Worker-Churn-based Task Assignment with Context-LSTM in Spatial Crowdsourcing

Yan Zhao, Tinghao Lai, Ziwei Wang, Kaixuan Chen, Huan Li, and Kai Zheng*, *Senior Member, IEEE*

Abstract—The pervasiveness of GPS-enabled devices and wireless communication technologies flourish the market of Spatial Crowdsourcing (SC), which consists of location-based tasks and requires workers to be at specific locations physically to complete them. In this work, we study the problem of worker-churn-based task assignment in SC, where tasks are assigned by considering workers' churn. In particular, we aim to maximize the total rewards of task assignments based on the worker churn prediction. To solve the problem, we propose a two-phase framework, which consists of a worker churn prediction and a task assignment phase. In the first phase, we use an LSTM-based model to extract latent feelings of workers based on historical data and then estimate idle time intervals of workers. In the assignment phase, we design an efficient greedy algorithm and a Kuhn-Munkras-based algorithm that can achieve the optimal task assignment. To improve the accuracy of the idle time interval estimation for workers, we adopt a context-dependent LSTM model, which involves interactions between inputs and their context. We further optimize the original task assignment framework by proposing a travel distance optimization strategy to reduce the overall travel distance. Extensive experiments offer insight into the effectiveness and efficiency of the proposed solutions.

Index Terms—Spatial Crowdsourcing, Task Assignment, Worker Churn.

1 Introduction

Crowdsourcing is a computing paradigm, where humans actively or passively participate in the procedure of computing, especially for tasks that are intrinsically easier for humans than for computers [38]. Many successful crowdsourcing platforms exist, e.g., Amazon Mechanical Turk (MTurk)[1] and Wikipedia[2]. Along with the ubiquity of GPS-equipped networked devices, e.g., smartphones, a new class of crowdsourcing, called Spatial Crowdsourcing (SC), has drawn increasing attention in both academia and industry. With SC, requesters can issue spatial tasks (e.g., monitoring traffic conditions and picking up passengers) to SC servers that then assign workers to these tasks (called *task assignment*). Workers complete their tasks by moving to the specified locations. Spatio-temporal information (e.g., location, mobility, and the associated contexts) plays a crucial role in SC. Due to its natural connection to the physical world, SC is relevant to a wide spectrum of daily applications, including real-time ride-hailing services (e.g., Uber[3]), and on-wheel meal-ordering services (e.g., GrubHub[4]).

- *Y. Zhao, K. Chen and H. Li are with the Department of Computer Science, Aalborg University, Aalborg 9220, Denmark. Email: {yanz, kchen, lihuan}@cs.aau.dk.*
- *T. Lai, Z. Wang and K. Zheng are with the School of Computer Science and Engineering, Yangtze Delta Region Institute (Quzhou), Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, China. Email: {tinghao, ziwei}@std.uestc.edu.cn, zhengkai@uestc.edu.cn. *K. Zheng is the corresponding author of the paper.*

1. https://www.mturk.com/
2. https://www.wikipedia.com/
3. https://www.uber.com/
4. https://get.grubhub.com/

Research on SC [6]–[8], [11], [38], [41], [44], [45], [47]–[51] has gained momentum in recent years; consequently, many techniques of task assignment are proposed for different application scenarios. Cheng et al. [10] study a reliable diversity-based spatial crowdsourcing (RDB-SC) problem in SC, which aims to maximize the diversity score of assignments. Zhao et al. [46] propose a tensor-decomposition-based algorithm to learn worker preference, based on which they assign tasks by transforming the assignment problem into a Minimum Cost Maximum Flow problem. The study [18] aims to maximize the number of performed tasks for a worker with an optimal schedule, which combines two optimization problems, i.e., task-matching and task-scheduling.

However, the existing studies focus mainly on spatio-temporal availability of workers and tasks, thus leaving challenges related to effective and efficient task assignment largely unaddressed. For example, the above studies fail to consider user churn (i.e., worker churn) in task assignment, which describes worker defection from an SC service provider. Studies on user churn started from Customer Relation Management and have been proposed in various service fields [1], [13], [19], [22], [37]. Considering user churn as a real and serious business problem, several machine learning methods and artificial neural networks have been proposed to address the problem by telecommunication companies [2], [13], [25]. Besides, studies on user churn prediction in these fields, such as banks and websites, have been carried out as well [1], [19], [22]. Traditionally, the user churn prediction is treated as a classification problem based on labeled data and feature engineering, where users are divided into the churn and non-churn categories generally [40]. As the SC market saturates due to the globalization of services and fierce competition, the cost of worker acquisition has been rising

rapidly. Therefore, it is crucial to predict worker churn and take some measures to retain workers.

In this work, we investigate a task assignment problem in SC, called Worker-Churn-based Task Assignment (WC-TA). To be more specific, given a set of workers and a set of tasks, it aims to achieve the highest total rewards of task assignments based on the worker churn prediction.

In order to tackle the proposed WC-TA problem, we propose a two-phase framework, which includes a worker churn prediction and a task assignment phase. The first phase aims to predict the worker churn in the future. More specifically, we capture the latent feelings of workers using an LSTM-based model based on the historical data and then predict the idle time interval of workers. In the assignment phase, we propose a greedy and a Kuhn Munkras (KM) based method to achieve the task assignment. Specifically, the greedy method aims to assign each task to the most churn-prone worker from the unassigned workers, until all the tasks are assigned or all the workers are exhausted. The KM-based method is to find the maximum weight matching on a bipartite graph (composed of workers and tasks) in which the workers that are more likely to be churned are given higher priority.

The conference version [39] of this work provides a standard LSTM model to predict worker churn. However, the model does not consider the context when modeling the latent feelings of workers, where the contextual information can make a model more expressive and explainable. Inspired by the success of Mogrifier LSTM [31], a context-LSTM model, in modeling Natural Language Processing, we adopt it to predict worker churn. Mogrifier LSTM is an expressive model that explores how inputs interact with the context in which they occur. In particular, it uses a mutual gating strategy to integrate the current input and the previous output, which provides a richer space of interactions between inputs and their context. Mogrifier LSTM adapts to our worker churn prediction problem by modifying the number of iteration rounds based on workers' satisfaction and passion.

The second limitation of the conference version [39] is that it fails to consider the travel distance of workers in task assignment, which is a key factor in SC since workers must go to the designated locations physically to perform the assigned tasks. However, the goals of maximizing the total reward of assigned tasks and minimizing the travel distance are often conflicting, which means optimizing both simultaneously could be difficult. To address this issue, in Section 6 of this extension, we incorporate a travel distance optimization strategy into the task assignment phase, which tries to minimize the overall travel distance of workers while keeping the total reward of task assignment unchanged by giving priority to the performable task set with lower travel distance for each worker.

The major value-added extensions over our preliminary work [39] are four-fold.

1) We identify and study in depth two limitations in our previous algorithms, which do not consider the contextual information when modeling worker churn and the travel distance when assigning tasks.

2) We propose a satisfaction-based and a passion-based model to estimate each worker's idle time interval based on both inputs and their context.

3) We redesign the KM-based task assignment algorithm by incorporating a travel distance optimization strategy into the task assignment process, which tries to re-assign workers the performable tasks with less travel distance whenever possible as long as the overall reward of task assignment remains optimized.

4) Extensive experiments are conducted to study the impact of the key parameters, demonstrating the efficiency and effectiveness of the proposed methods.

The remainder of this paper is organized as follows. Section 2 introduces the related work, and Section 3 provides the proposed problem. In Section 4, a brief introduction of the framework overview is given, followed by the worker churn prediction models and two task assignment algorithms in Sections 5 and 6. We give the experimental results in Section 7 and conclude the paper in Section 8.

2 Related Work

2.1 Task Assignment in Spatial Crowdsourcing

Spatial Crowdsourcing (SC) can be deemed as one of the main enablers to complete location-based tasks [5], [12], [14], [26]–[30], [33]–[36], [42]. According to the task publish mode, SC can be classified into two categories, namely *server assigned tasks* (SAT) mode and *worker selected tasks* (WST) mode [23]. In SAT mode, the SC server assigns proper tasks to nearby workers based on the system optimization goals, e.g., maximizing the number of assigned tasks [9], [18], [23], [24], and maximizing the total reward of task assignment [4], [39]. While in WST mode, the server publishes various spatial tasks online, and workers can select any tasks based on their own preferences without the need to coordinate with the server [17], [18].

Most existing studies adopt the SAT mode, where an SC server takes charge of the task assignment process. For example, Cheng et al. [10] propose a reliable diversity-based spatial crowdsourcing (RDB-SC) problem in SC, where an SC server assigns tasks to suitable workers to maximize the diversity score of assignments. Zhao et al. [46] propose a preference-based task assignment problem and design a tensor-decomposition-based algorithm to capture worker's temporal preferences, based on which tasks are assigned. However, the above studies focus mainly on spatio-temporal availability of workers and tasks, which do not consider worker churn that describes worker defection from an SC service provider.

2.2 User/Worker Churn Prediction

User/Worker churn prediction is a hot spot in both academia and industry. Traditionally, the problem of user churn prediction is treated as a classification problem, where users are generally divided into the churn and non-churn categories.

To solve the problem of user churn prediction, it is necessary to take into account users' characteristics, including state sequences, behavior sequences, and other features extracted from the historical user profile. Recent studies make great efforts to predict user churn. For

instance, Bahnsen et al. [3] introduce a new finance-based approach and develop a cost-sensitive customer churn prediction model, which enables classification algorithms to serve business objectives. Hudaib et al. [21] hybrid a K-means algorithm, Multilayer Perceptron Artificial Neural Networks, and Self-Organizing Maps to establish a two-stage loss prediction model to predict user churn, where the effectiveness of the solution is demonstrated on real data. However, due to the data sparsity, the spatio-temporal characteristic, and uncertain churn criteria in SC applications, the aforementioned methods cannot be applied to the worker churn in SC directly. In this work, we combine different models (i.e., LSTM and fully connected neural network) under different criteria to address the worker churn prediction problem by considering SC characteristics.

