Personalized Location-Preference Learning for Federated Task Assignment in Spatial Crowdsourcing

Xiaolong Zhong
University of Electronic Science and Technology of China
xiaolongzhong2000@std.uestc.edu.cn

Hao Miao*
Department of Computer Science, Aalborg University
haom@cs.aau.dk

Dazhuo Qiu
Department of Computer Science, Aalborg University
dazhuoq@cs.aau.dk

Yan Zhao*
Department of Computer Science, Aalborg University
yanz@cs.aau.dk

Kai Zheng†
University of Electronic Science and Technology of China
zhengkai@uestc.edu.cn

ABSTRACT

With the proliferation of wireless and mobile devices, Spatial Crowdsourcing (SC) attracts increasing attention, where task assignment plays a critically important role. However, recent task assignment solutions in SC often assume that data is stored in a central station while ignoring the issue of privacy leakage. To enable decentralized training and privacy protection, we propose a federated task assignment framework with personalized location-preference learning, which performs efficient task assignment while keeping the data decentralized and private in each platform center (e.g., a delivery center of an SC company). The framework consists of two phases: personalized federated location-preference learning and task assignment. Specifically, in the first phase, we design a personalized location-preference learning model for each platform center by simultaneously considering the location information and data heterogeneity across platform centers. Based on workers' location preference, the task assignment phase aims to achieve effective and efficient task assignment by means of the Kuhn-Munkres (KM) algorithm and the newly proposed conditional degree-reduction algorithm. Extensive experiments on real-world data show the effectiveness of the proposed framework.

CCS CONCEPTS

• **Networks** → **Location based services**; • **Information systems** → *Geographic information systems*; • **Security and privacy**;

KEYWORDS

preference; task assignment; federated learning; spatial crowdsourcing

*Corresponding author: Hao Miao and Yan Zhao.
†Yangtze Delta Region Institute (Quzhou), School of Computer Science and Engineering, and Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3615008

ACM Reference Format:
Xiaolong Zhong, Hao Miao, Dazhuo Qiu, Yan Zhao, and Kai Zheng. 2023. Personalized Location-Preference Learning for Federated Task Assignment in Spatial Crowdsourcing. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3615008

1 INTRODUCTION

With the proliferation of edge devices (e.g., sensors), a new form of crowdsourcing, namely Spatial Crowdsourcing (SC), has emerged recently [7, 11, 31, 37, 47] including three important components: tasks, workers, and the platform. The SC platform requires workers with GPS devices to reach a specific location physically under certain spatio-temporal restrictions to perform spatial tasks. Thus, task assignment is one of the crucial tasks in SC, which benefits a range of real-world applications, such as disaster response [50], ride-hailing [21, 24, 32], and food delivery [26].

Privacy protection is critically concerned in task assignment [2, 18, 34]. Workers or platform centers are usually required to disclose their raw information (e.g., workers' locations), which can reveal their identity, for effective SC services. However, it is dangerous that real data is available to untrustworthy entities. Consequently, people will be reluctant to voluntarily share their data on an SC platform, resulting in low worker engagement. Previous studies on privacy protection in SC mainly focus on providing location privacy protection of workers or tasks [13, 27, 38], as well as secure computation of distance [18, 25]. However, most existing studies on privacy protection for task assignment in SC assume that the model is trained with centralized data gathered from edge devices and fails to handle the decentralized setting. To enable privacy protection and support data access restrictions due to existing licensing agreements, it is critical to utilize decentralized data in SC with the potential gains in lower latency, which calls for a new decentralized model that can perform effective task assignment, as well as protect privacy.

Some recent efforts have been made to use Federated Learning (FL) to perform decentralized training and protect the privacy of task assignment in SC [21, 28, 29]. FL is a machine learning setting where many clients (e.g., edge devices, or platforms), keeping local training data, collaboratively train a model under the orchestration of a central server, which can mitigate systematic privacy risks [14,

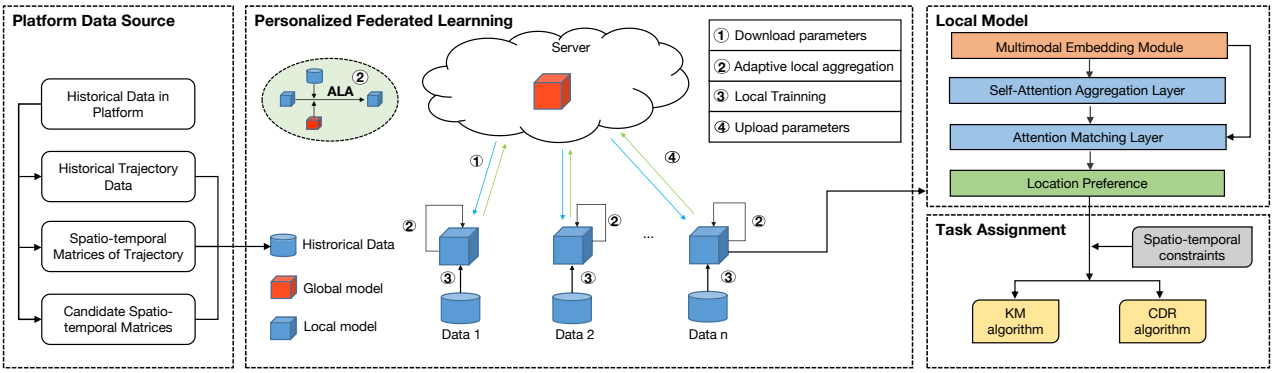


Figure 1: PTA-FLP Framework Overview

30]. It is critical for an SC platform to figure worker preference out, which is a key to ensuring continuous worker participation and satisfaction. A recent study [21] focuses on the worker preference of task category for each platform while ignoring the individual location preference, especially the destination information, for each worker. It is challenging to satisfy the preference of workers if task assignment are only based on the workers' current spatial-temporal (e.g., location) information and task requirements. Our insight is that different workers have different preferences for task destinations under specific spatiotemporal conditions, which may heavily impact their work performance. Intuitively, if workers are assigned tasks with their preferred locations, they are willing and dedicated to completing these tasks. For example, in the context of ride-hailing, workers prefer to finish their tasks at locations close to their homes, especially during the mid-night.

In addition, there lacks an off-the-shelf method that can handle data heterogeneity across platforms for federated task assignment in SC. Due to various rules and regulations across different platforms, the distributions of the data collected across different platforms are usually not independent and identically distributed (Non-IID). For example, the distributions of two platforms vary where one platform is with the workers for food delivery, while the other is with the workers for ride-hailing. In such cases, data heterogeneity across platforms will not lead to an optimal global model, thus degrading the model performance. Moreover, it is challenging to develop an efficient task assignment method in SC, which can help the platforms earn more revenue and motivate the workers.

To tackle the aforementioned problems, we formulate a location-aware task assignment problem and propose a personalized task assignment framework with federated location preference learning (PTA-FLP). As shown in Figure 1, the PTA-FLP framework consists of a personalized federated location-preference learning (PLP) and a task assignment (TA) phase. In the PLP phase, to model personalized location preference, we design a federated location preference model for each local platform center, and all local models are combined with a central server. The local model contains a multi-modal embedding module, a self-attention aggregation layer, and an attention matching layer, aiming to learn more reliable and effective location preferences. Additionally, we employ an adaptive local aggregation method to alleviate the effect of data heterogeneity across platforms. Specifically, this approach enables learning platform-specific parameters and adaptively aggregating local models on each platform to update the global model, allowing each platform to learn distinct model parameters. In the TA phase, we propose a location-aware task assignment algorithm that considers certain spatio-temporal constraints. The task assignment algorithm includes a location-aware KM algorithm and conditional degree-reduction (CDR) algorithm, by taking into account the location preference predicted by PLP and the distance to the task destination, improving efficiency. In particular, the location-aware KM algorithm assigns weights based on geographical distances, while the CDR algorithm prioritizes tasks with shorter distances.

