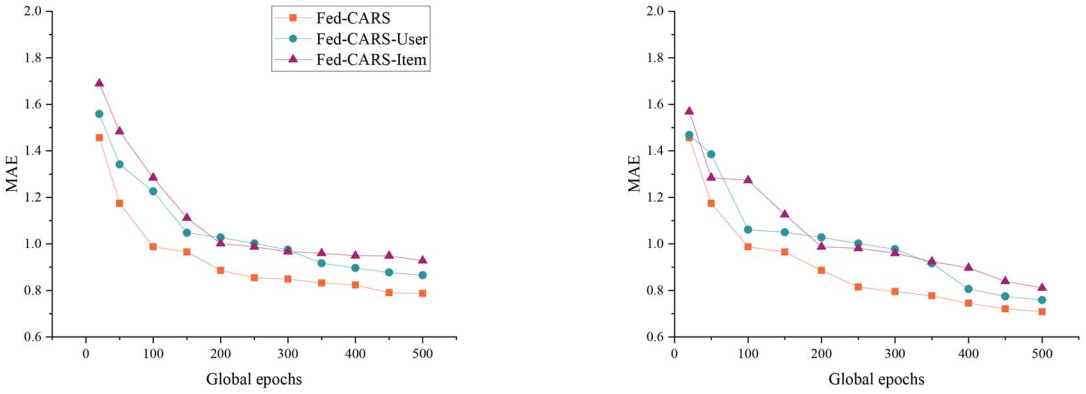
Here, N is the number of instances in the query set taken for each user as top-n considered ratings, whereas the factor m_i is 1 if the item exists at appropriate position in the predicted top-n list, and 0 otherwise. Ideal discounted cumulative gain (IDCG) is calculated as shown in Equation (4):

$$\text{IDCG} = \sum_{i=1}^{R}\frac{2^{m_i}-1}{\log_2(i+1)}, \tag{12}$$

where R represents the actual list of items ordered by their relevance. In our case, we considered a constant number of items, (i.e. 10) for top-n item generation. The overall nDCG value for all the ranking results can be an average obtained through all the recommendation results. It acts as a measure of the ranking performance of the algorithm in consideration.

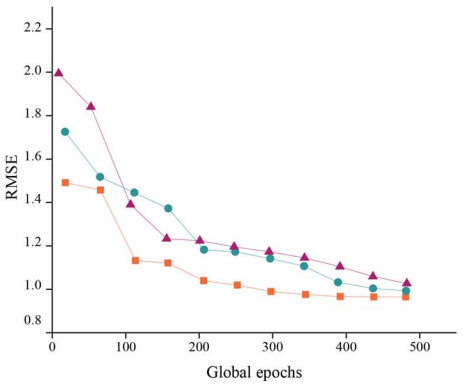
4.4. Result discussions

As for the objective of this research is concerned, we are more caring about privacy instead of recommendation performance. Inherently, federated-based recommendation architecture is a strong witness for users' privacy by keeping data locally at

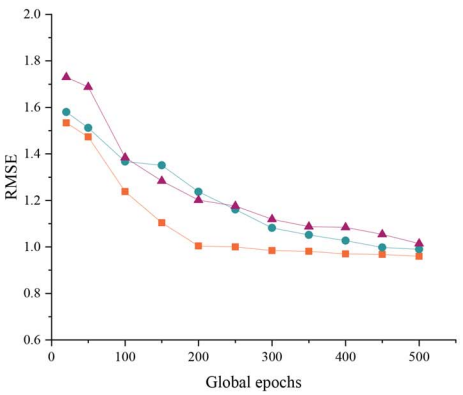


(a) MAE @ ML-100K

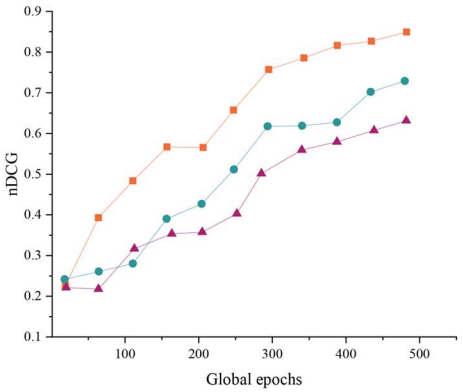
(b) MAE @ ML-latest-small



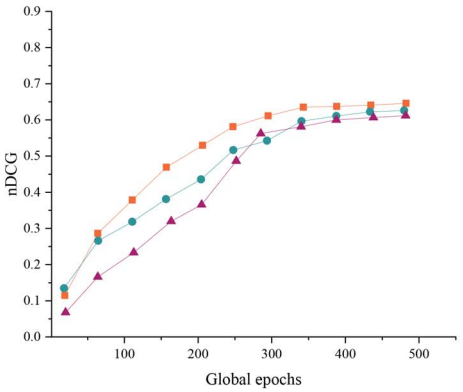
(c) RMSE @ ML-100K



(d) RMSE @ ML-latest-small



(e) nDCG @ ML-100K



(f) nDCG @ ML-latest-small

FIGURE 4. An illustration for the significance of context information with ML-100K and ML-latest-small datasets, it is clearly demonstrated that Fed-CARS has achieved better performance in term of MAE (a, b), RMSE (c, d), and nDCG (e, f) than the models with partial context information.

