Task Recommendation in Spatial Crowdsourcing: A Trade-off between Diversity and Coverage

Liwei Deng[1], Yan Zhao[2,✉], Yue Cui[3], Yuyang Xia[1], Jin Chen[1], Kai Zheng[1,✉]
[1]University of Electronic Science and Technology of China, China [2]Aalborg University, Denmark
[3] The Hong Kong University of Science and Technology, Hong Kong SAR, China
{deng_liwei, xiayuyang, chenjin}@std.uestc.edu.cn, yanz@cs.aau.dk, ycuias@cse.ust.hk, zhengkai@uestc.edu.cn

***Abstract*—The popularity of mobile devices has led to the increased attention of Spatial Crowdsourcing (SC), a framework that assigns location-sensitive tasks to mobile workers. Task recommendation is crucial in helping workers discover attractive tasks. Existing studies have focused on modeling workers' preferences from past task-performing patterns, but their performance is sub-optimal due to the strong coupling of sequentiality, spatiality, and temporality. Moreover, achieving the highest preference-based utility of workers in most of the existing task recommendation studies is inferior to the benefits of the SC platform and the satisfaction of workers in a long range, due to the lower task coverage rate and the poor diversity in a worker's recommended list. To address these problems, we propose a Diversity-Coverage Balanced Task Recommendation ($DCBTaskRec$) framework. Specifically, we first introduce a decoupled worker preference learning model that adopts self-attention networks as the backbone and decouples the modeling of multiple factors in attention scores. Additionally, we provide an optimal diveristy-aware approach to maximize the recommendation diversity while keeping high preference-based utility of workers to satisfy the multiple tastes of workers. From the side of the SC platform, we also provide two approaches (i.e., greedy coverage-aware approach and diversity-coverage balanced approach) to achieve high coverage and provide a trade-off between diversity and coverage, respectively. Extensive experiments offer insight into the effectiveness of the proposed framework.**

***Index Terms*—Spatial Crowdsourcing, Diversity, Coverage, Task Recommendation**

I. Introduction

With the increased popularity of GPS-enable smart devices and the accompanying deployment of sensing technologies, Spatial Crowdsourcing (SC) has attracted great attention from both academia and industry, where task requesters can issue spatial tasks (e.g., taking a scenic photo or reporting a hot spot) to the SC platform and a crowd of workers are recruited to perform by physically moving to the specified locations. Different task assignments may induce different behaviors of workers. For example, a worker will be passive in completing the assigned task when the task does not match the worker's preference, resulting in a degradation in the quality of the task assignment. Thus, to achieve high-quality SC services, an SC platform should model the workers' preferences and recommend a set of suitable tasks to workers. Compared with mandatory task assignment, i.e., assigning a task to each worker at a time and the worker is forced to perform, a task recommendation system provides a list of available tasks, in which the worker can select his/her most preferred one from the recommended list.

✉ The corresponding author, Kai Zheng, is with Yangtze Delta Region Institute(Quzhou), University of Electronic Science and Technology of China

Existing studies [1]–[5] on the task recommendation in SC are proposed for different scenario applications. For example, Alamer et al. design a privacy-preserving location matching mechanism with the aims of secure task recommendation and protecting location privacy for workers, in which the preference modeling is ignored (i.e., a worker equally prefers the tasks in their reachable task set) [2]. Li et al. propose a preference-aware group task assignment framework to maximize the overall number of assigned tasks while giving priority to the groups of workers that are more interested in the tasks [5]. Zhao et al. propose an adaptive task recommendation framework, focusing on the workers' preference modeling in both hometown and out-of-town areas and the fairness in task recommendation phase [1]. In spite of the exploration of different scenario applications, we still face three main challenges to achieving effective task recommendations.

Challenge I: How to effectively model workers' preference for spatial tasks? Most SC studies focus on modeling workers' preference based on three key aspects: sequentiality [5], spatiality [1], and temporality [6]. These aspects consider the factors that influence human behavior patterns [7]–[9]. For instance, MobTCast [10] concatenates the spatial and temporal embedding with task embeddings. STiSAN [11] incorporates sequential information and spatio-temporal intervals by adding task embedding and attention scores, respectively. However, we argue that such treatments may lead to inaccurate modeling of workers' preferences due to simple and arbitrary coupling of the factors.

Challenge II: How to take diversity into consideration to meet the multi-interests of workers? Traditional recommendation methods focus solely on maximizing preference-based utility of workers by suggesting their top k tasks of interest. However, this approach often leads to the recommendation of highly similar tasks, negatively impacting the overall user experience due to the lack of consideration for task diversity. For instance, consider a scenario where a worker's preferences lie in both taking pictures and writing food reviews, but his/her current behavior predominantly involves taking pictures. In such a case, a Spatial Crowdsourcing (SC) platform that maximizes the workers' preference might generate a task list exclusively comprised of various picture-taking tasks from different locations. However, continuously completing a multitude of similar tasks may quickly bore the worker, causing the increase of worker churn. A healthy task recommendation system should, therefore, account for task diversity, ensuring a varied selection of task categories to cater to the multi-interests of workers.

Challenge III: How to recommend tasks to workers to achieve a trade-off between task coverage rate and diversity? The primary objective of a Spatial Crowdsourcing (SC) platform is to maximize benefits, achieved when workers successfully complete tasks and the platform receives a portion of the task reward. However, solely focusing on enhancing worker satisfaction may jeopardize these benefits. To establish a mutually beneficial relationship between the SC platform and workers, it is essential to simultaneously consider the trade-off between task coverage rate and diversity in task recommendations.

Observing these unmet challenges, we propose an SC framework, named Diversity-Coverage Balanced Task Recommendation ($DCBTaskRec$), for effective task recommendation. The framework takes into account multiple factors influencing workers' preferences, the diversity of tasks in a worker's recommended list, as well as the overall task coverage rate in the entire recommended set. It consists of two phases: a worker preference learning phases and a task recommendation phase. In the first phase, we design a Decoupled Spatial-Tempoal Self-Attention Network (DSTSAN). Specifically, we first separately model the spatial and temporal proximity among the historically performed tasks of workers, in which the similarities of spatiality and temporality are measured by a neural network and a periodic function, respectively. Then, sequentiality and the current interest are modeled through the similarity among the position and task embeddings. Due to the different contributions of each component, we adopt vanilla attention to adaptively integrate them to obtain the final attention matrix.

In the task recommendation phase, our objective is to maximize the diversity in the recommended list while maintaining a high preference-based utility for workers. To achieve this, we first propose an Optimal Diversity-Aware (ODA) approach. It considers the preference score and the category of a task when recommending to a worker. To enable a high task coverage rate, we consider the degree of a task (i.e., how many workers it can be recommended) and design a Greedy Coverage-Aware (GCA) approach, in which the tasks will be sequentially recommeneded according to the minimum number of available workers. To strike a balance between diversity in the worker's recommendation list and the task coverage rate, we further present a Diversity-Coverage Balanced (DCB) approach, which is designed based on the coarse-finetuning mechanism.

In general, our contributions can be summarized as follows:

- We propose a novel SANs-based workers' preference modeling method, i.e., DSTSAN, in which the spatio-temporal information and sequentiality from the historically performed tasks of workers can be uniformly modeled via the self-attention mechanism.
- We propose three task recommendation strategies that consider different task recommendation concerns, i.e., task coverage rate and task diversity.
- We conduct extensive experiments on two real datasets. The experimental results demonstrate the effectiveness and efficiency of the proposed solution.

II. Preliminaries

We proceed to give the necessary preliminaries and then define the problem addressed.

Definition 1 (Worker): A worker, denoted as $w = (l, d)$, is able to perform spatial tasks. A worker can be either online or offline. A worker is online when the worker is ready to accept tasks and offline when unavailable to perform tasks. An online worker w is associated with a current location $w.l$ and a reachable distance $w.d$. The reachable range of worker w is a circle with center $w.l$ and radius $w.d$, within which w can accept tasks.

Definition 2 (POI): A point of interest (POI), denoted by $p = (l, S)$, consists of a location $p.l$, and a set of tasks $p.S$ that are associated with the POI, i.e., the tasks in $p.S$ are located at $p.l$.

Definition 3 (Spatial Task): A spatial task, denoted by $s = (p, e, c, r)$, encompasses a POI $s.p$, a task expiration deadline $s.e$, a category of the task $s.c$, a reward $s.r$ that the worker completing s will obtain.

A spatial task s is said to be finished only if a worker can physically move to its location (i.e., $s.p$) before its expiration time (i.e., $s.e$). Although the SC platform can recommend multiple tasks, e.g., k, to a worker, a worker can only choose one task to perform at a time according to the single-task assignment mode. The state of a worker, i.e., online and offline, will be switched to offline once a worker chooses a task to perform.

Definition 4 (Reachable Task Set): Given an online worker w and a set of tasks to be recommended in the vicinity of w, a reachable task set for worker w, denoted as $RS(w)$, satisfy two conditions: $\forall s \in RS(w)$, 1) worker w is able to arrive at the location of task s before its expiration time, i.e., $t_{now} + t(w.l, s.l) < s.e$; and 2) task s is located in the reachable range of worker w, i.e., $d(w.l, s.l) \leq w.d$, where t_{now} is the current time, $t(w.l, s.l)$ is the travel time from worker w's location $w.l$ to tasks s's location $s.l$, and $d(w.l, s.l)$ is the travel distance from location $w.l$ to location $s.l$.