In summary, our main contributions can be outlined as follows.

- We explore and investigate a novel problem of location preference based on spatial crowdsourcing, which utilizes workers' location preferences, especially the information of the destination.
- We propose a personalized task-assignment framework with federated location preference learning to ensure privacy protection, as well as handle data heterogeneity across platforms, by means of adaptive local aggregation.
- An efficient task assignment algorithm is proposed, which includes a location-aware KM and a conditional degree-reduction algorithm.
- We conduct extensive experiments on real-world data, offering evidence of the effectiveness and efficiency of the paper's proposals.

The remainder of this paper is organized as follows. Section 2 provides an overview of related works. Preliminary concepts and notations are introduced in Section 3. We then present the proposed personalized location-aware algorithm in Section 4. The experimental results are reported in Section 5, and Section 6 offers conclusions.

2 RELATED WORK

Spatial Crowdsourcing (SC) emerges recently, which involves three components: tasks, workers, and the platform, requiring workers with location-based (e.g., GPS) devices to reach a specific location physically under certain restrictions to perform spatial tasks [8, 11, 31, 36, 42]. Task assignment is one of the core algorithmic issues in SC [4, 5, 20, 33, 41, 48, 49], which can be divided into two categories: Worker-Selected Tasks (WST) and Server-Assigned Tasks (SAT) [15]. Regarding WST in SC, the platform server first publishes spatially aware tasks. Then, the online workers are allowed to independently select nearby tasks without negotiating with the server [9, 16]. Cheng et al. [6] study a prediction-based task assignment to improve global task assignment by considering both present

and future aspects of workers and tasks through predictions. In SAT, the platform server assigns suitable tasks to workers with the consideration of their locations [3, 5, 19, 43, 44, 46]. Lai et al. [17] investigated loyalty-aware task assignment in spatial crowdsourcing, while Xia et al. [35] focused on profit-oriented task assignment in spatial crowdsourcing. Nonetheless, these studies ignore the effects of task destination, which may reflect workers' preference. Privacy protection is a critical issue in SC. Existing studies mainly focus on location mask during task assignment [2, 27, 34, 40]. Further, Liu et al. proposed TA-FPL to find the optimal task assignment while considering worker's preference and protect worker's raw data [21]. However, data heterogeneity across clients (i.e., platforms) is not well-considered, which may cause performance degradation of task assignment.

3 PROBLEM DEFINITION

We proceed to present the necessary preliminaries and then define the problem addressed. Table 1 lists the notations used throughout the paper.

Table 1: Summary of Notation

Symbol	Definition	Symbol	Definition
s	Spatial task	$w.r$	Reachable radius of w
$s.o$	Origin Location of s	$w.speed$	Speed of worker w
$s.d$	Destination Location of s	$w.S$	A set of historical tasks of w
$s.p$	Publication time of s	pc	Platform center
$s.e$	Expiration time of s	$pc.l$	Location of pc
w	Worker	$pc.W$	A worker set of pc
$w.l$	Current location of w	A	A spatial task assignment
$w.t$	Current timestamp of w	$A.S$	Allocated task set of S
$w.pc$	Platform center of w	$\mathbb{A}$	Task assignment set

Definition 1 (Spatial Task). *A spatial task, denoted by $s = (o, d, p, e)$, has an origin location $s.or$, a destination location $s.de$, a publication time $s.p$, and an expiration time $s.e$.*

Unlike previous works that only focus on the origin of a task, we specify the location of the current task with a corresponding time instance and identify an origin and a destination. The origin is considered as a spatial constraint and the destination is a factor for task assignment.

Definition 2 (Worker). *A worker, which is denoted by $w = (l, t, PC, r, speed, S)$, has a current location $w.l$, associated with a current timestamp $w.t$, a platform center $w.pc$ that the worker w works for, a reachable radius $w.r$, a traveling speed $w.speed$, and a set of historical tasks $w.S$.*

The reachable area of worker w is centered around $w.l$ and constrained by a circle with a radius of $w.r$. Assignments within this reachable area are considered feasible for the worker. In practical scenarios, there is a strict constraint that a worker can only undertake a single task at any given time instance and can be associated with just one platform [15]. Note that we follow this restriction in this study.

Definition 3 (Platform Center). *A platform center, denoted by $pc = (l, W)$, includes a location $pc.l$ and a set of workers $pc.W$.*

Definition 4 (Spatial Task Assignment). *Given a set of platform centers PC, a set of tasks S, and a set of workers W, the spatial task assignment A is defined as a set of tuples $A = (pc, w, s)$. The spatial task assignment allocates each spatial task s to a worker w employed by a platform center pc, while all the spatial-temporal constraints of the workers and tasks are guaranteed.*

Based on the above definitions, the formal problem definition is formulated as follows.

Location-Aware Task Assignment with Privacy Protection. Consider a set of platforms PC that possess private local data (i.e., workers' historical task records and workers' locations) and workers' location preferences. Given a set of online workers W, and a set of tasks S at the current time instance, our problem is to determine an optimal task assignment A_{opt} that maximizes the total number of assigned tasks, i.e.,

$$\forall A_i \in \mathbb{A}, \quad |A_{opt}.S| \geq |A_i.S|, \tag{1}$$

where $\mathbb{A}$ denotes the set of all possible assignments and S denotes the task set of corresponding assignment. Privacy protection is guaranteed by federated learning.

4 ALGORITHM

In this section, we propose a personalized task assignment framework with federated location-preference learning, namely PTA-FLP. PTA-FLP consists of two phases: a personalized federated location-preference learning (PLP), described in Section 4.1, and a task assignment phase 4.2.

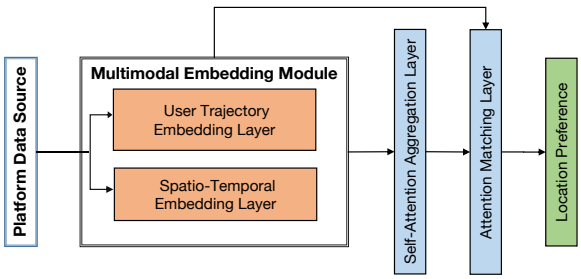


Figure 2: Location Preference Model

4.1 Personalized Federated Location-Preference Learning

Personalized Federated Location-Preference Learning consists of two components: Local Location-Preference Modeling for each Platform Center and Personalized Federated Training. Local Location-Preference Modeling aims to extract spatiotemporal features to predict workers' location preferences, while Personalized Federated Training protects the raw data of each platform and trains platform-specific model parameters. We will elaborate on the procedure of location-preferences learning with local data for each platform, and the proposed adaptive local aggregation method for personalized federated learning, respectively.

4.1.1 Local Location-Preference Learning. It is crucial to properly utilize the local platform data to learn effective workers' location preferences, which benefits the downstream task assignments. In preference learning, we design a local location-preference model by means of a multi-modal embedding module and attention mechanisms. As shown in Figure 2, the local location preference model

comprises three modules: multi-modal embedding module, self-attention aggregation layer, and attention matching layer. The details are illustrated as follows.

Data Preprocessing. Given a group of workers W with their time-ordered completed tasks S^c, we denote a task record as $tr = (o, d, ot, dt)$, where o and d refer to origin and destination, ot and dt are their corresponding timestamps. We transform the task location records into trajectories including an origin $tr^o = (o, ot)$ and a destination $tr^d = (d, dt)$. For a worker w, we denote his historical trajectory as $HT = \{(tr_1^o, tr_1^d), (tr_2^o, tr_2^d), ..., (tr_N^o, tr_N^d)\}$, which consists of a sequence of task records (whose length is N) sorted by chronological order.