edge device. It is difficult to fully exploit CF because of the fact that all user–item interactions are not at a central location, which affects the performance in general. To eliminate this effect, we took advantage of meta-learning algorithm MAML that makes it possible to achieve a considerable recommendation performance while maintaining the federated architecture. Table 1 shows an overall comparison of Fed-CARS and selective baseline techniques. It is worth considering that despite

federated cost for collaborative data, the overall values against all measures are comparable and satisfactory while providing privacy guarantees.

Additionally, to observe the impact of data provision for global model updates in a federated environment, we recorded time-wise error through an additional set of experiments. Trends in Figure 3 show a positive impact on the number of users participating in the recommendation process. Figure 3 (a) highlights that the proposed model got an optimal MAE only with 50% users participation with ML-100K dataset. Similarly, Figure 3(b) shows that the model obtained significantly low error with only 60% of users data for ML-latest-small dataset. It indicates the significance of the proposed model also effectively working for a limited number of users such as 10 or 20% of the total users. Since, as discussed in Section 3.3, each user is independent to access its full contextual log of item usage, the proposed model got a considerable performance only with 10% of users in the system. This suggests that, as expected from a few-shot learning framework, it is highly appropriate for cold-start users as well.

4.5. Impact of contextual information

In a federated learning environment, each user could not fully access the overall user–item rating matrix. Each client-side model has access to its own item usage with contextual information. With the help of self-maintained user logs and item vectors publicly available on the server side, the federated model can estimate rating preferences and ranking levels. What if only the user-context information provided or only the item conditional usage incorporated with a local and global model? To address these questions and analyze the impact of contextual information separately, we constructed two additional versions of Fed-CARS, denoted as Fed-CARS-User and Fed-CARS-Item that incorporate only user-context information and item-context information, respectively, but not both. Figure 4 shows the results of additional experiments to validate whether the contextual information described in Section 3.3 plays a vital role in performance gain or not. We can observe through Figure 4 that the provision of complete contextual information has a marginal significance in all the performance measures. The trends highlight this significance on all global rounds of the model update, the Fed-CARS model with both the user-context and item-context achieved a significantly lower error rate in terms of MAE and RMSE on both datasets. For ranking performance, Figure 4 shows that Fed-CARS has gained a marginal improvement in nDCG compared with the versions without contextual information.

5. CONCLUSION

Privacy protection is one of the key concerns of users in recommender systems. In this research, we proposed a federated learning-based privacy-preserving CF technique for context-aware recommender systems that works with a UDCP to ensure users' privacy. Instead of crawling users' personal information into a central server, the proposed federated architecture Fed-CARS allows the individual user to keep personal data at edge device and train lightweight local recommendation model by utilizing personal rating behavior and global item characteristic vector maintained at server side. The inbuilt power of federated learning ensures users' privacy concerns while providing considerably accurate recommendation outcomes. We evaluated the performance of the proposed Fed-CARS algorithm through both the prediction and ranking perspectives. Despite federated cost and lack of open collaboration, the overall performance gain is comparable with popular recommendation models and satisfactory while providing privacy guarantees.

DATA AVAILABILITY

The datasets and evaluation framework underlying this work are publicly available at https://grouplens.org/datasets/movielens/ and https://github.com/caserec/CaseRecommender, respectively.

FUNDING

National Natural Science Foundation of China (61832001, 61672133) and Sichuan Science and Technology Program (2019YFG0535).

REFERENCES

[1] Narayanan, A. and Shmatikov, V. (2008) Robust de-anonymization of large sparse datasets. *2008 IEEE Symposium on Security and Privacy (S&P 2008)*, May 18–21, pp. 111–125. Oakland, CA.

[2] Nguyen, L. and Ishigaki, T. (2019) Collaborative multi-key learning with an anonymization dataset for a recommender system. *International Joint Conference on Neural Networks, IJCNN 2019*, Budapest, Hungary, July 14–19, pp. 1–9.

[3] Majeed, A. and Lee, S. (2020) Attribute susceptibility and entropy based data anonymization to improve users community privacy and utility in publishing data. *Appl. Intell.*, 50, 2555–2574.