Definition 5 (k-Recommended-Task-Set): Given an online worker w and the reachable task set $RS(w)$, a recommended task set with k tasks, denoted as $RTS_k(w)$, is a subset of $RS(w)$, where the tasks in $RTS_k(w)$ are ranked according to workers' preferences, and k can be specified by the SC platform.

The utility of a worker from a recommended task set $RTS_k(w)$ is proportional to the sum of preference scores of workers. We follow AdaTaskRec [1] to define the preference-based utility.

Definition 6 (Preference-based Utility): The preference-based utility of worker w can be defined as the ranking metric, Normalized Discounted Cumulative Gain (NDCG) [12], over the recommended task set $RTS_k(w)$:

$$U(RTS_k(w)) = \frac{DCG(RTS_k(w))}{DCG(RTS_k^*(w)))}$$
$$DCG(RTS_k(w))) = \sum_{s \in RTS_k(w)} \frac{2^{c_w(s)} - 1}{log_2(rank(w, s|RTS_k(w)) + 1)} \tag{1}$$

where $DCG(RTS_k(w))$ denotes the discounted cumulative gain of worker w over $RTS(w)$, $RTS_k^*(w)$ is the expected optimal recommended task set for w, i.e., the k tasks with the highest preference scores of w, $c_w(s)$ is w's preference score for task $s \in RTS_s(w)$, and $rank(w, s|RTS_k(w))$ is the position that task s is placed at in the ranking task set $RTS_k(w)$ for w.

It should be noted that recommending the k most interested tasks will give the maximum possible utility according to definition 6. However, due to the intrinsic shortcoming of neural-based recommender systems (RSs), i.e., RSs usually predict similar tasks compared to the workers' performed history [13], only maximizing utility may not satisfy the workers' multiple interests and thus damage the benefit of the platform in a long range. Similar to previous studies [13], we also use diversity in terms of task category to measure how the recommended set support the multiple interests of workers.

Definition 7 (Diversity): Diversity measures the number of distinct categories the recommendation set contains, which can be formulated as follows:

$$Diversity(S_u) = 1 - \frac{\sum_{i,j \in S_u, i \neq j} sim(i,j)}{|S_u|(|S_u| - 1)} \tag{2}$$

, where S_u presents a recommendation set to the user u, $|S_u|$, indicates the cardinality of S_u, and $sim(i,j) = 1$ if i and j belong to the same category (such as taking a photo and reporting a hot pot), and 0 otherwise.

SC platform obtains the benefits only if a worker finishes a task. Thus, to maximize the benefits, the platform needs to recommend as many as possible tasks to workers to increase the task completion rate. Due to the number of tasks to be recommended to a worker being limited to k, the platform should instead increase the task coverage rate [1].

Definition 8 (Coverage): Coverage indicates how many distinct POIs are contained in the recommended sets, which can be formulated as follows:

$$Coverage(R) = \frac{|R|}{|I|} \tag{3}$$

where $R = \cup_{u \in U} S_u$, and I is the set of total tasks.

$Diversity$ and $Coverage$ are two different metrics to show the different aspects of the recommended sets, i.e., the former measure the results from the individual level, while the latter from the global level. To achieve high diversity and coverage simultaneously, we combine them to define new metrics, namely DC, as follows:

$$DC(R) = \lambda\, Coverage(R) + (1 - \lambda)\frac{1}{|U|}\sum_{u \in U} Diversity(S_u) \tag{4}$$

where $\lambda \in [0, 1]$ is a balance weight to control contributions of different components. When λ is set to 0 (or 1), DC is degenerated to Diversity (or Coverage).

Definition 9 (Spatial Task Recommendation): Given a set of workers W and a set of tasks S, a spatial task recommendation, denoted by R, consists of a set of pairs of a worker and a k-recommended-task-set for the worker: $(w_1, RTS_k(w_1)), ..., (w_{|W|}, RTS_k(w_{|W|}))$, where $|W|$ denotes the number of the worker set.

Problem Statement. Given a worker set W and a task set S, the task recommendation problem in SC is to find an optimal task recommendation R_{opt} that achieves the following goals: 1) primary optimization goal: maximize the task coverage rate (cf. Equation 3) and diversity (cf. Equation 2), with a limited k ($k \ll |S|$) value; and 2) secondary optimization goal: maximize the average worker preference-based utility (cf. Equation 1).

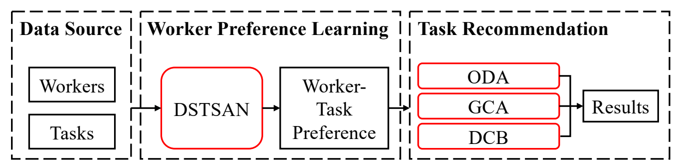


Fig. 1. Framework Overview

III. Methodology

We present the overview of *DCBTaskRec* in Figure 1, which consists of two phases, i.e., a worker preference learning and a task recommendation. In the first phase, we design a Decoupled Spatio-Temporal Self-Attention Network (DSTSAN) that learns workers' preferences based on the performed tasks in terms of sequentiality and spatio-temporality, where each task is associated with a POI. The learned preference model is used in the second phase to make the recommendation utility-aware. In the task recommendation phase, to maximize the diversity of recommended task list for each worker, we propose an optimal diversity-aware approach (i.e., ODA). However, it achieves almost the lowest task coverage rate, which does not fit the benefits of the SC platform. To enable a high task coverage rate, a greedy coverage-aware (GCA) approach is proposed. Based on this method, we propose a diveristy-coverage balanced approach (i.e., DCB) based on coarse-finetuning framework, to balance the task coverage rate and diversity with slightly sacrificing the utility of workers. We elaborate on each proposal as follows.

A. Worker Preference Learning

We design a model to learn the worker's preference on tasks, where the overview of DSTSAN is shown in Figure 2. It

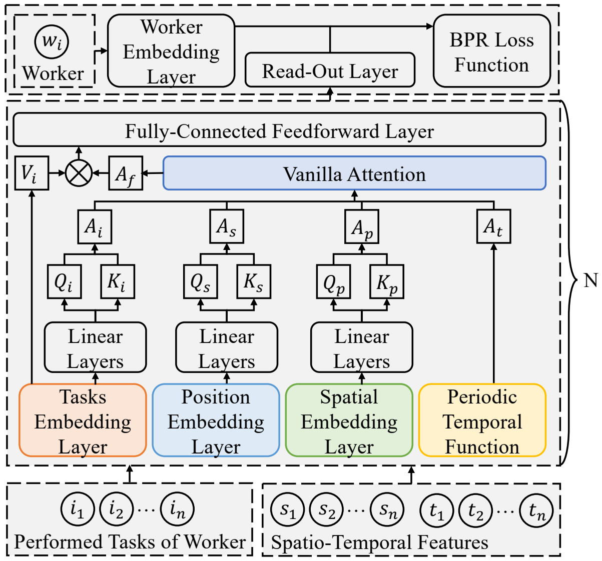


Fig. 2. DSTSAN Model Overview

is an end-to-end model based on SANs, where we modify the calculation of the attention matrix in the self-attention mechanism through the decoupled modeling of tasks, sequentiality, spatiality, and temporality. Then, we integrate these attention matrices through a vanilla attention layer to fit the different contributions of these properties. Finally, a read-out layer is adopted to summarize the worker's preference from the historically performed tasks, integrating with the embedding of the worker to calculate the preference scores for the candidate tasks. We train the model by minimizing the Bayesian Personalized Ranking (BPR) loss to the ground-truth. Next, we first present the workflow of DSTSAN and then elaborate on the details of each component.

1) Workflow of DSTSAN: The overall framework of DSTSAN consists of multiple SAN blocks followed by a read-out layer to summarize the worker preference from the performed tasks. The i-th block of DSTSAN can be presented as follows:

$$\begin{aligned} E_i &= DSAN(E_i, P, G, T) \\ E_i &= FFN(E_i) \end{aligned} \tag{5}$$

where $DSAN$ presents the decoupled SAN, E, and P are the embeddings of POI where the tasks in, and position for the modeling of worker's interests and sequentiality, $G = (g_1, g_2, ..., g_n)$ and $T = (t_1, t_2, ..., t_n)$ present the GPS locations of tasks (e.g., $g_1 = (s_1^{lat}, s_1^{lon})$, where s_1^{lat} and s_1^{lon} stand for the latitude and longitude of location s_1,) and the performing time (e.g., the worker performs task i_1 at t_1), FNN is a two-layer feedforward layer with $GELU$ as activation. N blocks are stacked in DSTSAN, where $N = 2$ in our experiments. Then, a read-out layer is followed to extract the worker's preference from the sequence of tasks. Following the well-known sequential recommendation model, SASRec [14], we use the most recent embedding of POI (i.e., e_n^N, the embedding of e_n after N blocks) as the worker embedding. Despite the embeddings from the performed tasks that can present the workers' preferences dynamically, the long-term preference of workers is ignored. To fill this gap, we integrate the embedding of worker identity (e.g., e_{w_i}) and the embedding from the performed tasks through additive operation (i.e., $e_{w_i}^f = e_{w_i} + e_n^N$) to obtain the final embedding of worker w_i. To measure the preference score (i.e., y_{ij}) of worker w_i on the task in POI p_j, the dot-product on their latent embeddings is adopted (i.e., $y_{ij} = {e_{w_i}^f}^T E_{p_j}$). In the model training phase, the widely-used BPR loss function is used to measure the difference between the predictions and the ground-truths.