Multi-modal Embedding Module. The multi-modal embedding module is designed to extract spatio-temporal correlations of the processed trajectory data and learn its high-dimensional representations, which includes two sub-layers: the worker trajectory embedding layer and the spatio-temporal embedding layer.

In the worker trajectory embedding layer, each node tr in HT of a worker w is encoded to learn its latent representation by mapping the worker, location, and time into embedding vectors $e^w \in \mathbb{R}^d$, $e^l \in \mathbb{R}^d$, and $e^t \in \mathbb{R}^d$ respectively. In addition, periodicity plays a vital role in preference learning. To capture the periodicity (i.e., a week), we map continuous timestamps into 168 = 24×7 dimensions, representing the number of hours for one week. For each node in the trajectory, we denote its embedding e^{tr} as the sum of above learned embedding vectors $e^{tr} = e^w + e^l + e^t \in \mathbb{R}^d$. Thus, the trajectory embedding for each worker is represented as $E(w) = \{e_1^{tr}, e_2^{tr}, ..., e_{2N}^{tr}\} \in \mathbb{R}^{2N \times d}$.

In the Spatio-Temporal Embedding Layer, we utilize two matrices: the spatio-temporal matrices of trajectory and the candidate spatio-temporal matrices, which are generated from the historical trajectory. The aim of the two matrices is to fully leverage spatio-temporal information. The spatio-temporal matrices of trajectory calculate the pairwise spatio-temporal relation between every two trajectory nodes tr_i and tr_j in HT, which preserves both the time intervals and the spatial distances. Specifically, the temporal interval between the i-th and j-th nodes in HT is denoted as $\triangle_{ij}^t = |t_i - t_j|$ and the spatial GPS distance between them is denoted as $\triangle_{ij}^s = Haversine(GPS_i, GPS_j)$.

$$Haversine(i,j) = 2 \cdot R \cdot \arcsin\left(Degree(i,j)\right),$$
$$Degree(i,j) = \sqrt{\sin^2\left(\frac{\phi_j - \phi_i}{2}\right) + \cos(\phi_i) \cdot \cos(\phi_j) \cdot \sin^2\left(\frac{\lambda_j - \lambda_i}{2}\right)} \quad (2)$$

where R represents the radius of the Earth, ϕ_i and λ_i correspond to the latitude and longitude of GPS_i, and ϕ_j and λ_j correspond to the latitude and longitude of GPS_j.

The Candidate Spatio-Temporal Relation Matrix represents the time intervals between t_{m+1} and $t_1, t_2, ..., t_m$, where each element in the matrix represents the absolute difference between t_{m+1} and t_j. The matrix captures the temporal distances between t_{m+1} and each of the L candidate locations with respect to the reference point (tr). It is noted that t_{m+1} corresponds to the last point in the trajectory.

The candidate spatio-temporal matrices calculate the spatio-temporal relation between a pair of location candidate $lc \in [1, L]$ and trajectory node $tr \in [1, 2N]$. The candidate spatial relation is denoted as $N_{lc,tr}^s = Haversine(GPS_{lc}, GPS_{tr})$, which represents the spatial distance between location candidate lc and trajectory node tr.

The candidate temporal relation is denoted as $N_{lc,tr}^t = |t_{m+1} - t_{tr}|$, where it represents the time interval between t_{m+1} and $t_1, t_2, ..., t_m$.In particular, the time interval values for each lc are given by $|t_{m+1} - t_{tr}|$, where $m + 1$ represents the length of the trajectory. Finally, we obtain the spatio-temporal matrices of trajectory $\triangle^s \in \mathbb{R}^{2N \times 2N}$ and $\triangle^t \in \mathbb{R}^{2N \times 2N}$. Meanwhile, we prepare the candidate spatio-temporal relation matrices $N^s \in \mathbb{R}^{L \times 2N}$ and $N^t \in \mathbb{R}^{L \times 2N}$.

Self-Attention Aggregation Layer. This module applies the self-attention mechanism [10, 12, 22] to integrate the spatial distance and temporal interval, as well as aggregate relevant visit tr and update the representation of each visit. The layer constructs a matrix $M \in \mathbb{R}^{2N \times 2N}$, where the top-left element $\mathbb{R}^{m \times m}$ is a unit matrix and the rest of the elements are zeros. The multimodal embedding layer outputs worker-embedded trajectory matrices $E(w)$ and spatiotemporal relation matrices $E(\triangle)$. The layer calculates a new sequence S through these parameter matrices$W_Q, W_K, W_V \in \mathbb{R}^{d \times d}$ using Eq. 3 and Eq. 4.

$$S(w) = Attention(E(w)W_Q, E(w)W_K, E(w)W_V, E(\triangle), M). \quad (3)$$

$$Attention(Q, K, V, \triangle, M) = \left(M \times softmax\left(\frac{Q \cdot K^T + \triangle}{\sqrt{d}}\right)\right)V. \quad (4)$$

Attention Matching Layer. Utilizing the trajectories generated by the previous modules, this module matches the trajectory with L candidate locations to find the most preferred location. The workers' next preferred location is predicted by three embeddings: embeddings of generated trajectory $TR(w) \in \mathbb{R}^{2N \times d}$, embeddings of candidate position $E(l) = \{e_1^l, e_2^l, ..., e_L^l\} \in \mathbb{R}^{L \times d}$, and embeddings of candidate spatiotemporal relation matrix $E(N) \in \mathbb{R}^{L \times d}$. The preference $A(w) \in \mathbb{R}^L$ for the next position is calculated as follows:

$$A(w) = Matching(E(l), S(w), E(N)). \quad (5)$$

$$Matching(Q, K, N) = Sum\left(softmax\left(\frac{Q \cdot K^T + N}{\sqrt{d}}\right)\right). \quad (6)$$

4.1.2 Personalized Federated Training. In this section, we will show the personalized federated training for preference learning, which can alleviate the effect of data heterogeneity, as illustrated in Algorithm 1. Specifically, we propose a Balanced Sampler based loss to balance the positive and negative samples and an adaptive local aggregation method for personalized federated training.

Balanced Sampler based Loss. Given a worker's historical trajectory HT, and the probabilities of each candidate location $a_j \in A(w)$, the standard cross-entropy loss, as shown in Eq. 7, requires computing $L - 1$ negative samples. However, due to the biased distribution of positive and negative samples in $A(w)$, the loss function can lead to low efficiency with high computational costs. In our paper, we propose a balanced sampler that randomly samples negative samples at each training step to improve the efficiency, as shown in Eq. 8.

$$L_1 = -\sum_i \sum_{m_i} \left(log\sigma(a_k) + \sum_{j=1, j \neq k}^{L} log(1 - \sigma(a_j))\right) \quad (7)$$

$$L_2 = -\sum_i \sum_{m_i} \left(log\sigma(a_k) + \sum_{j_1, j_2, ..., j_S \in [1,L], (j_1, j_2, ..., j_S) \neq k} log(1 - \sigma(a_j))\right) \quad (8)$$

where L represents the cross-entropy loss function, and σ denotes a commonly used activation function.