[4] Fouad, M.R., Elbassioni, K.M. and Bertino, E. (2014) A supermodularity-based differential privacy preserving algorithm for data anonymization. *IEEE Trans. Knowl. Data Eng.*, 26, 1591–1601.

[5] Ogunseyi, T. B. and Yang, C. (2018) Survey and analysis of cryptographic techniques for privacy protection in recommender systems. *Cloud Computing and Security—4th International Conference, ICCCS 2018*, Haikou, China, June 8–10, Revised Selected Papers, Part III, pp. 691–706.

[6] Kim, J., Koo, D., Kim, Y., Yoon, H., Shin, J. and Kim, S. (2018) Efficient privacy-preserving matrix factorization for recommendation via fully homomorphic encryption. *ACM Trans. Priv. Secur.*, 21, 1, 30–17.

[7] Wadhwa, S., Agrawal, S., Chaudhari, H., Sharma, D., and Achan, K. (2020) Data poisoning attacks against differentially private

recommender systems. *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, Virtual Event, China, July 25–30, pp. 1617–1620.

[8] Shin, H., Kim, S., Shin, J. and Xiao, X. (2018) Privacy enhanced matrix factorization for recommendation with local differential privacy. *IEEE Trans. Knowl. Data Eng.*, 30, 1770–1782.

[9] Yang, Q., Liu, Y., Chen, T. and Tong, Y. (2019) Federated machine learning: concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10, 1, 19–12.

[10] Adomavicius, G., Sankaranarayanan, R., Sen, S. and Tuzhilin, A. (2005) Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans. Inf. Syst.*, 23, 103–145.

[11] Koren, Y. (2009) Collaborative filtering with temporal dynamics. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, France, June 28–July 1, pp. 447–456.

[12] Yuan, W., Wang, H., Yu, X., Liu, N. and Li, Z. (2020) Attention-based context-aware sequential recommendation model. *Inf. Sci.*, 510, 122–134.

[13] Xu, H. and Jiang, C. (2020) Research on context-aware group recommendation based on deep learning. *Neural Comput. Appl.*, 32, 1745–1754.

[14] Bhattacharya, M. and Barapatre, A. (2020) Query as context for item-to-item recommendation. *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, Virtual Event, Brazil, September 22–26, pp. 575–576.

[15] Guo, W., Zhang, C., Guo, H., Tang, R., and He, X. (2020) Multi-branch convolutional network for context-aware recommendation. *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, Virtual Event, China, July 25–30, pp. 1709–1712.

[16] Song, K., Ji, M., Park, S., and Moon, I. (2019) Hierarchical context enabled recurrent neural network for recommendation. *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, Honolulu, Hawaii, January 27–February 1, pp. 4983–4991.

[17] Wu, L., Quan, C., Li, C., Wang, Q., Zheng, B. and Luo, X. (2019) A context-aware user-item representation learning for item recommendation. *ACM Trans. Inf. Syst.*, 37, 1, 29–22.

[18] Waqar, A., Shao, J., Aman, K.A. and Saifullah, T. (2019) Context-aware recommender systems: challenges and opportunities. *J. Univ. Electron. Sci. Technol. China*, 48, 655–673.

[19] Qi, L., Wang, R., Hu, C., Li, S., He, Q. and Xu, X. (2019) Time-aware distributed service recommendation with privacy-preservation. *Inf. Sci.*, 480, 354–364.

[20] Mazeh, I. and Shmueli, E. (2020) A personal data store approach for recommender systems: enhancing privacy without sacrificing accuracy. *Expert Syst. Appl.*, 139, 112858.

[21] Knijnenburg, B.P. and Kobsa, A. (2013) Making decisions about privacy: information disclosure in context-aware recommender systems. *ACM Trans. Interact. Intell. Syst.*, 3, 1, 23–20.

[22] Konecný, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T. and Bacon, D. (2016) Federated learning: strategies for improving communication efficiency. *arXiv*, *preprint*, **1610.05492**.

[23] Konecný, J., McMahan, H.B., Ramage, D. and Richtárik, P. (2016) Federated optimization: distributed machine learning for on-device intelligence. *arXiv*, *preprint*, **1610.02527**.

[24] Fang, C., Guo, Y., Wang, N. and Ju, A. (2020) Highly efficient federated learning with strong privacy preservation in cloud computing. *Comput. Secur.*, 96, 101889.

[25] Hao, M., Li, H., Luo, X., Xu, G., Yang, H. and Liu, S. (2020) Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Trans. Ind. Informatics*, 16, 6532–6542.

[26] Qian, Y., Hu, L., Chen, J., Guan, X., Hassan, M.M. and Alelaiwi, A. (2019) Privacy-aware service placement for mobile edge computing via federated learning. *Inf. Sci.*, 505, 562–570.