2) Details of DSAN: In vanilla SANs, the position embedding (e.g., P) is early-integration with POI embeddings (e.g., E) through additive operation before inputting into SANs (e.g., $E+P$). Thus, the attention value between the i-th POI and j-th POI is calculated as $E_i^T E_j + P_i^T P_j + E_i^T P_j + P_i^T E_j$, where the modeling of POI-to-position (i.e., $E_i^T P_j + P_i^T E_j$) may introduce the noise due to the low correlations between position and the content of POI [15]. Similarly, the POI-to-time and time-to-space have a similar problem if we do the early-integration. Thus, we decouple these things in the attention mechanism to prevent the invalid correlation calculation among different types of embedding. Specifically, four attention matrices are computed in our attention layer, i.e., POI-to-POI, position-to-position, space-to-space, and time-to-time. The calculation of the first three matrices has the same formulation with different parameters to be learned. The formulation of the POI-to-POI attention matrix is as follows.

$$\begin{aligned} A_i &= Softmax(Q_i K_i^T / \sqrt{d}) \\ Q_i = E_i W_i^q &\quad K_i = E_i W_i^k \end{aligned} \tag{6}$$

where d indicates the latent dimension, $A_i \in \mathcal{R}^{n \times n}$ is the attention matrix in terms of POI, W_i^q and $W_i^k \in \mathcal{R}^{d \times d}$ are the learnable parameters for query and key embeddings, respectively. Similarly, we can obtain the attention matrices in terms of position and space and denote them as A_p and A_s, respectively. The time-to-time correlation is not modeled in the same way due to the periodic patterns existing in the workers' behavior, which cannot be well modeled through the computation in Equation 6. Following previous studies [16], [17], we adopt a periodic function to obtain the attention matrix A_t in terms of time.

$$A_t[i,j] = \frac{1 + \cos(2\pi \Delta T_{ij})}{2} \cdot e^{-\alpha \Delta T_{ij}} \tag{7}$$

where ΔT_{ij} indicates the time interval between the performing time of the i-th and j-th tasks, α is a temporal decay rate, which controls how fast the weight decreases over time ΔT_{ij}. Considering the different contributions of these attention matrices, we leverage a vanilla attention layer to adaptively integrate them. For the k-th task, the vanilla attention can be formulated as follows.

$$\begin{aligned} A_f[k,:] &= \sum_{* \in \{i,p,s,t\}} A_*[k,:] \\ w_* &= Softmax_{* \in \{i,p,s,t\}}(A_*[k,:] W^a + b^a) \end{aligned} \tag{8}$$

Algorithm 1: ODA approach

Input: Reachable task sets RS, all workers W, a specified value k
Output: Task recommendation R

1 $R \leftarrow \emptyset$;
2 **for** w *in* W **do**
3 $\quad k_{min} = min(len(RS(w)), k)$;
4 $\quad$ group tasks in $RS(w)$ according to its category;
5 $\quad$ **while** $len(R(w)) < k_{min}$ **do**
6 $\quad\quad$ descendingly sort the groups according to the maximal preference score in the category;
7 $\quad\quad$ **for** *all categories in* $RS(w)$ **do**
8 $\quad\quad\quad$ select the most interested task s in this group;
9 $\quad\quad\quad$ **if** $len(R(w)) < k_{min}$ **then**
10 $\quad\quad\quad\quad R(w) \leftarrow R(w) \cup (w, s)$;
11 $\quad\quad\quad$ **end**
12 $\quad\quad$ **end**
13 $\quad\quad RS(w) \leftarrow RS(w) - R(w)$;
14 $\quad$ **end**
15 **end**
16 **Return** R;

Algorithm 2: GCA approach

Input: Reachable task sets RS, available worker sets AW, all tasks S, all workers W, a specified value k
Output: Task recommendation R

1 $R \leftarrow \emptyset$;
2 **while** $|S|$ *is not empty* **do**
3 $\quad$ ascendingly sort the tasks S according to the number of available workers for each task;
4 $\quad$ select the task s with the minimum number of degree;
5 $\quad$ descendingly sort the workers in $AW(s)$ according to the preference score;
6 $\quad$ **for** w *in* $AW(s)$ **do**
7 $\quad\quad$ **if** $len(R(w)) < k_{min}$ **then**
8 $\quad\quad\quad R(w) \leftarrow R(w) \cup (w, s)$; break;
9 $\quad\quad$ **end**
10 $\quad\quad AW \leftarrow AW - w$;
11 $\quad$ **end**
12 $\quad S \leftarrow S - s$;
13 $\quad$ eliminate the task $s \in S$ whose available workers is empty;
14 **end**
15 fill the recommended list of each worker to k_{min} with the highest preference scores in RS;
16 **Return** R;

where $A_*[k,:]$ presents the k-th row of the matrix A_*, W^a and b^a are the learnable parameters, w_* is the weight of the attention matrix A_*. Compared with STiSAN [11] that integrates the spatio-temporal information by directly adding the spatio-temporal interval matrix with the POI-to-POI attention matrix, our method provides a more flexible and adaptive way to model the different contributions of the multiple influences.

B. Task Recommendation

The typical task recommendation method is the TOPK recommendation after obtaining of the workers' preferences on tasks, which recommends the k most interested tasks to a worker from the worker's available task set. However, this strategy only focuses on maximizing the preference-based utility of workers, which ignores the benefits of the SC platform (i.e., the task coverage rate) and harms the workers' multi-interests in a long-range (i.e., diversity in the recommended list). To deal with these, we propose three task recommendation strategies, i.e., optimal diversity-aware (ODA), greedy coverage-aware (GCA), and diversity-coverage balanced (DCB) approaches, to meet the requirements of different applications. We elaborate on each method in the following parts.

1) ODA approach: We first focus on maximizing the diversity in the worker's recommended list. From the definition of diversity (cf. Equation 2), two facts can be found: 1) The diversity of worker w_i will not affect the diversity of w_j, which means we can obtain the global optima if each worker gets the maximal diversity on their recommended list; 2) More even recommendation on different categories achieve the best diversity. For instance, the SC platform aims to recommend two tasks to a worker w_i, where the available tasks set of w_i is (s_1, s_2, s_3) and s_1 and s_2 share the same category while s_3 is different. Recommending (s_1, s_2) to the worker achieves the lowest diversity, i.e., $diversity(w_i) = 1 - \frac{2}{2 \times 1} = 0$, otherwise (e.g., (s_1, s_3)), we can achieve the highest diversity, (e.g., $diversity(w_i) = 1 - \frac{0}{2} = 1$). Based on these facts, we propose an optimal approach with diversity as the first goal and preference-based utility of worker as the second goal, namely ODA, as shown in Algorithm 1, which takes the reachable task sets RS for all workers, all workers W, and a k value as input. After initialization (line 1), for each worker, we aggregate the available tasks according to the task's category (lines 2-4). If the number of recommended tasks is less than k_{min}, where k_{min} indicates the most number of recommendations of the looped worker, we sort the groups according to the maximal preference score in these groups (line 6). Then, we loop each group and recommend the worker's most interested task until it reaches the exit condition (i.e., $len(R(w)) = k_{min}$) (line 7-13).

2) GCA approach: Despite the ODA approach achieving the optimal diversity, the task coverage rate will be extremely affected (i.e., it performs the worst as TOPK) and far from the satisfaction of the SC platform (cf. Section IV-B2). Here, we propose a task-degree-reduction-greedy method, namely GCA, to maximize the task coverage rate while achieving a high utility for workers. The core idea of GCA is to recommend the task with a minimum number of available workers since it has the highest probability that cannot be recommended if the number of recommended tasks of its available workers is full. Algorithm 2 depicts the details of the proposed method, which takes the reachable task sets RS, the available worker sets AW, and a k value as input, where $AW(s)$ records the available workers for task s. Firstly, the recommended set R is initialized (line 1). Then, while the task set S is not empty (line 2), we sort the tasks in S according to the number of available workers for each task (i.e., degree of tasks) and select the task with the minimum degree (lines 3-4), in which we select the task whose preference score is highest if there are

Algorithm 3: DCB-Finetuning

Input: Reachable task sets RS, task recommendation R from the coarse phase, all workers W
Output: Task recommendation R

1 **while** R *is not changed* **do**
2   **for** w *in* W **do**
3         $ImproveDiversity(RS, R, W)$;
4   **end**
5 **end**
6 **Return** R;

Algorithm 4: ImproveDiversity

Input: Reachable task sets RS, task recommendation R from the coarse phase, all workers W
Output: Task recommendation R

1 $\delta_c = \lambda/|S|$;
2 **for** c_1, c_2 *in all categories of* $RS(w)$ **do**
3   **if** $N_w(c_1) - N_w(c_2) < 2$ **then**
4   continue;
5   **end**
6   group the tasks in $RS(w)$ with category c_1 into D_s and ND_s, where D_s records the tasks that will cause the decrease of Coverage if it is removed from R;
7     $ND_s \leftarrow ND_s \cap R(w)$; $D_s \leftarrow D_s \cap R(w)$;
8   **if** ND_s *is not empty* **then**
9   select a task s_1 in ND_s with the lowest preference score;
10   **end**
11   **else**
12   calculate δ_d using Equation 9;
13   **if** $\delta_d > \delta_c$ **then**
14   select a task s_1 in N_s with the lowest preference score;
15   **end**
16   **end**
17   select an unrecommended c_2 category task s_2 with the highest preference score;
18     $R \leftarrow R - s_1 + s_2$;
19 **end**
20 **Return** R;

multiple tasks with the same minimum degree. After that, we sort and loop the task available workers $AW(s)$ according to the preference scores (lines 5-6). If the recommended list of the looped worker is full (i.e., $len(R(w)) = k_{min}$), we remove this worker from the task available workers AW (lines 10). Otherwise, the recommendation (w, s) is recorded into R. At the end of the loop, we remove the task from all tasks set S and eliminate the task whose available workers are empty (lines 12-13). Finally, the worker whose number of recommended tasks is less than k_{min} is filled by the tasks with the highest preference scores in the reachable task set RS (line 15).