Personalized Federated Training. We adopt personalized federated training with the adaptive local aggregation method, i.e., each platform individually trains a model with different parameter weights, and the weights are integrated by the central server to improve prediction accuracy and guarantee privacy protection. Before jumping into the personalized federated algorithm, we first introduce the Adaptive Layer Aggregation (ALA) mechanism [39], where the parameters of the local models are updated through differential aggregation with the parameters of the global model. Each platform center downloads the global model θ^{t-1} from the central server at iteration t, and each platform center updates the local model $\hat{\theta}_i^t$ for subsequent local model training, calculated as shown in Eq. 9, where W_i is the aggregating weights. Before aggregation, we first initialize W, where each element in W is initialized to one at the beginning, and the value of W is iteratively updated. To reduce computational overhead, we randomly sample $s\%$ of local data in each iteration, and the W_i of each platform center is trained as Eq. 10.

$$\hat{\theta}_i^t := \theta_i^{t-1} + (\theta^{t-1} - \theta_i^{t-1}) \odot W_i, \tag{9}$$

$$W_i^p \leftarrow W_i^p - \eta \nabla_{W_i^p} L(\hat{\theta}_i^t; D_i^{s,t}; \theta^{t-1}), \tag{10}$$

The process of proposed Federated Personalized Learning is shown in Algorithm 1, which can be divided into two parts in each communication round. The first part is for the central server, where the central server transmits the trained model to each platform center. In each iteration, we randomly sample a certain ratio of platforms, and the central model parameters of the last iteration are sent to these platform centers. After the completion of local training, the central server aggregates parameters from clients based on FedAvg [23]. The second part is for the platform center, where an Adaptive Local Aggregation (ALA) method is used to train local models with the model. When $t = 1$, we will train W_i^p to convergence; when $t > 2$, we only train W_i^p for one epoch to adapt to changing model parameters. Since $\theta^0 = \theta_t^0, \forall i \in [T]$, ALA is not used in the first iteration. Besides, each active platform center computes its local gradient to satisfy local data, then transmits the updated parameters to the central server.

Algorithm 1: Personalized Federated Location Preference Learning

Input: N,T, θ^0, α, $s\%$
Output: Local models $\hat{\theta}_1, ..., \hat{\theta}_N$

```
The central server sends θ^0 to all clients to initialize local models.
Platform centers initialize W_i, ∀i ∈ [N] to ones.
for each iteration t in [1, .., T] do
    Server samples platform centers P_t;
    Server transmits θ^{t-1} to each selected platform center.
    for each platform center pc_k ∈ P_t in parallel do
        Platform center i samples s % of local data.
        if t = 2 then
            while W_i dose not converge do
                Platform center i updates W_i by Eq. 10.
        else if t > 2 then
            Platform center i updates W_i by Eq. 10.
        Platform center i obtains θ̂_t^i by
        θ̂_i^t ← θ̂_i^t − α∇_{θ̂_i^t} L(θ̂_i^t; D^{i;θ^{t-1}}),
        Transmit platform center model θ_i^t to central server.
    Server obtains θ^t by θ^t = 1/|P_t| Σ_{k∈P_t} θ_k^t .
return θ̂_1, ..., θ̂_N;
```

4.2 Task Assignment

In this section, we first obtain available workers and reachable tasks under spatiotemporal constraints. Each platform center trains a personalized federated learning model to predict the top$-k$ location preferences of the employed workers. To maximize the number of workers assigned to each platform and match the predicted location preferences of workers with the task destination, we propose two location-aware assignment algorithms: the location-aware KM algorithm and the conditional degree-reduction algorithm.

4.2.1 Location-aware KM algorithm. We leverage the predicted location preferences and prioritize tasks based on the distance between the task destination and the predicted location. Specifically, we assign higher weights to tasks that have shorter distances between their destination and the worker's location preference. In this way, we transform the task assignment problem into a maximum-weight matching problem, which can be solved by the KM algorithm.

Before constructing the bipartite graph, which serves in KM algorithm solving the maximum-weight matching problem, it is necessary to obtain the available workers $AW(s)$ and reachable tasks $RT(w)$. Given a set of online workers $W = w_1, w_2, ..., w_{|W|}$ and a set of currently online and unassigned tasks $S = s_1, s_2, ..., s_{|S|}$, $AW(s)$ $(\forall w \in AW(s), s \in S)$ and $RT(w)$ $(\forall s \in RT(w), w \in W)$ should both satisfy the following conditions: first, the distance between worker w and task s is shorter than the worker's reachable radius w_d; second, the worker w can reach the task location before the task's expiration time $s.e$.

The constructed undirected bipartite graph, denoted as $G = (V, E)$, consists of a set of vertices V and a set of edges E. The vertices of the bipartite graph are partitioned into G_W and G_S based on W and S. Each edge between vertices can be mapped into the reachable tasks of workers $RT(w)$, and $|E|$ is equal to the number of reachable tasks of all workers in W. Specifically, we rank the distances between the endpoint position and the top-K $(K = 1, 2, 3)$ preferred locations and choose the top-K positions with the nearest distance between the task endpoint and the worker's preferred location. Then, we divide the weights of the edges into several intervals according to the distances, i.e., $0 - 0.5$, $0.5 - 1$, $1 - 2$, $2 - 5$, $5 - 10$, and 10+ kilometers, and assign weights of 6, 5, 4, 3, 2, and 1, respectively.

After constructing the undirected bipartite graph, we introduce the KM algorithm for task assignment, which is shown in Algorithm 2. First, we initialize the expectations of each vertex $v \in V$ in graph G to the maximum weight among the edges associated with it. Then, we call the *AvailableTaskGeneration* algorithm (Algorithm 3), which is a traversal algorithm that recursively calculates the difference between the weight of edges associated with two

Algorithm 2: Location-aware KM Algorithm

Input: graph G
Output: A
1 Initialize A, sv_{task} and $slack$;
2 **for** *each worker* $w \in W$ **do**
3 $wv_{worker}[w] \leftarrow max(weight(v_w^W, v_s^S))$;
4 **for** *each worker* $w \in W$ **do**
5 **while** $wv_{worker}[w] > 0$ **do**
6 Set $traverse_{task}$ and $traverse_{worker}$ to $False$;
7 **if** *AvailableTaskGeneration(w)* **then**
8 break ;
9 **else**
10 d=INF;
11 **for** *each task* $s \in S$ **do**
12 **if** ! $traverse_{task}[s]$ **then**
13 d=min(d,$slack[s]$);
14 **for** *each worker* $w \in W$ **do**
15 **if** $traverse_{worker}[w]$ **then**
16 $wv_{worker}[w] -= d$;
17 **for** *each task* $s \in S$ **do**
18 **if** $traverse_{task}[s]$ **then**
19 $sv_{task}[s] += d$;
20 **else**
21 $slack[s] -= d$;
22 **return** A;

Algorithm 3: AvailableTaskGeneration Algorithm

Input: *worker* w
Output: *Bool*
1 Set $traverse_{worker}[w]$ to $True$;
2 **for** *each task* s *is adjacent to* w *in* G **do**
3 **if** $traverse_{task}[s]$ **then** continue ;
4 $diff \leftarrow wv_{worker}[w] + sv_{task}[s] - weight(v_w^W, v_s^S)$;
5 **if** $diff == 0$ **then**
6 **if** *A[s] = -1 or AvailableTaskGeneration(A[s])* **then**
7 A[s]=w;
8 **return** $True$;
9 **else**
10 $slack[s] = min(slack[s], diff)$;
11 **return** $False$;

vertices and the expected sum of workers and tasks (line 4). If the difference is 0, a task match is found for the worker. If no task can be matched for worker w, we update the expectations of the last matched worker and task, changing the competitive relationship among workers, and allowing for more choices between tasks and workers (lines 9–21). Finally, the total task allocation A is obtained after traversing all workers.

