PREFER: Point-of-interest REcommendation with efficiency and privacy-preservation via Federated Edge leaRning

YETING GUO, College of Computer, National University of Defense Technology, China
FANG LIU, School of Design, Hunan University, China
ZHIPING CAI, College of Computer, National University of Defense Technology, China
HUI ZENG, College of Computer, National University of Defense Technology, China
LI CHEN, Department of Computer Science, School of Computing and Informatics, University of Louisiana at Lafayette
TONGQING ZHOU, College of Computer, National University of Defense Technology, China
NONG XIAO, College of Computer, National University of Defense Technology, China

Point-of-Interest (POI) recommendation is significant in location-based social networks to help users discover new locations of interest. Previous studies on such recommendation mainly adopted a centralized learning framework where check-in data were uploaded, trained and predicted centrally in the cloud. However, such a framework suffers from privacy risks caused by check-in data exposure and fails to meet real-time recommendation needs when the data volume is huge and communication is blocked in crowded places. In this paper, we propose PREFER, an edge-accelerated federated learning framework for POI recommendation. It decouples the recommendation into two parts. Firstly, to protect privacy, users train local recommendation models and share multi-dimensional user-independent parameters instead of check-in data. Secondly, to improve recommendation efficiency, we aggregate these distributed parameters on edge servers in proximity to users (such as base stations) instead of remote cloud servers. We implement the PREFER prototype and evaluate its performance using two real-world datasets and two POI recommendation models. Extensive experiments demonstrate that PREFER strengthens privacy protection and improves efficiency with little sacrifice to recommendation quality compared to centralized learning. It achieves the best quality and efficiency and is more compatible with increasingly sophisticated POI recommendation models compared to other state-of-the-art privacy-preserving baselines.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; *Collaborative and social computing systems and tools*; • **Security and privacy** → *Social aspects of security and privacy*.

Additional Key Words and Phrases: POI recommendation, federated learning, edge computing

Authors' addresses: Yeting Guo, College of Computer, National University of Defense Technology, Hunan, China, guoyeting13@nudt.edu.cn; Fang Liu, School of Design, Hunan University, Hunan, China, fangl@hnu.edu.cn; Zhiping Cai, College of Computer, National University of Defense Technology, Hunan, China, zpcai@nudt.edu.cn; Hui Zeng, College of Computer, National University of Defense Technology, Hunan, China, zenghui116@nudt.edu.cn; Li Chen, Department of Computer Science, School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, Louisiana, li.chen@louisiana.edu; Tongqing Zhou, College of Computer, National University of Defense Technology, Hunan, China, zhoutongqing@nudt.edu.cn; Nong Xiao, College of Computer, National University of Defense Technology, Hunan, China, nongxiao@nudt.edu.cn.

2474-9567/2021/3-ART13 $15.00
https://doi.org/10.1145/3448099

ACM Reference Format:
Yeting Guo, Fang Liu, Zhiping Cai, Hui Zeng, Li Chen, Tongqing Zhou, and Nong Xiao. 2021. PREFER: Point-of-interest REcommendation with efficiency and privacy-preservation via Federated Edge leaRning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 1, Article 13 (March 2021), 25 pages. https://doi.org/10.1145/3448099

1 INTRODUCTION

1.1 Motivation

Location-based social networks (LBSNs), such as Foursquare, Gowalla and Wechat, have been very pervasive nowadays [16]. Point-of-Interest (POI) recommendation is one of the major tasks in LBSNs, which suggests new interesting locations to users by extracting user preference and location popularity from their historical check-in activities [53]. It not only significantly saves our effort in exploring new places (e.g. restaurants [35]) most likely to match personal interests for our weekends or vacations, but also benefits newly opened places to find their target customers for advertising[53].

Previous works on POI recommendation [14, 21, 52] mainly focus on building strong recommendation models to achieve high recommendation quality. Various POI-related factors, such as location category, distance, time and social relationship, are integrated into POI models. However, the whole recommendation procedure includes not only model building but also data collection and recommendation presentation. Most works neglect the impact of these two stages and simply adopt a centralized learning framework. As shown in Figure 1, user check-in data are collected and uploaded to the cloud for model training and prediction. However, there exist substantial social security risks and inefficiency problems elaborated as follows.

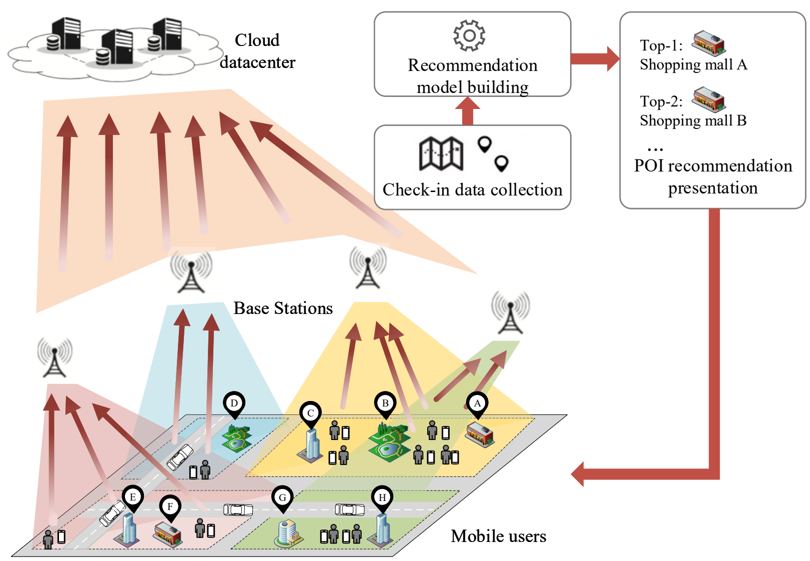

Fig. 1. POI recommendation procedure under centralized learning framework

- *Privacy risk:* Location-aware social data are highly sensitive. With these data obtained online, criminals and stalkers could easily speculate on the scope of victims' daily activities and the places that attract victims to go in the future, and then spot and track victims at these locations [4, 7].

- *Inefficiency problem:* Data collection and recommendation presentation are heavily dependent on network quality. When the network bandwidth between users and the cloud is limited and unstable, such as in an overcrowded attraction during holidays, the latency to send recommendation requests and receive feedback is always unacceptable. Besides, in the model building stage, the cloud suffers from high carbon footprint and energy cost on recommendation service platforms during peak periods [42].

Aware of the privacy risks, existing works have applied data encryption mechanisms, such as k-anonymity [38] and differential privacy [3], to enhance privacy protection. But they have the following limits:

- *Data exposure:* When applying data encryption mechanisms in geographical data, the effectiveness of preventing data exposure dynamically fluctuates with environmental factors. For example, suppose you are on an island, the noised location information shows that you are in the middle of the sea now. It hardly prevents the attacker from obtaining your real locations.
- *Inefficiency:* These solutions need to take extra time to add noise to their check-in data. And they still cannot get rid of the dependency on network quality when uploading these noised data to the cloud.
- *Incompatibility:* All of them evaluate the effect of noise on performance only using the most basic recommendation model, matrix factorization method. In our follow-up experimental study, we found that the performance degradation is quite different when different models are used. The evaluation should keep pace with the development of recommendation models. The recommendation system should be designed to be compatible with different models.

1.2 Our Contribution

In view of these limits, we propose PREFER, an edge-accelerated federated learning framework for POI recommendation. Following the federated learning paradigm [47], user data are kept and trained locally. Users collaborate to construct a shared representing model by having their parameter updates aggregated round by round. However, in the POI recommendation scenario, some model parameters are still of high sensitivity to be shared, based on our comprehensive investigation and analysis on the POI recommendation models [11, 14, 21]. In particular, most models are based on the matrix factorization method, which decouples the recommendation matrix into user latent matrix and location item latent matrix. Then time&distance-related matrices are introduced and analyzed by the promising artificial intelligence technology. They enrich the dimension of POI models and improve recommendation quality. Given that the user latent matrix is extremely sensitive, PREFER shares multi-dimensional user-independent parameters, *i.e.*, item latent matrix, time&distance-related matrices and the weights of model among users, while keeping user latent matrix at local. Thus, different from existing federated learning , PREFER is customized for the POI recommendation setting which further enhances privacy protection.

With respect to improving efficiency, we further propose to aggregate user-independent parameters close to mobile users. Traditional federated learning performs the aggregation in the remote cloud, causing large network consumption and high latency. Nowadays, the resource capability at the network edge has been gradually strengthened. Thus, edge computing is proposed to leverage such edge resources to reduce the transmission distance and improve real-time response capability [23, 25]. Operator China Telecom has officially launched the pilot project of edge computing nationwide and the scale construction of thousands of edge data centers [40]. Edge data centers are close to mobile users and often co-located with base stations. The network bandwidth is usually sufficient and stable, contributing to low request latency. And in the POI recommendation scenario, people whose check-in data rarely intersect have very limited reference to each other's location preference. Thus, in our system, we propose to offload the aggregation tasks from the remote cloud to the edge server which can be a server in an edge data center or a base station in a city or district. The edge server aggregates user-independent parameters from users in the same region. Owing to user mobility, the edge server is able to capture enough valuable information from these mobile users and provides high-quality and real-time recommendation service.

The above design optimizes the privacy and efficiency issues in POI recommendation from the perspective of the system framework. It does not require modifying the algorithm design when deploying POI models in our framework. And note that since our discussions are for advanced POI recommendation models that consider various POI-related factors, our framework is more practical and compatible with existing works on POI models. We have implemented the prototype, conducted case studies on two advanced POI models (PRME-G [15] and Distance2Pre [11]), and evaluated PREFER in recommendation quality, efficiency and compatibility.

The contributions of PREFER are summarized as follows:

- We propose a privacy-preserving federated framework for POI recommendation, especially to support more multi-dimensional content aggregation. In this framework, we keep the sensitive check-in data at local devices, assign these devices to build recommendation models in parallel, and then aggregate multi-dimensional user-independent parameters to build a federated POI recommendation model.
- We propose an edge-based efficient POI recommendation scheme. We benefit from the proximity of the edge server, and aggregate parameters in the edge server instead of the remote cloud to improve the real-time responsiveness of the recommendation system.
- We customize two advanced POI recommendation models in PREFER. As far as we know, we are the first to prove the recommendation system framework's compatibility with existing advanced models.
- We implement the PREFER prototype and evaluate its performance using two real-world check-in datasets. The results demonstrate that our proposal is compatible with these advanced models, and achieves the state-of-the-art recommendation quality and efficiency without the exposure of check-in data.

The remainder of this paper is organized as follows. We review the background and related works in Section 2, followed by some related preliminaries in Section 3. We elaborate the core design of PREFER and two case studies in Section 4. Finally, we evaluate the system in Section 5 and conclude in Section 6.

2 RELATED WORK

In this section, we first present the background of POI recommendation models, and then review the related work about federated learning, edge computing and existing privacy-preserving recommendation systems.

2.1 Point-of-Interest Recommendation Model

Point-of-Interest recommendation problem has been extensively researched in location-based social network [53]. POI recommendation models always adopt matrix factorization to make recommendations [5]. The recommendation matrix could be composed of two matrices, user latent matrix and location item latent matrix. In this way, it avoids the sparsity problem and improves the computation efficiency. Gradually, researchers study the use of auxiliary POI-related information to enhance the POI recommendation model. Feng *et al.* and Cui *et al.* investigate the geographical clustering phenomenon in users' check-in activities [11, 14]. Hosseini *et al.* and Liu *et al.* argue that the temporal influence also benefits recommendation performance since the user's preference for check-in is different when the user is in different temporal states [21, 27]. Griesner *et al.* combine geographical and temporal influences into matrix factorization [17]. Cheng et al. and Zeng et al. suppose that friends share more common interest in POI than non-friends, and make a recommendation with the consideration of social relationships [10, 52]. Such multi-dimensional auxiliary information improves and complicates the recommendation matrix. And with the development of machine learning, some promising methods, such as Recurrent Neural Network [11], Markov Chain [15], are applied in the recommendation model to make better use of auxiliary information.