3) DCB approach: The previously mentioned methods, namely ODA and GCA, each excel in a specific aspect, either diversity or task coverage rate (see Section IV-B2). However, finding a way to simultaneously achieve both objectives to satisfy the requirements of the SC platform and workers still poses a challenging problem. To address this, we use DC (cf. Equation 4) as our final decision metric and design a coarse-finetuning framework, which achieves near optimal value in terms of diversity and coverage with tolerable sacrifice on the preference-based utility in our empirical studies. Specifically, the DCB approach consists of two phases: coarse phase and finetuning phase. In the coarse phase, we employ the GCA algorithm. Subsequently, we propose a finetuning algorithm in the finetuning phase, which maximizes DC with a specified λ (e.g., $\lambda = 0.5$). From the definition of DC, we can obtain a higher value of DC by improving diversity or coverage. Since the GCA approach is used as the algorithm in the coarse phase, we only focus on the improvement of diversity in the finetuning phase.

We first provide a quantitative metric of replacing a task in the recommended list in terms of the diversity in DC, which can be used to quickly filter out groups of tasks that are impossible to make an improvement of DC, and then give an example of the computation of δ_d. Suppose $N_w(c_1)$ is a function that returns the number of recommended tasks of worker w with category c_1, k_{min} (i.e., $k_{min} = min(len(RS(w)), k)$) is the maximal number of tasks can be recommended for w, $|W|$ is the number of workers. Replacing a task whose category is c_1 in R with a task whose category is c_2, the variation of DC in terms of diversity δ_d is as follows:

$$\delta_d = 2(1-\lambda)(N_w(c_1) - N_w(c_2) - 1)/(k_{min}(k_{min}-1)|W|) \quad (9)$$

Here, we provide a running example as follows, to demonstrate the consistency between computing the variation of DC score from scratch and using above equation after replacing a task recommendation.

Example 1: Let $(s_1, s_2, s_3, s_4, s_5)$ is the available tasks of worker w, in which the categories of (s_1, s_2), (s_3, s_4), and (s_5) are c_1, c_2, and c_3, respectively. The number of workers $|W|$ and λ are 10 and 0. Replacing a task recommendation $R(w) = (s_1, s_2, s_3)$ with $R'(w) = (s_1, s_3, s_5)$, where $N_w(c_1)$, $N_w(c_3)$, and k_{min} are 2, 0, and 3, the variation of DC in terms of diversity δ_d is $2 * (2 - 0 - 1)/(3 * 2 * 10) = 1/30$. This variation can also be obtained through the exact computation the difference between $Diversity(R(w))$ and $Diversity(R'(w))$, where $Diversity(R(w)) = 1 - \frac{2}{3*2} = \frac{2}{3}$ and $Diversity(R'(w)) = 1 - \frac{0}{3} = 1$. The induced variation of DC in terms of diversity is $(1-\lambda)(1-2/3)/|W|$, which equals to $(1-0)1/3/10 = 1/30$.

The core idea of the DCB approach is to iterate all the workers and find the possible replacements to improve DC in terms of the diversity of each worker. Algorithm 3 shows the finetuning process, which takes reachable task set RS, task recommendation R from the coarse phase, and all workers W as inputs. It iterates the workers W and calls the sub-procedure $ImproveDiversity$ until the task recommendation R keeps unchanged. In $ImproveDiversity$, the variation of DC in terms of coverage δ_c is first computed, which presents the decrease of DC by replacing a task that is recommended once (line 1). Then, we loop all the categories in $RS(w)$ (line 2). If the difference in terms of number of recommendation between the two selected categories less

TABLE I
STATISTICS (AFTER PREPROCESSING)

Dataset	#Users	#POIs	#Category	#Check-ins	Sparsity
NYC	1084	5136	286	146855	0.9736%
TKY	2294	7874	314	445277	0.9753%

than 2 (i.e., $N_w(c_1) - N_w(c_2) < 2$), we can assert that replacing any task between these categories, DC in terms of diversity will not be improved according to Equation 9 (i.e., $(N_w(c_1) - N_w(c_2) - 1) \leq 0$ always holds). Thus, we directly jump these category pairs (line 5). Next, we try to replace a task from category c_1 with a task from category c_2. We separate the category c_1 tasks in $RS(w)$ into two groups, ND_s and D_s, where ND_s records the tasks that are removed from this recommended list will not decrease the task coverage rate and D_s records the rests (line 6). We guarantee that the tasks to be replaced out in ND_s and D_s are in the recommended list $R(w)$ (line 7). We select a task in ND_s with the lowest preference score to replace out if ND_s is not empty (lines 8-10). Otherwise, we compare the variation of DC in terms of coverage and diversity (i.e., δ_c and δ_d). If the gain from the improvement of diversity is greater than the decrease of coverage, we select a task in D_s with the lowest preference to replace out (lines 11-16). Finally, an unrecommended task s_2 whose category is c_2 with the highest preference score is selected, and the task recommendation R is updated (lines 17-18).

IV. EXPERIMENTS

We evaluate the performance of the worker preference learning and the task recommendation on real data. We conduct the experiments on Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz with 128 RB RAM and a GeForce GTX 1080 GPU.

A. Experimental Setup

Due to the lack of benchmark for task recommendation algorithms in SC, we use two real check-in datasets from Foursquare, i.e., New York City (NYC) and Tokyo (TKY), to simulate the task recommendation scenario, where NYC and TKY are collected from 12 April 2012 to 16 February 2013 [18]. To ensure the data quality, we filter out the POIs that are visited less than 10 times and the users whose check-ins are less than 10 times. After that, we chronologically sort users' all the check-ins. The first 60% check-ins of each user are split into multiple length-equally (e.g., 20) sequences, which are chosen as training sets. The following 20% and the left are used for validation and testing, respectively. Table I shows the statistics of the two datasets after preprocessing used in our experiments.

For the task recommendation experiments, tasks are generated randomly on POIs, which means that each POI may have several tasks. The speed of workers in both datasets is set to 5km/h. Since the number of users in both datasets is insufficient, we generate workers based on the long-term check-in POIs. Specifically, we take each sub-sequence to simulate the travel record of a worker. For example, suppose (p_0, p_1, p_2) is a sequence of check-ins. We can generate four workers from it (i.e., (p_0), (p_0, p_1), and (p_0, p_1, p_2)) which enables us to have enough workers to study the scalability of the proposed methods. For simplicity and without loss of generality, we assume that the processing time of a task is 0, which means that a worker will proceed to the location of the next task immediately upon finishing the current one [1], [19]. Moreover, we run the task recommendation methods over 10 rounds and report the average results. In each round, a worker selects a task randomly from the recommended task list.

B. Experimental Results

1) Performance of Worker Preference Learning: In this experiment, we evaluate the performance of the worker preference learning phase.

Evaluation Methods. We compare the proposed method with seven representative competitors and two variants of ours.

- GRU4Rec [20]: employ GRU to model the sequential information in user's check-ins.
- STAMP [21]: captures users' general interests of the current session and current interests of the last click.
- SRGNN [22]: models the complex item transition relations through the constructed session graph and generates the user embedding through the attentive technique.
- NARM [23]: employs RNNs with an attention mechanism to capture the users' main purpose and sequential behavior.
- SASRec [14]: stacks multiple self-attention modules followed by fully-connected layers to model the long-term sequential information and relationship among POIs.
- Flashback [17]: leverages RNNs with spatial and temporal similarity matrix to model the users' spatio-temporal behavior.
- STiSAN [11]: incorporates spatio-temporal interval into the self-attention technique to precisely enhance sequence representations to reflect spatio-temporal proximity among checked POIs.
- DSTSAN-E: a variant of the proposed model, which fuses positional embeddings before inputting into self-attention.
- DSTSAN-ST: a variant of the proposed model, which eliminates the spatio-temporal similarity in self-attention.

Metrics. We adopt two widely-used metrics, Hit Rate (HIT), and Normalized Discounted Cumulative Gain (NDCG) [12], to evaluate how well the target POIs in the test set are ranked. We report three metrics at $k = 5$ and $k = 10$ in our experiments. The larger the reported values are, the better the performance the model achieves.