4.2.2 Conditional Degree-reduction algorithm (CDR). To explain the CDR algorithm, we visualize task distributions on the Foursquare dataset, as illustrated in Figure 3. Workers are divided into three categories: red, blue, and orange. The center represents the simulated location of the workers, and the radius represents their reachable distance. Consequently, workers in different categories have access to different task distributions within a fixed radius. During the task assignment process, our objective is to ensure efficient allocation of

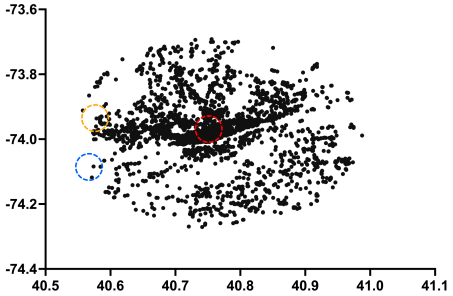


Figure 3: Various Tasks Distribution on Foursquare

Algorithm 4: Conditional Location-aware Degree-reduction Algorithm

Input: W, AS,AW
Output: A
1 Initialize $flag$ to $True$;
2 **while** $|W| > 0$ *and* $flag$ **do**
3 $flag = False$;
4 **for** *each worker* w *in* W **do**
5 Calculate the degree $D(w)$ of worker w from $AS(w)$;
6 **if** $!D(w)$ **then**
7 remove w from W;
8 **else if** $!flag$ *and* $D(w) < dt$ **then**
9 $flag = True$;
10 $w_{min} \leftarrow w$ in $min(D(w))$;
11 $s_{min} \leftarrow$ the task s with the minimum distance in $AS(w_{min})$;
12 $A[s_{min}] = w_{min}$;
13 remove w from W;
14 $w_{list} = AW(s_{min})$;
15 **for** *each worker* $w \in w_{list}$ **do**
16 **if** $s_{min} \in AS(w)$ **then**
17 remove s_{min} from $AS(w)$;
18 **while** $|W| > 0$ **do**
19 $s_{min} \leftarrow$ the task s with the minimum distance in $AS(w)$;
20 **if** $s_{min} < 0$ **then**
21 remove w from W and continue;
22 $A[s_{min}] = w$;
23 remove w from W;
24 $w_{list} = AW(s_{min})$;
25 **for** *each worker* $w \in w_{list}$ **do**
26 **if** $s_{min} \in AS(w)$ **then**
27 remove s_{min} from $AS(w)$;
28 **return** A;

tasks. The Degree-Reduction-based greedy algorithm [17, 45] suggests selecting workers with the minimum degree for assignment. However, we emphasize allocating tasks to workers in the blue and orange categories as much as possible because they have fewer reachable tasks. On the other hand, workers in the red category can almost guarantee the completion of tasks assigned to them due to the large number of reachable tasks. To achieve this goal and improve task assignment efficiency, we propose a conditional degree-reduction algorithm by considering the distribution bias of reachable tasks and the distance between the destination and preferred locations.

Derived from Degree-Reduction-based (DR) greedy algorithm, we apply location-based conditions to the algorithm, as shown in Algorithm 4. Before W is empty, we calculate the degree $D(w)$ of each vertex w, and if $D(w_i) = 0$, there is no task that can be assigned to the worker, so we remove w_i from W. Then, we find the smallest $|D(w_i)|$ and assign the task s_j, where s_j is the task with the minimum distance to the worker w_i among the assignable tasks. Finally, we allocate the task assigned to the worker, update AS, where $u_i \in AW(s_j)$, and delete the node s_j from $AS(u_i)$. Our novelty is setting a degree threshold dt for the degree greedy strategy(line 8), and if we find that all degrees of w in W satisfy $D(w) > dt$ after performing the degree greedy strategy for a period of time, we will no longer calculate the degree of each w or find the smallest $|D(w_i)|$(line 9). Instead, we perform the task assignment phase among the remaining workers(lines 18–27).

5 EXPERIMENTAL EVALUATION

5.1 Experimental Setup

We use the check-in dataset from Foursquare to simulate our spatio-temporal crowdsourcing scenarios, which is a common practice in evaluating SC platforms. FourSquare has a user base of 11,326 and a total number of 1,385,223 check-in records between January to December 2011. We consider all users as workers and the latest check-in location of each worker is considered as the current location. Then, we generate 10 SC platform centers, and all workers are randomly and uniformly assigned to the generated platforms. Each worker is restricted to belonging to one platform. For each check-in, two adjacent check-ins of a worker represent a completed task. The origin of the task is set as the check-in location of the first check-in information, and the destination is set as the check-in location of the second check-in information. The publication time of the task is set as the earliest time of the first check-in information. Each platform individually possesses all historical data of the employed workers. In addition, we set the ideal distance id and assume that the distance between the task ending point and the predicted location preference is d. When $d < id$, we consider the task as a correctly assigned task. All the experiments are implemented on an Intel (R) Xeon (R) CPU E5-2650 v4 @ 2.20GHz and NVidia TITAN XP GPU.

5.2 Results

5.2.1 Performance of Personalized Federated Location Preference Learning. We first elaborate on the performance of the proposed Personalized Federated Location Preference Learning.

Baselines. We select the four most representative state-of-the-art methods for comparison.

- POISeqPop [21]: The POISeqPop model ranks locations in descending order based on the popularity of their origins in the target worker's sequence of tasks.
- CTP [21]: The CTP model assumes that all workers belong to a platform center and all data is centralized for training.
- FedAvg [23]: FedAvg aggregates parameters collected from clients by averaging, which is used to update the global model.
- FedDyn [1]: The FedDyn is a dynamic regularization method, which can address the non-iid data and concept drift in FL.

Metrics. To evaluate the accuracy of workers' location preference prediction, we use *Recall@K* as the evaluation metric, which measures the proportion of relevant task positions retrieved among the top-K predicted workers' location preference. We filter workers with less than 50 historical check-in sequences and fixed the length of workers' historical trajectories to 50. We conduct local training in 10 platform centers separately, where the dataset was split into 60% for training, 20% for validation, and 20% for testing.

Table 2: Performance of Different Models

Methods	Recall@1	Recall@5	Recall@10
POISeqPop	0.0069	0.0125	0.0236
CTP	**0.1520**	0.2914	0.3687
FedAvg	0.1316	0.2760	0.3642
FedDyn	0.1335	0.2838	0.3670
PLP	0.1495	**0.3053**	**0.3859**

Results. As shown in Table 2, PLP significantly outperforms the POISeqPop model, which indicates that PLP is capable of predicting more accurate worker location preferences. Among baselines, CTP achieves the best result on *Recall*@1, This situation may occur because training under CTP allows for more training data from candidate locations, while using federated learning focuses more on predicting the candidate locations in the platform data, which may not be sufficient compared to CTP. Due to the proposed location-preference learning and adaptive local aggregation method, PLP achieves the best results in most cases, which performs better than the baselines by up to 4.77% - 10.65%, with the exception of POISeqPop. The results demonstrate the superiority of the proposed PLP to handle data heterogeneity across clients.

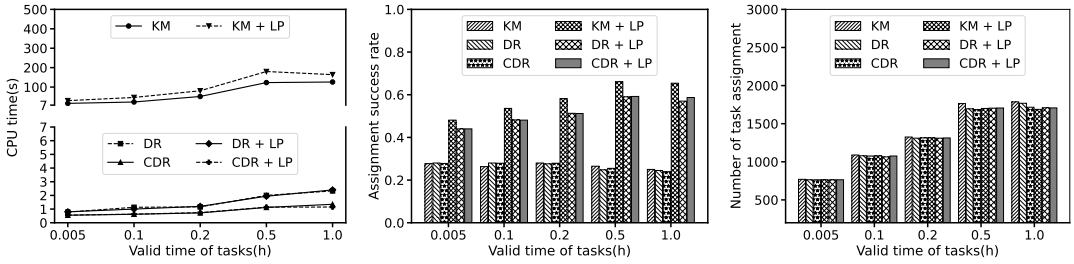
5.2.2 Performance of Task Assignment. We proceed to study the performance of task assignment. Table 3 shows our experimental settings, where the default values of all parameters are underlined.

