functioning of the SC platform and worker participation. However, it is difficult to calculate and predict worker loyalty in SC because of two challenges. First, traditional user/worker loyalty calculation relies on a large amount of business data, and the data of each user is massive and multivariate. But in SC, the data of workers is sparse and univariate, which cannot well reflect workers' characteristics. Second, previous studies calculate worker loyalty only based on workers' existing historical behavioral data and fail to capture the latest worker loyalty dynamically, while in SC, workers' loyalty is fast-changing and depends on workers' earnings and satisfaction with the SC platform.

To address these unmet challenges, this study goes beyond state of the art and develops a spatial crowdsourcing framework called Task Assignment with Worker Loyalty Prediction (TA-WLP), which consists of a worker loyalty prediction and a task assignment phase. In the worker loyalty prediction phase, we regard the historical data of workers and tasks as continuous time series. As a result, the worker loyalty prediction problem can be modeled as a related time series prediction problem. This phase is equipped with an efficient time series prediction algorithm, Prophet, which can handle the case where some outliers exist and can automatically predict the future trend of the time series of workers performing tasks. We then adopt an Entropy Weighting (EW) method to obtain workers' short-term and long-term loyalty scores and combine them to get the latest worker loyalty. In the task assignment phase, we achieve the optimal task assignment based on the Kuhn-Munkras (KM) algorithm. To further improve the computational efficiency, we design a Degree-Reduction-based algorithm, which utilizes a minority first scheme for efficient task assignment.

Our main contributions can be summarized as follows:

1) We identify and study in-depth a new loyalty-based task assignment problem in the context of spatial crowdsourcing by taking workers' loyalty into account.

2) A loyalty prediction calculation method based on Prophet and Entropy Weighting (EW) method is developed to estimate worker loyalty considering workers' short-term and long-term loyalty.

3) We propose two task assignment methods, i.e., the KM-based and Degree-Reduction-based algorithms, to achieve the optimal and efficient task assignment.

4) We report extensive experiments that offer insight into the impact of key parameters and into the effectiveness and efficiency of the proposed techniques.

The remainder of this paper is organized as follows. Section 5 surveys the related work, and Section 2 provides notation and the proposed problem. In Section 3, a brief introduction of the framework overview is given first, followed by a worker loyalty prediction model and two task assignment algorithms. Section 4 gives experimental results, and we conclude the paper in Section 6.

2 PROBLEM DEFINITION

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

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 $s.r$.*

Table 1: Summary of Notations

Notations	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 radius of worker w
A	A spatial task assignment
$A.r$	Total reward in task assignment A
$\mathbb{A}$	Spatial task assignment set

With SC, the query of a spatial task s can be answered only if space task s is within the reachable radius $w.d$ of the worker w Besides, taking into account the expiration time of the spatial task, the spatial task s can be accepted and completed only if the online worker arrives at $s.l$ before its deadline $s.e$. Note that with the single task assignment mode [14], an SC server assigns each spatial task to a 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.*

Definition 3 (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 a spatial task assignment that 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.

Loyalty-based Task Assignment (LTA). Given a set of online workers W and a set of tasks S at the current time instance on an SC platform, our 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 total loyalty score among workers.

3 ALGORITHM

We propose a Task Assignment with Worker Loyalty Prediction (TA-WLP) framework to solve the LTA problem. We first give an overview of the framework and then provide specifics on each component in the framework.

3.1 Framework Overview

The Task Assignment with Worker Loyalty Prediction (TA-WLP) framework (cf. Figure 1) is comprised of two components: worker loyalty prediction and task assignment.

The first component is equipped with a Worker Loyalty Prediction (WLP) model, which aims to predict the loyalty score for each worker based on the worker's historical task-performing data. To this end, we first use the Prophet algorithm to estimate the

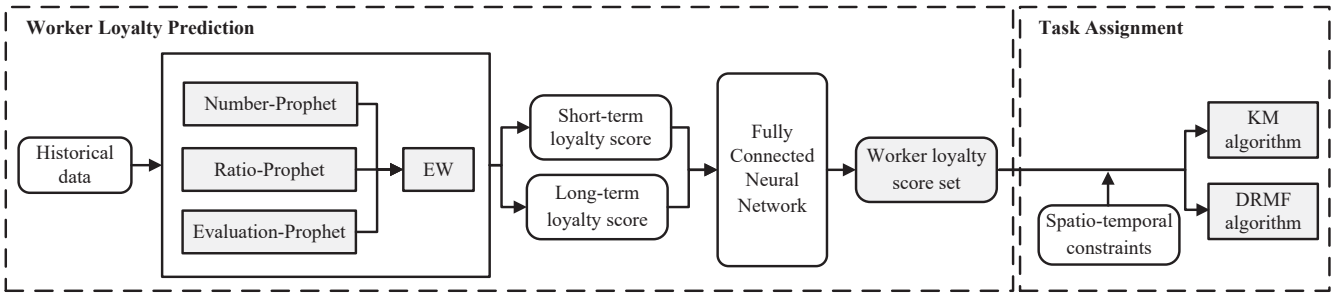


Figure 1: Framework Overview

number of completed tasks, task completion ratio, and task completion evaluation score of each worker. We weight the three metrics by introducing an Entropy Weighting (EW) method to assess the short-term and long-term loyalty scores of workers. Finally, we predict the loyalty score for each worker by a fully connected neural network layer.

In practice, an SC server must take measures to maintain worker loyalty to ensure continuous and high worker participation and satisfaction. The measures include assigning the high-loyalty workers high-value tasks, such as tasks with high rewards or good positions. Therefore, in the task assignment component, considering spatio-temporal constraints, i.e., workers' reachable region and tasks' expiration time, we assign tasks to suitable workers while giving high priority to high-loyalty workers. Specifically, we propose a Kuhn-Munkras (KM) algorithm that tries to achieve the optimal task assignment that maximizes the total reward among workers while prioritizing high-loyalty workers when assigning tasks. We also develop a Degree-Reduction-based algorithm with Minority First scheme (DRMF) to improve the efficiency of task assignment.

3.2 Worker Loyalty Prediction

In this section, we introduce the Worker Loyalty Prediction (WLP) model, which consists of three parts: worker performance prediction based on Three-way Prophet, worker loyalty assessment with Entropy Weighting (EW), and loyalty score prediction, the structure of which is shown in Figure 2.