Parameter Settings. For all the models to be evaluated, the hidden size and batch size are fixed at 64 and 2048, respectively. For the proposed model and the variants, we set the number of layers and the number of heads in attention to 2 and 4, respectively. We adopt Adam optimizer and L2 regularization with weight 1e-5 to train all the learning models. For the baselines, we follow their original settings to conduct experiments. Specifically, for Flashback, we set their spatial factor λ_t and the temporal factor λ_t to 1000 and 0.1, respectively. For the attention-based competitors, such as

TABLE II
OVERALL PERFORMANCE

Method	Dataset / Metrics	NYC					Forsquare-TKY				
		HIT@1	HIT@5	HIT@10	NDCG@5	NDCG@10	HIT@1	HIT@5	HIT@10	NDCG@5	NDCG@10
GRU4Rec		0.1136	0.3055	0.4045	0.2137	0.2457	0.1070	0.2745	0.3690	0.1931	0.2237
STAMP		0.1465	0.3880	0.4850	0.2715	0.3029	0.1202	0.3181	0.4130	0.2227	0.2534
SRGNN		0.1525	0.3866	0.5048	0.2747	0.3097	0.1216	0.3633	0.4557	0.2423	0.2622
NARM		0.1554	0.4106	0.5198	0.2888	0.3243	0.1287	0.3401	0.4339	0.2387	0.2598
SASRec		0.1549	0.4304	0.5342	0.3042	0.3379	0.1307	0.3679	0.4708	0.2547	0.2881
Flashback		0.1644	0.4376	0.5472	0.3073	0.3428	0.1258	0.3323	0.4387	0.2382	0.2471
STiSAN		<u>0.1743</u>	<u>0.4481</u>	<u>0.5557</u>	<u>0.3102</u>	<u>0.3492</u>	<u>0.1312</u>	<u>0.3876</u>	<u>0.4932</u>	<u>0.2589</u>	<u>0.2933</u>
DSTSAN-E		0.1705	0.4473	0.5543	0.3157	0.3505	0.1319	0.3955	0.5037	0.2646	0.3056
DSTSAN-ST		0.1749	0.4392	0.5492	0.3086	0.3444	0.1292	0.3994	0.5205	0.2694	0.3097
DSTSAN		**0.1810**	**0.4633**	**0.5749**	**0.3286**	**0.3648**	**0.1370**	**0.4077**	**0.5291**	**0.2757**	**0.3141**

TABLE III
EXPERIMENT PARAMETERS

Parameter	Value
Number of tasks, $\|S\|$	2K, 4K, <u>6K</u>, 8K, 10K
Number of workers, $\|W\|$	1K, 2K, <u>3K</u>, 4K, 5K
Number of POIs, $\|P\|$	1K, 2K, <u>3K</u>, 4K, 5K
Expiration time of tasks (h), e	0.5, <u>1.0</u>, 1.5, 2.0, 2.5
Reachable distance of workers (km), d	1, 2, 3, 4, <u>5</u>
Number of recommended tasks, k	4, 6, 8, <u>10</u>, 12, 14

SASRec and STiSAN, we set the number of layers and heads in line with ours for fair comparison.

Accuracy. We compare the worker preference learning models on both datasets, i.e., NYC and TKY, and report the HIT and NDCG values in Table II. Note that we test all baselines for 10 rounds, and take the average value as their final performance. The best performance by an existing method is underlined, and the overall best performance is marked in bold. For both datasets, DSTSAN achieves the highest HIT@N, which outperforms the best among the baseline methods by up to 0.5749 and 0.5291 in NYC and TKY, respectively. In terms of NCDG, DSTSAN performs the best among all methods, followed by its variants (DSTSAN-E in NYC and DSTSAN-ST in TKY) and other methods in both datasets. Moreover, DSTSAN always achieves better accuracy than its two variants regardless of metrics and datasets, which demonstrates the superiority of the decoupled positional embedding and spatio-temporal proximity in the attention matrix.

2) Performance of Task Recommendation: Next we evaluate the performance of task recommendation in spatial crowdsoucing.

Evaluation Methods. We study the following methods.

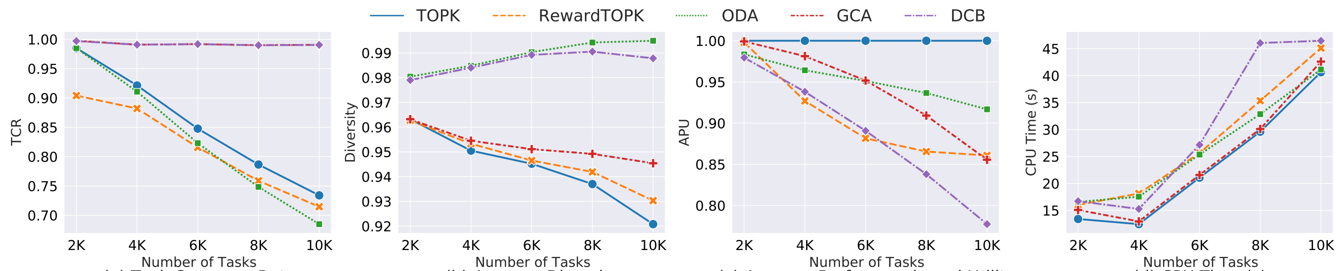
- TOPK: The traditional TOPK method, recommends each worker k most interested tasks (i.e., k tasks with the highest preference scores) from reachable tasks of the worker.
- RewardTOPK [1]: It gives priority to the ignored tasks in previous task recommendation iterations.
- GCA, ODA and DCB: The proposed methods.

Metrics. Four main metrics are compared for the above algorithms, i.e., Task Coverage Rate (TCR), Average Diversity (AD), Average Preferenced-based Utility (APU), and CPU time for finding task recommendation, in which TCR is the ratio between the number of recommended tasks and the total number of tasks, AD is the average diversity of workers, and APU is the average preference-based utility of workers. A larger TCR, AD, or APU implies better task recommendation. The formal definition of AD and APU are as follows.

$$AD = \frac{\sum_{w \in W}(Diversity_w)}{|W|}$$
$$APU = \frac{\sum_{w \in W}(RTS_k(w))}{|W|} \tag{10}$$

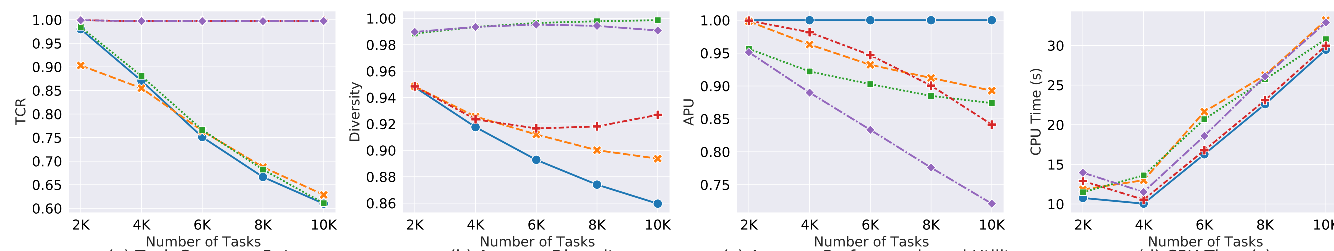
Parameter Settings. Table III shows our experimental settings, where the default values of all parameters are underlined.

Effect of $|S|$. We first study the effect of the number of tasks $|S|$ on both datasets. From Figures 3(a) and 4(a), we can see that the GCA and DCB approaches always achieve a higher task coverage rate compared with TOPK by up to around 25% and 35%, respectively, and the ODA approach is comparable with TOPK in most cases, which shows the superiority of our methods in terms of TCR. Besides, the superiority of the GCA and DCB approaches is more prominent when the number of tasks increases, i.e., the performance gaps of our methods and TOPK are increasing with $|S|$ grows. Moreover, the TCR of all methods decline with increasing $|S|$, but the TCR of the GCA and DCB approaches is rather stable with a slight increase of CPU time in Figures 3(d) and 4(d), which demonstrates their good scalability. Figure 3(b) and 4(b) show the average diversity of workers, in which the ODA approach always achieves the best. Besides, the DCB approach is comparable with it, which shows that we can slightly sacrifice the performance of average diversity to largely improve the task coverage rate to satisfy the multiple tastes of workers. When it comes to the APU of workers in Figure 3(c) and Figure 4(c), since TOPK recommends the k most interested tasks for each worker, it achieves the highest APU (i.e., optimal APU) followed by the GCA, ODA, and DCB approaches in most cases on both datasets. For efficiency, TOPK is the fastest due to its simplest strategy of task recommendation, while the other four methods are comparable and cost slightly more than TOPK, which demonstrates the efficiency of the proposed methods. RewardTOPK can achieve consistently improvement in terms of diversity compared with TOPK. However, its TCR cannot preserve the consistency, i.e., TCR of RewardTOPK is usually outperforms than it of TOPK in TKY dataset while the situation is reversed in NYC dataset, which is also shown in the later experimental results. This finding shows that the robustness should be concerned when



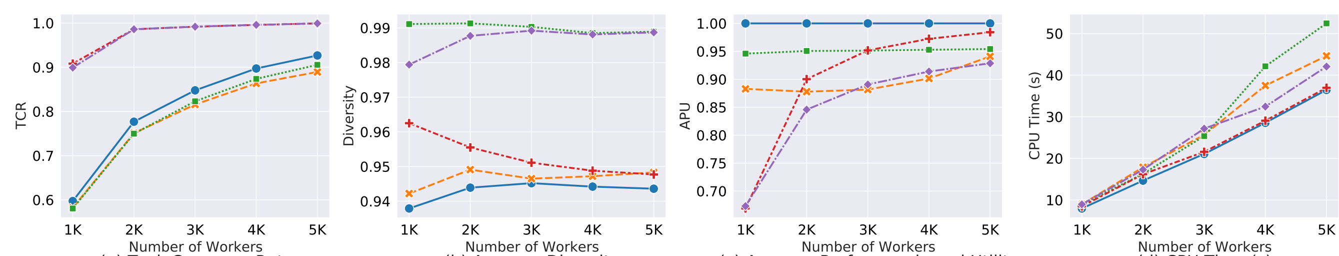
(a) Task Coverage Rate (b) Average Diversity (c) Average Preference-based Utility (d) CPU Time (s)

Fig. 3. Effect of $|S|$ on NYC



(a) Task Coverage Rate (b) Average Diversity (c) Average Preference-based Utility (d) CPU Time (s)

Fig. 4. Effect of $|S|$ on TKY



(a) Task Coverage Rate (b) Average Diversity (c) Average Preference-based Utility (d) CPU Time (s)

Fig. 5. Effect of $|W|$ on NYC

applying it into practice. Comparing with RewardTOPK, our proposals is more robustness in terms of TCR and diversity as shown in Figure 3 and 4.