3 Problem Definition

We proceed to present necessary preliminaries and then define the problem addressed. Table 1 lists the notation used throughout the paper. We use capital letters (e.g., A) to denote finite sets and use blackboard bold letters (e.g., $\mathbb{A}$) to denote sets of finite sets.

TABLE 1
Summary of Notation

Notation	Definition
s	Spatial task
$s.l$	Location of spatial task s
$s.p$	Publication time of spatial task s
$s.e$	Expiration time of spatial task s
$s.r$	Reward of spatial task s
w	Worker
$w.l$	Current location of worker w
$w.d$	Reachable distance of worker w
S_w	A historical task-performing sequence of w
$w.\tau$	Idle time interval of worker w
A	A spatial task assignment
$A.r$	Total reward in task assignment A
$\mathbb{A}$	Spatial task assignment set
μ	Temporal threshold

Definition 1 (Spatial Task). A spatial task, denoted by $s = (l, p, e, r)$, has a location $s.l$, a publication time $s.p$, an expiration time $s.e$, and a reward $s.r$.

Figure 1 shows a task assignment example, where two tasks, s_1 and s_2, exist. Each task has a publication time, an expiration time and a reward, e.g., the publication time and the expiration time of s_1 are 0 and 2, respectively, and the reward of s_1 is 1. With SC, the query of a spatial task s can be answered only if a worker is physically located at that location $s.l$. Besides, considering the expiration time, a spatial task s can be completed only if a worker arrives at $s.l$ before its deadline $s.e$. Note that with the single task assignment mode [23], an SC server should assign each spatial task to only one worker.

Definition 2 (Worker). A worker, denoted by $w = (l, d)$, has a location $w.l$ and a reachable distance $w.d$. The reachable range of worker w is a circle with $w.l$ as the center and $w.d$ as the radius, within which w can accept assignments.

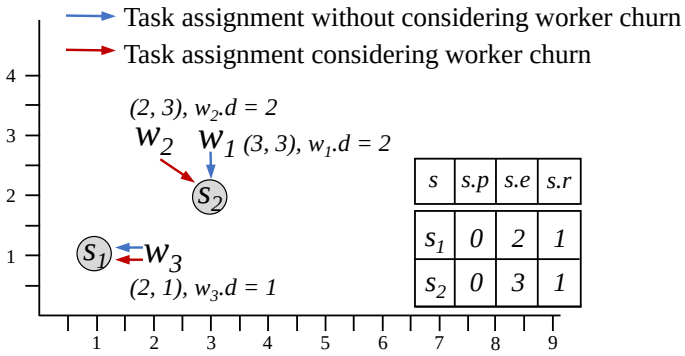


Fig. 1. Running Example

In Figure 1, three workers, w_1, w_2 and w_3, exist, which are associated with their reachable distance, e.g., the reachable distance of w_1 is 2. A worker can be in either online or offline mode. A worker is online when being ready to accept tasks. In our work, a worker can handle only one task at a certain time instance, which is reasonable in practice. Once the SC server assigns a task to a worker, the worker is considered being offline until the assigned task is completed.

Definition 3 (Idle Time Interval). Given a worker w who has performed n tasks in a time period, we define the task-performing history of w as a time-ordered task sequence, $S_w = (s_1, s_2, ..., s_n)$. The idle time interval of worker w is the time interval between two adjacent performed tasks, i.e., $w.\tau_i = s_{i+1}.t_s - s_i.t_e$ $(i > 0)$, where $w.\tau_i$ denotes the idle time interval of w between s_i and s_{i+1}, $s_{i+1}.t_s$ denotes the start time (i.e., time of assignment) of s_{i+1}, and $s_i.t_e$ is the completion time of s_i.

In the rest of the paper, we will use the terms *idle time interval* and *idle interval* interchangeably.

Definition 4 (Churn-prone Worker). Given a temporal threshold μ, if the idle time interval of worker w exceeds μ, worker w is regarded as a churn-prone worker.

Definition 5 (Spatial Task Assignment). Given a set of workers $W = \{w_1, w_2, ..., w_{|W|}\}$ and a set of tasks $S = \{s_1, s_2, ..., s_{|S|}\}$, we define A as the spatial task assignment. Next, A consists of a set of tuples of form (w, s), where a spatial task s is assigned to worker w, satisfying all the workers' and tasks' spatio-temporal constraints.

We use $A.r$ to denote the total reward in task assignment A. The problem investigated can be stated as follows.

Worker Churn based Task Assignment (WC-TA) Problem Statement. Given a set of online workers W and a set of tasks S at the current time instance on an SC platform, the WC-TA problem is to find an optimal task assignment A_{opt} that achieves the following goals:

1) primary optimization goal: maximize the total reward among workers, i.e., $\forall A_i \in \mathbb{A}$ ($A_{opt}.r \geq A_i.r$), where $\mathbb{A}$ denotes all possible assignments; and

2) secondary optimization goal: maximize the assignment ratio of churn-prone workers, which is the ratio between the number of assigned churn-prone workers and the total number of churn-prone workers.

We assume that worker w_2 is a churn-prone worker in Figure 1. When we only aim to maximize the total reward among workers without considering worker churn, we can assign the closet worker to each task and get the task assignment $\{(w_1, s_2), (w_3, s_1)\}$ with the total reward of 2. However, such a task assignment might lead to the churn of w_2 since the worker is not assigned any task and cannot get any reward. When we consider worker churn, we can also achieve a total reward of 2 with the assignment $\{(w_2, s_2), (w_3, s_1)\}$, showing the necessity of considering worker churn.

4 Framework Overview

Our framework (cf. Figure 2) is comprised of two components: worker churn prediction and task assignment.

The first component aims to predict worker churn by calculating the idle time intervals for all workers based on workers' historical task-performing data. To this end, we utilize an LSTM-based Latent Feeling Capture (LFC) model, which is a regression model that predicts the idle time intervals of workers. More specifically, we extract the latent feeling of each worker from historical data, including worker ID, reward, time interval, and spatial distance between two adjacent completed tasks, as well as geographic locations of tasks. Taking the data related to latent feelings of each worker as input, the LFC model generates a worker idle time interval tensor, where each entry is the prediction of a worker's idle time interval in a certain time slot. Given a temporal threshold μ, e.g., a month, if the predicted idle time interval of a worker exceeds μ, the worker is regarded as a churn-prone worker; otherwise, the worker is regarded as an active one.

In practice, it is necessary for an SC server to take measures to retain the churn-prone workers to ensure continuous and high worker participation and satisfaction. The measures include assigning the churn-prone workers high-value tasks, e.g., tasks with high rewards or nearby tasks. Therefore, in the assignment component, by considering spatio-temporal constraints including workers' reachable region and tasks' expiration time, we assign tasks to the suitable workers by giving priority to the churn-prone workers. For the sake of efficiency, we propose a churn-aware greedy algorithm that tries to assign tasks to the workers who are most likely to churn. We also develop a churn-aware KM algorithm to maximize the total reward while giving priority to the churn-prone workers when assigning tasks.

5 Worker Churn Prediction

We adopt a behavior-based modeling method to help the LSTM-based LFC model predict worker churn, which turns the classification problem of churn prediction into a regression problem. Then we estimate the idle time intervals for each worker using the LSTM-based LFC model, based on which we obtain the churn-prone workers by introducing a temporal threshold.

5.1 Behavior-based Modeling

Traditionally, the problem of user churn prediction is treated as a classification problem with labeled data [40]. However, in the fields like SC, the standards of worker churn are not always the same, and it is hard to tell if a worker is really churned. Besides, we cannot get labeled data in SC. To solve these issues, we transform the worker churn problem into a behavior-based problem and introduce a BM-UCP (Behavior-based Modeling for User Churn Prediction) method [40] to solve it. Different from traditional methods of user churn prediction, BM-UCP converts the classification problem into a regression problem, which predicts the idle time intervals of users. More specifically, given a temporal threshold μ, BM-UCP trains a model that maximizes the accuracy defined in Equation 1.

$$accuracy = \frac{\sum_i^{|W|} l((\tau_i > \mu \wedge \tau_i^* > \mu) \vee (\tau_i \leq \mu \wedge \tau_i^* \leq \mu))}{|W|},$$
$$l(x) = \begin{cases} 1 & \text{if } x = True \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $l(x)$ is an indication function, τ_i is the predicted idle time interval of worker w_i, τ_i^* is the true value of the time interval, $\wedge$ is logical AND operation, $\vee$ is logical OR operation, and $|W|$ is the number of workers. Given a temporal threshold μ, we train the model by maximizing the churn prediction accuracy, i.e., minimizing the variance between τ and τ^* (cf. Equation 1).

5.2 LSTM-based Latent Feeling Capturing Model

We introduce the proposed LSTM-based Latent Feeling Capturing (LFC) model from three aspects, i.e., input, latent feeling capturing, and output, the structure of which is shown in Figure 3. In the following, we provide specifics on each aspect.