3.2.1 Worker Performance Prediction based on Three-way Prophet. We introduce the Prophet algorithm for predicting worker performance. In practice, worker loyalty is related to the worker's historical performance, such as the number of completed tasks, task completion ratio, and task completion evaluation score. Therefore, we use three different vectors as the inputs for Prophet, i.e., the vector of the completed task number, the vector of task completion ratio, and the vector of the task completion evaluation score.

For worker w, the vector of the completed task number is denoted as (t_i, f_i), where t_i represents a time interval and takes one day as the unit. Next, f_i represents the number of tasks completed at time interval t_i. The vector of task completion ratio reflects how many tasks were not completed and is represented by (t_i, f_i, n_i), where n_i represents the number of accepted tasks at time interval t_i (i.e., one day) in the historical data, and the remaining two elements (i.e., t_i and f_i) have the same meaning with the corresponding elements in the vector of the completed task number. The vector of task completion evaluation score is received by worker w after completing tasks and is denoted as (t_i, f_i, s_i), where s_i represents the average task evaluation score at time interval t_i in the historical data.

The historical task performance data for workers can be regarded as time series data. In this time-series data, clear seasonal and other cyclical effects exist, such as weekly and yearly cyclical changes. These effects are naturally occurring and can be expected from time series of historical human behavioral data. Considering that Prophet [26] addresses these issues well and has a good performance when dealing with sequential data, we introduce it to our model to predict the number of task completion, the task completion ratio, and the task completion evaluation score.

The Prophet model divides the time series into a superposition of three components. The first component represents the trend function, which is used to represent the non-periodic variation in the data. The second component is used to represent some periodic variations, such as weekly, monthly. The last component represents unpredictable and estimated variations caused by special reasons such as specific holidays, festivals, etc. Finally, a noise term is added, which is used to represent random fluctuations. We divide the historical task performance data of workers into three components: the number of tasks completed, task completion ratio, and task completion evaluation score. Then we implement worker performance prediction using a three-way prophet, as shown in Figure 2.

Next, we illustrate worker performance prediction based on Prophet, which is a decomposable time-series model [10] with three main model components, i.e., trend, seasonality, and holidays. They are combined in Equation 1.

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t, \tag{1}$$

where $g(t)$ is a trend function that fits non-periodic changes in a time series, $s(t)$ represents periodic variation, and $h(t)$ represents the effect of a special event (e.g., holidays). The error term ϵ_t represents any special variation that the established model does not fit.

The calculation of $g(t)$ is shown in the following equations.

$$a_j(t) = \begin{cases} 1 & \text{if } t \geq t_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$g(t) = \left(k + \mathbf{a}(t)^\top \boldsymbol{\delta}\right) t + \left(m + \mathbf{a}(t)^\top \boldsymbol{\gamma}\right), \tag{3}$$

where k is the base growth rate, δ denotes the amount of change in the growth rate, and m is the offset parameter. In a real-time series, the curve's trend does not always remain the same, but at some specific time or with some potential cycles, the curve will change,

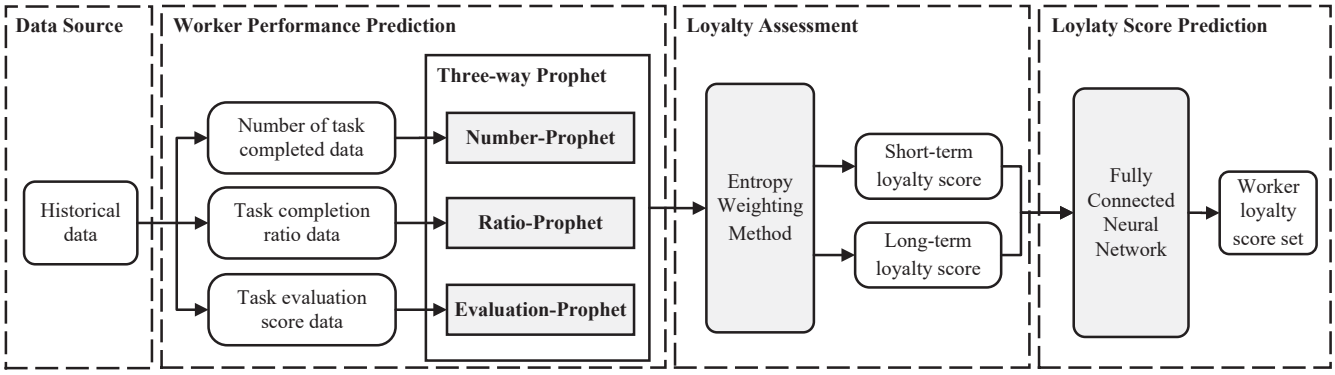


Figure 2: Worker Loyalty Prediction Model

and the model defines the points corresponding to the change in the growth rate k, called change points. When the positions of change points are assumed to be at timestamp $S = \{s_1, s_2, ..., s_{|S|}\}$, γ_j is set to $-s_j\delta_j$ to make the function continuous. The growth rate at any time t is expressed as $k + \sum_{j:t>s_j} \delta_j$. The rate at any time instance t is the sum of the base rate k and all of the adjustments before t, $k + \sum_{s_j<t} \delta_j$, where δ_j denotes the adjustment at time instance s_j. This is represented more clearly by defining a vector $a(t)$, which is shown in Equation 2.

Due to the periodic behavior of humans, time series usually have periodicity. For example, a five-day work week. To accommodate and predict these effects, We have to specify a seasonal model as a periodic function of t, so we can use the Fourier series to generate a suitable periodic model $X(t)$, as shown in Equation 4, and then obtain $s(t)$, as shown in Equation 5.

$$X(t) = \left[\cos\left(\frac{2\pi(1)t}{365.25}\right), \ldots, \sin\left(\frac{2\pi(10)t}{365.25}\right)\right], \tag{4}$$

$$s(t) = X(t)\boldsymbol{\beta}, \tag{5}$$

where $\boldsymbol{\beta} \sim \text{Normal}\left(0, \sigma^2\right)$, imposing a smoothing prior on the seasonality.

Holidays and special events have an enormous impact on many time series and often do not follow a cyclical pattern, so their impact cannot be well modeled with smoothed cycles. The effect of a particular holiday or event on a time series is often repeated, so it is essential to include it in the forecast. Holiday effect model $h(t)$ is calculated in the following equations.

$$Z(t) = \left[\mathbf{1}\left(t \in D_1\right), \ldots, \mathbf{1}\left(t \in D_L\right)\right], \tag{6}$$

$$h(t) = Z(t)\boldsymbol{\kappa}, \tag{7}$$

where D_i records the holidays and significant events that recur in time and assigns a parameter κ_i to each holiday and event. As with seasonality model, we use a prior $\boldsymbol{\kappa} \sim \text{Normal}\left(0, \nu^2\right)$.