Effect of $|W|$. Next, we study the effect of $|W|$, the number of workers to be recommended. As shown in Figures 5(a) and 6(a), our proposed approaches, i.e., GCA, and DCB, can always achieve higher TCR than the traditional Top-k and RewardTOPK method, which can improve the TCR by up to around 30% and 40%, respectively. In Figure 5(b) and 6(b), with the increases of $|W|$, the AD of Controllable gradually increases until near the theoretical optimum (i.e., AD of the ODA approach) which shows that the DCB approach is good at simultaneously satisfying the benefit of platform (i.e., TCR) and workers (i.e., AD). The APU of TOPK is the highest, but it cannot achieve a good task coverage rate and average diversity of workers as shown in Figures 5(c) and 6(c). For efficiency, TOPK still runs faster than the proposed methods in Figures 5(d) and 6(d), but the running time of others is also acceptable.

Effect of $|P|$. Figures 7 and 8 show the effect of the number of POIs, $|P|$, on the performance of all methods. When the number of POIs increases, the TCR of all methods is stable, as shown in Figures 7(a) and 8(a). The GCA and DCB approaches can obtain higher task coverage rate than the other three methods while sacrificing some diversity and utility of workers, as shown in the later two subfigures. In addition, we can see an upward trend of AD and APU for all methods due to the generation of diversified tasks on different POIs. Figures 7(d) and 8(d) show that the running time of these methods is randomly affected by the varies of POIs, which may come from the random generation strategy of tasks, i.e., a task is randomly generated from one of POIs at each timestamp.

Effect of e. Next, we study the effect of expiration time (e) on the recommendation performance. Figures 9(a) and 10(a) show that RewardTOPK and TOPK approach perform the worst compared with others and show a downward trend with increasing e, especially in the TKY dataset. For the inferior performance of RewardTOPK on TCR, we speculate that the reason is as follows: the default setting has 3K workers and 6K tasks, which cause there are too many tasks that cannot be done. With the time goes on, the left tasks in previous iterations will have extreme large chance to be recommended to the workers due to the reward mechanism. Thus, the TCR of

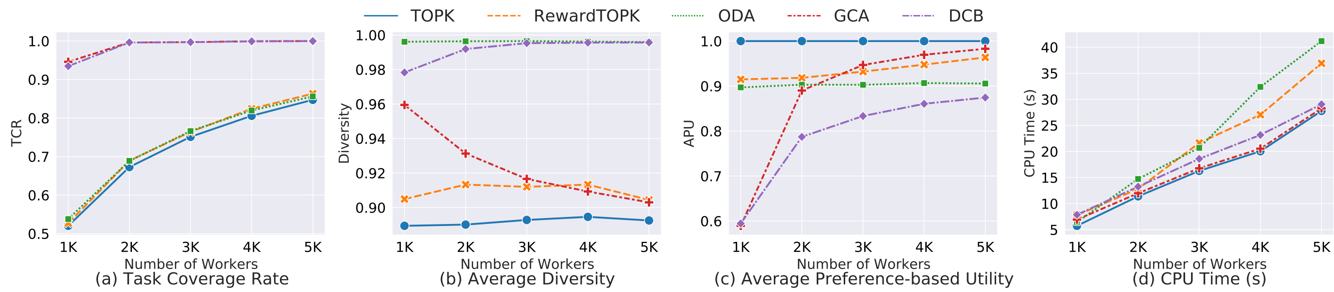


Fig. 6. Effect of $|W|$ on TKY

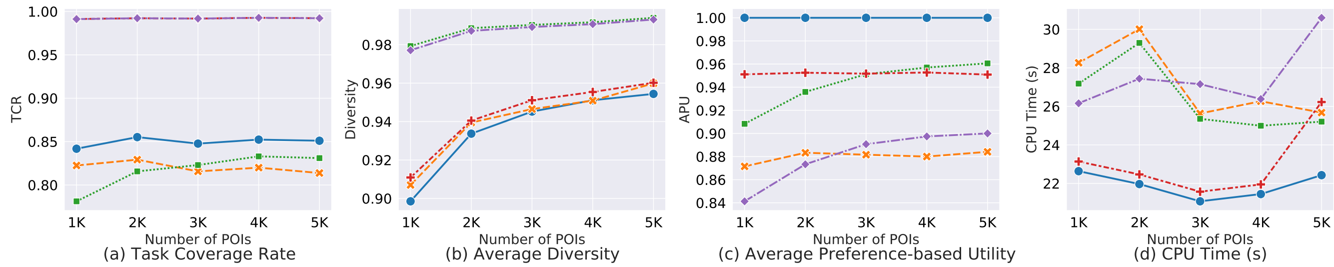


Fig. 7. Effect of $|P|$ on NYC

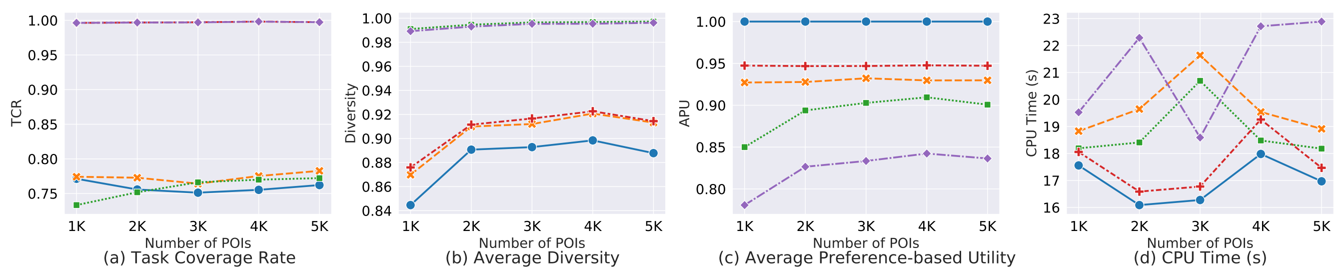


Fig. 8. Effect of $|P|$ on TKY

RewardTOPK will be unusually worse than it of TOPK, which also demonstrates that our proposed method is more robust than RewardTOPK in terms of TCR. When e gets larger, the AD of ODA and DCB show an upward trend as shown in Figures 9(b) and 10(b) since each worker may have more reachable tasks with more relaxed valid time. The CPU time of all methods increases with increasing expiration time e in Figures 9(d) and 10(d) due to the implicit increase of the number of tasks after several rounds of task recommendation.

Effect of d. We study the effect of reachable distance d of workers. Figures 11(a) and 12(a) show that the TCR of GCA and DCB increases gradually with increasing d, which is because the larger d is, the more tasks can be reachable and recommended for workers. TOPK, RewardTOPK and the ODA approach without the design of considering task coverage achieve their best at around 3 in NYC and 2 in TKY, respectively, and then deteriorate fast after further increasing of d. From Figures 11(b) and 12(b), we can observe that the DCB approach can always achieve the near-optimal AD with negligible gaps. Besides, the DCB approach performs the worst in APU metric, which is the cost for achieving excellent TCR and AD. Moreover, Figures 11(d) and 12(d) show an ascending trend due to the similar reason of increasing e, i.e., a larger d means more reachable tasks to access and processing more tasks will cost more CPU time.

Effect of k. We also study the effect of the number of recommendations k in Figures 13 and 14. We can see that the DCB approach always achieves near-optimal TCR and AD, e.g., around 98% in TCR and 99% when $k = 4$. Besides, we also notice that APU is much affected when varying k for the GCA and DCB approaches, which shows that achieving high TCR should sacrifice much APU when the k is small.