5.2.1 Input

In this part, we introduce the input of the LSTM-based LFC model. In practice, it is obvious that workers' idle time intervals are affected by workers' latent feelings such as satisfaction and passion about the assigned tasks. Specifically, satisfaction indicates the pleasure a worker gets from the task assignment or the fulfillment degree of the expectations of the worker, which determines the completion time of a performed task to some extent. Passion indicates the wish or the desire to perform a task, which determines the start time of a task to be performed. Both satisfaction and passion are used to express the latent feelings of a worker from different aspects. For example, after performing tasks for a long time period, the passion of a worker to perform one more tasks may be decreased. In this process, the satisfaction will be changed to affect the completion time of tasks due to the rewards or the difficulty of tasks. The idle time interval of a worker w is

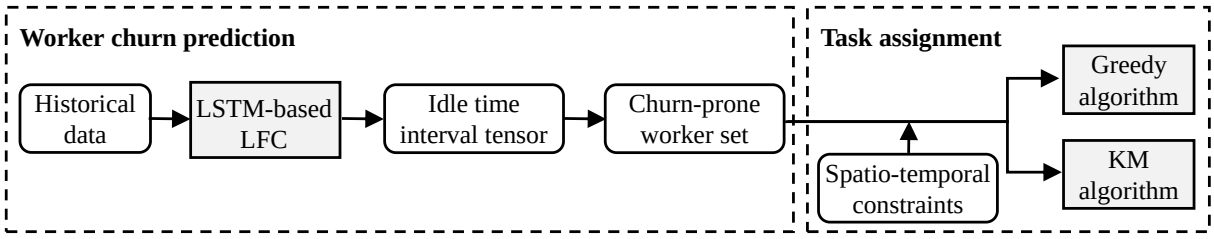


Fig. 2. Framework Overview

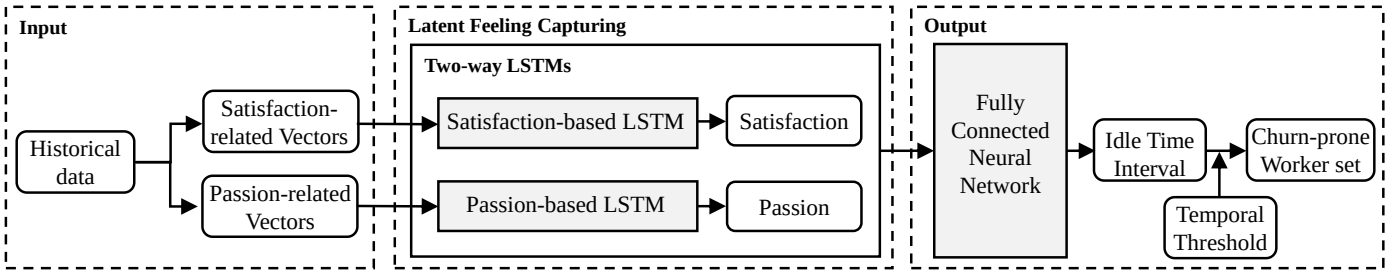


Fig. 3. Latent Feeling Capturing Model

the time interval between two adjacent performed tasks (see Definition 3), i.e., $w.\tau_i = s_{i+1}.t_s - s_i.t_e$, where $w.\tau_i$ denotes the idle time interval of w between two adjacent performed tasks (s_i and s_{i+1}), $s_{i+1}.t_s$ denotes the start time of s_{i+1} that is primarily determined by the worker passion, and $s_i.t_e$ is the completion time of s_i that is primarily determined by the worker satisfaction. Therefore, workers' satisfaction and passion about tasks can affect their idle time intervals.

We introduce two kinds of vectors as the input of LSTM-based LFC, i.e., satisfaction-related and passion-related vectors. Besides the rewards and the locations of tasks, we assume that satisfaction-related and passion-related vectors are primarily determined by the geographical distance (affecting the completion time of tasks to a certain extent) and the time interval between two adjacent tasks, respectively. Specifically, for worker w, the satisfaction-related vector $\boldsymbol{v}_t^s$, denoted by $\boldsymbol{v}_t^s = (r_t, d_t, lo_t, la_t)$, where r_t represents the reward of the tth task in the historical data of w, d_t represents the geographical distance between the tth task and the $(t-1)$th task, lo_t represents the longitude of the tth task, and la_t represents the latitude of the tth task. Passion is a feeling similar to satisfaction, which is denoted by $\boldsymbol{v}_t^p = (r_t, wt_t, lo_t, la_t)$, where wt_t represents the time interval between the tth task and the $(t-1)$th task in the historical data of w, and the remaining three elements (i.e., r_t, lo_t, and la_t) have the same meanings as the corresponding elements in the satisfaction-related vector.

The model is an LSTM-based model, and we set the time-step length to 10, i.e., an input instance consists of 10 satisfaction-related vectors or 10 passion-related vectors.

5.2.2 Latent Feeling Capturing

As we discussed above, the idle time interval is related to a worker's latent feelings (i.e., satisfaction and passion). A worker's satisfaction/passion after performing the tth task is related to the satisfaction/passion before the tth task and the features of the tth task. Therefore, we regard the latent feelings as sequential data, which means that the latent feelings have sequential dependencies. As shown in Figure 3, LFC contains two-way LSTMs, where one way is used to capture a worker's satisfaction and the other way is used to capture a worker's passion.

We will first introduce the process of capturing worker satisfaction. We take NL as a non-linear translation. Given the historical data of worker w, denoted by $S_w = \{s_1, s_2, ..., s_n\}$, w's satisfaction after performing task s_t depends on features of task s_t and the satisfaction after performing previous assigned tasks. As a result, we can calculate the worker's satisfaction in the current moment by the input satisfaction-related vector and the satisfaction gotten from performing previous assigned tasks, as shown in Equation 2.

$$\mathcal{S}_t = NL_{st}(s_t) + NL_{ss}(\mathcal{S}_{t-1}), \tag{2}$$

where $\mathcal{S}_t$ is worker w's satisfaction after performing the tth task, $NL_{st}(s_t)$ denotes the non-linear translation for w's satisfaction of the current task s_t that reflects the impact of s_t on worker satisfaction, and $NL_{ss}(\mathcal{S}_{t-1})$ denotes the non-linear translation for w's satisfaction after completing the $(t-1)$th task.

The process of calculating passion is similar to that of calculating satisfaction, which is shown in the following.

$$\mathcal{P}_t = NL_{pt}(s_t) + NL_{pp}(\mathcal{P}_{t-1}), \tag{3}$$

where $\mathcal{P}_t$ is worker w's passion after performing the tth task, $NL_{pt}(s_t)$ denotes the non-linear translation for w's passion of the current task s_t that reflects the impact of s_t on worker passion, and $NL_{pp}(\mathcal{S}_{t-1})$ denotes the non-linear translation for w's passion after completing the $(t-1)$th task.

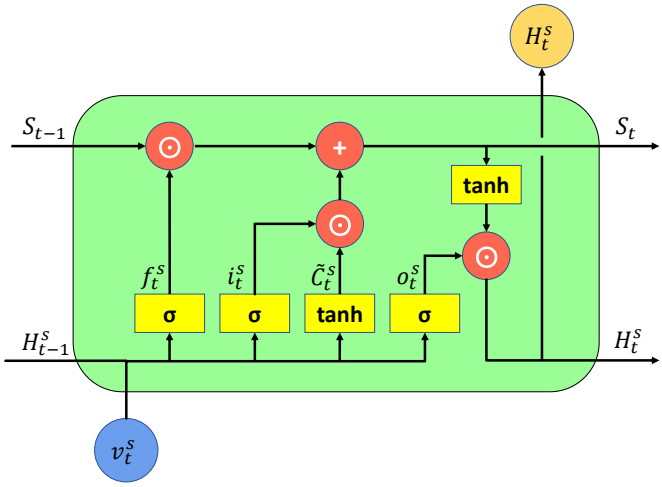


Fig. 4. LSTM-based Satisfaction Capturing

As discussed above, both satisfaction and passion have sequential dependencies. LSTM solves the long-term dependency problem through three gate mechanisms and has excellent performance when processing sequential data. Therefore, we introduce LSTM [20] into our model to capture satisfaction and passion.

Next, we illustrate the LSTM-based satisfaction capturing, which is shown in the following equations.

$$f_t^s = \sigma(\mathcal{W}_f^s \cdot [H_{t-1}^s, \boldsymbol{v}_t^s] + b_f^s), \tag{4}$$

$$i_t^s = \sigma(\mathcal{W}_i^s \cdot [H_{t-1}^s, \boldsymbol{v}_t^s] + b_i^s), \tag{5}$$

$$\tilde{C}_t^s = tanh(\mathcal{W}_c^s \cdot [H_{t-1}^s, \boldsymbol{v}_t^s] + b_c^s), \tag{6}$$

$$o_t^s = \sigma(\mathcal{W}_o^s \cdot [H_{t-1}^s, \boldsymbol{v}_t^s] + b_o^s), \tag{7}$$

$$\mathcal{S}_t = f_t^s \odot \mathcal{S}_{t-1} + i_t^s \odot \tilde{C}_t^s, \tag{8}$$

$$H_t^s = o_t^s \odot tanh(\mathcal{S}_t), \tag{9}$$

where f_t^s indicates what to forget, and the concatenation of satisfaction-related vector $\boldsymbol{v}_t^s$ and the filtered satisfaction H_{t-1}^s from the last time step are taken as input. We use a sigmoid function $\sigma(\cdot)$ as the activation function when calculating f_t^s. Note that the $\mathcal{W}$ terms denote weight matrices, e.g., $\mathcal{W}_f^s$ is the matrix of weights from the input to the forget gate f, and the b terms denote bias vectors, e.g., b_f^s is the forget gate bias vector. Next, i_t^s indicates what to update, the input and activation function are the same as those of calculating f_t^s but the parameters are different. Further, $\tilde{C}_t^s$ is a candidate satisfaction, which is calculated by taking the concatenation of $\boldsymbol{v}_t^s$ and filtered satisfaction H_{t-1}^s as input and selecting a tanh function $tanh(\cdot)$ for activation. Then, the new satisfaction is an addition of the values remembered from $\mathcal{S}_{t-1}$ (calculated by $f_t^s \odot \mathcal{S}_{t-1}$), and the values that need to be updated from the candidate satisfaction $\tilde{C}_t^s$ (calculated by $i_t^s \odot \tilde{C}_t^s$), where $\odot$ is the elementwise product. Finally, o_t^s indicates what to output, which is based on our cell state. The filtered satisfaction H_{last}^s of the last time step will be used as the output of LSTM for idle time interval calculation, and the satisfaction of worker w, denoted by $\mathcal{S}_w$, is the satisfaction of the last time step. The structure of the LSTM-based satisfaction capturing is shown in Figure 4.

The process of capturing passion is similar to that of satisfaction.