Table 3: Experiment Parameters

Parameter	Values
Valid time of tasks (h) $e - p$	0.05, 0.1, 0.2, 0.5, 1
Reachable distance of workers (km) r	1, 2, 3, 4, 5
Number of workers $\|W\|$	2600, 2800, 3000, 3200, 3400
Number of tasks $\|S\|$	3200, 3400 3400, 3600, 3800
Degree threshold dt	1, 5, 10, 20, 50

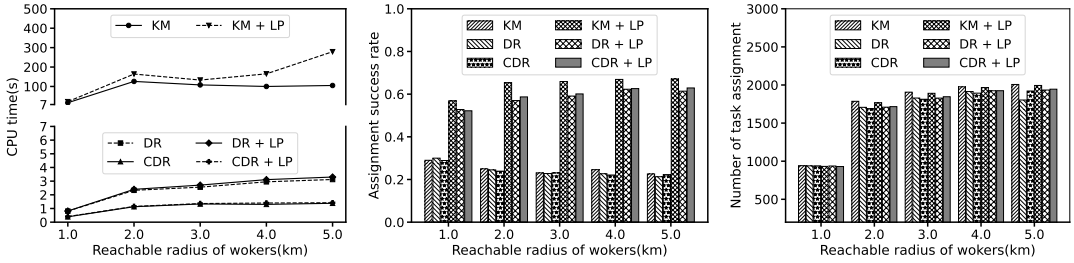
Evaluation Methods. We study the following task assignment algorithms.

- KM: The original Kuhn-Munkres (KM) algorithm for finding a maximum cardinality perfect matching in a bipartite graph.
- DR: The Degree-Reduction-based (DR) greedy algorithm gradually constructs a solution by continuously reducing the degrees of nodes.
- CDR: The Conditional Degree-Reduction-based (CDR) algorithm improves upon DR by introducing a degree threshold, which controls the number of iterations for the degree greedy strategy.
- LP+KM: The LP+KM algorithm considers the workers' location preferences (LP), predicted by the proposed PLP, in KM.
- LP+DR: The LP+DR algorithm incorporates the predicted workers' location preferences by the proposed PLP into DR.
- LP+CDR: An improved CDR algorithm considers workers' location preferences predicted by the proposed PLP.



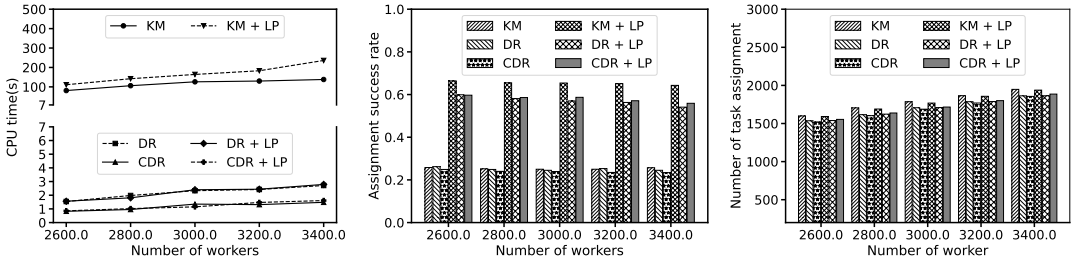
(a) CPU Time (b) Assignment Success Rate (c) Number of Task Assignments

Figure 4: Performance of Task Assignment: Effect of $e - p$



(a) CPU Time (b) Assignment Success Rate (c) Number of Task Assignments

Figure 5: Performance of Task Assignment: Effect of r



(a) CPU Time (b) Assignment Success Rate (c) Number of Task Assignments

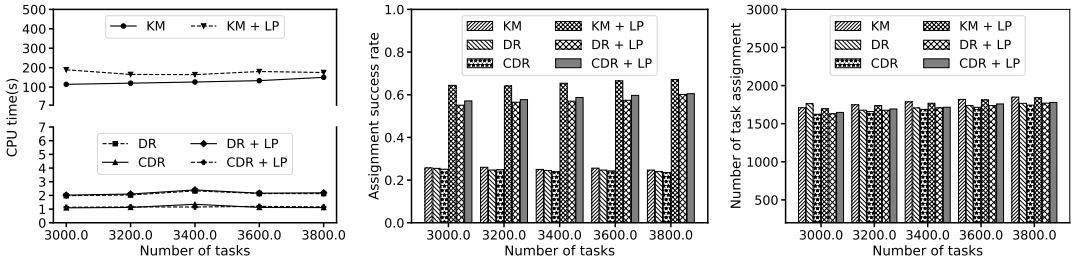
Figure 6: Performance of Task Assignment: Effect of $|W|$

Metrics. We apply three widely used metrics for task assignment, including CPU time, Assignment Success Rate (ASR), and the number of task assignments. The CPU time is the time cost for finding the task assignment. ASR is the ratio between the number of successfully assigned tasks and the total number of assigned tasks. In our experiments, we consider a task assignment succeeds when the assigned task destination is less than 2 km from the worker's preferred location.

Effect of $e - p$. The impact of tasks' valid time $e - p$ on task assignment is shown in Figure 4. As the valid time of tasks increases, the CPU time and the number of task assignments for all algorithms significantly increase (Figures 4(a) and 4(c)). The reason is that as the valid time increases, more tasks are feasible for the workers and the search space in the assignment process extends, but the CPU time increases correspondingly. At the same time, the probability of assigning new tasks to the workers increases. Additionally, as shown in Figure 4(b), all preference-based algorithms obtain decent ASR scores, which leads to an increase in the expansion of the valid time.

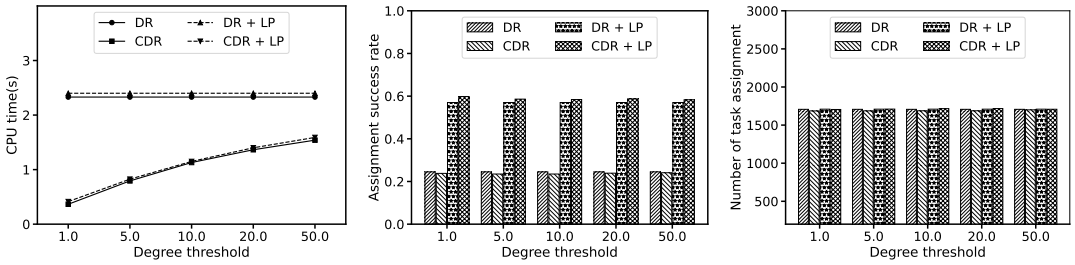
Effect of r. Next, we study the effect of workers' reachable distance r. As shown in Figure 5(a), with the increase of r, the CPU time for all algorithms shows an increasing trend, and both ASR and the number of task assignments increase correspondingly (see Figures 5(b) and 5(c)). This is because as the reachable distance r of workers increases, the number of reachable tasks for the workers also increases, leading to a larger search space during the task assignment process. In our simulated dataset, the increase in reachable distance does not improve the probability of obtaining workers' preferred tasks, which results in a substantial increase in KM+LP time as well.