As we have discussed above, we combine the three models mentioned as Prophet. With the Three-way Prophet algorithm, we can predict three workers' metrics by three vectors, i.e., a vector of the number of completed tasks, a vector of the task completion ratio, and a vector of task completion evaluation score.

3.2.2 Worker Loyalty Assessment with Entropy Weighting (EW). Traditionally, User/worker loyalty is related to the quality of service received and satisfaction [30, 33, 35]. However, it cannot be well applied in a field like SC since data related to workers is sparse and single. It is difficult to determine whether a worker is loyal or not. In addition, we cannot obtain tagged data in SC. To solve these problems, we transform the worker loyalty problem into a problem based on behavioral performance and introduce an Entropy Weighting (EW) method [23, 28]. Unlike traditional methods of calculating user/worker loyalty, we use EW to calculate worker loyalty in the sparse data of SC. More specifically, given three performance metrics, i.e., the number of completed tasks, task completion ratio, and task completion evaluation score of a worker, EW calculates the weight of each worker to assess the level of loyalty a worker in the whole worker group.

Each worker has a feature set, $P_{w_i} = \{p_{i1}, p_{i2}, p_{i3}\}$, which represent the three performance indexes of the worker, respectively. Next, we introduce the calculation process of the EW method. Firstly, the feature data are normalized to the unit interval using min-max scaling, as shown in Equation 8. Then we calculate the entropy and the weight of each indicator for worker w_i, where the proportion y_{ij} of the j-th indicator p_{ij} of worker w_i is calculated in Equation 9. We continue to calculate the information entropy e_j of p_{ij}. Finally, we can get the weight k_j of each indicator of worker w_i in the final calculation in Equation 11.

$$p'_{ij} = \frac{p_{ij} - \min\left(p_j\right)}{\max\left(p_j\right) - \min\left(p_j\right)} \tag{8}$$

$$y_{ij} = \frac{p'_{ij}}{\sum_{i=1}^{m} p'_{ij}} \tag{9}$$

$$e_j = \frac{1}{\ln m} \sum_{i=1}^{m} y_{ij} \ln y_{ij} \tag{10}$$

$$k_j = \frac{1 - e_j}{\sum_j 1 - e_j} \tag{11}$$

Finally, each worker's loyalty is calculated with weighted sum, where l is the loyalty score of each worker (cf. Equation 12).

$$l = \sum_j 100 y_{ij} k_j \tag{12}$$

The Prophet uses data of different lengths to predict the three performance data of workers, which EW assesses to obtain worker loyalty. For assessing short-term and long-term worker loyalty, we set the size of input data to 90 and 365, which correspond to three months and one year of data, respectively. For predicting short-term loyalty l_s, an input instance consists of 90 task completion number vectors, 90 task completion ratio vectors, and 90 task completion

evaluation score vectors. When predicting long-term loyalty l_l, the number is changed to 365.

3.2.3 Loyalty Score Prediction. As mentioned before, the loyalty scores are related to the three performance data in workers' historical data as well as the short-term and long-term performance of workers. Therefore, EW calculates the data predicted by the Prophet to assess the short-term and long-term loyalty of workers would be concatenated and then passed through a fully connected layer, by which the predicted loyalty scores are output. Finally, the loyalty score tensor LS is obtained. The detailed steps are shown in Equations 13 and 14.

$$L_w = W_s l_s + W_l l_l + b_w, \tag{13}$$

$$LS = [L_1, L_2, L_3, ...], \tag{14}$$

where L_w is the predicted loyalty score of worker w, W_s and W_l are the weight matrix, and b_w is the bias.

3.3 Task Assignment

In the real-time scenario of spatial crowdsourcing problem, the dynamic arrival of workers and tasks requires the immediate response and assignment of SC servers, and it is challenging to achieve a global optimal solution to the worker loyalty-based task assignment (LTA) problem. To improve worker loyalty and worker performance, we will optimize worker task assignment at each instance by maximizing the current assigned total reward and worker loyalty score and giving higher priorities to loyal workers. We propose two task assignment algorithms: a Loyalty-aware KM Algorithm and a Loyalty-aware Degree-Reduction-based algorithm with Minority First scheme.

3.3.1 Loyalty-aware KM Algorithm. Taking workers' loyalty scores as the priority of task assignment, we transform the LTA problem into a Bipartite Maximum Weight Matching problem and apply the KM algorithm to solve it.

For an undirected bipartite graph, it is represented by $G = (V, E)$, where V corresponds to the vertex set and E corresponds to the edge set. Given a set of online workers $W = \{w_1, w_2, ..., w_{|W|}\}$ and a set of currently online unassigned tasks $S = \{s_1, s_2, ..., s_{|S|}\}$, the number of V is equal to the sum of the numbers of W and S. The assignable task of worker w_i is represented by $AS(w_i)$ and is positively related to the predicted loyalty score of w_i, where $AS(w)$ should satisfy two conditions: $\forall s \in AS(w), w \in W$, the distance between worker w and task s less than the worker's reachable radius w_d, and the worker w can reach the task location and complete the task before the deadline s_e of the task s.

Since our goal is to maximize worker loyalty scores and total task reward. If space task s_j can be assigned to worker w_i , i.e., $s_j \in AS(w_i)$. Then for each edge (v_i^W, v_j^S), its weight (denoted by $weight(v_i^W, v_j^S)$) can be measured as worker loyalty score $S[i]$ and the reward of the spatial task s_j, i.e., $weight(v_i^W, v_j^S) = w_s * S[i] + w_r * s_j.r$, where w_s is the weight of the loyalty score, w_r is the weight of the reward s_j .

Before introducing the Loyalty-aware KM algorithm, we first introduce the FindTask algorithm. The FindTask algorithm is a traversal algorithm used to find suitable tasks for workers. In the algorithm, we compute the difference between the weight of the edge associated with two vertices and the expected sum of workers and tasks. If the difference is equal to 0, it means that the current task assigned to this worker is the best choice (line 5–6). If the task has been assigned to another worker, we try to assign another task to that worker (lines 7–9).