Context-based Optimization. Although the original LSTM has shown promising performance in modeling dependencies and dynamics in sequential data, it does not have strong and explicit capability for capturing the contextual information in the sequential data. In particular, the original LSTM does not fully interact with the inputs and the hidden states at multiple prior moments, but only correlates with the input at the current moment and the prior hidden state, leading to poor accuracy in capturing workers' long- and short-term satisfaction and passion. To tackle this issue and inspired by the success of Mogrifier LSTM [31] in modeling contexts in Natural Language Processing, we introduce two context-LSTMs (i.e., satisfaction-based and passion-based context-LSTMs) to replace the original LSTMs in the LFC model, which involves interactions between inputs and their contexts. Specifically, the context-LSTM equips the LSTM with gates that scale its weight matrices $\mathcal{W}$ in a context-dependent manner. Mogrifier LSTM is a general model for sequential data, but it only processes the worker's task-performing sequence sequentially and cannot integrate spatial information to consider the relationship between tasks. Our satisfaction-related vectors, which is the input of Mogrifier LSTM, consider the spatial distance between two adjacent tasks, which integrates spatial information into the Mogrifier LSTM and makes up for its defect of ignoring spatial information.

Based on Equations 4–9 that illustrate the satisfaction-based LSTM, we can simplify it as $LSTM(\boldsymbol{v}_t^s, \mathcal{S}_{t-1}, H_{t-1}^s)$. In the satisfaction-based context-LSTM, the two inputs $\boldsymbol{v}_t^s$ and H_{t-1}^s modulate one another in an alternating way before the usual LSTM computation, which is shown in Figure 5. Specifically, the satisfaction-based context-LSTM can be represented by $CLSTM(\boldsymbol{v}_t^s, \mathcal{S}_{t-1}, H_{t-1}^s) = LSTM(\boldsymbol{v}_{t,last}^s, \mathcal{S}_{t-1}, H_{t-1,last}^s)$, where the modulated inputs $\boldsymbol{v}_{t,last}^s$ and $H_{t-1,last}^s$ are the final ones after the modulation. Taking Figure 5 as a case, $CLSTM(\boldsymbol{v}_t^s, \mathcal{S}_{t-1}, H_{t-1}^s) = LSTM(\boldsymbol{v}_{t,5}^s, \mathcal{S}_{t-1}, H_{t-1,4}^s)$. Formally,

$$\boldsymbol{v}_{t,j}^s = 2\sigma(\mathbf{Q}^j H_{t-1,j-1}^s) \odot \boldsymbol{v}_{t,j-2}^s, \text{ for odd } j \in [1, \ldots, r], \tag{10}$$

$$H_{t-1,j}^s = 2\sigma(\mathbf{R}^j v_{t,j-1}^s) \odot H_{t-1,j-2}^s, \text{ for even } j \in [1, \ldots, r], \tag{11}$$

where $\boldsymbol{v}_{t,-1}^s = \boldsymbol{v}_t^s$ and $H_{t-1,0}^s = H_{t-1}^s$. Next, r denotes the number of modulation rounds that is calculated by $r = \frac{1}{d_t+1}N$, where d_t denotes the geographical distance between the tth task and the $(t-1)$th task, and N is a hyperparameter. We should notice that the context-LSTM is converted back into LSTM when $r = 0$.

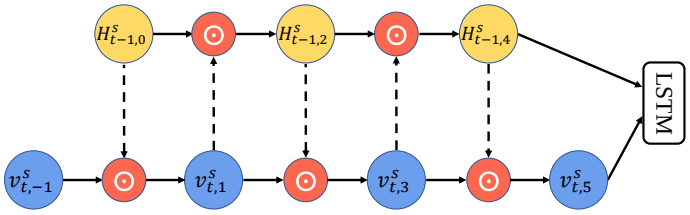


Fig. 5. Five Modulation Rounds of Satisfaction-based Context-LSTM

The passion-based LSTM can be optimized in the same way but with different number of modulation rounds, which is set to $\frac{1}{wt_t+1}N$, where wt_t is the time interval between the tth task and the $(t-1)$th task in the historical data of worker w.

5.3 Output

Since the idle time interval of worker w is related to the latent feelings including satisfaction and passion, the satisfaction and passion output by the two-way LSTMs can be concatenated and then passed through a fully connected layer, by which the predicted idle time interval of w is output, as shown in Equation 12.

$$\tau_w = \mathcal{W}_{sp} \cdot [\mathcal{S}_w, \mathcal{P}_w] + b_{sp}, \tag{12}$$

where τ_w is the predicted idle time interval of worker w, $\mathcal{W}_{sp}$ is the weight matrix, and b_{sp} is the bias.

Given a temporal threshold μ, if the predicted idle time interval of worker w, τ_w, exceeds μ, w is judged as a churn-prone worker.

6 Task Assignment

In a real-time scenario, where workers and tasks arrive dynamically and require immediate responses from an SC server, it is challenging to achieve the global optimal solution for the WC-TA problem. Since an SC server only has local knowledge of available tasks and workers at any instance of time, we optimize the task assignment locally at every time instance by maximizing the current assignments and giving higher priorities to workers who are more likely to be churned. We propose two task assignment algorithms on this basis, a Churn-aware Greedy algorithm and a Churn-aware KM algorithm.

6.1 Churn-aware Greedy Algorithm

Taking workers' idle time intervals as the priority of task assignment, we propose a basic greedy solution to solve the WC-TA problem.

Specifically, given a set of online workers, $W = \{w_1, w_2, ..., w_{|W|}\}$, and a set of available tasks, $S = \{s_1, s_2, ..., s_{|S|}\}$, at the current time, the available workers for spatial task s ($s \in S$), denoted as $AW(s)$, should satisfy the following two conditions: $\forall w \in AW(s), s \in S$,

1) $dis(w.l, s.l) \leq w.d$, and
2) $t_{now} + t(w.l, s.l) \leq s.e$,

where $dis(w.l, s.l)$ is the distance (e.g., Euclidean distance) between $w.l$ and $s.l$, t_{now} is the current time, and $t(w.l, s.l)$ is the travel time from $w.l$ to $s.l$. For the sake of simplicity, we assume that all the workers share the same speed, so the travel time between two locations can be estimated with their Euclidean distance, e.g., $t(w.l, s.l) = dis(w.l, s.l)$. However, our proposed algorithms are not dependent on this assumption and can handle the case where workers are moving at different speeds.

The churn-aware greedy algorithm is shown in Algorithm 1. Given a worker set W and a task set S, we calculate the set of available workers for each task according to the conditions described above (line 3). Then, we sort the workers in the available worker set for each task in descending order according to worker-task greedy value ρ (line 4), where $\rho = \mathcal{W}_c \cdot w.\tau + \mathcal{W}_r \cdot s.r$, $\mathcal{W}_c$ is the weight of w's predicted idle time interval $w.\tau$, and $\mathcal{W}_r$ is the weight of s's reward $s.r$. Finally, each task is assigned to the worker $\tilde{AW}(s)[0]$ with the largest greedy value (line 5).

Algorithm 1: Churn-aware Greedy Algorithm

Input: W, S
Output: A
1 $A \leftarrow \emptyset$;
2 **for** *each task* $s \in S$ **do**
3 $\quad AW(s) \leftarrow$ Find the set of available workers of s;
4 $\quad \tilde{AW}(s) \leftarrow$ Sort $AW(s)$ in descending order of the worker-task greedy value ρ;
5 $\quad A \cup [(s, \tilde{AW}(s)[0])]$;
6 **return** A;

The time complexity of Algorithm 1 is $O(|S| \cdot (|W| + |maxAW| \cdot \log|maxAW|))$, where $|S|$ denotes the number of tasks, $|W|$ denotes the number of workers, and $|maxAW|$ denotes the maximum number of available workers among all tasks, i.e., $|maxAW| = \max_{s \in S} |AW(s)|$.

6.2 Churn-aware KM Algorithm

In this part, we transform the WC-TA problem to a Bipartite Maximum Weight Matching problem and apply the KM algorithm to solve it. The Bipartite Maximum Weight Matching is based on a graph, which is represented by $G = (V, E)$ with V corresponding to the set of vertices and E the set of edges. Given a set of online workers, $W = \{w_1, w_2, ..., w_{|W|}\}$, and a set of available tasks, $S = \{s_1, s_2, ..., s_{|S|}\}$, the number of V and the number of E are fixed to $|W|+|S|$ and $\sum_{w \in W} m_w$, respectively, where m_w is the number of worker w's adaptive available assignments, which is a subset of worker w's available tasks $AS(w)$ and positively related to the predicted idle time interval of w. Each task s in $AS(w)$ should meet the two conditions of $AW(s)$ in Section 6.1, i.e., $\forall s \in AS(w)(dis(w.l, s.l) \leq w.d, t_{now} + t(w.l, s.l) \leq s.e)$. For the vertex construction, the entire point set V is divided into two sets V^W and V^S, where $V^W \cap V^S = \emptyset$. Each worker w maps to a vertex, $v_w \in V^W$, and each spatial task s maps to a vertex, $v_s \in V^S$.

Due to the spatio-temporal constraints, we add an edge from v_w mapped from $w \in W$ to the vertex v_s mapped from $s \in S$ if s can be assigned to w, i.e., $s \in AS(w)$. For each edge (v_w, v_s), its weight (denoted by $weight(v_w, v_s)$) can be measured as the weighted sum of the time interval $w.\tau$ and the reward of the spatial task s, i.e., $weight(v_w, v_s) =$

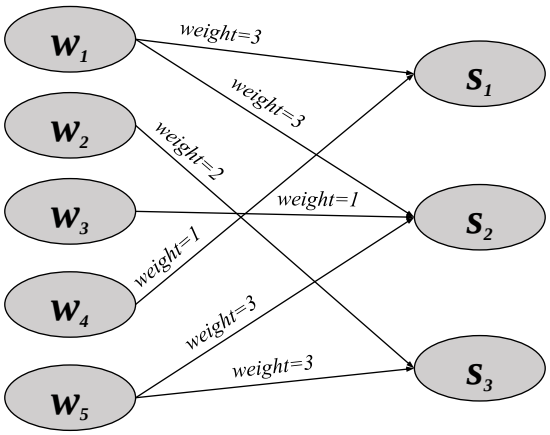


Fig. 6. Worker-Task Bipartite Graph

$\mathcal{W}_c \cdot w.\tau + \mathcal{W}_r \cdot s.r$, where $weight(v_w, v_s)$ is the worker-task greedy value ρ (see Section 6.1), $\mathcal{W}_c$ is the weight of w's predicted idle time interval $w.\tau$, and $\mathcal{W}_r$ is the weight of s's reward $s.r$. Figure 6 depicts an example of a graph for five workers and three tasks.