2.2 Federated Learning and Edge Computing

Federated learning is an emerging privacy-preserving distributed learning paradigm [47]. It has been applied in some data privacy-sensitive areas. In 2019, Google has implemented the first product-level federated learning system [6]. Tens of millions of mobile phones have been adopted in the system to develop machine learning models for on-device item ranking, new word prediction and so on. NVIDIA has announced to work with medical institutions to build the first federated AI platform for medical diagnosis and drug researches [28]. Open-source projects, such as Webank FATA[46], are released to support the federated AI ecosystem. In addition, Feng *et al.* follow federated learning paradigm to predict human mobility and designs a group optimization method for the training on local devices to achieve a trade-off between performance and privacy [13]. Samarakoon *et al.* and Lu *et al.* design a distributed federated learning system for connected vehicles [29, 34].

Edge computing is becoming an important part of ubiquitous computing infrastructure in the Internet-of-Things (IoT) era. Edge computing data centers are co-located with mobile access networks, connecting cloud and end devices [23]. Delay-sensitive applications, such as AR/VR [37] and disaster relief applications [20], process user requests at the edge to provide real-time services. Applications with large data transmission always suffer from the limited backhaul capacity of the cellular core network. Hung *et al.* deploy live video streaming service in an edge-enabled cellular system [22]. Qiao *et al.* build a joint content placement and content delivery in vehicular edge caching network[32]. In terms of the privacy issues in edge computing, [49] has discussed it in detail. Xiao *et al.* design a hierarchical edge computing architecture to provide smart privacy protection for video data storage [48]. Intel also designs a series of edge security policies for the newly released HERO-X platform [2].

Recently some work has been conducted for the integration of the above two technologies. *In-Edge AI* [44] is such an integration framework and greatly alleviates the privacy and efficiency issues in mobile edge computing, caching and communication. Liu *et al.* propose a cloud-edge-end hierarchical federated learning system which allows edge data centers to perform partial model aggregation and release the aggregation burden in the cloud [26]. Guo *et al.* also propose a three-tier system but assigns edge data centers to execute the partial training tasks from the terminals [18]. On the basis of these frameworks, many researchers have further carried out some module optimization. Nishio *et al.* [31] and Ye *et al.* [51] design novel client selection algorithms for federated learning. They allow the edge data center to aggregate as many client updates as possible from the heterogeneous resources in mobile edge and accelerate the learning process. Wang *et al.* [43] adaptively adjust the number of local training epochs and global aggregation rounds with the awareness of network dynamics and computation resources on the end devices. Tao *et al.*[39] propose to exchange model parameters only when the changes of parameters are above a certain threshold. In this way, the number of uploads of model parameters is reduced. These works highly emphasize the importance and visibility of this integration. But they mainly apply to the general machine learning models whose parameters are less sensitive. For POI recommendation, some model parameters are still highly sensitive to be shared and the recommendation presents a certain locality, that is, people whose check-in data rarely intersect have very limited reference to each other's location preference while people who are active in the same region have relatively strong references. As far as we know, the edge-based federated learning system for POI recommendation remains unexplored.

2.3 Privacy-preserving Recommendation System

Privacy leakage has great social security risks. The recommendation system usually needs to obtain the user's daily access data, such as product browsing records and location check-in data, and then extract the user's preference features and item's rating features. Existing recommendation systems mainly focus on how to accurately extract useful features to improve recommendation performance, but neglect the potential privacy issues. Protection mechanisms, such as k-anonymity [38], l-diversity [30], and t-closeness [24], are applied. k-anonymity screens the true location by the other neighboring k-1 users' dummy locations. But when these users are in the same location, the location privacy is leaked. Then l-diversity is proposed to diversify query locations and prevent attackers from identifying the true location from l-1 different locations. t-closeness further protects users' location privacy when different locations actually belong to very close types. Tu *et al.* designs an algorithm to generalize user trajectories to resist semantic attack and re-identification attack [41]. It meets the demands of the above three mechanisms at the same time. Gao *et al.* and Andrés *et al.* adopt geographic differential privacy [3, 16] that disturbs the real location according to a certain probability. It lacks the definition of adjacent datasets and adopts geo-indistinguishability when extending traditional differential privacy into geography. Adjacent datasets refer to a pair of datasets whose difference is reflected in a single sample. The probability that adjacent data sets get the same output is used to evaluate whether the traditional differential privacy is satisfied. Geo-indistinguishability defines a circle centered on the real location, and the closer the location point to the real location, the higher the probability that it will be released as the noisy location point. These methods are easily affected by the geographical environment factors and damage the recommendation quality. Su *et al.* apply differential privacy and random perturbation to protect users' relationship data when using these data as auxiliary information [36]. It's still unable to protect user geographic location data. Cao *et al.* highlight that urban morphology can lead to location privacy leakage even when the person only simply reveals the nearby POI types [8]. Xu *et al.* demonstrate that aggregated mobility data also cause individuals' privacy breaches [50]. The aggregated data here are always some basic statistics, such as the number of users with a certain region at a specific timestamp. They are different from the aggregated model parameters in our paper.

Thus, some researches suggest keeping and training geographic location data on the user's own device. Specifically, Chen *et al.* propose a decentralized matrix factorization framework [9]. It decomposes the item latent vector into global and personal vectors, and users share the global vector with their k-nearest neighbors and keep the personal vectors locally to preserve privacy. But it is easy to cause low performance due to data sparseness. Wang *et al.* apply a teacher-student training framework [42]. The cloud trains a teacher model from available contextual data, and the end device trains a simplified student model with the teacher model. The collaboration is unidirectional, that is, the teacher model cannot perceive the characteristics of the student model well. Dolui *et al.* apply a federated learning framework [12]. The end devices upload both the user latent vector and the item latent vector to the cloud. Wang *et al.* propose to learn user latent vectors based on user group preference instead of individual user preference [45]. It makes these vectors less sensitive to be shared. Considering the sensitivity of these vectors, the authors apply differential privacy or data anonymization mechanism. Hegedüs *et al.* also propose a decentralized privacy-preserving recommendation framework [19]. But it still lacks the consideration of the multidimensionality and analysis methods in the POI recommendation model.

Inspired by these works, we develop them in the following ways: 1) using federated learning paradigm and sharing multi-dimensional user-independent parameters (including item latent matrices, time&distance matrices and weights in some applied machine learning models) to optimize the collaboration among devices; 2) making the edge server as the coordinator to efficiently aggregate the models of devices in the same region; 3) discussing and evaluating the compatibility of our framework in different POI recommendation models, rather than only for the simplest matrix factorization model. More design details would be given in Section 4. And we have compared with some above related works in our evaluation, and summarized the differences in Table 3.

3 PRELIMINARY

In this section, we briefly overview the definition of POI recommendation and the essential concepts of the model.

Definition 1 (POI recommendation): POI recommendation is devoted to extracting the user preference from their check-in data and then recommending a list of POIs that has never been visited by the user before.

Here we would first introduce two related terms, check-in activity and check-in sequence. Assume that U and L is the set of users and locations, respectively. Each item l in the set L has two basic geographical attributes, longitude and latitude. A check-in activity is denoted by a triplet $s =< u, l, t >, u \in U, l \in L$, which depicts that user u visited location l at time t. A check-in sequence is a set of user u's consecutive check-in activities. We denote this term as $S_u = \{< l_1, t_1 >, < l_2, t_2 >, \cdots < l_{|S_u|}, t_{|S_u|} >\}$. Based on users' check-in sequences, POI recommendation would provide user u with a series of POIs that have not been visited by user u before but are very likely to be visited in the near future.

Definition 2 (Matrix Factorization): POI recommendation could be regarded as a matrix completion task. As shown in Figure 2, we represent users and location items as a two-dimensional matrix $A \in \mathbb{R}^{U \times L}$ and fill this matrix with users' check-in activities. If user u has visited location l, the value of $A_{u,l}$ would be set as 1. The matrix is usually extremely sparse for the limited check-in activity records and hard to fill the remaining blank information. Matrix factorization is one of the most popular technologies used in recommendation systems. Matrix A can be the product of $X \in \mathbb{R}^{U \times k}$ and $Y \in \mathbb{R}^{L \times k}$. X and Y denote the user and location item factor matrices, respectively, and their column vectors x_u and y_l are the k-dimensional latent factors for user u and item l. In this way, it effectively gets rid of the data sparsity problem and improves the operation efficiency.

Definition 3 (Recommendation Model): POI recommendation models have been advanced with various POI-related factors. It makes A become a more practical and complex function, $Function(X, Y, D, S, W)$. D and S are distance-related matrix and time-related matrix that represent the preference on distance and time in check-in activities. Time-related matrix can also refer to the sequential matrix in some works to indicate the sequential preference. When using some machine learning model to obtain D and S or making a trade-off between different POI-related factors, some weights W are introduced in POI recommendation model.

Figure 2 presents the model building procedure in advanced POI recommendation. Check-in activity analysis module is the core component. Based on historical check-in sequences, the corresponding user latent vector x_u and item latent vector y_l are fed into this module as input, and the distance interval between two sequential check-in activities and the check-in time sequence are also calculated and fed as input with D and S. Various methods, such as Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) and Markov Chain (MC), are applied to analyze these data. The weights of these models are stored in W. This module constantly revises these parameters according to the designed loss functions. At last, the module outputs with the possibilities to visit each unvisited location, and the ranking of unvisited locations is obtained.

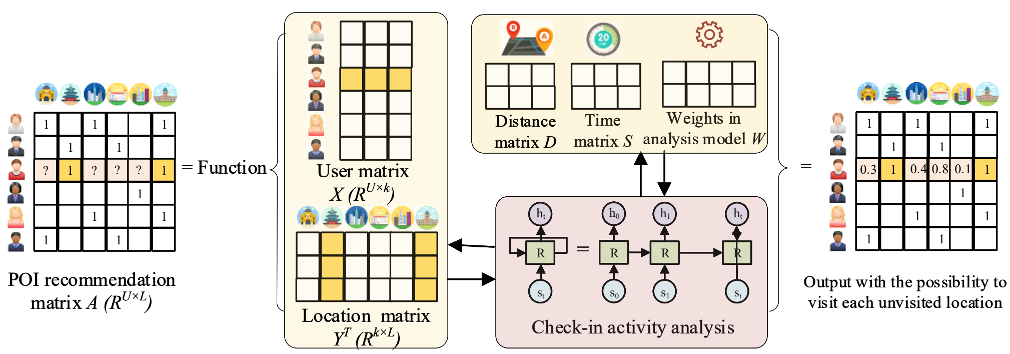

Fig. 2. Advanced POI recommendation model building procedure

4 SYSTEM DESIGN AND CASE STUDIES

In this section, faced with the privacy risk and inefficient problem in POI recommendation, we propose PREFER, a federated edge learning system. We first give an overview of the system framework, then describe each module in the system in detail, and finally take two advanced recommendation models as case studies in our system.