Algorithm 1: FindTask Algorithm

```
   Input: worker w
   Output: Bool
1  S_worker[w] = True;
2  for each task s is adjacent to w in G do
3      if S_task[s] then
4          continue;
5      tmp ← ex_worker[w] + ex_task[s] − weight(v_w^W, v_s^S);
6      if tmp = 0 then
7          if A[s]=-1 or FindTask(A[s]) then
8              A[s]=w;
9              return True;
10     else
11         slack[s] = min(slack[s], tmp);
12     return False;
```

Algorithm 2: Loyalty-aware KM Algorithm

```
   Input: G
   Output: A
1  A ← [−1, −1, ...];
2  T_task ← [0, 0, ...];
3  slack ← [INF, INF, ...];
4  for each worker w ∈ W do
5      T_worker[w] ← max(weight(v_w^W, v_s^S));
6  for each worker w ∈ W do
7      Set both left_task and right_worker to False;
8      if FindTask(w) then
9          break;
10     else
11         d=INF;
12         for each task s ∈ S do
13             if ! left_task[s] then
14                 d=min(d,slack[s]);
15         for each worker w ∈ W do
16             if right_worker[w] then
17                 T_worker[w]− = d;
18         for each task s ∈ S do
19             if left_task[s] then
20                 T_task[s]+ = d;
21             else
22                 slack[s]− = d;
23 return A;
```

The KM task assignment algorithm is shown in Algorithm 2, which is used to determine the complete match with the largest sum of weights of the bipartite graph, that is, the best match. For

a given weighted bipartite graph G, it consists of two vertex sets, V_S and V_W. First, for each vertex in V_W, its expectation is equal to the maximum weight among the edges associated with it in the graph G (lines 4–5). Second, Algorithm 2 performs a traversal search for worker w through the FindTask function (line 8) to find a current task match. Third, if w fails to match a task, we adjust the expectations of workers and the last matched tasks and adjust the expected relationship between workers and tasks, so that workers and tasks have more choices (lines 10–22). Finally get the total task allocation A (line 23).

3.3.2 Loyalty-aware Degree-Reduction-based Algorithm With Minority First Scheme (DRMF) . Taking workers' loyalty scores and tasks' rewards as the priority of task assignment, we propose a Degree-Reduction-based algorithm with Minority First scheme (DRMF) to solve it.

We first transform the task assignment problem into a directed weighted bipartite graph. The graph 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, $S = \{s_1, s_2, ..., s_{|S|}\}$ at the current time, the available workers for spatial task s ($s \in S$), denoted as $AW(s)$, and the available tasks for worker w ($w \in W$), denoted as $AS(w)$. The other settings are the same as mentioned in Section 3.3.1. Specifically, in our directed weighted bipartite graph, for each edge (v_i^W, v_j^S), its weight (denoted by $weight(v_i^W, v_j^S)$) with different directions correspond to different weights. For each spatial task $s_j \in S$, the edge from v_j^S mapped from $s_j \in S$ to the vertex v_i^W mapped from $w_i \in W$, its weight can be measured as the loyalty score of the worker w_i. For each worker $w_i \in W$, the edge from v_i^W mapped from $w_i \in W$ to the vertex v_j^S mapped from $s_j \in S$, its weight can be measured as the reward of the spatial task s_j.

Algorithm 3: Degree-Reduction-based Greedy Algorithm

Input: W, AS, AW
Output: A
1 $A \leftarrow \emptyset$;
2 **while** $|W| > 0$ **do**
3 **for** *each worker* $w \in W$ **do**
4 calculate the degree $C(w)$ of w;
5 **if** ! $C(w)$ **then**
6 remove w from W;
7 $w_{tmp} \leftarrow min(C(u))$;
8 $\tilde{AS}(w_{tmp}) \leftarrow$ Sort $AS(w_{tmp})$ in descending order of $reward$;
9 $t_{tmp} = \tilde{AS}(w_{tmp})[0]$
10 $A \cup [(w_{tmp}, t_{tmp})]$;
11 $w_{list} = AW(t_{tmp})$
12 **for** *each worker* $w \in w_{list}$ **do**
13 **if** $t_{tmp} \in AS(w)$ **then**
14 remove t_{tmp} from $AS(w)$
15 clear $AS(w_{tmp})$
16 **return** A;

Before introducing the DRMF algorithm, we will first introduce the Degree-Reduction-based (DR) greedy algorithm. It is a greedy algorithm based on a bipartite graph. The greedy algorithm is to utilize a degree reduction strategy. It only allocates to one of the vertex sets of the bipartite graph.

The DR algorithm is shown in Algorithm 3. We introduce the example of priority allocation to workers. Before W is empty, we calculate the degree of each node $w \in W$, denoted as $C(w)$, and if $C(w_i)$ is zero, remove w_i from W, to reduces the search time of our algorithm and ensure that all the nodes w_i we find have edges connected to them (lines 3–6). Finding the smallest $|C(w_i)|$, the node v_i^W is mapped from $w_i \in W$. Those task nodes connected with $v_i^W \in W$ are $AS(v_i^W)$ (line 7). If $AS(v_i^W)$ is not empty, we can find the task node v_j^S mapped from $s_j \in S$ with the largest weight, and thus (v_i^W, v_j^S) is a worker-task matching (lines 8–10). Then we update $AS(v_j^S)$, where $u_i \in AW(v_j^S)$, and delete node v_j^S in $AS(u_i)$ to indicate that task v_j^S has been assigned. Such a result ensures that all node degrees in $AS(v_i^W)$ will be reduced by 1. Finally we clear $AS(v_i^W)$ (lines 11–15).

Algorithm 4: Degree-Reduction-based Algorithm With Minority First Scheme

Input: W, S
Output: A
1 $A \leftarrow \emptyset$;
2 **for** *each worker* $w \in W$ **do**
3 Predict the loyal score of w;
4 **for** *each task* $s \in S$ **do**
5 $AW(s) \leftarrow$ Find the set of available workers of s;
6 **for** *each worker* $w \in W$ **do**
7 $AS(w) \leftarrow$ Find the set of available tasks of w;
8 **if** $|W| > |S|$ **then**
9 A = DR(S, AS, AW)
10 **else**
11 A = DR(W, AS, AW)
12 **return** A;

The Degree-Reduction-based algorithm with Minority First scheme is shown in Algorithm 4. Given a bipartite graph G, which is composed of two vertices sets V_S and V_W. We first determine the executable tasks for each worker and the available workers for each task (lines 4–7). Then we determine whether there are more workers or more tasks and prioritize the allocation of the few, trying to have as many worker-task matches as possible at the end of the allocation, by using the DR algorithm (lines 8–11).