Effect of $|W|$. We also evaluate the impact of the number of workers $|W|$. As shown in Figure 6(a), as $|W|$ increases, the CPU time for all algorithms increases accordingly. This is caused by the need of assigning more available workers and leads to a significant increase in assigning task assignments to workers. Additionally, for the KM and KM+LP algorithms, limited tasks are available for assignment, causing extra search time overhead. As shown in Figure 6(b), for all location-preference-based algorithms, the ASR slightly decreases as the number of workers increases, while maintaining high ASR values, and the number of task assignments also increases concurrently (see Figure 6(c)).



(a) CPU Time (b) Assignment Success Rate (c) Number of Task Assignments

Figure 7: Performance of Task Assignment: Effect of $|S|$



(a) CPU Time (b) Assignment Success Rate (c) Number of Task Assignments

Figure 8: Performance of Task Assignment: Effect of dt

Effect of $|S|$. Figure 7(a) shows the effect of the number of tasks $|S|$. As the number of tasks increases, the CPU time for KM increases, but decreases for KM+LP. This is because, with an increased number of tasks, the probability of the occurrence of tasks that workers are most interested in also increases, reducing competition among workers in the KM+LP algorithm. Meanwhile, for the KM algorithm, the number of feasible tasks increases. Simultaneously, as observed in Figures 7(b) and 7(c), with an increase in the number of tasks, the ASR for KM+LP, DR+LP, and CDR+LP all increase, and the number of assignments for all algorithms increases. This is due to the fact that an increase in the number of tasks expands the reachable number of tasks to workers and increases the probability of occurring interest tasks.

Effect of dt. Finally, we study the effect of dt, which controls the range value in the degree reduction greedy phase of CDR. We focus on degree-based algorithms (i.e., DR, DR+LP, CDR, and CDR+LP) in the experiments. As illustrated in Figure 8, dt has no significant effect on the DR algorithm. The results of DR and DR+LP also remain unchanged. As shown in Figure 8(a), when dt increases, the CPU time consumed by CDR and CDR+LP algorithms gradually increases but is still less than that of the Degree-Reduction-based (DR) greedy algorithms.

Summary of our empirical study. During task assignment, we experimented with five controlled variables ($e - p$, r, $|W|$, $|S|$, dt), observing substantial variations in CPU time, ASR, and task assignment numbers. From the Figures 4– 8, we can conclude that in general, the basic allocation algorithms (i.e., KM, DR, and CDR) consume less CPU time compared to the LP-based algorithms (i.e., KM+LP, DR+LP, and CDR+LP). Furthermore, the KM-based algorithm consumes significantly more time compared to the degree-based algorithms. In terms of ASR, the LP-based algorithms outperform the basic allocation algorithms, even though there was a probability of a drop in the number of task assignments. Particularly, CDR and CDR+LP show high efficiency, as they reduced CPU time while maintaining ASR and task assignment numbers at a level comparable to DR and DR+LP. The results show the efficiency of the proposed CDR.

6 CONCLUSION

In this paper, we study a location-aware task assignment problem and propose a personalized task assignment framework with federated location preference learning (PTA-FLP). PTA-FLP consists of two phases: a personalized federated location-preference learning (PLP) and a task assignment phase. In the PLP phase, we propose a multi-modal embedding module to learn local location-preference for each platform center. In addition, an adaptive local aggregation method is proposed to alleviate the influence of data heterogeneity across various platform centers (i.e., clients). In the phase of task assignment, a location-aware KM and a conditional degree-reduction algorithm are proposed to ensure effective and efficient task assignment, which considers workers' location preference. We conducted extensive experiments on real data to demonstrate the superiority of the proposed algorithms.

ACKNOWLEDGMENTS

This work is partially supported by NSFC (No. 61972069, 61836007, 61832017, 62272086), Shenzhen Municipal Science and Technology R&D Funding Basic Research Program (JCYJ20210324133607021), Municipal Government of Quzhou under Grant No. 2022D037, and Key Laboratory of Data Intelligence and Cognitive Computing, Longhua District, Shenzhen.

REFERENCES

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N Whatmough, and Venkatesh Saligrama. 2021. Federated learning based