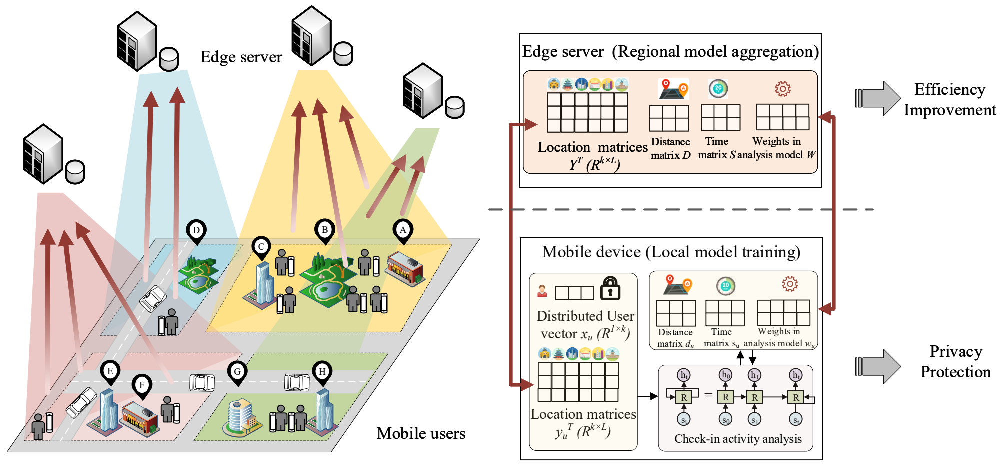

Fig. 3. PREFER shows the distributed POI recommendation model building at the network edge without check-in data transfer

4.1 System Design

The system overview is shown in Figure 3. Different with the typical centralized learning framework, we need not collect and process user check-in activities in a centralized cloud. Instead, we keep these user data on their local mobile devices. Thus, we directly step into the model building stage. In this stage, we leverage the computing resources in mobile devices and edge servers to perform distributed model building at the network edge. And in the recommendation presentation stage, we leverage the recommendation model in the mobile device to infer the POI recommendation list for their owners. PREFER is featured with two special designs, namely *local model training* and *regional model aggregation*. The two modules are deployed in distributed mobile devices and edge servers, with the purpose of privacy protection and efficiency improvement respectively. We next describe the details of each design in the following subsections.

4.1.1 Model Training at Distributed Mobile Devices. In this module, we propose to offload model building tasks from the cloud to users' own local devices. Each mobile device keeps user check-in sequences local and builds its local recommendation model based on local sequences. The procedure is presented in Algorithm 1.

To begin with, the mobile device needs initialize its user latent vector x_u or load local available x_u (line 1-3). Then it requests the nearby edge server to send location latent matrix Y, distance-related matrix D, time-related matrix S and model weights W (line 4), more details would be given in the next subsection. These downloaded parameters are used to initialize the local recommendation model in the device (line 5) and benefit the model building for cold-start users. Considering different works adopt different data analysis methods and loss functions, here we briefly denote the function of data analysis and loss calculation as *Analysis* and don't discuss their design details in this subsection. In each epoch, each sequential pair of check-in activities performs function *Analysis* with model parameter set $\{x_u, y_u, d_u, s_u, w_u\}$, and *Analysis* returns the value of the loss for the corresponding sequential pair (line 8-10). Then based on the gradient to the accumulated value of the loss, the model parameter set $\{x_u, y_u, d_u, s_u, w_u\}$ is updated with learning rate η (line 11-12). When the number of epochs reaches the preset value *Epoch*, that is, the model training is completed, we would keep and store sensitive user latent vector x_u at local to protect user privacy (line 14). Distinct from other recommendation models, such as movie or book recommendations, the distance and time factors are very important in POI-specific recommendation models.

Algorithm 1: Local model training

Input: user check-in sequence $S_u = \{< l_1, t_1 >, < l_2, t_2 >, \cdots < l_{|S_u|}, t_{|S_u|} >\}$, local user vector x_u
Output: user-independent parameters $\{y_u, d_u, s_u, w_u\}$
if $x_u == Null$ **then**
    initialize x_u;
download Y, D, S and W from the edge server in proximity;
$y_u \leftarrow Y, d_u \leftarrow D, s_u \leftarrow S, w_u \leftarrow W$;
$loss \leftarrow 0$;
for *each epoch* $i \in [1, Epochs]$ **do**
    for *each check-in activity* $< l_j, t_j > \in S_u$ *and* $j! = |S_u|$ **do**
        $loss \leftarrow loss + Analysis(< l_j, t_j >, < l_{j+1}, t_{j+1} >, x_u, y_u, d_u, s_u, w_u)$;
    $x_u \leftarrow x_u - \eta \cdot \triangledown loss(x_u)$;
    $y_u \leftarrow y_u - \eta \cdot \triangledown loss(y_u)$, $d_u \leftarrow d_u - \eta \cdot \triangledown loss(d_u)$, $s_u \leftarrow s_u - \eta \cdot \triangledown loss(s_u)$, $w_u \leftarrow w_u - \eta \cdot \triangledown loss(w_u)$;
store x_u at local;
return $\{y_u, d_u, s_u, w_u\}$;

Thus, for the availability and quality of POI recommendation models, we not only share the basic location item latent metric y_u, but also share the auxiliary parameters d_u, s_u and w_u (line 15).

In this way, the model building tasks are offloaded to the mobile devices, the user check-in data are kept at the local devices without being transferred to the third party. It greatly strengthens privacy protection. Besides, the efficiency in the model building stage is also improved because the amount of data in each mobile device is relatively small compared to the numerous data collected by the cloud. However, the scarce and biased data in one single device also results in very poor-quality recommendations. Faced with this problem, through analyzing the composition of parameters in POI-specific model, we propose to share only necessary model parameters (location latent matrix, distance&time-related matrices and model weights) except for sensitive user latent matrix. It takes into account both privacy concerns and the quality of the model, minimizing the shared information used to build a well-performed POI recommendation model.

4.1.2 Regional Model Aggregation in the Edge Server. Since each mobile device has obtained its own recommendation model, we then discuss the aggregation of these distributed models. We propose to assign the edge server in replace of the cloud to aggregate models from users in the same region. The procedure is shown in Algorithm 2.

As shown in Figure 1, base stations forward user requests to the cloud. The edge server is always co-located with the base stations, thus can sense the existence of users in the service area. The server records users' IP addresses within its service coverage in list U, which would be updated when a user leaves or enters the corresponding area. Note that the edge server only knows whether users are in its coverage area (usually as large as a district scale, or even a city scale), instead of the users' detailed addresses or check-in activities. In the beginning of model aggregation, the edge server creates a model base and initializes the necessary global model parameters $\{Y, D, S, W\}$ (line 1). Then to optimize the model base, the edge server interacts with users in U, and queries whether they are willing to participate in the POI recommendation model co-construction and improve their own model's quality by the way. When obtaining the availability of users, it starts several rounds of interactions with users to make the model convergence (line 2). In each round of interactions, the edge server randomly selects a subset U_t' from all available users (line 3) and sends the global model parameters to each selected user (line 6). Each user would perform local model training based on the received global model parameters and local user latent

Algorithm 2: Regional model aggregation

Input: users in its covered region U
Output: aggregated model parameters $\{Y, D, S, W\}$
1 initialize model parameters $\{Y, D, S, W\}$;
2 **for** *each round* $t \in [1, Rounds]$ **do**
3 $\quad U'_t \leftarrow$ random subset of available users in U;
4 $\quad \Delta Y,\ \Delta D,\ \Delta S,\ \Delta W = \emptyset,\ \emptyset,\ \emptyset, \emptyset$;
5 $\quad$ **for** *each user* $u \in U'_t$ **do**
6 $\quad\quad$ send $\{Y, D, S, W\}$ to u ;
7 $\quad\quad$ user u perform local training as shown in Algorithm 1;
8 $\quad\quad$ receive $\{y_u, d_u, s_u, w_u\}$ from user u;
9 $\quad\quad$ $\Delta Y \leftarrow \Delta Y \cup y_u,\ \Delta D \leftarrow \Delta D \cup d_u,\ \Delta S \leftarrow \Delta S \cup s_u, \Delta W \leftarrow \Delta W \cup w_u$;
10 $\quad Y \leftarrow\ Average(\Delta Y), D \leftarrow\ Average(\Delta D), S \leftarrow\ Average(\Delta S), W \leftarrow\ Average(\Delta W)$;
11 return $\{Y, D, S, W\}$;

vector as shown in Algorithm 1 (line 7), and provide feedback with their updated parameters $\{y_u,\ d_u,\ s_u, w_u\}$ (line 8). The server collects these distributed updated parameters into the corresponding set $\{\Delta Y,\ \Delta D,\ \Delta S\ \Delta W\}$ (line 9). In the end of each round, the server calculates the average value of each parameter set to update the global model parameter $\{Y, D, S, W\}$ (line 11). At last, the global model parameters are stored in the edge server (line 13) and continue to be called in the next aggregation.

In this way, the edge server serves as an aggregation point to connect the distributed mobile devices within the same region. It acquires valuable check-in knowledge from the POI recommendation models of these devices rather than directly access raw check-in activities. It not only protects user privacy, but also gets rid of the low-quality models caused by data sparsity on a single device. Although the edge server need to record users' area information (not an exact GPS location), considering that this geographic area is usually very large, our scheme can still efficiently protect user privacy. The uppermost challenge in the model aggregation stage is the heavy communication burden. Even though the dimension of location matrix has been reduced through matrix factorization method, the amount of data transferred is still large because we need to perform multiple rounds of interaction. It exacerbates the dependency on network quality, described in Section 1. Faced with this problem, we propose to perform regional model aggregation in the edge server in replace of global model aggregation at the cloud. Benefit from the proximity to users, the edge server could aggregate models more efficiently than the cloud. But since the edge server covers limited users, the recommendation quality decreases to some extent compared to the global aggregation. Later our experiments have approved that the quality decrease is acceptable compared to the great decrease in transmission latency.

4.2 Case Studies

In terms of the lack of discussion on advanced recommendation models in most existing POI recommendation systems, we have investigated two advanced POI recommendation models (PRME-G [15] and Distance2Pre [11]) and customized them into our system as case studies.

4.2.1 PRME-G. PRME-G resorts to the Metric Embedding method to model personalized check-in sequences and integrate sequential information, individual preference and spatial influence.

In the *local model training* module, user u constructs three matrices, X, Y, S. X and Y are user and location latent matrices as we described above. S is the sequential transition matrix. Each location l has one latent vector

s_l in S to indicate the transition relationship with other locations. This work calculates the Euclidean distance of each pair of locations and the Euclidean distance of users and locations, as shown in Equation 1. The matrix D^G_{u,l_i,l_j} defined in Equation 2 integrates the spatial and sequential influence. $\Delta(l_i, l_j)$ is the time difference between two successive check-in activities and τ is a threshold, and d_{l_i,l_j} is the geographical distance and α controls the weight of $D^S_{l_i,l_j}$ and $D^P_{l_i,l_j}$. The loss function is shown in Equation 3, where $\Theta = \{X, Y, S\}$, σ refers to *sigmoid* function and λ is a parameter controlling the regularization term. These calculation tasks are performed at distributed mobile devices rather than the centralized cloud.

$$D^S_{l_i,l_j} = \|s_{l_i} - s_{l_j}\|^2, \quad D^P_{u,l} = \|x_u - y_l\|^2 \tag{1}$$

$$D^G_{u,l_i,l_j} = \begin{cases} D^P_{u,l_j} & if \;\; \Delta(l_i, l_j) > \tau \\ (1 + d_{l_i,l_j})^{0.25} \cdot (\alpha D^P_{u,l_j} + (1-\alpha) D^S_{l_i,l_j}) & otherwise \end{cases} \tag{2}$$

$$\Theta^* = argmin_\Theta \sum_{l_c \in L} \sum_{l_i \in L} \sum_{l_j \in L} (-ln\sigma(D^G_{u,l_c,l_i} - D^G_{u,l_c,l_j})) + \lambda \|\Theta\|^2 \tag{3}$$

In *regional model aggregation* module, according to our above design, to further protect user privacy, we aggregate location matrix and sequential transition matrix $\{Y, S\}$ in the edge server except for user matrix X.