Summary of our empirical study. First, decoupled modeling of the multiple factors, such as sequentiality, spatiality, and temporality, can help the model to capture the preference of workers more accurately. Second, for task recommendation, there is no approach whose performance can dominate the others on all metrics. The GCA approach, the ODA approach, and TOPK always perform the highest TCR, AD, and APU, respectively, compared with others. Third, the DCB approach is a method for balancing the performance of TCR and AD, which can obtain around 98% of the optimal AD and near-

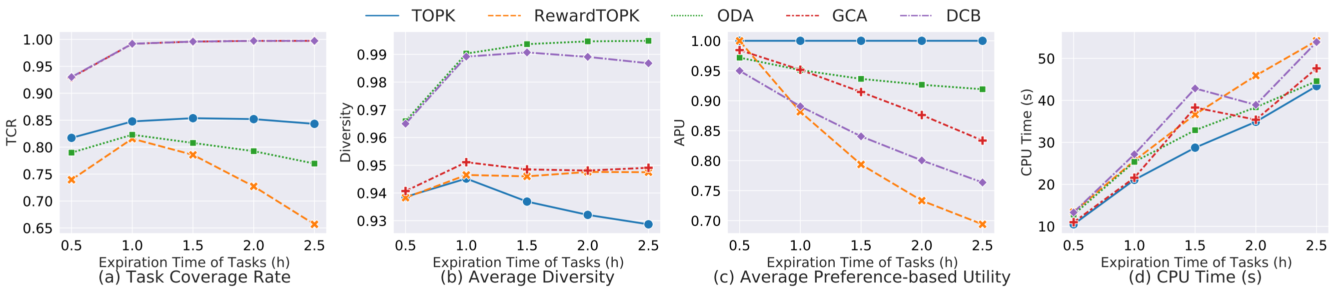


Fig. 9. Effect of e on NYC

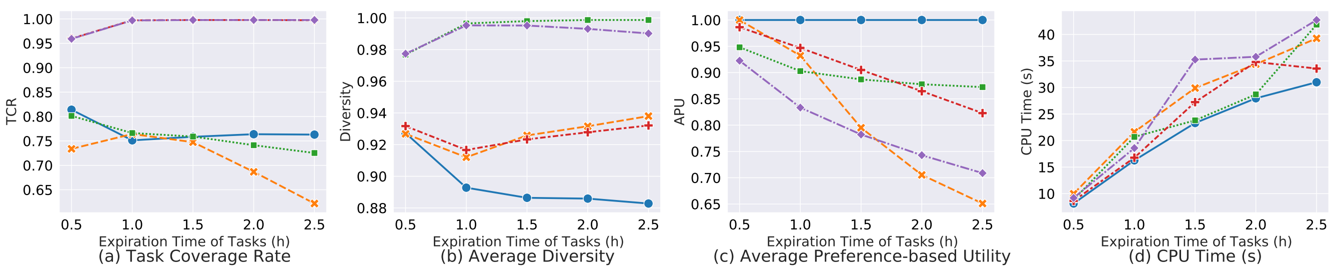


Fig. 10. Effect of e on TKY

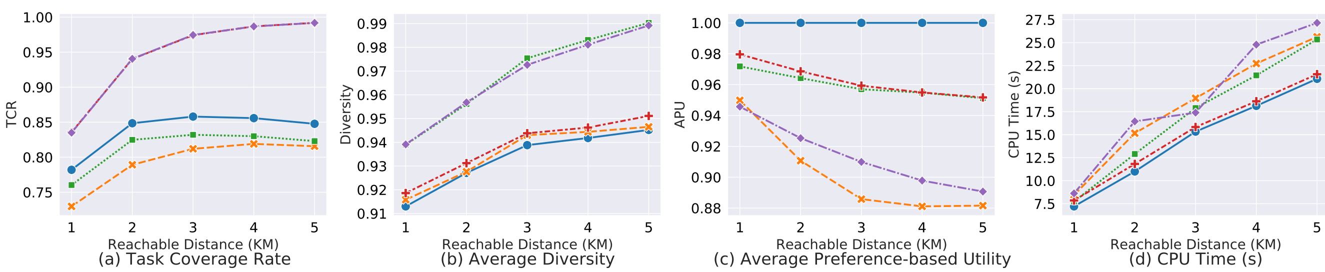


Fig. 11. Effect of d on NYC

optimal TCR with sacrificing slight APU. Our proposed methods can be applied to different applications according to their requirements.

V. Related Works

Spatial Crowdsourcing (SC) engages individuals (i.e., workers) to collect and process social, environmental, and other information with spatio-temporal features [2], reducing the production cost and making the data collection efficient and smart, where workers perform spatial tasks that involve traveling to specified locations [19], [24]–[45]. According to the way of task assignment to workers, SC can be classified into Server Assigned Tasks (SAT) mode and Worker Selected Tasks (WST) mode [1], [46]. SAT is the most widely-adopted assumption in current studies [19], [27], [30], [47]–[53], in which the server (i.e., the SC platform) takes charge of the task assignment. For example, Zhao et al. propose a preference-aware task assignment method that maximizes the total number of task assignments [6]. Lai et al. design a prediction model to forecast the loyalty of workers and a loyalty-aware Kuhn-Munkras (KM) algorithm to maximize the overall rewards of workers [30]. Tong et al. focus on the global online micro-task allocation problem in SC and propose a two-phase-based framework to maximize the total utility under a set of constraints [50]. Cheng et al. propose three effective approximation approaches, including greedy, sampling, and divide-and-conquer algorithms, to assign workers to spatial tasks such that the completion reliability and the spatial or temporal diversities of spatial tasks are maximized [47]. However, the flexibility of workers in this mode is limited (i.e., the SC server assigns tasks to workers compulsively without the consideration of the workers' willingness), which cannot stimulate the enthusiasm of the workers and thus affect the quality of the task result [1]. Therefore, we adopt the WST mode in this work, in which an online worker can select any tasks in their vicinity. However, the huge volume of tasks in a worker's vicinity brings challenges in terms of efficiency, i.e., a worker may cost much time to select the most suitable task to perform. To alleviate this problem, task recommendation plays a vital role in SC under the WST mode, which can help the worker quickly filter the tasks that are far from his satisfaction. For example, Chen et al. [54] propose a multi-agent task recommendation framework considering

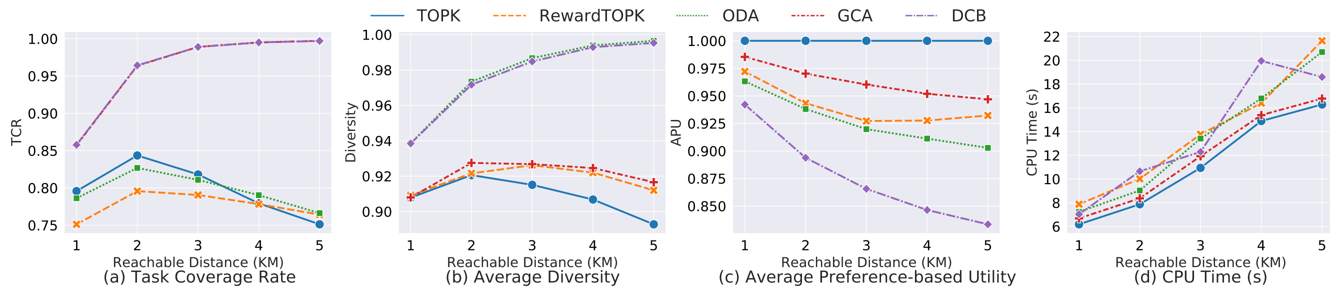


Fig. 12. Effect of d on TKY

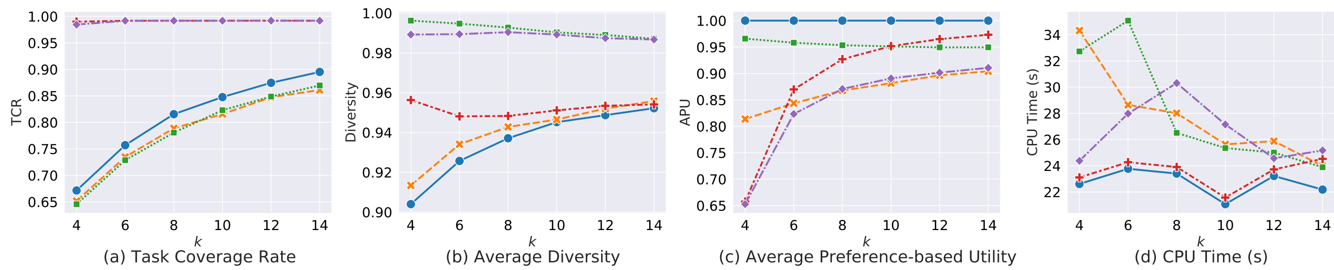


Fig. 13. Effect of k on NYC

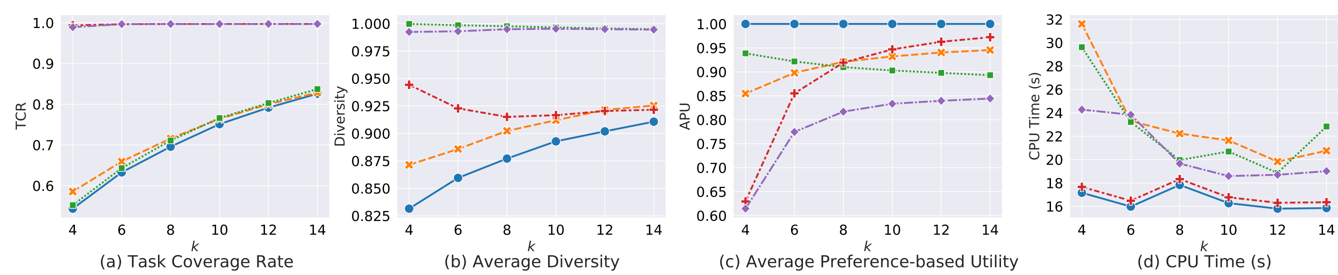


Fig. 14. Effect of k on TKY

stochastic spatio-temporal uncertainty. The task recommendation shares many common grounds with existing studies about POI recommender systems [11], [55], which models the workers' preference in terms of spatio-temporality and sequentiality. For instance, MobTCast [10] models the spatio-temporal behaviors of users through early-integration with POI embeddings. STiSAN [11] integrates the interval of space and time into SANs through the additive operation. However, such naive treatment may introduce noise [15] that achieves the sub-optimal performance in the modeling of preference accuracy. Besides, the task recommendation in SC usually considers the utility of workers [54], the task coverage rate, or the fairness of recommended tasks [1], which ignores the multiple interests of workers. To deal with that, in this work, the diversity in the recommended list of workers is taken into consideration.