4 EXPERIMENTAL EVALUATION

4.1 Data Preparation

We use the check-in dataset from Yelp to simulate the problem we want to solve, which is a common choice and practice in studying SC platform problems [2, 5, 7, 14, 36]. We simply filter the data, here we select the user data with the total number of comments more than 20 and the number of comments more than 10 before 2019-09-15 23:23:59 in the entire dataset so that the experimental data we get will be more real. For the filtered dataset, it includes

156,483 POIs and 16,262 users. In our experiments, we assume that all tasks are assigned to online workers at some time period in the near future, i.e. 2019-12-15 23:23:59. We assume that the users are workers in the SC system, and their latest location is the location of the most recent check-in point, and each POI is assumed to be a task to be assigned in the SC system. 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. For each worker (i.e., each user), we use the average rating stars of other workers when a worker checks in a POI (i.e., a task) as the task completion ratio of the worker. We use the linear weighted sum of the amount of change in a worker's follower count and the total follower count to denote the worker's true loyalty score. The distance is calculated by the Euclidian distance. All the algorithms are implemented on an Intel (R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz, and NVidia GeForce RTX 1080Ti GPU.

4.2 Experiment Results

4.2.1 Performance of Worker Loyalty Prediction. We first evaluate the performance of worker loyalty prediction.

Evaluation Methods. Seven baseline algorithms are introduced to compare with our Worker Loyalty Prediction (WLP) model.

1) LR: A Linear Regression algorithm [9], which is a statistical analysis method that uses regression analysis in mathematical statistics to determine the quantitative relationship between two or more variables. The expression is: $y = w_1x_1 + w_2x_2 + w_3x_3 + b$, where x_1, x_2, x_3 are historical data (i.e., the number of completed tasks, task completion ratio, and task completion evaluation score) and y represents the worker loyalty score predicted in our problem.

2) k-means: A k-means algorithm [1] that is a division-based clustering algorithm constructing k divisional clusters based on a given data set with n data objects. We set the value of k to 5, which denotes 5 levels of worker loyalty, respectively, and input the historical task performance data of workers into K-means to get 5 clusters, where the data similarity of workers in the same cluster is higher, which means the workers' loyalty is similar.

3) WLP-S: A variant of WLP, which removes the short-term loyalty and fully connected layer component.

4) WLP-L: A variant of WLP, which removes the long-term loyalty and fully connected layer component.

5) WLP-E: A variant of WLP, which removes the Evaluation-Prophet when predicting worker performance.

6) WLP-R: A variant of WLP, which removes the Ratio-Prophet when predicting worker performance.

7) WLP-N: A variant of WLP, which removes the Number-Prophet when predicting worker performance.

Metrics. To evaluate the accuracy of worker loyalty prediction, we adopt a standard evaluation metric, Mean Square Error (MSE). We set the number of workers to 2000, 4000, 6000, 8000, and 10000 for testing. For simplicity, we assume that all the workers share the same speed.

Results. We report the MSE values of the methods in Table 2. Regardless of the number of workers, we observe that the WLP model always outperforms the two baselines and several of its variants, compared with k-means, WLP outperforms k-means by 85%–102%, thus showing the benefits of predicting three performance data for workers and assessing short-term and long-term loyalty, and demonstrating the superiority of WLP in predicting worker loyalty.

Table 2: Accuracy of Worker Loyalty Prediction

Methods	Number of workers				
	2000	4000	6000	8000	10000
LR	4.56	4.43	4.43	4.37	4.22
K-means	6.31	6.17	6.26	6.09	5.91
WLP-S	3.19	3.23	3.20	3.18	3.20
WLP-L	3.12	3.21	3.13	3.15	3.18
WLP-E	3.27	3.32	3.29	3.30	3.36
WLP-R	3.17	3.25	3.23	3.22	3.28
WLP-N	3.38	3.47	3.47	3.44	3.51
WLP	**3.11**	**3.17**	**3.09**	**3.11**	**3.15**

4.2.2 Performance of Task Assignment. In this set of experiments, we evaluate the performance of task assignment.

Evaluation Methods. We study the following algorithms.

1) KM: The KM task assignment algorithm that does not consider worker loyalty.

2) DRMF: The Degree-Reduction-based algorithm with Minority First scheme that does not consider worker loyalty.

3) KM+WL: The KM algorithm based on the worker loyalty predicted by WLP.

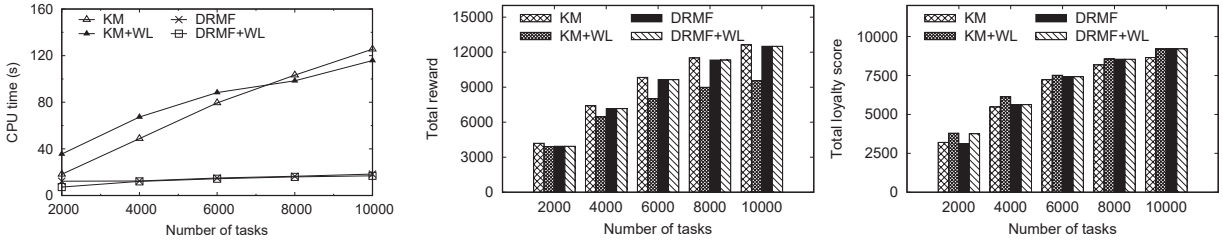
4) DRMF+WL: The DRMF algorithm based on the worker loyalty predicted by WLP.

Metrics. Three metrics are compared among the methods: i.e., *CPU Time* for finding the task assignment, *Total Reward*, and *Total Loyalty Score* of workers. Table 3 shows our experimental settings, where default values are underlined.