4.2.2 Distance2Pre. Distance2Pre introduces sequential and spatial factors into the POI recommendation model by modeling user check-in sequences and distances between successive check-in activities in RNN method.

In *local model training* module, user u calculates the distances between successive activities, and maps each distance value to a preset distance interval set $[0, \delta, 2\delta, \cdots, M_D]$. δ and M_D refer to the minimal and maximum intervals, respectively. Each interval $n\delta$ is also mapped to a distance latent vector $d_n \in \mathbb{R}^{1\times k}$. Thus, given the user' check-in sequences S_u, the sequence of mapped distance vectors is obtained, denoted as $[d^1_l, d^2_l \cdots]$.

This work applies RNN method to model check-in sequences. In Equation 4, f refers to the RNN function, and U', W', b are the parameters in RNN function. For each unit in RNN, the inputs are the location and distance vectors corresponding to the current visited location $[y^t_l, d^t_l]$, and the previous user latent matrix x^{t-1}_u, and the output is the updated user latent matrix x^t_u. s^t in Equation 5 denotes the spatial preference of all distance intervals, V_s and b_s are parameters in $SoftReLU$ function. The spatial and sequential preferences are integrated as shown in Equation 6. The loss function is shown in Equation 7, and Θ denotes a set of parameters $\Theta = \{Y, D, W\}$ and $\widehat{x}^t_{ul'}$ is the negative preference obtained by Bayesian Personalized Ranking [33]. As we described in the above section 4.1, D is the set of distance latent vectors, and $W = \{U', W', b, V_s, b_s, w_d\}$ is the set of model weights in this work.

$$x^t_u = f(U'[y^t_l; d^t_l], W' x^{t-1}_u, b) \tag{4}$$

$$s^t = SoftReLU(V_s x^t_u + b_s) \tag{5}$$

$$\widehat{x}^t_{ul} = (x^t_u)^T y^{t+1}_l + w_d s^t (d^{t+1}_p) \tag{6}$$

$$\Theta^* = argmin_\Theta \sum_{t=1}^{t=|S_u|} (-ln\sigma(\widehat{x}^t_{ul} - \widehat{x}^t_{ul'})) + \frac{\lambda}{2} \|\Theta\|^2 \tag{7}$$

Different with the centralized learning framework, we update Θ at the user local devices based on the loss value of S_u, instead of updating Θ at the cloud based on the sum of the loss value of all users.

Table 1. Experiment Platform

Configuration	Entities		
	Mobile End Device	Edge Server	Cloud
CPU	8*Snapdragon 660@2.2GHZ	4*Intel(R)Core(TM) i5-4590 CPU@3.30GHZ	20*Intel(R) Xeon(R) CPU E5-2660 v3@2.60GHz
Memory	4GB	8GB	62GB
System	Android 7.1	Windows 10	Ubuntu 16.04

In *regional model aggregation* module, the edge server iteratively collects parameter sets $\Theta = \{Y, D, W\}$ from users in coverage and performs the aggregation. Different with the centralized learning framework, our proposal keeps user latent matrix X invisible to the third party. Each mobile device trains POI recommendation model based on the aggregated Θ received from the edge server in each round of interactions in replace of the Θ obtained by gradient update in centralized learning.

5 EVALUATION

In this section, we conduct extensive experiments on two real-world datasets and two state-of-the-art POI recommendation models to answer the following research questions:

RQ1: PREFER strengthens privacy protection by not exposing user check-in data to third parties. Compared with the centralized recommendation framework that requires all user check-in data to be exposed and uploaded, will our recommendation quality be affected? Compared with the existing privacy protection recommendation frameworks, how does PREFER perform in terms of recommendation quality and compatibility with advanced POI models?

RQ2: PREFER aggregates model parameters in the edge server. How efficient is PREFER compared with the traditional recommendation framework that trains model at the cloud and the traditional federated method that aggregates users' model parameters at the cloud?

In the following, we first describe the settings of our experiments and then evaluate our system framework based on the above questions.

5.1 Experimental Settings

5.1.1 Experiment Platform. We implement a prototype system to conduct our experiments and measure recommendation efficiency. The experiment platform is shown in Table 1. The battery capacity of the smartphone is 3000mAh. And the software AidLearning [1] is installed on the device to provide a python programming platform for training recommendation models. Totally one smartphone, one laptop and one remote server are involved to act as the terminal device, the edge server and the cloud, respectively. When conducting the experiment, both the smartphone and the laptop are endowed with multi-identities to virtually simulate a group of end devices and edge servers. By loading different user data for different identities, the virtual group is able to perform different computation tasks. During the evaluation, we attain and record each virtual entity's performance metrics, and calculate the average value as the overall performance of PREFER. As for the network connection, the smartphone and the laptop are in the same local area network, and the network between the smartphone and the cloud center is a wide area network. The network bandwidth is 10M/s and 1M/s, respectively.

5.1.2 Datasets. We conduct experiments on two real-world datasets, Foursquare and Gowalla. In the preprocessing, we remove the locations with less than 10 visitors, and users with less than 10 check-in activities by following the existing works [11]. The Foursquare dataset is composed of 142105 check-ins made by 1954 users over 2589 positions, and the Gowalla dataset is composed of 130594 check-ins made by 2606 users over 2561 positions.

Table 2. Statistics of Data sets

Dataset	Region ID	User#	Location#	Record#	Sparsity	The number of trainable model parameters	
						PRME-G	Distance2Pre
Foursquare	1	1883	2589	138466	0.9716	155400	90557
	2	886	2582	64679	0.9717	154980	90340
	3	1048	2583	71614	0.9735	155040	90371
	4	728	2556	55695	0.9719	153420	89534
Gowalla	1	2065	2550	111289	0.9789	153060	99505
	2	484	2080	19686	0.9804	124860	84965
	3	837	2327	32990	0.9831	139680	92622

Their sparsity is 0.9719 and 0.9804. In the training and testing stage, we apply the leave-one-out technique, that is, leave the latest check-in activity for evaluation and the other activities for model training. Note that each user's test location is not contained in his/her train location set [16]. To simulate geographical division, we use K-means method to divide the whole region into 4 or 3 non-overlapping small regions based on the coordinate data. And we set that if the frequency of a user's visit to a region is less than 0.1, the region will not obtain the user's characteristics, that is, the user does not participate in the federated learning process of the region, otherwise the region may obtain the user's characteristics. The dataset information of each region is presented in Table 2.

5.1.3 Recommendation Models. We reconstruct two advanced POI recommendation models to prove our system framework's compatibility. The two models are 1) PRME-G [15]: The method improves the traditional matrix factorization method by using Markov chain model to model the sequential preference and incorporating spatial preference. 2) Distance2Pre [11]: The method is also based on matrix factorization and uses RNN model to analyze user sequential and spatial preference and improve its quality. The programming language is Python. The latent size k of two models is set to 30, and the number of trainable parameters of two models in different regions is shown in Table 2.

5.1.4 Metrics. To evaluate recommendation quality, We require each algorithm to provide each user with top-K recommendation locations among all his/her remaining unvisited locations and these locations' rank scores. The value of K is set to 5, 10, 15 and 20, respectively.

We use two popular metrics, HR and NDCG. Users usually have limited time and attention to view the list of recommendations, so we should show the top recommendations first, maybe we can promote them more actively. HR reflects whether the test location is in our recommendation list, and NDCG further reflects the ranking of the test location in the recommendation list. The definition of HR@K and NDCG@K are given as follows: 1) HR@K: It refers to Hit Ratio. $rel_i(u)$ refers to whether the i_{th} item in user u's recommendation list hits the test location (1 for yes and 0 for no). 2) NDCG: It refers to Normalized Discounted Cumulative Gain. It assigns higher scores to the hits at higher positions of the ranking list. It's an extension of DCG (Discounted Cumulative Gain) for recommendation ranking scenario. We give the definition of DCG and NDCG in Equation 8 and 9. IDCG@K refers to the ideal DCG.

$$DCG@K(u) = \sum_{i=1}^{K} \frac{2^{rel_i(u)} - 1}{log_2(i+1)} \tag{8}$$

$$NDCG@K = \frac{1}{U} \sum_{u=1}^{U} \frac{DCG@K(u)}{IDCG@K(u)} \tag{9}$$

Table 3. Comparison with state-of-the-art works

	CCL	CCMF	LLRec	PartialFL _cloud	NoisedFL _cloud	FL_cloud	PREFER
Where to train	cloud	cloud	end (student)& cloud (teacher)	end	end	end	end
What to train	user data	target domain data + noised auxiliary domain data	public user data (cloud) + private user data(end)	user data	user data	user data	user data
Where to aggregate	/	/	/	cloud	cloud	cloud	edge
What to aggregate	/	/	/	location matrix	noised user vector + noised location matrix	location matrix + time&distance -related matrices + model weights	location matrix + time&distance -related matrices + model weights

To evaluate recommendation efficiency, we measure the time taken to complete the whole procedure. In our proposal, this procedure can be divided into local training in the mobile end device, parameter transmission and model aggregation in the edge server. The time taken by these three sub-processes is defined as t_{train_end}, t_{trans_ee} and t_{aggre_edge}, respectively. The total time is the sum of the above three, notated as T. For baselines, t_{train_cloud} refers to the time taken to train model at the cloud, and t_{trans_ec} refers to the time taken to transmit user data or parameters to the cloud, t_{aggre_cloud} refers to the time taken to aggregate model parameters in the cloud. And some baselines need to disturb the interactive information, thus we notate the time required for this step as t_{noise}. Besides we also measure the end device's resource consumption on memory and energy.

5.1.5 Baselines. We compare our scheme with the following baselines. The major differences among them lie in four aspects: *where to train*, *what to train*, *where to aggregate* and *what to aggregate*. We summarize their differences in Table 3. The mobile device is abbreviated to *end*.

(1) CCL[11, 15]: It means Centralized Learning at the Cloud. In this framework, the raw user check-in data is uploaded to the cloud, and then the cloud will execute the recommendation model. It represents the most ideal recommendation quality without any consideration of user privacy.
(2) CCMF [16]: It means Without Category Confidence-aware Collective Matrix Factorization. In this framework, the raw user data is also uploaded to the cloud, but belongs to two domains, auxiliary domain and target domain. The auxiliary domain applies geographic differential privacy to disturb the raw data and share the noised data to target domain for privacy security. And the target domain constructs the confidence matrix for the noised data to reduce the noise interference. Due to the lack of category information of location, we are reluctant to apply CCMF without category information. But it has little effect on our evaluation comparison because the input data of all the frameworks is the same. The evaluation of [16] is done by randomly selecting 100 locations from the unvisited locations and ranking the test locations among the 100 locations. In order to unify the evaluation method, we will rank all unvisited locations and judge whether the test location is in the top-K list.
(3) LLRec[42]: This framework adopts teacher-student training framework. The cloud trains a teacher RNN-based model with some public data and sends it to the end device to assist the local student model training. In this way, end devices keep private user data local without exposure to the cloud and the student model is simplified based on the teacher model to reduce the heavy computation burden for end devices. The

evaluation of [42] is done by randomly selecting 300 unvisited locations and ranking the test location among the 300 locations. We also unify the evaluation method in our comparison.