on dynamic regularization. *arXiv preprint arXiv:2111.04263* (2021).
[2] Raed S Alharthi, Esam Aloufi, Ibrahim Alrashdi, Ali Alqazzaz, Mohamed A Zohdy, and Julian L Rrushi. 2020. Protecting location privacy for crowd workers in spatial crowdsourcing using a novel dummy-based mechanism. *IEEE Access* 8 (2020), 114608–114622.
[3] Xuanhao Chen, Yan Zhao, Kai Zheng, Bin Yang, and Christian S Jensen. 2022. Influence-aware Task Assignment in Spatial Crowdsourcing. In *ICDE*. IEEE, 2141–2153.
[4] Zhao Chen, Peng Cheng, Lei Chen, Xuemin Lin, and Cyrus Shahabi. 2020. Fair task assignment in spatial crowdsourcing. *PVLDB* 13, 12 (2020).
[5] Peng Cheng, Lei Chen, and Jieping Ye. 2019. Cooperation-aware task assignment in spatial crowdsourcing. In *ICDE*. IEEE, 1442–1453.
[6] Peng Cheng, Xiang Lian, Lei Chen, and Cyrus Shahabi. 2017. Prediction-based task assignment in spatial crowdsourcing. In *ICDE*. IEEE, 997–1008.
[7] Yurong Cheng, Lei Chen, Ye Yuan, Guoren Wang, Boyang Li, and Fusheng Jin. 2020. Strict and flexible rule-based graph repairing. *TKDE* 34, 7 (2020), 3521–3535.
[8] Yurong Cheng, Boyang Li, Xiangmin Zhou, Ye Yuan, Guoren Wang, and Lei Chen. 2020. Real-time cross online matching in spatial crowdsourcing. In *ICDE*. IEEE, 1–12.
[9] Dingxiong Deng, Cyrus Shahabi, and Linhong Zhu. 2015. Task matching and scheduling for multiple workers in spatial crowdsourcing. In *SIGSPATIAL*. 1–10.
[10] Jie Feng, Yong Li, Zeyu Yang, Qiang Qiu, and Depeng Jin. 2020. Predicting human mobility with semantic motivation via multi-task attentional recurrent networks. *TKDE* 34, 5 (2020), 2360–2374.
[11] Srinivasa Raghavendra Bhuvan Gummidi, Xike Xie, and Torben Bach Pedersen. 2019. A survey of spatial crowdsourcing. *TODS* 44, 2 (2019), 1–46.
[12] Qing Guo, Zhu Sun, Jie Zhang, and Yin-Leng Theng. 2020. An attentional recurrent neural network for personalized next location recommendation. In *AAAI*, Vol. 34. 83–90.
[13] Weishan Huang, Xinyu Lei, and Hongyu Huang. 2021. PTA-SC: Privacy-Preserving Task Allocation for Spatial Crowdsourcing. In *WCNC*. 1–7.
[14] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning* 14, 1–2 (2021), 1–210.
[15] Leyla Kazemi and Cyrus Shahabi. 2012. Geocrowd: enabling query answering with spatial crowdsourcing. In *SIGSPATIAL*. 189–198.
[16] Leyla Kazemi, Cyrus Shahabi, and Lei Chen. 2013. Geotrucrowd: trustworthy query answering with spatial crowdsourcing. In *SIGSPATIAL*. 314–323.
[17] Tinghao Lai, Yan Zhao, Weizhu Qian, and Kai Zheng. 2022. Loyalty-based Task Assignment in Spatial Crowdsourcing. In *CIKM*. 1014–1023.
[18] Maocheng Li, Jiachuan Wang, Libin Zheng, Han Wu, Peng Cheng, Lei Chen, and Xuemin Lin. 2021. Privacy-Preserving Batch-based Task Assignment in Spatial Crowdsourcing with Untrusted Server. In *CIKM*. 947–956.
[19] Xiang Li, Yan Zhao, Xiaofang Zhou, and Kai Zheng. 2020. Consensus-based group task assignment with social impact in spatial crowdsourcing. *DSE* 5 (2020), 375–390.
[20] Yunchuan Li, Yan Zhao, and Kai Zheng. 2021. Preference-aware Group Task Assignment in Spatial Crowdsourcing: A Mutual Information-based Approach. In *ICDM*. 350–359.
[21] Jiaxin Liu, Liwei Deng, Hao Miao, Yan Zhao, and Kai Zheng. 2022. Task Assignment with Federated Preference Learning in Spatial Crowdsourcing. In *CIKM*. 1279–1288.
[22] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. Stan: Spatio-temporal attention network for next location recommendation. In *WWW*. 2177–2185.
[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.
[24] Hao Miao, Jiaxing Shen, Jiannong Cao, Jiangnan Xia, and Senzhang Wang. 2022. MBA-STNet: Bayes-enhanced discriminative multi-task learning for flow prediction. *TKDE* (2022).
[25] Chenxi Qiu and Anna Cinzia Squicciarini. 2019. Location Privacy Protection in Vehicle-Based Spatial Crowdsourcing Via Geo-Indistinguishability. In *ICDCS*. 1061–1071.
[26] Dezhi Sun, Ke Xu, Hao Cheng, Yuanyuan Zhang, Tianshu Song, Rui Liu, and Yi Xu. 2019. Online delivery route recommendation in spatial crowdsourcing. *WWWJ* 22 (2019), 2083–2104.
[27] Hien To, Gabriel Ghinita, and Cyrus Shahabi. 2014. A framework for protecting worker location privacy in spatial crowdsourcing. *PVLDB* 7, 10 (2014), 919–930.
[28] Yongxin Tong, Yansheng Wang, and Dingyuan Shi. 2020. Federated Learning in the Lens of Crowdsourcing. *IEEE Data Eng. Bull.* 43, 3 (2020), 26–36.
[29] Yongxin Tong, Yuxiang Zeng, Bolin Ding, Libin Wang, and Lei Chen. 2019. Two-sided online micro-task assignment in spatial crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering* 33, 5 (2019), 2295–2309.
[30] Yongxin Tong, Yuxiang Zeng, Zimu Zhou, Boyi Liu, Yexuan Shi, Shuyuan Li, Ke Xu, and Weifeng Lv. 2023. Federated Computing: Query, Learning, and Beyond. *IEEE Data Eng. Bull.* 46, 1 (2023), 9–26.
[31] Yongxin Tong, Zimu Zhou, Yuxiang Zeng, Lei Chen, and Cyrus Shahabi. 2020. Spatial crowdsourcing: a survey. *PVLDB* 29, 1 (2020), 217–250.
[32] Jiachuan Wang, Peng Cheng, Libin Zheng, Lei Chen, and Wenjie Zhang. 2022. Online Ridesharing with Meeting Points. *PVLDB* 15, 13 (2022), 3963–3975.
[33] Ziwei Wang, Yan Zhao, Xuanhao Chen, and Kai Zheng. 2021. Task assignment with worker churn prediction in spatial crowdsourcing. In *CIKM*. 2070–2079.
[34] Jianhao Wei, Yaping Lin, Xin Yao, and Jin Zhang. 2019. Differential privacy-based location protection in spatial crowdsourcing. *TSC* 15, 1 (2019), 45–58.
[35] Jinfu Xia, Yan Zhao, Guanfeng Liu, Jiajie Xu, Min Zhang, and Kai Zheng. 2019. Profit-driven Task Assignment in Spatial Crowdsourcing.. In *IJCAI*. 1914–1920.
[36] Yi Yang, Yurong Cheng, Ye Yuan, Guoren Wang, Lei Chen, and Yongjiao Sun. 2022. Privacy-preserving cooperative online matching over spatial crowdsourcing platforms. *PVLDB* 16, 1 (2022), 51–63.
[37] Guanyu Ye, Yan Zhao, Xuanhao Chen, and Kai Zheng. 2021. Task Allocation with Geographic Partition in Spatial Crowdsourcing. In *CIKM*. 2404–2413.
[38] Xun Yi, Fang-Yu Rao, Gabriel Ghinita, and Elisa Bertino. 2018. Privacy-preserving spatial crowdsourcing based on anonymous credentials. In *MDM*. 187–196.
[39] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. 2023. FedALA: Adaptive Local Aggregation for Personalized Federated Learning. In *AAAI*.
[40] Junwei Zhang, Fan Yang, Zhuo Ma, Zhuzhu Wang, Ximeng Liu, and Jianfeng Ma. 2020. A decentralized location privacy-preserving spatial crowdsourcing for internet of vehicles. *TITS* 22, 4 (2020), 2299–2313.
[41] Yan Zhao, Jiannan Guo, Xuanhao Chen, Jianye Hao, Xiaofang Zhou, and Kai Zheng. 2021. Coalition-based task assignment in spatial crowdsourcing. In *ICDE*. 241–252.
[42] Yan Zhao, Tinghao Lai, Ziwei Wang, Kaixuan Chen, Huan Li, and Kai Zheng. 2023. Worker-Churn-based Task Assignment with Context-LSTM in Spatial Crowdsourcing. *TKDE* (2023).
[43] Yan Zhao, Yang Li, Yu Wang, Han Su, and Kai Zheng. 2017. Destination-aware task assignment in spatial crowdsourcing. In *CIKM*. 297–306.
[44] Yan Zhao, Jiaxin Liu, Yunchuan Li, Dalin Zhang, Christian S. Jensen, and Kai Zheng. 2023. Preference-aware Group Task Assignment in Spatial Crowdsourcing: Effectiveness and Efficiency. *TKDE* (2023).
[45] Yan Zhao, Kai Zheng, Yue Cui, Han Su, Feida Zhu, and Xiaofang Zhou. 2020. Predictive task assignment in spatial crowdsourcing: a data-driven approach. In *ICDE*. 13–24.
[46] Yan Zhao, Kai Zheng, Yang Li, Han Su, Jiajun Liu, and Xiaofang Zhou. 2019. Destination-aware task assignment in spatial crowdsourcing: A worker decomposition approach. *TKDE* 32, 12 (2019), 2336–2350.
[47] Yan Zhao, Kai Zheng, Yunchuan Li, Jinfu Xia, Bin Yang, Torben Bach Pedersen, Rui Mao, Christian S. Jensen, and Xiaofang Zhou. 2022. Profit Optimization in Spatial Crowdsourcing: Effectiveness and Efficiency. *TKDE* (2022).
[48] Yan Zhao, Kai Zheng, Ziwei Wang, Liwei Deng, Bin Yang, Torben Bach Pedersen, Christian S. Jensen, and Xiaofang Zhou. 2023. Coalition-based Task Assignment with Priority-aware Fairness in Spatial Crowdsourcing. *VLDBJ* (2023).
[49] Yan Zhao, Kai Zheng, Hongzhi Yin, Guanfeng Liu, Junhua Fang, and Xiaofang Zhou. 2020. Preference-aware task assignment in spatial crowdsourcing: from individuals to groups. *TKDE* 34, 7 (2020), 3461–3477.
[50] Matthew Zook, Mark Graham, Taylor Shelton, and Sean Gorman. 2010. Volunteered geographic information and crowdsourcing disaster relief: a case study of the Haitian earthquake. *World Medical & Health Policy* 2, 2 (2010), 7–33.