Table 3: Experimental Parameters

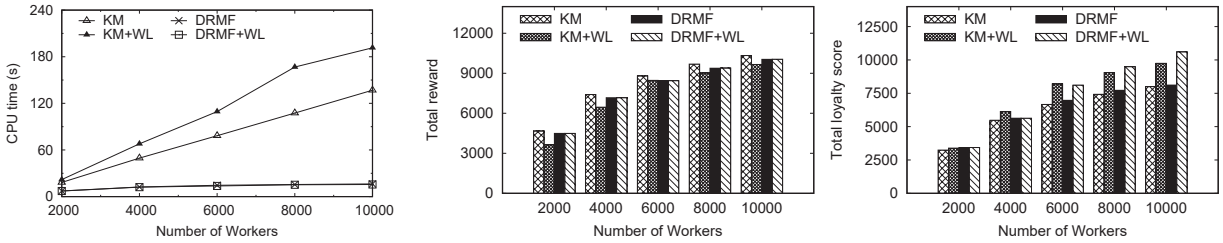
Parameters	Values
Number of tasks $\|S\|$	2000, <u>4000</u>, 6000, 8000, 10000
Number of workers $\|W\|$	2000, <u>4000</u>, 6000, 8000,10000
Reachable distance of workers d (km)	0.05, 0.1, <u>0.5</u>, 1, 5
Valid time of tasks $e - p$ (h)	0.05, 0.1, <u>0.3</u>, 0.5, 0.7

Effect of $|S|$. To study the scalability of all algorithms, we generate five datasets containing 2000 to 10000 tasks by random selection from the Yelp dataset. As shown in Figure 3(a), the CPU time of all methods increases when $|S|$ grows. The methods associated with DRMF run faster than those associated with KM. In addition, the CPU time of DRMF and DRMF+WL increases slowly when $|S|$ grows, which shows the adaptability and good scalability of DRMF-related methods in scenarios with different data volumes. Figure 3(b) shows the total rewards of all methods, which increase with the increasing $|S|$. We observe that the total rewards of DRMF and DRMF+WL are similar to that of KM, which further reflects the reliability of DRMF-related algorithms. DRMF and DRMF+WL are neck-and-neck when $|S|$ is large (i.e., $|S| > 4000$) due to the fact that the number of tasks is more than the number of workers, DRMF and DRMF+WL use the same strategy, i.e., assigning tasks to workers. KM+WL performs worse than others in terms of the total reward since KM+WL gives priority to high-loyalty workers when assigning high-reward tasks. At the same time, as depicted in Figure 3(c), KM+WL and DRMF+WL that consider loyalty scores of workers have better performance than their counterparts (i.e., KM and DRMF) in terms of the total loyalty score. As the number of tasks increases, more workers can be assigned to suitable tasks, so the total loyalty score of all algorithms increases. We should



(a) CPU Time (b) Total Reward (c) Total Loyalty Score

Figure 3: Performance of Task Assignment: Effect of $|S|$



(a) CPU Time (b) Total Reward (c) Total Loyalty Score

Figure 4: Performance of Task Assignment: Effect of $|W|$

also note that when $|S|$ is large (i.e., $|S| > 4000$), the DRMF+WL and DRMF algorithms use the same assignment strategy since the number of tasks is more than the number of workers, so they have the same trend.

Effect of $|W|$. To study the effect of $|W|$, we generate five datasets containing 2000 to 10000 workers by random selection from the testing set. As depicted in Figures 4(a) and 4(b), as $|W|$ increases, more tasks can be assigned to workers, and more workers can get tasks, so the total rewards increase. Although the total rewards of the DRMF and DRMF+WL algorithms are close to those of the KM-related algorithms (including KM and KM+WL), they run much faster than them. The CPU time of DRMF and DRMF+WL is only 11.28%–34.12% of that of KM and only 6.26%–31.09% of that of KM+WL. It proves the efficiency of the DRMF-related algorithms. Meanwhile, as shown in Figure 4(c), KM+WL and DRMF+WL have higher total loyalty scores than the other methods. In both DFMF+WL and KM+WL, as $|W|$ increases, more workers are involved in the assignment, and thus more workers with high-loyalty scores get the task. It is worth mentioning that when $|W|$ is small (i.e., $|W| < 4000$), DRMF and DRMF+WL use the same allocation strategy for the same reason as $|S|$ is large, namely that the number of workers is less than the number of tasks, assign workers to tasks with high rewards, as a result of which both algorithms achieve the same result.

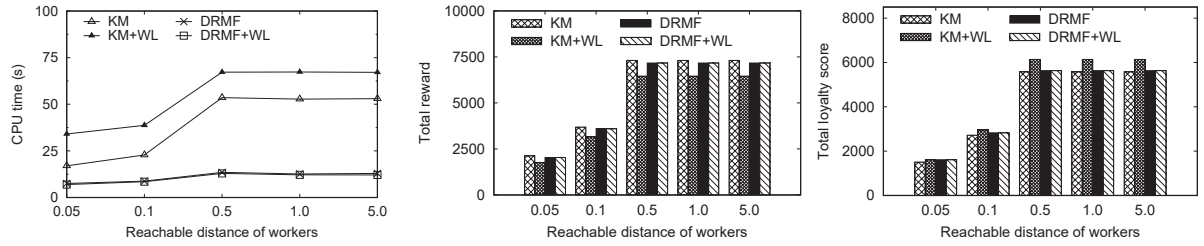
Effect of d. We also study the effect of workers' reachable distances d. From Figure 5(a), we can see that the CPU time of all methods increases when $d \leq 0.5km$ KM and KM+WL are more time-consuming than DRMF and DRMF+WL. This is because the number of reachable tasks per worker becomes larger and the number of available workers per task also gets larger when d grows, which leads to more edges in the graphs of KM and KM+WL, and the competition among workers is more intense causing more searching time. With d increases, the total loyalty scores and total rewards of all methods increase, which is shown in Figures 5(b) and 5(c). However, limited by the number of tasks and workers, the impact of d on the algorithms gradually becomes stable as d increases. As shown in Figure 5, all the four methods remain unchanged after d exceeds $0.5km$.

Effect of $e - p$. Finally, we study the effect of the valid time $e - p$ of tasks. In Figure 6, when $e - p$ increases, there are more eligible workers and tasks at the same time, a greater probability of a worker being assigned to a suitable task, and a greater probability of a task matching an appropriate worker, which increase both total rewards and total loyalty scores. As shown in Figure 6(a), the CPU time of all methods increases when $e - p \leq 0.5h$. The CPU time of KM and KM+WL algorithms increase faster, with a similar reason for the effect of d. Limited by $|S|$ and $|W|$, all the four methods remain unchanged after $e - p$ exceeds $0.5h$.