(4) PartialFL_cloud[19]: It means Partial Federated Learning in the cloud. In this framework, users train model locally and upload only the model's item matrix to the cloud for aggregation.
(5) NoisedFL_cloud[12]: It means Noised Federated Learning in the cloud. Users also train model locally but upload both user vector and item matrix to the cloud. It applied differential privacy to protect these vectors.
(6) FL_cloud: It means Federated Learning in the cloud. In this framework, users share item matrix, time&distance-related matrices and model weights to the cloud. It's our initial version of PREFER which does not take efficiency into consideration and is not accelerated by edge computing.

LLRec is specific to its own recommendation model, and other data protection ideas can be applied to any recommendation model based on matrix factorization. Thus, to eliminate the interference of recommendation model, we will reconstruct PRME-G and Distance2Pre based on these frameworks except for LLRec and compare the recommendation quality and efficiency under different frameworks.

5.1.6 Parameter Settings. In baseline CCL, the number of training epochs for the two above recommendation models is 100. In baseline CCMF, we set 70% of the data as the auxiliary domain and the remaining 30% as the target domain, and set the parameter ϵ in geographic differential privacy algorithm as 10. These settings are referred to [16]. The number of training epochs for recommendation models is also 100. In baseline LLRec, we set 70% of the data as public user data stored in the cloud and refer to the data segmentation preprocessing step in [42]. And we set the number of teacher training epochs and student training epochs as 10 and 93, respectively. In the latter three baselines and our proposal PREFER, we randomly select 50% of participants in each interaction round and the selected participants download the model base from the server and then update the model locally based on their local data. We set the number of interaction rounds as 100, the number of local training epochs as 2. In this way, we control the possibility of each user check-in data participating in training in each scheme is the same.

5.2 Results and Discussion

5.2.1 Quality Comparison (RQ1). Firstly we evaluate the top-K recommendation quality of our proposal PREFER and the above baselines. We set the value of K to 5, 10, 15 and 20. Due to the space limitations, we only present the top-20 results in figures, and the remaining results are given in Appendix A. Figure 4 and Figure 5 demonstrate the top-20 recommendation quality comparison on two datasets using PRME-G model, and Figure 6 and Figure 7 demonstrate the quality using Distance2Pre model. From these results, we have the following observations:

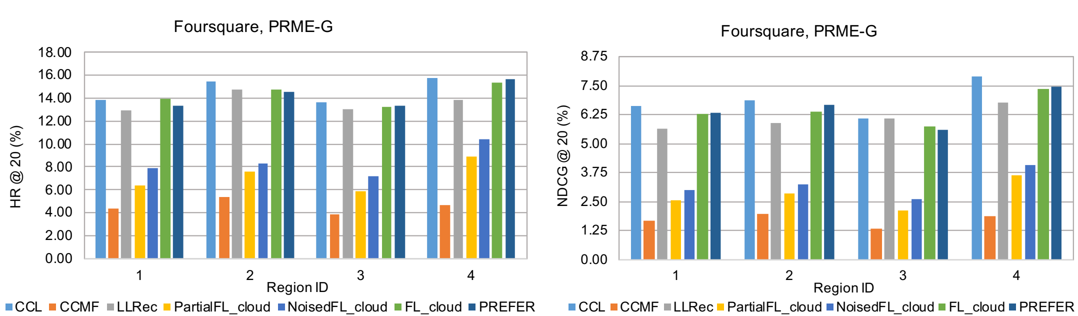

Fig. 4. Top-K recommendation quality comparison in different regions on Foursquare dataset using PRME-G model (K = 20)

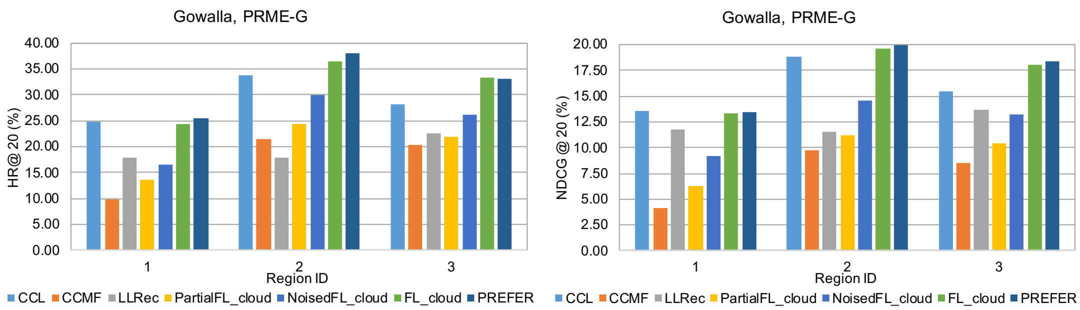

Fig. 5. Top-K recommendation quality comparison in different regions on Gowalla dataset using PRME-G model (K = 20)

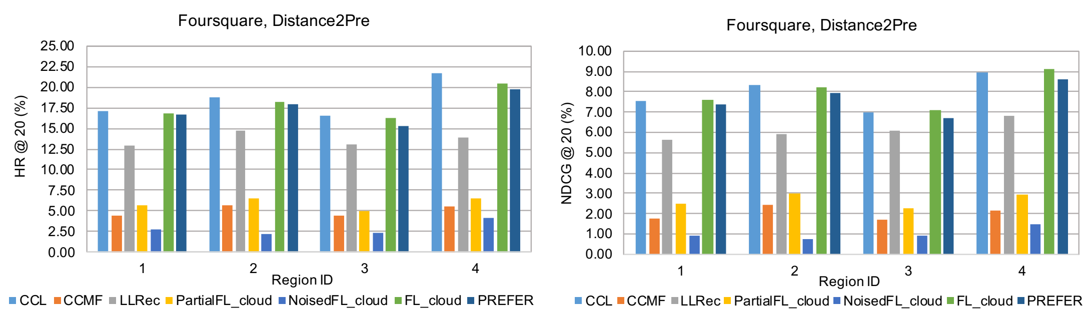

Fig. 6. Top-K recommendation quality comparison in different regions on Foursquare dataset using Distance2Pre model (K = 20)

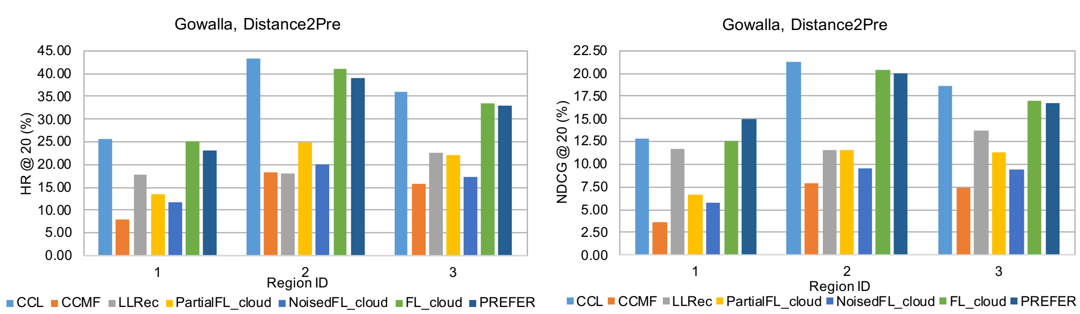

Fig. 7. Top-K recommendation quality comparison in different regions on Gowalla dataset using Distance2Pre model (K = 20)

Comparison to ideal but insecure baseline CCL. Baseline CCL represents the ideal and highest recommendation quality without discussion of recommendation privacy leakage and efficiency. And our proposed FL_cloud and PREFER take these factors into consideration and achieve near-optimal and even optimal recommendation quality in both two metrics. Specifically, the recommendation quality ratio of FL_cloud and CCL is 0.96−1.01 for HR@20 and 0.93−0.95 for NDCG@20 on Foursquare dataset using PRME-G model; 0.98−1.18 for HR@20 and 0.98−1.16 for NDCG@20 on Gowalla dataset using PRME-G model; 0.94−0.99 for HR@20 and 0.98−1.02 for NDCG@20 on Foursquare dataset using Distance2Pre model; 0.93−0.99 for HR@20 and 0.91−0.98 for NDCG@20 on Gowalla dataset using Distance2Pre model. And the recommendation quality of PREFER is also very close to CCL. The quality ratio of PREFER and CCL is above 0.90 no matter what real-world dataset, recommendation model and metrics are applied. **Thus, our proposed FL_cloud and PREFER strengthen user privacy protection with a little and acceptable sacrifice of recommendation quality.**

Comparison to existing privacy-preserving recommendation frameworks. Compared to baseline CCMF, our proposed PREFER improves NDCG@20 by 2.43−3.01 times and improves HR@20 by 1.69−2.35 times on Foursquare dataset using PRME-G. Compared to baseline LLRec, FL_cloud and PREFER still achieve better recommendation quality with an increase of 11%−34% on HR@20 and 17%−43% on NDCG@20 on Foursquare dataset using Distance2Pre. It's because FL_cloud and PREFER adopt multi-direction learning mode rather than one-direction teacher-student learning mode. The remaining results on other datasets and models also prove the significant quality improvement. In terms of privacy protection, FL_cloud and PREFER need not force any users to expose their check-in data in the cloud and shows stronger protection. Baseline PartialFL_cloud and NoisedFL_cloud present lower quality. For PRME-G model, PartialFL_cloud is worse than NoisedFL_cloud, while for Distance2Pre, PartialFL_cloud outperforms NoisedFL_cloud. It's because item vectors and user vectors have different importance on different models. Both of them are greatly lower than our proposal. It proves that the share of time&distance matrices is very important to guarantee recommendation quality. **Our proposed FL_cloud and PREFER achieve the highest recommendation quality among these privacy-preserving recommendation frameworks.**

Comparison to FL_cloud. Even that PREFER only aggregates model parameters from users who often visit a certain region while FL_cloud aggregates all users' models, PREFER performs almost the same as FL_cloud. The recommendation quality ratio of PREFER and FL_cloud in each region is always stable between 97% and 99%. It proves that it's not very necessary for collecting all users' model parameters at one point. The preference from users who do not visit in a certain region is less helpful to improve recommendation quality for users in the region. **Our proposed PREFER achieves almost the same recommendation quality as FL_cloud, but is more efficient than FL_cloud (as shown in 5.2.2).**

Compatibility analysis and comparison of different privacy protection frameworks for recommendation models. We tried two different recommendation models to evaluate different frameworks, and whatever the model is adopted, FL_cloud and PREFER always perform better. The quality gap between the ideal CCL and privacy-preserving baselines fluctuates when using different models, but our proposed FL_cloud and PREFER always remain higher and stabler quality. It proves that **our proposal is more compatible with the mature and complex recommendation model**.

In summary, experimental results on two datasets and two recommendation models demonstrate that our proposed FL_cloud and PREFER achieve near-optimal recommendation quality to the ideal training methods and outperform these existing state-of-the-art methods. They are also more practical for the gradually mature recommendation model and the growing awareness of privacy protection.