[27] Hsu, R., Wang, Y., Fan, C., Sun, B., Ban, T., Takahashi, T., Wu, T., and Kao, S. (2020) A privacy-preserving federated learning system for android malware detection based on edge computing. *15th Asia Joint Conference on Information Security, AsiaJCIS 2020*, Taipei, Taiwan, August 20–21, pp. 128–136.

[28] Aïvodji, U. M., Gambs, S., and Martin, A. (2019) IOTFLA: a secured and privacy-preserving smart home architecture implementing federated learning. *2019 IEEE Security and Privacy Workshops, SP Workshops 2019*, San Francisco, CA, May 19–23, pp. 175–180.

[29] Lu, Y., Huang, X., Dai, Y., Maharjan, S. and Zhang, Y. (2020) Blockchain and federated learning for privacy-preserved data sharing in industrial iot. *IEEE Trans. Ind. Informatics*, 16, 4177–4186.

[30] Zheng, H., Hu, H. and Han, Z. (2020) Preserving user privacy for machine learning: local differential privacy or federated machine learning? *IEEE Intell. Syst.*, 35, 5–14.

[31] Malle, B., Giuliani, N., Kieseberg, P., and Holzinger, A. (2017) The more the merrier—federated learning from local sphere recommendations. *Machine Learning and Knowledge Extraction-First IFIP TC 5, WG 8.4, 8.9, 12.9 International Cross-Domain Conference, CD-MAKE 2017*, Reggio di Calabria, Italy, August 29–September 1, Proceedings, pp. 367–373. Springer.

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

[33] Chen, F., Dong, Z., Li, Z. and He, X. (2019) Federated meta-learning with fast convergence and efficient communication. *arXiv*, *preprint*, **1802.07876**.

[34] Jalalirad, A., Scavuzzo, M., Capota, C., and Sprague, M. R. (2019) A simple and efficient federated recommender system. *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, BDCAT 2019,* Auckland, New Zealand, December 2–5, pp. 53–58.

[35] Tan, B., Liu, B., Zheng, V. W., and Yang, Q. (2020) A federated recommender system for online services. *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, Virtual Event, Brazil, September 22–26, pp. 579–581.

[36] Muhammad, K., Wang, Q., O'Reilly-Morgan, D., Tragos, E. Z., Smyth, B., Hurley, N., Geraci, J., and Lawlor, A. (2020) Fedfast: going beyond average for faster training of federated recommender systems. *KDD'20: The 26th ACM SIGKDD Conference*

on Knowledge Discovery and Data Mining, Virtual Event, CA, August 23–27, pp. 1234–1242.

[37] Zhang, Y., Feng, F., Wang, C., He, X., Wang, M., Li, Y., and Zhang, Y. (2020) How to retrain recommender system? A sequential meta-learning method. *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, Virtual Event, China, July 25–30, pp. 1479–1488.

[38] Fallah, A., Mokhtari, A. and Ozdaglar, A.E. (2020) Personalized federated learning: a meta-learning approach. *arXiv*, *preprint*, **2002.07948**.

[39] Jiang, Y., Konecný, J., Rush, K. and Kannan, S. (2019) Improving federated learning personalization via model agnostic meta learning. *arXiv*, *preprint*, **1909.12488**.

[40] Finn, C., Abbeel, P., and Levine, S. (2017) Model-agnostic meta-learning for fast adaptation of deep networks. *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, Sydney, NSW, Australia, August 6–11, pp. 1126–1135.

[41] Costa, A. F. D., Fressato, E. P., Neto, F. S. A., Manzato, M. G., and Campello, R. J. G. B. (2018) Case recommender: a flexible and extensible python framework for recommender systems. *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018*, Vancouver, BC, Canada, October 2–7, pp. 494–495.

[42] Vlachos, M., Fusco, F., Mavroforakis, C., Kyrillidis, A., and Vassiliadis, V. G. (2014) Improving co-cluster quality with application to product recommendations. *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014*, Shanghai, China, November 3–7, pp. 679–688.

[43] Zheng, L., Zhu, F. and Mohammed, A. (2017) Attribute and global boosting: a rating prediction method in context-aware recommendation. *Comput. J.*, 60, 957–968.

[44] Taghavi, M., Bentahar, J., Bakhtiyari, K. and Hanachi, C. (2018) New insights towards developing recommender systems. *Comput. J.*, 61, 319–348.

[45] Waqar, A., Din, S.U., Khan, A.A., Tumrani, S., Wang, X. and Shao, J. (2020) Context-aware collaborative filtering framework for rating prediction based on novel similarity estimation. *Comput. Mater. Continua*, 63, 1065–1078.