To improve efficiency, we limit the number of edges in the graph. Each worker node v_w has an adaptive upper limit u_w. It means that the number of edges associated with a worker node cannot exceed u_w, which is calculated by $\rho \cdot \frac{w.\tau}{\mathcal{T}}$, where ρ is a hyperparameter, called limit coefficient. Next, $\mathcal{T}$ is a time range (e.g., three months) used to narrow $w.\tau$ down to a certain computer-convenient range, which can be set by observation from a real application or specified by the SC platform. We retain the top-k edges with the maximal weights and remove the remaining ones for each worker node, where $k = \min\{u_w, |AS(w)|\}$, u_w is the upper limit, and $|AS(w)|$ denotes the number of w's available tasks. For the same consideration, the number of associated edges of each task node v_s cannot exceed u_s, in order to reduce the competition among workers and the recursion depth of the algorithm, where u_s is a hyperparameter.

The WC-TA problem is now converted into a Bipartite Maximum Matching problem in the direct bipartite graph G, which is to achieve the maximum weight matching of G. In our work, we use the KM algorithm with limited recursions to find the maximal weight matching.

The Churn-aware KM algorithm is shown in Algorithm 2. The input is the bipartite graph G, which is composed of two vertex sets V^S and V^W. The algorithm first initializes task assignment A, task expectation E_S, and a relaxation array $slack$ that is used to update worker and task expectations. Then, for each vertex w in W (i.e., $v_w \in V^W$), its expectation is set to the largest weight among the edges associated with it in graph G (lines 4–5). Next, Algorithm 2 recursively finds matching tasks for worker w through the FindTask function (lines 6–11). The FindTask function is shown in Algorithm 3, which will be introduced later. If w fails to match a task, we adjust the expectations of workers and tasks involved in the last matching to change the competitive relationship among workers so that more workers can be assigned (lines 12–24).

The original KM algorithm is used to find the

Algorithm 2: Churn-aware KM Algorithm

Input: G
Output: A
1 $A \leftarrow [-1, ..., -1]$;
/*$A = [A(s_1), ..., A(s_{|S|})]$;*/
2 $E_S \leftarrow [0, ..., 0]$;
/*$E_S = [E_S(s_1), ..., E_S(s_{|S|})]$;*/
3 $slack \leftarrow [INF, ..., INF]$;
/*$slack = [slack(s_1), ..., slack(s_{|S|})]$;*/
4 **for** *each worker* $w \in W$ **do**
5 $\quad E_W(w) \leftarrow max\{weight(v_w, v_s)\}$;
6 **for** *each worker* $w \in W$ **do**
7 $\quad$ **while** $E_W(w) > 0$ **do**
8 $\quad\quad vis_S \leftarrow [False, ..., False]$;
$\quad\quad$ /*$vis_S = [vis_S(s_1), ..., vis_S(s_{|S|})]$;*/
9 $\quad\quad vis_W \leftarrow [False, ..., False]$;
$\quad\quad$ /*$vis_W = [vis_W(w_1), ..., vis_W(w_{|W|})]$;*/
10 $\quad\quad$ **if** *FindTask(w,0)* **then**
11 $\quad\quad\quad$ break;
12 $\quad\quad$ **else**
13 $\quad\quad\quad d \leftarrow INF$;
14 $\quad\quad\quad$ **for** *each task* $s \in S$ **do**
15 $\quad\quad\quad\quad$ **if** ! $vis_S(s)$ **then**
16 $\quad\quad\quad\quad\quad d \leftarrow min\{d, slack(s)\}$;
17 $\quad\quad\quad$ **for** *each worker* $w \in W$ **do**
18 $\quad\quad\quad\quad$ **if** $vis_W(w)$ **then**
19 $\quad\quad\quad\quad\quad E_W(w) -= d$;
20 $\quad\quad\quad$ **for** *each task* $s \in S$ **do**
21 $\quad\quad\quad\quad$ **if** $vis_S(s)$ **then**
22 $\quad\quad\quad\quad\quad E_S(s) += d$;
23 $\quad\quad\quad\quad$ **else**
24 $\quad\quad\quad\quad\quad slack(s) -= d$;
25 **return** A;

Algorithm 3: FindTask Algorithm

Input: *worker* w, *recursion depth* RT
Output: $Bool$
1 $vis_W(w) \leftarrow True$;
2 **if** $RT > \lambda$ **then**
3 $\quad$ **return** $False$;
4 **else**
5 $\quad$ **for** *each task* s *is adjacent to* w *in* G **do**
6 $\quad\quad$ **if** $vis_S(s)$ **then**
7 $\quad\quad\quad$ continue;
8 $\quad\quad gap \leftarrow E_W(w) + E_S(s) - weight(v_w, v_s)$;
9 $\quad\quad$ **if** $gap = 0$ **then**
10 $\quad\quad\quad$ **if** $A(s) = -1$ *or FindTask(*$A(s)$*,*$RT+1$*)* **then**
11 $\quad\quad\quad\quad A(s) \leftarrow w$;
12 $\quad\quad\quad\quad$ **return** $True$;
13 $\quad\quad$ **else**
14 $\quad\quad\quad slack(s) \leftarrow min\{slack(s), gap\}$;
15 $\quad$ **return** $False$;

perfect matching of a weighted bipartite graph. However, considering that a perfect matching may not exist in a worker-task bipartite graph, we propose some optimization strategies to improve the KM algorithm. In the original KM algorithm, it does not stop trying to match tasks for a worker until a successful match, which may cause an endless loop in our problem. Therefore, in our algorithm, if the expectation of w is less than 0 (line 7), we stop matching tasks for the worker.

In the following, we introduce the FindTask algorithm (see Algorithm 3), which is a Depth First Search algorithm that finds a suitable task for each worker. The depth of recursion cannot exceed the upper recursion limit λ (lines 2–3). In the algorithm, we calculate the difference between the sum of expectations of the worker and the task and the weight of the edge associated with the two vertices (lines 5–8). If the difference is equal to 0, the task can be assigned to the worker (lines 9–11). Else, $slack(s)$ is updated to $min\{slack(s), gap\}$ (lines 13–14).

Algorithm 2 has time complexity $O(|W| \cdot (|S| + |W|))$, where $|W|$ and $|S|$ are the numbers of workers and tasks, respectively.

Travel Distance Optimization. An issue of the original KM task assignment algorithm is that it does not consider travel distance of workers during the assignment process. In the problem settings of SC, travel distance is a critical factor since workers must go to the task location physically to perform the task. To address this issue and considering the fact that a worker is more likely to accept nearby tasks, we propose a strategy, referred to as travel distance optimization, to improve the overall task assignment by giving priority to the worker-task assignments with lower travel distance. Specifically, we carefully design a distance discount factor $\delta(w.l, s.l)$ between worker w and task s, which is shown in Equation 13.

$$\delta(w.l, s.l) = 1 - \min\{1, dis(w.l, s.l)/w.d\}, \qquad (13)$$

where $w.l/s.l$ denotes the location of w/s, $dis(w.l, s.l)$ is the distance between w and s, and $w.d$ is the reachable distance of w. Next, $dis(w.l, s.l)/w.d$ indicates the willingness of worker w for performing task s based on travel distance. The lower $dis(w.l, s.l)/w.d$ is, the higher willingness the worker has. Accordingly, a higher $\delta(w.l, s.l)$ value means that worker w are more likely to perform task s.

We consider the travel distance optimization by applying the distance discount factor in the churn-aware KM algorithm. In particular, we replace the weight between worker node v_w and task node v_s, $weight(v_w, v_s)$, by $\delta(w.l, s.l) \cdot weight(v_w, v_s)$ in line 5 of Algorithm 2 and line 8 of Algorithm 3. We study the effect of the travel distance optimization strategy in our experimental part in Section 7.2.2. The studies show that the optimization strategy can result in nearly the same task assignment results (including the total reward and the assignment ratio of churn-prone workers) as does the KM method without travel distance optimization, and it can reduce the total travel distance of task assignment.

7 Experimental Evaluation

We evaluate the performance of the proposed methods on real data. The experimental setup is presented in Section 7.1, followed by a coverage of key experimental results in Section 7.2.

7.1 Experimental Setup

We use a check-in dataset from Yelp to simulate our problem, which is a common practice in evaluation of SC platforms [9], [16], [17]. In order to make the user data more representative, we filter the data, where we only use the data of users with the number of reviews exceeding 20 and the number of reviews before 2019-05-14 23:22:59 exceeding 10. The resulting dataset provides check-in data from eight metropolitan areas in the USA, which includes 160,585 POIs and 31,262 users. We assume that we assign tasks to workers in a certain time slot in the near future, i.e., 2019-05-14 23:22:59. We also assume that the users are the workers of an SC platform since users who check in to different spots are good candidates to perform spatial tasks in the vicinity of those spots, and their locations are those of the most recent check-in points. For each POI, we use its location and stars as the location and reward of a task, respectively. Checking in a POI is equivalent to accepting a task. The distance is calculated by the Euclidian distance. The experiments are implemented on an Intel(R) Xeon(R) CPU Silver 4214 @ 2.20GHz, and NVidia GeForce RTX 2080Ti GPU.

7.2 Experiment Results

7.2.1 Performance of Worker Churn Modeling

We first evaluate the performance of worker churn prediction.

Evaluation Methods. We study the performance of the following methods.

1) LR: A linear regression method [32], which is a statistical method that uses regression analysis to determine the quantitative relationship between two or more variables. Its expression is $y = wx + b$, where x is historical data, and y denotes the predicted idle time interval in our problem.

2) MC: A multi-layer fully connected neural network method [15], which is a computational model. MC is composed of a large number of nodes (i.e., neurons) connected to each other. Each node represents a specific output function, i.e., an activation function. Each connection between two nodes represents a weighted value for the signal passing through the connection, called a weight, which is equivalent to the memory of an artificial neural network. The output of the network is different depending on the connection method of the network, the weight value, and the activation function.