5.2.2 Efficiency Comparison (RQ2). Secondly, we focus on the recommendation efficiency of our proposed PREFER and the above baselines. We measure and compare each step's time consumption for each framework. The results are shown in Figure 8 and Figure 9. From these results, we have found the following points:

Comparison to the traditional cloud training framework (baselines CCL and CCMF). Baseline CCL and CCMF represent the traditional cloud training framework. The bulky calculation of matrix easily takes up lots of memory and CPU resources in the cloud server and requires a long time to be completed. It's measured that t_{train_cloud} for CCL and CCMF on Foursquare using Distance2Pre reached 4302.12s and 4349.86s, using PRME-G reached

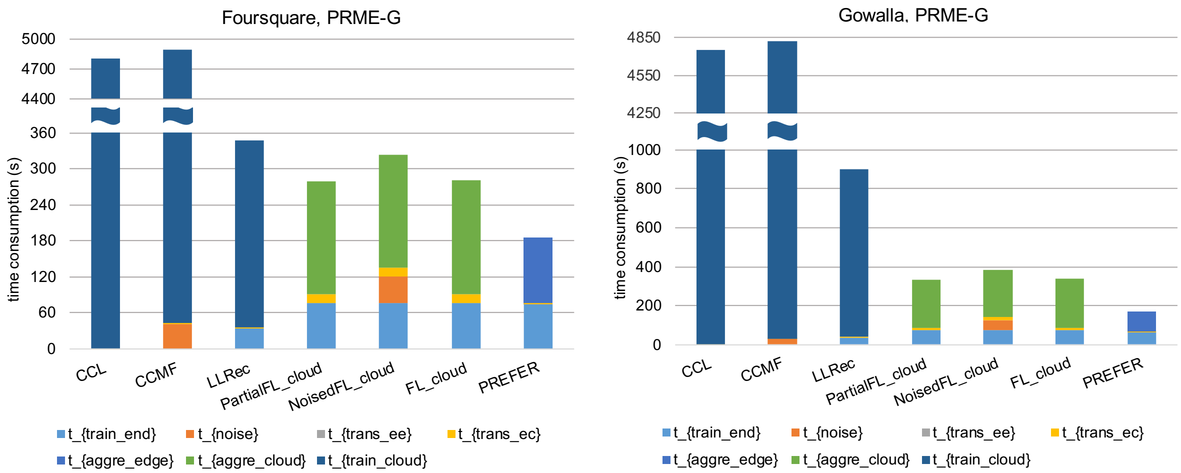

Fig. 8. Time consumption comparison in each step on two datasets using PRME-G model

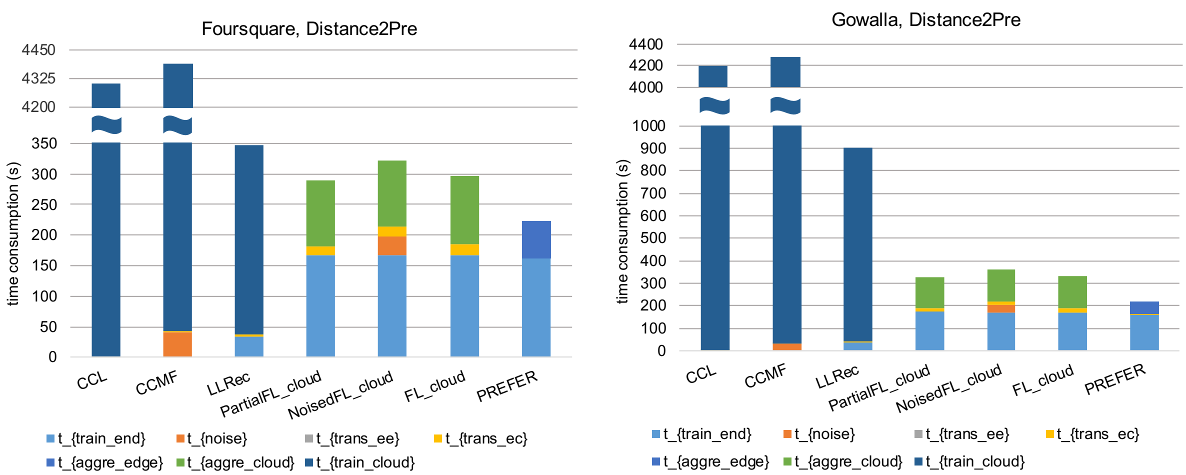

Fig. 9. Time consumption comparison in each step on two datasets using Distance2Pre model

4200.00s and 4243.50s, respectively. For Gowalla, t_{train_cloud} for CCL and CCMF using Distance2Pre reached 4800.02s and 4750.11s, using PRME-G reached 4850.86s and 4780.53s, respectively. CCMF need extra time to add noise when sharing to the target domain. This consumption is relatively small, 41.62s and 30.61s for Foursquare and Gowalla. And since users only need to upload check-in data to the cloud once, the transmission time is negligible. As for PREFER, it utilizes these distributed devices to perform tasks in parallel, and each calculation task in each device is affordable. The average overall time consumption is reduced by 96.29% and 96.36% for PRME-G, and 94.80% and 94.91% for Distance2Pre. Thus, **PREFER significantly reduces the overall time consumption of recommendations.**

Comparison to the unidirectional teacher(cloud)-student(end) training framework (baseline LLRec). Baseline LLRec is designed for reducing the computation burden on mobile end devices. We could found that the training time consumption is the smallest among these methods. But it still needs relatively long time to pre-train a teacher model in the cloud. The average overall time consumption in PREFER is reduced by 78.66% for PRME-G and 41.37% for Distance2Pre. Thus, **PREFER also shows better recommendation efficiency.**

Comparison to the existing bidirectional cloud-end federated framework (baselines PartialFL_cloud, NoisedFL_cloud and FL_cloud). One major difference between them is the aggregation point. These baseline methods require end devices to upload and aggregate model parameters in the cloud, and we assign edge servers to replace the cloud. We observed that the local training time t_{train_end} is roughly equal among them, the transmission time t_{trans_ee} is greatly less than t_{trans_ec} owing to the short transmission distance, and the aggregation time t_{aggre_edge} is less than t_{aggre_cloud} because the edge server only aggregates parameters from users in its service region rather than users in all regions. With the number of rounds of parameter exchange and the total number of users increase, their gap will further increase. Compared to these baselines, PREFER reduces the average overall time consumption by 41.77%, 49.8% and 42.6% for PRME-G, and 28.43%, 35.27% and 29.65% for Distance2Pre. Thus, **PREFER shortens the time consumption on parameter transmission and aggregation with the benefit of the edge server.**

We also evaluate the resource consumption of the end devices. For CCL and CCMF, they expose user privacy to the cloud and the main resource consumption is generated in the cloud. The end device only uploads local check-in data and receives the feedback. The resource consumption of the device is negligible and not affected by the adopted model. For LLRec, training the local student model occupies 1.7% of the memory and consumes 3.81mAh. For PartialFL_cloud, NoisedFL_cloud and FL_cloud, they all take up 3% of the memory in PRME-G, 7% of the memory of Distance2Pre. Their energy consumption is 56.5mAh, 75.5mAh, 57.1mAh in PRME-G, and 47.9mAh, 48.9mAh, 57.5mAh in Distance2Pre. The differences are caused by different amounts of transmission parameters. These baselines have the same amount of calculation for local training which consumes 7.9mAh in PRME-G and 12.7mAh in Distance2Pre. For PREFER, the memory consumption is the same with FL_cloud series. The edge device consumes 16.7mAh in PRME-G and 37.7mAh in Distance2Pre. The energy consumption is saved compared to FL_cloud series owing to the shorter transmission distance. The local training in PREFER requires 7.3mAh in PRME-G and 12.3mAh in Distance2Pre. End devices in PREFER have a lighter computation burden compared to FL_cloud series because PREFER considers the locations that users in the same region have visited rather than all locations. Considering the resource condition of the end device and the benefit of privacy protection, the resource consumption of PREFER is acceptable.

In summary, the evaluation results show that our proposed PREFER is more efficient than these baseline methods in completing complex recommendation tasks.

6 CONCLUSION

In this paper, we propose PREFER, an edge-accelerated federated learning framework for POI recommendation. In the system, in view of the privacy concerns on user locations, users collect and keep their sensitive check-in activities in their local devices, and then train their local recommendation models. Then, considering high dependency on network quality and demand for real-time response, users upload multi-dimensional user-independent model parameters with the edge server in the same region. The edge server aggregates these parameters to assist to build a more representative recommendation model. Experimental results demonstrate that our proposal achieves very similar recommendation quality to the centralized learning framework but greatly strengthens user privacy protection and improves the recommendation efficiency. And our proposal is more efficient and compatible with advanced recommendation models than existing state-of-the-art frameworks.

In the future, we will discuss more sophisticated POI recommendation models that take into consideration social relationship factors. Specifically, we expect to improve the aggregation step by setting different weights to each user's model parameters based on their social relationship. We strongly believe that the compatibility with existing advanced recommendation models should be paid more attention to system design. And we will also discuss how to perform cross-domain model parameter aggregations among multiple adjacent edge servers when the edge server gets more close to users and covers few users. It could be a promising direction for optimizing POI recommendation systems.

ACKNOWLEDGMENTS

This work is supported by National Natural Science Foundation of China-Guangdong Joint Fund (NSFCU1811461), National Natural Science Foundation of China (61832020, 62072465), National Key Research and Development Program of China (2018YFB0204301), Natural Science Foundation of Guangdong Province (2018B030312002) and Louisiana Board of Regents under Contract LEQSF(2019-22)-RD-A-21. Fang Liu (fangl@hnu.edu.cn) and Nong Xiao (nongxiao@nudt.edu.cn) are the corresponding authors.

REFERENCES

[1] [n.d.]. *AidLearning*. Retrieved May 27, 2020 from http://www.aidlearning.net/index.html

[2] 2019. *Robot 4.0 white paper: robot system and architecture of cloud-edge-end integration.* Technical Report. Intel. http://robotplaces.mikecrm.com/kdQ63Ad

[3] Miguel E. Andrés, Nicolás Emilio Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: differential privacy for location-based systems. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*. 901–914. https://doi.org/10.1145/2508859.2516735

[4] Louise Barkhuus. 2012. The mismeasurement of privacy: using contextual integrity to reconsider privacy in HCI. In *CHI Conference on Human Factors in Computing Systems, CHI '12, Austin, TX, USA - May 05 - 10, 2012*. 367–376. https://doi.org/10.1145/2207676.2207727

[5] Betim Berjani and Thorsten Strufe. 2011. A recommendation system for spots in location-based online social networks. In *Proceedings of the 4th Workshop on Social Network Systems, Salzburg, Austria, April 10, 2011*. 4. https://doi.org/10.1145/1989656.1989660

[6] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konecný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards Federated Learning at Scale: System Design. *CoRR* abs/1902.01046 (2019). arXiv:1902.01046 http://arxiv.org/abs/1902.01046

[7] Ioannis Boutsis and Vana Kalogeraki. 2016. Location privacy for crowdsourcing applications. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016, Heidelberg, Germany, September 12-16, 2016*. 694–705. https://doi.org/10.1145/2971648.2971741

[8] Hancheng Cao, Jie Feng, Yong Li, and Vassilis Kostakos. 2018. Uniqueness in the City: Urban Morphology and Location Privacy. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2 (2018), 62:1–62:20. https://doi.org/10.1145/3214265

[9] Chaochao Chen, Ziqi Liu, Peilin Zhao, Jun Zhou, and Xiaolong Li. 2018. Privacy Preserving Point-of-Interest Recommendation Using Decentralized Matrix Factorization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, , the 30th innovative Applications of Artificial Intelligence , and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence , New Orleans, Louisiana, USA, February 2-7, 2018*. 257–264. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16123

[10] Chen Cheng, Haiqin Yang, Irwin King, and Michael R. Lyu. 2012. Fused Matrix Factorization with Geographical and Social Influence in Location-Based Social Networks. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.* http://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/view/4748