5 RELATED WORK

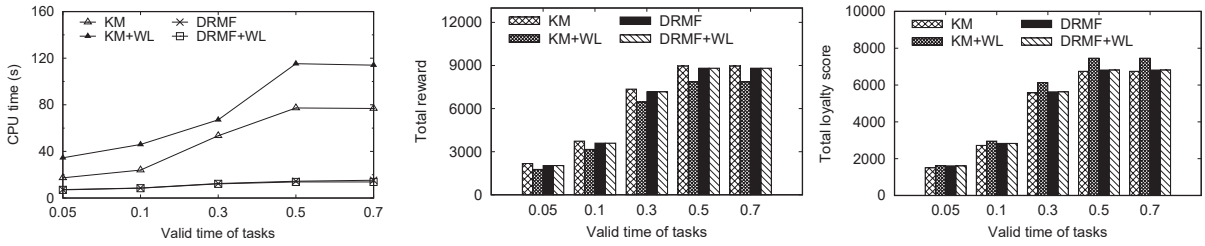
5.1 Task Assignment in Spatial Crowdsourcing

Spatial Crowdsourcing (SC) can be deemed as one of the main enablers to complete location-based tasks [3, 4, 17, 21, 24, 25, 37, 42]. Two task distribution modes are defined in SC, namely *worker selected tasks* (WST) [14] and *server assigned tasks* (SAT). In the WST mode, the platform server publishes spatially aware tasks, and online workers can select any nearby spatial task without negotiating with the server [2, 8, 14, 15]. In the SAT mode, the platform server does not post spatial tasks directly to workers. Instead, any online worker sends a location to the server. After



(a) CPU Time (b) Total Reward (c) Total Loyalty Score

Figure 5: Performance of Task Assignment: Effect of d



(a) CPU Time (b) Total Reward (c) Total Loyalty Score

Figure 6: Performance of Task Assignment: Effect of $e - p$

receiving the locations of all online workers, the server assigns a suitable task to each worker.

Zhao et al. [40] focus on the fairness-aware task assignment problem in SC, and design two algorithms to solve the problem, namely, a fairness-aware game theory algorithm and an improved evolutionary game theory algorithm, their first goal is to minimize the difference in earnings between workers, while the second objective is to maximize average worker compensation. However, the studies mentioned above mainly focus on the Spatio-temporal availability of workers and tasks. They do not consider worker loyalty, which reflects the productivity and reliability of workers with SC service providers.

5.2 User/Worker Loyalty Prediction

User/worker loyalty analysis has long been an essential part of business and a popular research topic in academia. Traditionally, user loyalty is a multidimensional concept involving behavioral and attitudinal approaches. It refers to how a user becomes attached to a product or service, develops a preference and dependence, and repeatedly buys that product or service over time. Recent studies have made many efforts to analyze and calculate user loyalty. For example, Zhang et al. [35] introduce a machine learning method to reveal the relationship between users' loyalty behavior and user relationship management by which the phenomenon of user churn is improved. Wu et al. [30] develop a Partial Least Squares Structural Equation Model (PLS-SEM). The PLS-SEM model can also analyze the role of each data indicator and the internal relationship between each indicator. Therefore, it provides an accurate reference for customer loyalty metrics.

However, due to the sparsity of data, and the lack of explicit definitions and criteria for worker loyalty in SC applications, the above methods that only summarize the personal historical behavior cannot represent the future loyalty of workers and thus cannot directly applied to predict the loyalty of workers in SC. In this paper, combining the historical behavioral data of workers, we propose a worker loyalty prediction model, which is tailor-made to predict workers' loyalty by considering the short-term and long-term behavior of workers as well as Spatio-temporal characteristics.

6 CONCLUSION

The prevalence of mobile devices with high-accuracy positioning and the dramatic decrease in the cost of wireless network usage have led to the rapid growth and massive demand for Spatial Crowdsourcing (SC) market, which requires workers to complete spatial tasks at specified locations within the valid time of tasks. In this paper, we investigate a new task assignment problem in SC, namely Loyalty-based Task Assignment (LTA). We address several challenges by proposing different strategies to assess and predict worker loyalty. In task assignment, we prioritize the assignment of high-loyalty workers so that they can gain more benefits and be satisfied with the SC platform. As far as we know, we are the first to consider worker loyalty in SC. Extensive experiments on real data demonstrate the effectiveness and efficiency of our proposed solutions.

ACKNOWLEDGMENTS

This work is partially supported by NSFC (No. 61972069, 61836007 and 61832017), and Shenzhen Municipal Science and Technology RD Funding Basic Research Program (JCYJ20210324133607021).

REFERENCES

[1] Dahlan Abdullah, S Susilo, Ansari Saleh Ahmar, R Rusli, and Rahmat Hidayat. 2021. The application of K-means clustering for province clustering in Indonesia of the risk of the COVID-19 pandemic based on COVID-19 data. *QUAL QUANT* (2021), 1–9.

[2] Peng Cheng, Xiang Lian, Lei Chen, and Cyrus Shahabi. 2017. Prediction-Based Task Assignment in Spatial Crowdsourcing. In *ICDE*. 997–1008.

[3] Yurong Cheng, Boyang Li, Xiangmin Zhou, Ye Yuan, Guoren Wang, and Lei Chen. 2020. Real-time cross online matching in spatial crowdsourcing. In *ICDE*. 1–12.

[4] Yue Cui, Liwei Deng, Yan Zhao, Bin Yao, Vincent W Zheng, and Kai Zheng. 2019. Hidden poi ranking with spatial crowdsourcing. In *KDD*. 814–824.

[5] Hung Vi Dang, Tuan Anh Nguyen, and Hien To. 2013. Maximum Complex Task Assignment: Towards Tasks Correlation in Spatial Crowdsourcing. In *IIWAS '13*.

[6] Didit Darmawan, Rahayu Mardikaningsih, Ella Anastasya Sinambela, Samsul Arifin, Arif Rachman Putra, Mila Hariani, Yusuf Rahman Al Hakim, and Mochamad Irfan. 2020. The Quality of Human Resources, Job Performance and Employee Loyalty. *Int. J. Psychosoc* 24, 3 (2020), 2580–2592.

[7] Dingxiong Deng, Cyrus Shahabi, and Ugur Demiryurek. 2013. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing. In *SIGSPATIAL*. 314–323.

[8] Dingxiong Deng, Cyrus Shahabi, and Linhong Zhu. 2015. Task matching and scheduling for multiple workers in spatial crowdsourcing. In *SIGSPATIAL*. 1–10.