3) LFC: Our latent feeling capture model, which is based on two-way LSTMs.

4) CLFC: Our LFC model with two-way context-LSTMs.

Metrics. To evaluate the accuracy of worker churn prediction, we adopt the metric, $accuracy$, shown in Equation 1. We randomly select 20% of the reviews from the dataset, which are used as the testing data to evaluate the inferred values. The remaining 80% are used as the training

TABLE 2
Accuracy of Worker Churn Prediction

Methods	Temporal threshold (%)			
	half a month	1 month	2 months	3 months
LR	38.04±0.38	57.15±0.64	77.70±0.47	84.13±0.28
MC	37.90±0.45	56.91±0.58	77.76±0.43	84.46±1.20
LFC	59.90±1.02	67.96±0.70	78.97±0.44	84.73±0.32
CLFC	**70.78±0.28**	**76.89±0.26**	**83.74±0.24**	**87.78±0.22**

data. We divide the testing dataset into 20 subsets evenly, run the methods on each subset, and report all results.

Accuracy. We report the accuracy values of all methods in Table 2, where the accuracy values consist of the mean and standard deviation of all results, e.g., $a \pm b$ denotes that a is the mean and b is the standard deviation of all results. Under different temporal thresholds, i.e., half a month, one month, two months, and three months, CLFC always achieves the highest mean accuracy, followed by LFC. CLFC outperforms LFC by 3.60%–18.16% in terms of the mean accuracy, which demonstrates the superiority of CLFC for predicting the worker churn. We also observe that CLFC always has the lowest standard deviations, which indicates that the accuracy results tend to be close to the mean, i.e., the results are relatively consistent, showing the reliability of CLFC for worker churn prediction.

7.2.2 Performance of Task Assignment

In this set of experiments, we evaluate the performance of task assignment.

Evaluation Methods. We study the following methods.

1) Greedy: The greedy task assignment method that does not consider the worker churn but considers the travel distance instead.

2) Greedy+WC: The greedy method based on the worker churn predicted by LFC.

3) DR: The Degree-Reduction-based task assignment method [43] that utilizes a degree reduction strategy. It is based on the same bipartite graph with the KM method, which finds the worker/task node with the minimal degree (i.e., the node with least edges) and assigns its connected node with the maximal weight. The intuition is that the nodes with less edges are more likely to be assigned unsuccessfully when they are assigned later, so DR gives priority to them.

4) KM: The KM task assignment method that does not consider the worker churn but considers the travel distance instead.

5) KM+WC: The KM method based on the worker churn predicted by LFC.

6) KM+CWC: The KM method based on the worker churn predicted by CLFC.

7) KM+CWC+Dis: The KM+CWC method, where the travel distance optimization strategy is adopted in task assignment.

Metrics. Three metrics are compared among the methods, i.e., *CPU Time* for finding task assignments, *Total Reward*, *Assignment Ratio* of churn-prone workers (marked as *AR*), and *Average Travel Distance* for workers. *AR* can be computed in Equation 14.

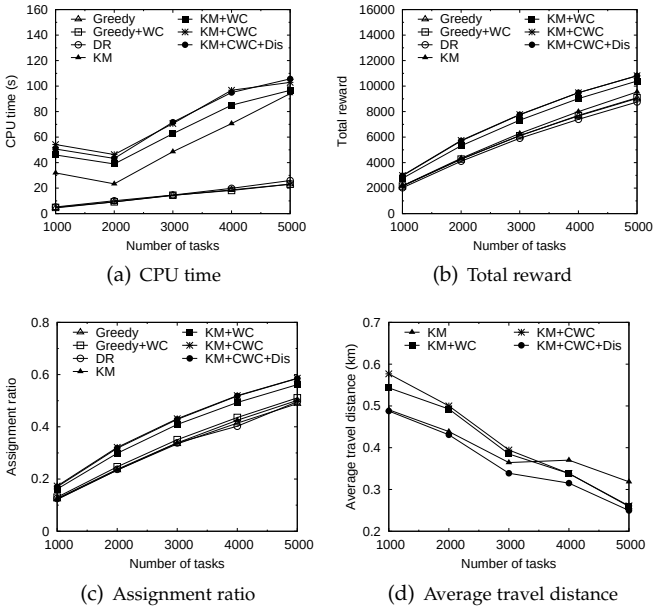
$$AR = \frac{N(W_{assign} \cap W_{churn})}{N(W_{churn})}, \tag{14}$$

where $N(W_{assign} \cap W_{churn})$ denotes the number of the assigned churn-prone workers, and $N(W_{churn})$ denotes the number of churn-prone workers. A larger *AR* implies more effective task assignments that give priority to churn-prone workers when assigning tasks. Table 3 shows our experimental settings, where default values are underlined.

TABLE 3
Experimental Parameters

Parameter	Values
Number of tasks $\|S\|$	1000, <u>2000</u>, 3000, 4000, 5000
Number of workers $\|W\|$	1000, 2000 , 3000, 4000, <u>5000</u>
Reachable distance of workers d (km)	0.05, 0.1, <u>0.5</u>, 1.0, 5.0
Valid time of tasks $e-p$ (h)	0.05, 0.1, <u>0.3</u>, 0.5, 0.7
Limit coefficient ρ	2, 4, <u>6</u>, 8, 10

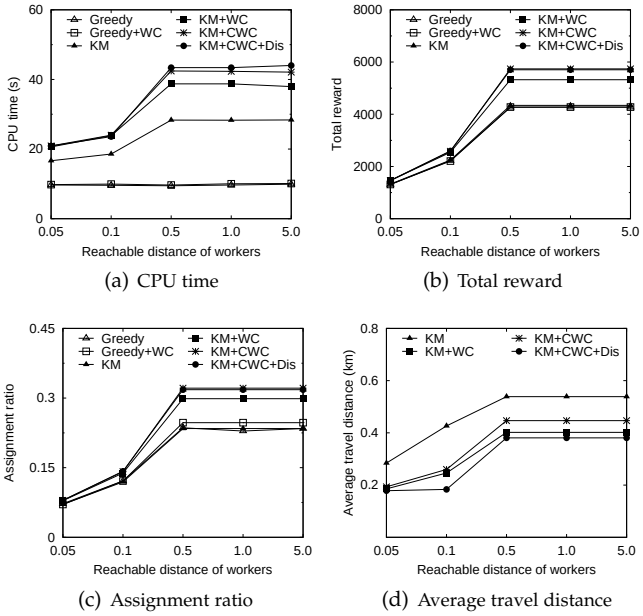
Effect of $|S|$. To study the scalability of the proposed methods, we generate five datasets containing 1,000 to 5,000 tasks by random selection from the Yelp dataset. As shown in Figure 7(a), for all greedy-related methods (including Greedy, Greedy+WC and DR), the CPU time exhibits a similar increasing trend as $|S|$ increases. However, for KM-related methods (i.e., KM, KM+WC, KM+CWC, and KM+CWC+Dis), the CPU time is not simply positively correlated with $|S|$. When $|S|$ is small (i.e., $|S| = 1,000$), the competition among workers is intense, which may increase the recursion depth of KM-related methods. When $|S|$ is larger than 2,000 and gets larger, the CPU time of KM-related methods increases since they need to search more tasks. Although KM+CWC and KM+CWC+Dis are more time-consuming than KM and KM+WC, they are able to obtain the most rewards and achieve the highest assignment ratio, as shown in Figures 7(b) and 7(c), which demonstrates the effectiveness of the proposed context-LSTM model. As expected in Figure 7(b), the total rewards obtained by all methods increase with increasing $|S|$. Since Greedy and Greedy+WC are greedy solutions, they perform worse than their counterparts (i.e., KM and KM+WC), respectively. DR performs worse than Greedy and Greedy+WC in terms of the total reward since DR gives priority to the worker/task nodes with the minimal degree, i.e., workers/tasks that have less associated tasks/workers, and then considers task reward. For the KM-related methods that considering worker churn, i.e., KM+WC, KM+CWC, and KM+CWC+Dis, churn-prone workers can be associated with more task vertices in the bipartite graphs, which changes the competitive relationship among workers. Therefore, the total rewards of KM+WC, KM+CWC, and KM+CWC+Dis are more than that of the original KM method. When it comes to assignment ratio in Figure 7(c), since Greedy+WC and KM+WC consider worker churn and give higher priority to churn-prone workers, they outperform their counterparts, i.e., Greedy, DR and KM that do not consider worker churn, respectively. As the number of tasks increases, the assignment ratio of all



(a) CPU time (b) Total reward

(c) Assignment ratio (d) Average travel distance

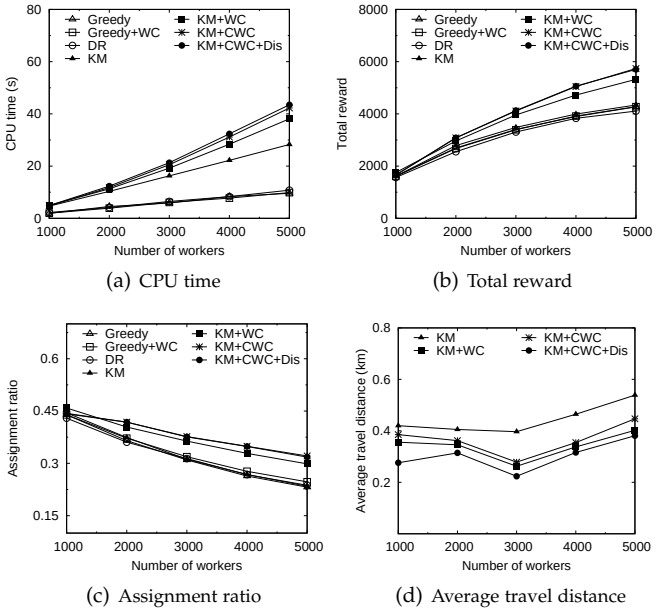
Fig. 7. Performance of Task Assignment: Effect of $|S|$