[11] Qiang Cui, Yuyuan Tang, Shu Wu, and Liang Wang. 2019. Distance2Pre: Personalized Spatial Preference for Next Point-of-Interest Prediction. In *Advances in Knowledge Discovery and Data Mining - 23rd Pacific-Asia Conference, PAKDD 2019, Macau, China, April 14-17, 2019, Proceedings, Part III*. 289–301. https://doi.org/10.1007/978-3-030-16142-2_23

[12] Koustabh Dolui, Illapha Cuba Gyllensten, Dietwig Lowet, Sam Michiels, Hans Hallez, and Danny Hughes. 2019. Towards Privacy-preserving Mobile Applications with Federated Learning: The Case of Matrix Factorization. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys 2019, Seoul, Republic of Korea, June 17-21, 2019*. 624–625. https://doi.org/10.1145/3307334.3328657

[13] Jie Feng, Can Rong, Funing Sun, Diansheng Guo, and Yong Li. 2020. PMF: A Privacy-preserving Human Mobility Prediction Framework via Federated Learning. *IMWUT* 4, 1 (2020), 10:1–10:21. https://doi.org/10.1145/3381006

[14] Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. POI2Vec: Geographical Latent Representation for Predicting Future Visitors. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. 102–108. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14902

[15] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, Qiang Yang and Michael J. Wooldridge (Eds.). AAAI Press, 2069–2075. http://ijcai.org/Abstract/15/293

[16] Chen Gao, Chao Huang, Yue Yu, Huandong Wang, Yong Li, and Depeng Jin. 2019. Privacy-preserving Cross-domain Location Recommendation. *IMWUT* 3, 1 (2019), 11:1–11:21. https://doi.org/10.1145/3314398

[17] Jean-Benoît Griesner, Talel Abdessalem, and Hubert Naacke. 2015. POI Recommendation: Towards Fused Matrix Factorization with Geographical and Temporal Influences. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015*. 301–304. https://dl.acm.org/citation.cfm?id=2799679

[18] Yeting Guo, Fang Liu, Zhiping Cai, Li Chen, and Nong Xiao. 2020. FEEL: A Federated Edge Learning System for Efficient and Privacy-Preserving Mobile Healthcare. In *ICPP 2020: 49th International Conference on Parallel Processing, Edmonton, AB, Canada, August 17-20, 2020*. 9:1–9:11. https://doi.org/10.1145/3404397.3404410

[19] István Hegedüs, Gábor Danner, and Márk Jelasity. 2019. Decentralized Recommendation Based on Matrix Factorization: A Comparison of Gossip and Federated Learning. In *Machine Learning and Knowledge Discovery in Databases - International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16-20, 2019, Proceedings, Part I*. 317–332. https://doi.org/10.1007/978-3-030-43823-4_27

[20] Teruo Higashino, Hirozumi Yamaguchi, Akihito Hiromori, Akira Uchiyama, and Keiichi Yasumoto. 2017. Edge Computing and IoT Based Research for Building Safe Smart Cities Resistant to Disasters. In *37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017, Atlanta, GA, USA, June 5-8, 2017*. 1729–1737. https://doi.org/10.1109/ICDCS.2017.160

[21] Saeid Hosseini and Lei Thor Li. 2016. Point-Of-Interest Recommendation Using Temporal Orientations of Users and Locations. In *Database Systems for Advanced Applications - 21st International Conference, DASFAA 2016, Dallas, TX, USA, April 16-19, 2016, Proceedings, Part I*. 330–347. https://doi.org/10.1007/978-3-319-32025-0_21

[22] Yi-Hsuan Hung, Chih-Yu Wang, and Ren-Hung Hwang. 2020. Optimizing Social Welfare of Live Video Streaming Services in Mobile Edge Computing. *IEEE Trans. Mob. Comput.* 19, 4 (2020), 922–934. https://doi.org/10.1109/TMC.2019.2901786

[23] Congfeng Jiang, Yeliang Qiu, Honghao Gao, Tiantian Fan, Kangkang Li, and Jian Wan. 2019. An Edge Computing Platform for Intelligent Operational Monitoring in Internet Data Centers. *IEEE Access* 7 (2019), 133375–133387. https://doi.org/10.1109/ACCESS.2019.2939614

[24] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. 2007. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*. 106–115. https://doi.org/10.1109/ICDE.2007.367856

[25] Fang Liu, Guoming Tang, Youhuizi Li, Zhiping Cai, Xingzhou Zhang, and Tongqing Zhou. 2019. A Survey on Edge Computing Systems and Tools. *Proc. IEEE* 107, 8 (2019), 1537–1562. https://doi.org/10.1109/JPROC.2019.2920341

[26] Lumin Liu, Jun Zhang, S. H. Song, and Khaled Ben Letaief. 2019. Edge-Assisted Hierarchical Federated Learning with Non-IID Data. *CoRR* abs/1905.06641 (2019). arXiv:1905.06641 http://arxiv.org/abs/1905.06641

[27] Yanchi Liu, Chuanren Liu, Bin Liu, Meng Qu, and Hui Xiong. 2016. Unified Point-of-Interest Recommendation with Temporal Interval Assessment. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 1015–1024. https://doi.org/10.1145/2939672.2939773

[28] King's College London. 2019. *King's and NVIDIA join forces to build UK's rst Al platform for hospitals.* Technical Report. King's College London. King's College London. https://www.kcl.ac.uk/news/kings-and-nvidia-to-build-uks-first-al-platform

[29] Sidi Lu, Yongtao Yao, and Weisong Shi. 2019. Collaborative Learning on the Edges: A Case Study on Connected Vehicles. In *2nd USENIX Workshop on Hot Topics in Edge Computing, Renton, WA, USA, July 9, 2019*. https://www.usenix.org/conference/hotedge19/presentation/lu

[30] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. 2006. l-Diversity: Privacy Beyond k-Anonymity. In *Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, 3-8 April 2006, Atlanta, GA, USA*. 24. https://doi.org/10.1109/ICDE.2006.1

[31] Takayuki Nishio and Ryo Yonetani. 2019. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. In *2019 IEEE International Conference on Communications, ICC 2019, Shanghai, China, May 20-24, 2019*. 1–7. https://doi.org/10.1109/ICC.2019.8761315

[32] Guanhua Qiao, Supeng Leng, Sabita Maharjan, Yan Zhang, and Nirwan Ansari. 2020. Deep Reinforcement Learning for Cooperative Content Caching in Vehicular Edge Computing and Networks. *IEEE Internet of Things Journal* 7, 1 (2020), 247–257. https://doi.org/10.1109/JIOT.2019.2945640

[33] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*. 452–461. https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25

[34] Sumudu Samarakoon, Mehdi Bennis, Walid Saad, and Mérouane Debbah. 2020. Distributed Federated Learning for Ultra-Reliable Low-Latency Vehicular Communications. *IEEE Trans. Communications* 68, 2 (2020), 1146–1159. https://doi.org/10.1109/TCOMM.2019.2956472

[35] Constantine Stephanidis (Ed.). 2019. *HCI International 2019 - Posters - 21st International Conference, HCII 2019, Orlando, FL, USA, July 26-31, 2019, Proceedings, Part III*. Communications in Computer and Information Science, Vol. 1034. Springer. https://doi.org/10.1007/978-3-030-23525-3

[36] Chang Su, Yumeng Chen, and Xianzhong Xie. 2019. Location Recommendation with Privacy Protection. *ISMSI 2019: Proceedings of the 2019 3rd International Conference on Intelligent Systems, Metaheuristics Swarm Intelligence*, 83–91. https://doi.org/10.1145/3325773.3325787

[37] Sukhmani Sukhmani, Mohammad Sadeghi, Melike Erol-Kantarci, and Abdulmotaleb El-Saddik. 2019. Edge Caching and Computing in 5G for Mobile AR/VR and Tactile Internet. *IEEE Multim.* 26, 1 (2019), 21–30. https://doi.org/10.1109/MMUL.2018.2879591

[38] Latanya Sweeney. 2002. k-Anonymity: A Model for Protecting Privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 5 (2002), 557–570. https://doi.org/10.1142/S0218488502001648

[39] Zeyi Tao and Qun Li. 2018. eSGD: Communication Efficient Distributed Deep Learning on the Edge. In *USENIX Workshop on Hot Topics in Edge Computing, HotEdge 2018, Boston, MA, July 10, 2018.* https://www.usenix.org/conference/hotedge18/presentation/tao

[40] China Telecom. [n.d.]. China Telecom edge computing technology white paper. ([n. d.]).

[41] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2019. Protecting Trajectory From Semantic Attack Considering ${k}$ -Anonymity, ${l}$ -Diversity, and ${t}$ -Closeness. *IEEE Trans. Netw. Serv. Manag.* 16, 1 (2019), 264–278. https://doi.org/10.1109/TNSM.2018.2877790

[42] Qinyong Wang, Hongzhi Yin, Tong Chen, Zi Huang, Hao Wang, Yanchang Zhao, and Nguyen Quoc Viet Hung. 2020. Next Point-of-Interest Recommendation on Resource-Constrained Mobile Devices. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020.* 906–916. https://doi.org/10.1145/3366423.3380170

[43] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K. Leung, Christian Makaya, Ting He, and Kevin Chan. 2018. When Edge Meets Learning: Adaptive Control for Resource-Constrained Distributed Machine Learning. In *2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018.* 63–71. https://doi.org/10.1109/INFOCOM.2018.8486403

[44] Xiaofei Wang, Yiwen Han, Chenyang Wang, Qiyang Zhao, Xu Chen, and Min Chen. 2019. In-Edge AI: Intelligentizing Mobile Edge Computing, Caching and Communication by Federated Learning. *IEEE Netw.* 33, 5 (2019), 156–165. https://doi.org/10.1109/MNET.2019.1800286

[45] Xiwei Wang, Hao Yang, and Kiho Lim. 2018. Privacy-Preserving POI Recommendation Using Nonnegative Matrix Factorization. In *2018 IEEE Symposium on Privacy-Aware Computing, PAC 2018, Washington, DC, USA, September 26-28, 2018.* 117–118. https://doi.org/10.1109/PAC.2018.00018

[46] Webank. 2019. *An Industrial Level Federated Learning Framework.* Retrieved February 28, 2020 from https://github.com/FederatedAI/FATE

[47] Xiguang Wei, Quan Li, Yang Liu, Han Yu, Tianjian Chen, and Qiang Yang. 2019. Multi-Agent Visualization for Explaining Federated Learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019.* 6572–6574. https://doi.org/10.24963/ijcai.2019/960

[48] Di Xiao, Min Li, and Hongying Zheng. 2020. Smart Privacy Protection for Big Video Data Storage Based on Hierarchical Edge Computing. *Sensors* 20, 5 (2020), 1517. https://doi.org/10.3390/s20051517

[49] Yinhao Xiao, Yizhen Jia, Chun-Chi Liu, Xiuzhen Cheng, Jiguo Yu, and Weifeng Lv. 2019. Edge Computing Security: State of the Art and Challenges. *Proc. IEEE* 107, 8 (2019), 1608–1631. https://doi.org/10.1109/JPROC.2019.2918437

[50] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. 2017. Trajectory Recovery From Ash: User Privacy Is NOT Preserved in Aggregated Mobility Data. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017.* 1241–1250. https://doi.org/10.1145/3038912.3052620

[51] Dongdong Ye, Rong Yu, Miao Pan, and Zhu Han. 2020. Federated Learning in Vehicular Edge Computing: A Selective Model Aggregation Approach. *IEEE Access* 8 (2020), 23920–23935. https://doi.org/10.1109/ACCESS.2020.2968399