VI. Conclusion

We propose a framework for task recommendation in spatial crowdsourcing, *DCBTaskRec*, which consists of two main phases, i.e., the workers' preference learning and the task recommendation. In the first phase, we propose a SANs-based model, DSTSAN, which decouples the sequentiality, spatiality, temporality, and dynamic interests of workers in the self-attention layer and adaptively aggregates them through a vanilla attention layer to model the different contributions of multiple aspects on the workers' preference. For task recommendation, we design three approaches, i.e., ODA, GCA, and DCB, to meet the needs of different applications, i.e., maximizing the diversity in worker's recommended list, maximizing task coverage rate, and achieving a trade-off between coverage and diversity. An empirical study with real data offers evidence that the framework is capable of advancing the state of the art in terms of preference learning accuracy, diversity in workers' recommended list, and task coverage rate of recommended tasks with limited k.

VII. Acknowledgement

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, 2023D044), and Key Laboratory of Data Intelligence and Cognitive Computing, Longhua District, Shenzhen.

REFERENCES

[1] Y. Zhao, L. Deng, and K. Zheng, "Adataskrec: An adaptive task recommendation framework in spatial crowdsourcing," *TOIS*, 2023.

[2] A. Alamer, J. Ni, X. Lin, and X. Shen, "Location privacy-aware task recommendation for spatial crowdsourcing," *WCSP*, pp. 1–6, 2017.

[3] D. Gao, Y. Tong, J. She, T. Song, L. Chen, and K. Xu, "Top-k team recommendation and its variants in spatial crowdsourcing," *DSE*, vol. 2, pp. 136–150, 2017.

[4] D. Gao, Y. Tong, J. She, T. Song, L. Chen, and K. Xu, "Top-k team recommendation in spatial crowdsourcing," in *WAIM*, 2016.

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

[6] Y. Zhao, J. Xia, G. Liu, H. Su, D. Lian, S. Shang, and K. Zheng, "Preference-aware task assignment in spatial crowdsourcing," in *AAAI*, 2019.

[7] M. C. González, C. A. Hidalgo, and A. L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 779–782, 2008.

[8] Y. Zhao, S. Shang, Y. Wang, B. Zheng, Q. V. H. Nguyen, and K. Zheng, "Rest: A reference-based framework for spatio-temporal trajectory compression," *KDD*, 2018.

[9] K. Zheng, Y. Zhao, D. Lian, B. Zheng, G. Liu, and X. Zhou, "Reference-based framework for spatio-temporal trajectory compression and query processing," *TKDE*, vol. 32, pp. 2227–2240, 2020.

[10] H. Xue, F. D. Salim, Y. Ren, and N. Oliver, "Mobtcast: Leveraging auxiliary trajectory forecasting for human mobility prediction," in *NIPS*, 2021.

[11] E. Wang, Y. Jiang, Y. Xu, L. Wang, and Y. Yang, "Spatial-temporal interval aware sequential poi recommendation," *ICDE*, pp. 2086–2098, 2022.

[12] M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola, "Cofi rank - maximum margin matrix factorization for collaborative ranking," in *NIPS*, 2007.

[13] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *WWW*, 2005.

[14] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," *ICDM*, pp. 197–206, 2018.

[15] X. Fan, Z. Liu, J. Lian, W. X. Zhao, X. Xie, and J.-R. Wen, "Lighter and better: Low-rank decomposed self-attention networks for next-item recommendation," *SIGIR*, 2021.

[16] X. Rao, L. Chen, Y. Liu, S. Shang, B. Yao, and P. Han, "Graph-flashback network for next location recommendation," *KDD*, 2022.

[17] D. Yang, B. Fankhauser, P. Rosso, and P. Cudré-Mauroux, "Location prediction over sparse user mobility traces using rnns: Flashback in hidden states!" in *IJCAI*, 2020.

[18] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns," *TSMC*, vol. 45, pp. 129–142, 2015.

[19] Y. Zhao, Y. Li, Y. Wang, H. Su, and K. Zheng, "Destination-aware task assignment in spatial crowdsourcing," *CIKM*, 2017.

[20] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *ICLR*, 2016.

[21] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: Short-term attention/memory priority model for session-based recommendation," *KDD*, 2018.

[22] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *AAAI*, vol. 33, no. 01, 2019, pp. 346–353.

[23] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," *CIKM*, 2017.

[24] B. Li, Y. Cheng, Y. Yuan, G. Wang, and L. Chen, "Three-dimensional stable matching problem for spatial crowdsourcing platforms," *KDD*, 2019.

[25] X. Chen, Y. Zhao, and K. Zheng, "Task publication time recommendation in spatial crowdsourcing," *CIKM*, 2022.

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

[27] Y. Cui, L. Deng, Y. Zhao, B. Yao, V. W. Zheng, and K. Zheng, "Hidden poi ranking with spatial crowdsourcing," *KDD*, 2019.

[28] B. Guo, Y. Liu, L. Wang, V. O. K. Li, J. C. K. Lam, and Z. Yu, "Task allocation in spatial crowdsourcing: Current state and future directions," *IoTJ*, vol. 5, pp. 1749–1764, 2018.

[29] D. Hettiachchi, V. Kostakos, and J. Gonçalves, "A survey on task assignment in crowdsourcing," *CSUR*, vol. 55, pp. 1 – 35, 2021.

[30] T. H. M. Lai, Y. Zhao, W. Qian, and K. Zheng, "Loyalty-based task assignment in spatial crowdsourcing," *CIKM*, 2022.

[31] L. Zheng and L. Chen, "Maximizing acceptance in rejection-aware spatial crowdsourcing," *ICDE*, pp. 71–72, 2017.

[32] L. Zheng and L. Chen, "Multi-campaign oriented spatial crowdsourcing," *ICDE*, pp. 1248–1251, 2018.

[33] Y. Yang, Y. Cheng, Y. Yuan, G. Wang, L. Chen, and Y. Sun, "Privacy-preserving cooperative online matching over spatial crowdsourcing platforms," *VLDB*, vol. 16, pp. 51–63, 2022.

[34] 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*, vol. 34, no. 7, pp. 3461–3477, 2022.

[35] Y. Zhao, K. Zheng, Y. Li, J. Xia, B. Yang, T. B. Pedersen, R. Mao, C. S. Jensen, and X. Zhou, "Profit optimization in spatial crowdsourcing: Effectiveness and efficiency," *TKDE*, 2022.

[36] 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*, 2021, pp. 265–276.

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

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

[39] 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*, 2020, pp. 13–24.

[40] 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*, pp. 2336–2350, 2019.

[41] 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*, 2022.

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

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

[44] Y. Zhao, K. Zheng, Z. Wang, L. Deng, B. Yang, T. B. Pedersen, C. S. Jensen, and X. Zhou, "Coalition-based task assignment with priority-aware fairness in spatial crowdsourcing," *VLDBJ*, 2023.

[45] Y. Zhao, T. Lai, Z. Wang, K. Chen, H. Li, and K. Zheng, "Worker-churn-based task assignment with context-lstm in spatial crowdsourcing," *TKDE*, 2023.

[46] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," *SIGSPATIAL*, 2012.

[47] P. Cheng, X. Lian, Z. Chen, L. Chen, J. Han, and J. Zhao, "Reliable diversity-based spatial crowdsourcing by moving workers," *VLDB*, vol. 8, pp. 1022–1033, 2014.

[48] P. Cheng, X. Lian, L. Chen, J. Han, and J. Zhao, "Task assignment on multi-skill oriented spatial crowdsourcing," *TKDE*, vol. 28, pp. 2201–2215, 2015.

[49] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," *ICDE*, pp. 997–1008, 2015.

[50] Y. Tong, Y. Zeng, B. Ding, L. Wang, and L. Chen, "Two-sided online micro-task assignment in spatial crowdsourcing," *TKDE*, vol. 33, pp. 2295–2309, 2021.

[51] Y. Tong, Y. Zeng, Z. Zhou, L. Chen, and K. Xu, "Unified route planning for shared mobility: An insertion-based framework," *TODS*, vol. 47, pp. 1 – 48, 2022.

[52] Y. Tong, Z. Zhou, Y. Zeng, L. Chen, and C. Shahabi, "Spatial crowdsourcing: a survey," *VLDBJ*, vol. 29, pp. 217 – 250, 2019.

[53] Y. Xu, Y. Tong, Y. Shi, Q. Tao, K. Xu, and W. Li, "An efficient insertion operator in dynamic ridesharing services," *ICDE*, pp. 1022–1033, 2019.

[54] C. Chen, S.-F. Cheng, H. C. Lau, and A. Misra, "Towards city-scale mobile crowdsourcing: Task recommendations under trajectory uncertainties," in *IJCAI*, 2015.

[55] Y. Luo, Q. Liu, and Z. Liu, "Stan: Spatio-temporal attention network for next location recommendation," *WWW*, 2021.