(a) CPU time (b) Total reward

(c) Assignment ratio (d) Average travel distance

Fig. 9. Performance of Task Assignment: Effect of d



(a) CPU time (b) Total reward

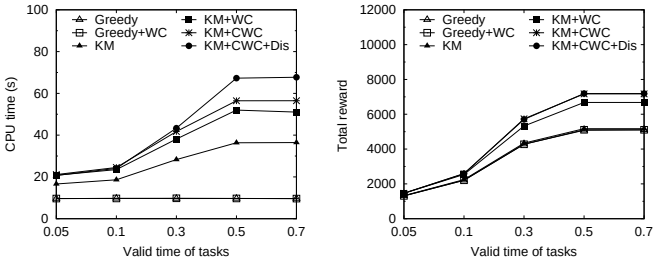
(c) Assignment ratio (d) Average travel distance

Fig. 8. Performance of Task Assignment: Effect of $|W|$

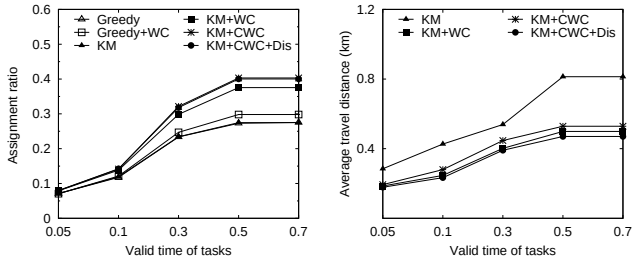
methods increases. The reason behind it is self-evident, that is, with more tasks to be assigned, each worker receives more available tasks such that the worker has more chance to be assigned a task. From Figure 7(d) we can see that KM+CWC+Dis achieves the minimal average travel distance, which shows the superiority of the travel distance optimization strategy.

Effect of $|W|$. To study the effect of $|W|$, we generate five datasets containing 1,000 to 5,000 workers by random selection from the testing set. As shown in Figure 8(a), the CPU time of all methods increase as $|W|$ increases since they need to traverse more workers. In Figures 8(b) and 8(c), KM+CWC and KM+CWC+Dis can achieve higher global rewards and assignment ratio than KM+WC while sacrificing some efficiency. However, the computational efficiency of KM+CWC and KM+CWC+Dis are acceptable. In Figure 8(b), as $|W|$ increases, more tasks can be assigned to workers, and thus the total rewards of all methods increase. We also observe that the total reward of DR is slightly less than those of other greedy methods for the same reason that DR prioritizes the workers/tasks that have less associated tasks/workers and then considers task reward. In Figure 8(c), the assignment ratio of all methods drops, but Greedy+WC and KM+WC always perform better than their own counterparts (i.e., Greedy and KM) without considering worker churn and are able to improve the assignment ratio by up to 6.56% and 27.48%, respectively. For the average travel distance shown in Figure 8(d), the average travel distance of all methods except KM+CWC+Dis decreases at first and then increases with $|W|$ getting larger. This is due to the fact that when the number of workers increases from 1000 to 3000, tasks are more likely to assign to nearby workers, thus reducing the average travel distance. When the number of workers increases from 3000 to 5000, the number of churn-prone workers also increases dramatically, which becomes the key factor in the average travel distance since all methods give priority to these churn-prone workers (that may be far away from the assigned tasks) in task assignment. We also observe that KM+CWC+Dis performs best, the average travel distance of which is 56.32%–94.75% of those of other methods, demonstrating the advantage of the travel distance optimization. To save space, in the following experiments, we do not report results of DR, as these are similar to those in the effect of $|S|$ and $|W|$, and reporting these results will impact the presentation of the whole experimental results.

Effect of d. We also study the effect of workers' reachable distance d. From Figure 9(a) we can see that, the CPU time of KM-related methods increase faster than those of greedy-related methods as d increases. This is because that as d increases, there are more available tasks for each worker and the competition among workers will be more intense,

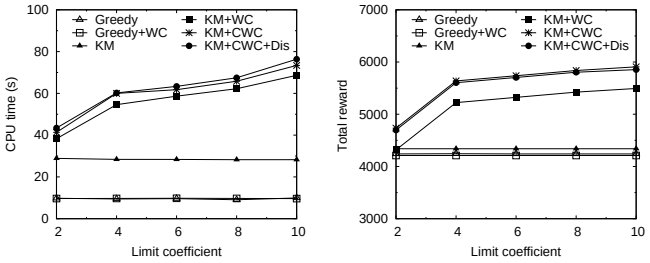


(a) CPU time (b) Total reward

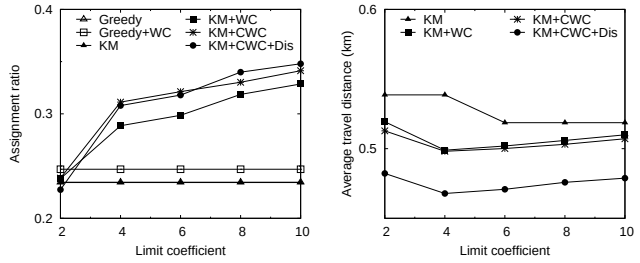


(c) Assignment ratio (d) Average travel distance

Fig. 10. Performance of Task Assignment: Effect of $e - p$



(a) CPU time (b) Total reward



(c) Assignment ratio (d) Average travel distance

Fig. 11. Performance of Task Assignment: Effect of ρ

which will increase the recursion depth of KM-related methods. Moreover, workers with larger reachable distances tend to have more available task assignments, which leads to more edges in the bipartite graphs of KM-related methods. The total rewards and assignment ratio of all methods increase as d increases, as shown in Figures 9(b) and 9(c). However, we also notice that limited by the number of tasks and workers, the effect of d on all methods becomes less pronounced when $d \geq 0.5$ km, i.e., all methods remain unchanged after d exceeds 0.5 km. Figure 9(d) shows that KM+CWC+Dis can reduce the average travel distance compared to other KM-related methods (i.e., KM, KM+WC, and KM+CWC), demonstrating the superiority of the travel distance optimization strategy. In particular, KM+CWC+Dis reduces the average travel distance by up to 29.38% compared to KM+CWC.

Effect of $e - p$. We then study the effect of the valid time $e - p$ of tasks in Figure 10. When $e - p$ increases, tasks can be assigned to more workers, which leads to more available worker-task matches and more CPU time, as shown in Figure 10(a). We also notice that the increase of CPU time of all methods become slower when $e - p \geq 0.5$ h since with longer task valid time there is increasing chance that most of the workers have already been added into the available worker sets, and thus few workers need to be added into the available worker sets. When $e - p$ gets larger, each task has a higher probability of being assigned to a suitable worker, so the total rewards and assignment ratio of churn-prone workers increase, which are shown in Figures 10(b) and 10(c). In Figure 10(d), KM+CWC+Dis still outperforms others in terms of the average travel distance regardless of $e - p$.

Effect of ρ. Finally, we study the effect of ρ, which limits the number of edges associated with worker vertices in KM+WC-related methods (i.e., KM+WC, KM+CWC, and KM+CWC+Dis). As ρ increases, each worker can be associated with more tasks, and the competition among workers is more intense, leading to increasing CPU time among the above KM+WC-related methods, which is shown in Figure 11(a). At the same time, as ρ increases, these methods can find a matching with greater weights. Therefore, the total rewards and the assignment ratio of these methods increase and are always better than those of other methods (i.e., Greedy, Greedy+WC, and KM) when $\rho > 2$, as shown in Figures 11(b) and 11(c). In Figure 11(d), KM+CWC+Dis can improve the average travel distance by up to 13.15% compared with others.

8 Conclusion and Future Work

The ubiquity of mobile devices with high-fidelity sensors and the sharp decreases in the cost of ultra-broadband wireless networks flourish the market of Spatial Crowdsourcing (SC), which consists of location-specific tasks and requires workers to physically be at specific locations to complete them. In this paper, we study a novel task assignment problem in SC, namely Worker Churn based Task Assignment (WC-TA). We address a few challenges by proposing different strategies to identify the workers who are easy to churn and consider their feelings when assigning tasks, so that they can get a better experience on SC platforms. Furthermore, we introduce two-way context-LSTMs to identify the churn-prone workers more accurately and design a travel distance optimization strategy to reduce the overall travel cost. Extensive experiments demonstrate the effectiveness of our proposed solutions.

Acknowledgment

This work is partially supported by NSFC (No. 61972069, 61836007 and 61832017), Shenzhen Municipal Science and Technology R&D Funding Basic Research Program (JCYJ20210324133607021), and Municipal Government of Quzhou under Grant No. 2022D037.

REFERENCES

[1] J. Ahn, J. Hwang, D. Kim, H. Choi, and S. Kang. A survey on churn analysis in various business domains. *Access*, 8:220816–220839, 2020.

[2] A. Amin, F. Al-Obeidat, B. Shah, M. A. Tae, C. Khan, H. U. R. Durrani, and S. Anwar. Just-in-time customer churn prediction in the telecommunication sector. *The Journal of Supercomputing*, 76(6):3924–3948, 2017.

[3] A. C. Bahnsen, D. Aouada, and B. Ottersten. A novel cost-sensitive framework for customer churn predictive modeling. *Decision Analytics*, 2(1):1–15, 2015.

[4] C. Cen, S. F. Cheng, A. Gunawan, A. Misra, K. Dasgupta, and D. Chander. Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing. In *AAAI*, pages 30–40, 2014.

[5] X. Chen, Y. Zhao, and K. Zheng. Task publication time recommendation in spatial crowdsourcing. In *CIKM*, pages 232–241, 2022.

[6] Z. Chen, P. Cheng, L. Chen, X. Lin, and C. Shahabi. Fair task assignment in spatial crowdsourcing. *PVLDB*, 13(12):2479–2492, 2020.

[7] Z. Chen, P. Cheng, Y. Zeng, and L. Chen. Minimizing maximum delay of task assignment in spatial crowdsourcing. In *ICDE*, pages 1454–1465, 2019.

[8] P. Cheng, L. Chen, and J. Ye. Cooperation-aware task assignment in spatial crowdsourcing. In *ICDE*, pages 1442–1453, 2019.

[9] P. Cheng, X. Lian, L. Chen, and C. Shahabi. Prediction-based task assignment in spatial crowdsourcing. In *ICDE*, pages 997–1008, 2017.