[52] Jun Zeng, Feng Li, Xin He, and Junhao Wen. 2019. Fused Collaborative Filtering With User Preference, Geographical and Social Influence for Point of Interest Recommendation. *Int. J. Web Service Res.* 16, 4 (2019), 40–52. https://doi.org/10.4018/IJWSR.2019100103

[53] Shenglin Zhao, Irwin King, and Michael R. Lyu. 2016. A Survey of Point-of-interest Recommendation in Location-based Social Networks. *CoRR* abs/1607.00647 (2016). arXiv:1607.00647 http://arxiv.org/abs/1607.00647

Table 4. Comparison on HR@k using Foursquare dataset and PRME-G model

	Foursquare Dataset, PRME-G model															
	HR@k in Region ID 1 (%)				HR@k in Region ID 2 (%)				HR@k in Region ID 3 (%)				HR@k Region ID 4 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20	5	10	15	20
CCL	7.01	9.88	12.16	13.81	7.22	10.61	13.66	15.46	6.11	9.16	11.74	13.65	8.79	12.09	13.74	15.80
CCMF	1.43	2.55	3.72	4.35	1.69	2.82	4.29	5.42	1.15	2.10	3.05	3.91	1.79	2.61	3.85	4.67
LLRec	6.05	8.28	10.83	12.96	6.32	9.82	12.98	14.79	6.01	7.54	10.88	13.07	7.69	10.03	12.77	13.87
PartialFL _cloud	2.28	4.30	5.52	6.43	2.14	5.19	6.55	7.56	1.81	3.34	4.77	5.92	3.43	5.91	7.97	8.93
NoisedFL _cloud	2.97	4.62	6.43	7.86	3.05	5.08	7.11	8.35	2.29	3.82	5.44	7.16	4.26	6.32	8.24	10.44
FL_cloud	6.32	9.19	12.06	13.97	6.55	10.16	12.87	14.79	5.73	8.59	11.93	13.26	7.55	10.85	13.60	15.38
PREFER	6.59	9.56	11.68	13.38	6.66	9.26	12.42	14.56	5.53	8.68	10.78	13.34	7.55	10.85	13.74	15.66

Table 5. Comparison on NDCG@k using Foursquare dataset and PRME-G model

	Foursquare Dataset, PRME-G model															
	NDCG@k in Region ID 1 (%)				NDCG@k in Region ID 2 (%)				NDCG@k in Region ID 3 (%)				NDCG@k in Region ID 4 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20	5	10	15	20
CCL	4.82	5.67	6.24	6.64	4.60	5.72	6.55	6.88	4.10	5.11	5.67	6.08	5.99	7.05	7.45	7.88
CCMF	0.93	1.22	1.50	1.66	1.03	1.33	1.68	1.95	0.64	0.92	1.17	1.35	1.21	1.46	1.64	1.86
LLRec	3.74	4.53	5.13	5.62	3.64	4.61	5.53	5.91	4.19	4.66	5.46	6.07	5.24	5.97	6.50	6.79
PartialFL _cloud	1.41	2.04	2.37	2.57	1.29	2.24	2.60	2.83	1.10	1.57	1.94	2.13	2.17	2.87	3.44	3.63
NoisedFL _cloud	1.75	2.27	2.71	3.00	1.80	2.33	2.87	3.24	1.44	1.86	2.23	2.62	2.75	3.37	3.76	4.08
FL_cloud	4.28	5.17	5.94	6.28	4.29	5.37	6.03	6.39	3.72	4.58	5.43	5.74	5.07	6.08	6.86	7.38
PREFER	4.45	5.26	5.90	6.32	4.51	5.30	6.10	6.68	3.71	4.73	5.19	5.61	5.36	6.43	7.05	7.47

A TOP-K RECOMMENDATION QUALITY COMPARISON

Table 6. Comparison on HR@k using Foursquare dataset and Distance2Pre model

	Foursquare Dataset, Distance2Pre model															
	HR@k in Region ID 1 (%)				HR@k in Region ID 2 (%)				HR@k in Region ID 3 (%)				HR@k Region ID 4 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20	5	10	15	20
CCL	7.49	11.42	14.45	17.05	8.92	13.32	16.59	18.74	6.97	10.50	14.22	16.60	8.79	14.01	19.09	21.70
CCMF	1.75	3.03	3.77	4.46	2.48	3.95	4.74	5.64	1.72	2.77	3.63	4.39	2.34	3.71	4.67	5.49
LLRec	6.05	8.28	10.83	12.96	6.32	9.82	12.98	14.79	6.01	7.54	10.88	13.07	7.69	10.03	12.77	13.87
PartialFL_cloud	2.60	3.88	5.15	5.74	3.39	4.97	6.43	6.55	2.19	3.24	4.29	4.96	3.16	3.98	5.49	6.46
NoisedFL_cloud	1.01	1.49	2.23	2.71	0.79	1.13	1.69	2.26	0.95	1.43	2.00	2.39	1.65	2.34	3.43	4.12
FL_cloud	7.49	11.15	14.50	16.83	8.35	12.41	15.35	18.28	6.87	10.40	13.36	16.22	8.93	13.74	17.86	20.47
PREFER	7.28	11.42	14.23	16.68	8.13	11.96	15.01	17.95	6.20	10.11	13.26	15.27	8.57	14.01	17.03	19.78

Table 7. Comparison on NDCG@k using Foursquare dataset and Distance2Pre model

	Foursquare Dataset, Distance2Pre model															
	NDCG@k in Region ID 1 (%)				NDCG@k in Region ID 2 (%)				NDCG@k in Region ID 3 (%)				NDCG@k in Region ID 4 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20	5	10	15	20
CCL	4.91	6.25	6.98	7.57	5.67	7.17	7.77	8.34	4.50	5.70	6.42	7.00	5.79	7.23	8.32	8.93
CCMF	1.04	1.39	1.64	1.76	1.64	2.03	2.24	2.42	0.95	1.30	1.51	1.73	1.30	1.68	1.94	2.13
LLRec	3.74	4.53	5.13	5.62	3.64	4.61	5.53	5.91	4.19	4.66	5.46	6.07	5.24	5.97	6.50	6.79
PartialFL_cloud	1.64	2.04	2.38	2.52	2.06	2.55	2.95	2.98	1.54	1.81	2.11	2.25	2.06	2.33	2.65	2.92
NoisedFL_cloud	0.53	0.68	0.82	0.91	0.46	0.54	0.69	0.76	0.55	0.69	0.82	0.91	0.89	1.08	1.29	1.45
FL_cloud	5.12	6.19	7.08	7.63	5.61	6.76	7.54	8.21	4.63	5.64	6.49	7.11	6.03	7.63	8.60	9.10
PREFER	5.00	6.22	6.84	7.38	5.50	6.69	7.37	7.96	4.36	5.31	6.18	6.72	5.75	7.03	8.05	8.64

Table 8. Comparison on HR@k using Gowalla dataset and PRME-G model

	Gowalla Dataset, PRME-G model											
	HR@k in Region ID 1 (%)				HR@k in Region ID 2 (%)				HR@k in Region ID 3 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20
CCL	15.01	19.66	22.86	24.94	21.51	26.76	30.82	33.81	17.37	22.29	25.71	28.24
CCMF	3.97	6.15	8.18	9.83	9.30	14.46	17.98	21.49	8.60	13.74	16.97	20.43
LLRec	12.45	15.06	16.56	17.92	14.46	16.94	17.36	17.98	14.22	21.15	22.10	22.70
PartialFL_cloud	6.30	9.30	12.06	13.61	10.33	14.67	19.83	24.38	10.04	14.81	19.00	21.98
NoisedFL_cloud	9.78	13.56	16.61	16.61	15.08	22.11	26.45	29.96	13.74	19.12	22.70	26.28
FL_cloud	14.77	19.27	22.28	24.41	20.87	29.13	33.47	36.57	20.19	26.64	30.11	33.45
PREFER	14.67	20.19	23.58	25.62	21.69	28.93	33.88	38.02	20.55	27.12	30.70	33.21

Table 9. Comparison on NDCG@k using Gowalla dataset and PRME-G model

	Gowalla Dataset, PRME-G model											
	NDCG@k in Region ID 1 (%)				NDCG@k in Region ID 2 (%)				NDCG@k in Region ID 3 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20
CCL	10.78	12.12	13.02	13.54	15.38	17.08	18.06	18.86	12.44	13.92	14.83	15.42
CCMF	2.61	3.20	3.73	4.15	6.20	7.80	8.75	9.74	5.48	7.16	7.90	8.53
LLRec	10.21	11.01	11.42	11.75	10.38	11.37	11.43	11.57	11.11	13.21	13.55	13.69
PartialFL_cloud	4.18	5.13	5.87	6.24	7.34	8.77	10.18	11.24	7.08	8.1	9.72	10.45
NoisedFL_cloud	6.67	7.89	8.65	9.16	10.44	12.74	13.83	14.56	9.75	11.48	12.43	13.17
FL_cloud	10.60	12.05	12.74	13.28	15.51	17.74	19.11	19.56	14.35	16.55	17.31	18.01
PREFER	10.20	12.01	12.95	13.45	15.43	17.70	18.90	19.89	14.73	16.78	17.67	18.34

Table 10. Comparison on HR@k using Gowalla dataset and Distance2Pre model

	Gowalla Dataset, Distance2Pre model											
	HR@k in Region ID 1 (%)				HR@k in Region ID 2 (%)				HR@k in Region ID 3 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20
CCL	14.43	19.32	22.76	25.52	23.35	32.64	39.26	43.39	19.95	27.12	31.78	36.08
CCMF	3.92	5.71	6.92	7.89	8.68	12.40	16.32	18.18	8.00	10.75	13.38	15.89
LLRec	12.45	15.06	16.56	17.92	14.46	16.94	17.36	17.98	14.22	21.15	22.10	22.70
PartialFL_cloud	7.51	10.80	12.40	13.41	11.16	17.15	20.87	25.00	11.59	17.32	19.71	22.10
NoisedFL_cloud	6.05	8.62	10.02	11.82	9.50	14.88	17.56	20.04	10.51	14.10	15.65	17.32
FL_cloud	14.00	18.89	22.62	25.23	22.73	31.61	36.98	41.12	19.24	25.69	30.59	33.57
PREFER	13.05	17.46	20.61	23.10	21.69	30.17	35.71	39.05	18.04	25.09	29.63	32.86

Table 11. Comparison on NDCG@k using Gowalla dataset and Distance2Pre model

	Gowalla Dataset, Distance2Pre model											
	NDCG@k in Region ID 1 (%)				NDCG@k in Region ID 2 (%)				NDCG@k in Region ID 3 (%)			
	5	10	15	20	5	10	15	20	5	10	15	20
CCL	9.71	11.44	12.27	12.83	15.98	18.84	20.54	21.28	14.44	16.46	17.59	18.69
CCMF	2.52	3.09	3.38	3.61	5.60	6.55	7.63	7.98	5.54	6.45	7.12	7.46
LLRec	10.21	11.01	11.42	11.75	10.38	11.37	11.43	11.57	11.11	13.21	13.55	13.69
PartialFL_cloud	5.00	6.04	6.46	6.71	7.80	9.58	10.63	11.60	8.38	10.15	10.82	11.34
NoisedFL_cloud	4.22	5.05	5.39	5.75	6.47	8.30	8.89	9.56	7.55	8.67	9.05	9.47
FL_cloud	9.66	11.25	12.07	12.61	15.08	18.15	19.31	20.33	13.16	15.44	16.32	17.03
PREFER	11.46	13.35	14.34	15.02	15.59	17.98	19.14	20.06	12.62	14.63	15.73	16.77