[9] J. Brian Gray. 2002. Introduction to Linear Regression Analysis. *Technometrics* 44, 2 (2002), 191–192.

[10] Andrew C Harvey and Simon Peters. 1990. Estimation procedures for structural time series models. *J FORECASTING* 9, 2 (1990), 89–108.

[11] Tahir Iqbal. 2020. An Assessment of the Impact that Service Quality and Customer Satisfaction Possess on Customer Loyalty in Internet Banking. *IJOM* 10, 1 (2020), 58–71.

[12] Rui Jin and Kai Chen. 2020. Impact of value cocreation on customer satisfaction and loyalty of online car-hailing services. *J. Theor* 16, 3 (2020), 432–444.

[13] Xiaopu Jin and Fang Xu. 2020. Examining the factors influencing user satisfaction and loyalty on paid knowledge platforms. *ASLIB J INFORM MANAG* (2020).

[14] Leyla Kazemi and Cyrus Shahabi. 2012. GeoCrowd:enabling query answering with spatial crowdsourcing. In *SIGSPATIAL*. 189–198.

[15] Leyla Kazemi, Cyrus Shahabi, and Lei Chen. 2013. GeoTruCrowd:trustworthy query answering with spatial crowdsourcing. In *SIGSPATIAL*. 314–323.

[16] Songhua Li, Leqian Zheng, and Victor Lee. 2020. Car-Sharing: Online Scheduling k Cars Between Two Locations. In *FAW*. 49–61.

[17] Xiang Li, Yan Zhao, Xiaofang Zhou, and Kai Zheng. 2020. Consensus-based group task assignment with social impact in spatial crowdsourcing. *DSE* 5, 4 (2020), 375–390.

[18] 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.

[19] Hui Lin, Sahil Garg, Jia Hu, Georges Kaddoum, Min Peng, and M. Shamim Hossain. 2021. Blockchain and Deep Reinforcement Learning Empowered Spatial Crowdsourcing in Software-Defined Internet of Vehicles. *T-ITS* 22, 6 (2021), 3755–3764.

[20] Fandy Nugroho Hardiknasiono Musa, Altje Tumbel, and Magdalena Wullur. 2021. Discipline Analysis Of Work, Motivation And Loyalty Towards Employee Performance (Case Study At Gorontalo State University). *Aksara: Jurnal Ilmu Pendidikan Nonformal* 7, 2 (2021), 449–462.

[21] Wangze Ni, Peng Cheng, Lei Chen, and Xuemin Lin. 2020. Task allocation in dependency-aware spatial crowdsourcing. In *ICDE*. 985–996.

[22] Yilei Pei, Wanxin Xue, Yong Yang, Dandan Li, and Yi Li. 2019. The Impacts of user experience on user loyalty based on O2O innovation platform. *JECO* 17, 2 (2019), 79–87.

[23] Qiao Shi, Honglv Wang, and Hailong Lu. 2020. Research and Application of AHP-EWM-based Comprehensive Evaluation of Data Quality. In *CSAE*. 152:1–152:5.

[24] Tianshu Song, Ke Xu, Jiangneng Li, Yiming Li, and Yongxin Tong. 2020. Multi-skill aware task assignment in real-time spatial crowdsourcing. *GeoInformatica* 24, 1 (2020), 153–173.

[25] Qian Tao, Yongxin Tong, Zimu Zhou, Yexuan Shi, Lei Chen, and Ke Xu. 2020. Differentially private online task assignment in spatial crowdsourcing: A tree-based approach. In *ICDE*. 517–528.

[26] Sean J. Taylor and Benjamin Letham. 2018. Forecasting at Scale. *Am Stat* 72, 1 (2018), 37–45.

[27] Yongxin Tong, Yuxiang Zeng, Bolin Ding, Libin Wang, and Lei Chen. 2021. Two-Sided Online Micro-Task Assignment in Spatial Crowdsourcing. *TKDE* 33, 5 (2021), 2295–2309.

[28] Wanyu Wang and Qiang Li. 2020. Research on the Impact Indicator System of E-Commerce Innovation Experimental Teaching: Deep Fusion Evaluation Method Based on EWM-AHP. In *IC4E*. 290–295.

[29] Ziwei Wang, Yan Zhao, Xuanhao Chen, and Kai Zheng. 2021. Task Assignment with Worker Churn Prediction in Spatial Crowdsourcing. In *CIKM*. 2070–2079.

[30] Kaibiao Wu, Shanshan Chen, and Yun Yuan. 2018. Research on the customer loyalty of bicycle-sharing company based on PLS-SEM model. In *ICMSS*. 68–72.

[31] Ya-Ling Wu and Kai-Hsien Chen. 2021. Exploring the Determinants of Customer Loyalty to Internet-Only Bank Services. In *AMCIS*.

[32] 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.

[33] Kai-Fu Yang, Yu-Ching Chiang, and Yi-Shen Lin. 2018. A Study on Service Quality, Customer Satisfaction, and Customer Loyalty: The Case of PChome. In *ICSET*. 88–93.

[34] Guanyu Ye, Yan Zhao, Xuanhao Chen, and Kai Zheng. 2021. Task Allocation with Geographic Partition in Spatial Crowdsourcing. In *CIKM*. 2404–2413.

[35] Sheng Zhang, Xueliang Tan, Jiawen Wang, Jianghang Chen, and Xinjun Lai. 2019. Modeling Customers' Loyalty Using Ten Years' Automobile Repair and Maintenance Data: Machine Learning Approaches. In *DSIT*. 242–248.

[36] Yan Zhao, Xuanhao Chen, Liwei Deng, Tung Kieu, Chenjuan Guo, Bin Yang, Kai Zheng, and Christian S Jensen. 2022. Outlier detection for streaming task assignment in crowdsourcing. In *WWW*. 1933–1943.

[37] 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.

[38] Yan Zhao, Yang Li, Yu Wang, Han Su, and Kai Zheng. 2017. Destination-aware Task Assignment in Spatial Crowdsourcing. In *CIKM*. 297–306.

[39] 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.

[40] Yan Zhao, Kai Zheng, Jiannan Guo, Bin Yang, Torben Bach Pedersen, and Christian S Jensen. 2021. Fairness-aware Task Assignment in Spatial Crowdsourcing: Game-Theoretic Approaches. In *ICDE*. 265–276.

[41] 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* (2019), 2336–2350.

[42] 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* (2020).