[10] P. Cheng, X. Lian, Z. Chen, L. Chen, J. Han, and J. Zhao. Reliable diversity-based spatial crowdsourcing by moving workers. *PVLDB*, 8(10):1022–1033, 2015.

[11] P. Cheng, X. Lian, X. Jian, and L. Chen. Frog: A fast and reliable crowdsourcing framework. *TKDE*, 31(5):894–908, 2019.

[12] Y. Cheng, B. Li, X. Zhou, Y. Yuan, G. Wang, and L. Chen. Real-time cross online matching in spatial crowdsourcing. In *ICDE*, pages 1–12, 2020.

[13] A. Chouiekh and E. Haj. Deep convolutional neural networks for customer churn prediction analysis. *Int. J. Cogn. Informatics Nat. Intell.*, 14(1):1–16, 2020.

[14] Y. Cui, L. Deng, Y. Zhao, B. Yao, V. W. Zheng, and K. Zheng. Hidden poi ranking with spatial crowdsourcing. In *SIGKDD*, pages 814–824, 2019.

[15] Y. L. Cun, L. Jackel, B. Boser, J. Denker, H. Graf, I. Guyon, D. Henderson, R. Howard, and W. Hubbard. Handwritten digit recognition: applications of neural network chips and automatic learning. *COMMUN MAG*, 27(11):41–46, 1989.

[16] H. Dang, T. A. Nguyen, and H. To. Maximum complex task assignment: Towards tasks correlation in spatial crowdsourcing. In *IIWAS*, pages 77–81, 2013.

[17] D. Deng, C. Shahabi, and U. Demiryurek. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing. In *SIGSPATIAL*, pages 324–333, 2013.

[18] D. Deng, C. Shahabi, and L. Zhu. Task matching and scheduling for multiple workers in spatial crowdsourcing. In *SIGSPATIAL*, pages 1–10, 2015.

[19] J. Dias, P. Godinho, and P. Torres. Machine learning for customer churn prediction in retail banking. In *ICCSA*, pages 576–589, 2020.

[20] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[21] A. Hudaib, R. Dannoun, O. Harfoushi, R. Obiedat, and H. Faris. Hybrid data mining models for predicting customer churn. *IJCNS*, 8(5):91–96, 2015.

[22] M. Karnstedt, M. Rowe, J. Chan, H. Alani, and C. Hayes. The effect of user features on churn in social networks. In *WebSci*, pages 1–8, 2011.

[23] L. Kazemi and C. Shahabi. Geocrowd:enabling query answering with spatial crowdsourcing. In *SIGSPATIAL*, pages 189–198, 2012.

[24] L. Kazemi, C. Shahabi, and L. Chen. Geotrucrowd:trustworthy query answering with spatial crowdsourcing. In *SIGSPATIAL*, pages 314–323, 2013.

[25] S. M. Kostic, M. Simic, and M. V. Kostic. Social network analysis and churn prediction in telecommunications using graph theory. *Entropy*, 22(7):753, 2020.

[26] T. Lai, Y. Zhao, W. Qian, and K. Zheng. Loyalty-based task assignment in spatial crowdsourcing. In *CIKM*, pages 1014–1023, 2022.

[27] X. Li, Y. Zhao, J. Guo, and K. Zheng. Group task assignment with social impact-based preference in spatial crowdsourcing. In *DASFAA*, pages 677–693, 2020.

[28] X. Li, Y. Zhao, X. Zhou, and K. Zheng. Consensus-based group task assignment with social impact in spatial crowdsourcing. *Data Science and Engineering*, 5(4):375–390, 2020.

[29] Y. Li, Y. Zhao, and K. Zheng. Preference-aware group task assignment in spatial crowdsourcing: A mutual information-based approach. In *ICDM*, pages 350–359, 2021.

[30] J. Liu, L. Deng, H. Miao, Y. Zhao, and K. Zheng. Task assignment with federated preference learning in spatial crowdsourcing. In *CIKM*, pages 1279–1288, 2022.

[31] G. Melis, T. Kočiskỳ, and P. Blunsom. Mogrifier lstm. *arXiv preprint arXiv:1909.01792*, 2019.

[32] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[33] W. Ni, P. Cheng, L. Chen, and X. Lin. Task allocation in dependency-aware spatial crowdsourcing. In *ICDE*, pages 985–996, 2020.

[34] T. Song, K. Xu, J. Li, Y. Li, and Y. Tong. Multi-skill aware task assignment in real-time spatial crowdsourcing. *GeoInformatica*, 24(1):153–173, 2019.

[35] Q. Tao, Y. Tong, Z. Zhou, Y. Shi, L. Chen, and K. Xu. Differentially private online task assignment in spatial crowdsourcing: A tree-based approach. In *ICDE*, pages 517–528, 2020.

[36] H. To, C. Shahabi, and L. Xiong. Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server. In *ICDE*, pages 833–844, 2018.

[37] Y. Tong, Y. xiang Zeng, B. Ding, L. Wang, and L. Chen. Two-sided online micro-task assignment in spatial crowdsourcing. *TKDE*, 33(5):2295–2309, 2021.

[38] Y. Tong, Z. Zhou, Y. Zeng, L. Chen, and C. Shahabi. Spatial crowdsourcing: a survey. *VLDBJ*, 29(1):217–250, 2020.

[39] Z. Wang, Y. Zhao, X. Chen, and K. Zheng. Task assignment with worker churn prediction in spatial crowdsourcing. In *CIKM*, pages 2070–2079, 2021.

[40] M. Xi, Z. Luo, N. Wang, and J. Yin. A latent feelings-aware rnn model for user churn prediction with behavioral data. *ArXiv*, abs/1911.02224, 2019.

[41] J. Xia, Y. Zhao, G. Liu, J. Xu, M. Zhang, and K. Zheng. Profit-driven task assignment in spatial crowdsourcing. In *IJCAI*, pages 1914–1920, 2019.

[42] G. Ye, Y. Zhao, X. Chen, and K. Zheng. Task allocation with geographic partition in spatial crowdsourcing. In *CIKM*, pages 2404–2413, 2021.

[43] Y. Zhao, X. Chen, L. Deng, T. Kieu, C. Guo, B. Yang, K. Zheng, and C. S. Jensen. Outlier detection for streaming task assignment in crowdsourcing. In *WWW*, pages 1933–1943, 2022.

[44] Y. Zhao, J. Guo, X. Chen, J. Hao, X. Zhou, and K. Zheng. Coalition-based task assignment in spatial crowdsourcing. In *ICDE*, pages 241–252, 2021.

[45] Y. Zhao, Y. Li, Y. Wang, H. Su, and K. Zheng. Destination-aware task assignment in spatial crowdsourcing. In *CIKM*, pages 297–306, 2017.

[46] Y. Zhao, J. Xia, G. Liu, H. Su, D. Lian, S. Shang, and K. Zheng. Preference-aware task assignment in spatial crowdsourcing. In *AAAI*, pages 2629–2636, 2019.

[47] Y. Zhao, K. Zheng, Y. Cui, H. Su, F. Zhu, and X. Zhou. Predictive task assignment in spatial crowdsourcing: a data-driven approach. In *ICDE*, pages 13–24, 2020.

[48] Y. Zhao, K. Zheng, J. Guo, B. Yang, T. B. Pedersen, and C. S. Jensen. Fairness-aware task assignment in spatial crowdsourcing: Game-theoretic approaches. In *ICDE*, pages 265–276, 2021.

[49] Y. Zhao, K. Zheng, Y. Li, H. Su, J. Liu, and X. Zhou. Destination-aware task assignment in spatial crowdsourcing: A worker decomposition approach. *TKDE*, 32(12):2336–2350, 2019.

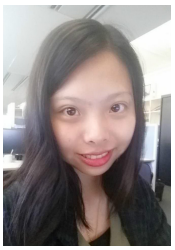
[50] Y. Zhao, K. Zheng, H. Yin, G. Liu, J. Fang, and X. Zhou. Preference-aware task assignment in spatial crowdsourcing: from individuals to groups. *TKDE*, 34(7):3461–3477, 2022.

[51] L. Zheng and L. Chen. Multi-campaign oriented spatial crowdsourcing. *TKDE*, 32(4):700–713, 2020.

Yan Zhao is an Assistant Professor with Aalborg University. She received the Doctoral Degree in Computer Science from Soochow University in 2020. Her research interests include spatial database and trajectory computing.

Tinghao Lai received the bachelor's degree in Computer Science and Technology from East China Jiaotong University, in 2021. He is currently studying for a master's degree in Computer Science and Technology at the University of Electronic Science and Technology of China. His research interests include spatial crowedsourcing.

Ziwei Wang received the bachelor's degree in Computer Science and Technology from Dalian University of Technology, in 2020. He is currently studying for a master's degree in Computer Science and Technology at the University of Electronic Science and Technology of China. His research interests include spatial crowedsourcing.



Kaixuan Chen Kaixuan Chen is an assistant professor of Computer Science at Aalborg University, Denmark. She received her Ph.D. degree from the School of Computer Science and Engineering at the University of New South Wales (UNSW), Australia in 2020. Her current research interests include data mining, machine learning, time-series analysis, and their applications to Internet of Things, human activity recognition, and braincomputer interface.

Huan Li is an EU Marie Curie IF Fellow and Assistant Professor with the Department of Computer Science, Aalborg University, Denmark. He received the PhD degree in computer science from Zhejiang University, China in 2018. His research interests include spatiotemporal data management, mobility analytics, and mobile/pervasive computing. Most of his research work are published in top-tier database venues. He is a member of the IEEE and the ACM.

Kai Zheng is a Professor of Computer Science with University of Electronic Science and Technology of China. He received his PhD degree in Computer Science from The University of Queensland in 2012. He has been working in the area of spatial-temporal databases, uncertain databases, social-media analysis, in-memory computing and blockchain technologies. He has published over 100 papers in prestigious journals and conferences in data management field such as SIGMOD, ICDE, VLDB Journal, ACM Transactions and IEEE Transactions. He is a senior member of IEEE.