AdaTaskRec: An Adaptive Task Recommendation Framework in Spatial Crowdsourcing

YAN ZHAO, Department of Computer Science, Aalborg University, Denmark

LIWEI DENG, School of Computer Science and Engineering, University of Electronic Science and Technology of China, China

KAI ZHENG*, Yangtze Delta Region Institute (Quzhou), School of Computer Science and Engineering, Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, China

Spatial crowdsourcing is one of the prime movers for the orchestration of location-based tasks, and task recommendation is a crucial means to help workers discover attractive tasks. While a number of existing studies place their focuses on modeling workers' geographical preferences in task recommendation, they ignore the phenomenon of workers' travel intention drifts across geographical areas, i.e., workers tend to have different intentions when they travel in different areas, which discounts the task recommendation quality of existing methods especially for workers that travel in unfamiliar out-of-town areas. To address this problem, we propose an Adaptive Task Recommendation (*AdaTaskRec*) framework. Specifically, we first give a novel two-module worker preference learning architecture that can calculate workers' preferences for POIs (that tasks are associated with) in different areas adaptively based on workers' current locations. If we detect that a worker is in the hometown area, we apply the hometown preference learning module, which hybrids different strategies to aggregate workers' travel intentions into their preferences while considering the transition and the sequence patterns among locations. Otherwise, we invoke the out-of-town preference learning module, which is to capture workers' preferences by learning their travel intentions and transferring their hometown preferences into their out-of-town ones. Additionally, to improve task recommendation effectiveness, we propose a dynamic top-k recommendation method that sets different k values dynamically according to the numbers of neighboring workers and tasks. We also give an extra-reward-based and a fair top-k recommendation method, which introduce the extra rewards for tasks based on their recommendation rounds and consider exposure-based fairness of tasks, respectively. Extensive experiments offer insight into the effectiveness of the proposed framework.

CCS Concepts: • **Information systems** → **Location based services**; • **Computing methodologies** → *Machine learning*; • **Human-centered computing** → *Empirical studies in collaborative and social computing*.

Additional Key Words and Phrases: task recommendation, travel intention, spatial crowdsourcing

ACM Reference Format:

Yan Zhao, Liwei Deng, and Kai Zheng. 2018. AdaTaskRec: An Adaptive Task Recommendation Framework in Spatial Crowdsourcing. In . ACM, New York, NY, USA, 33 pages. https://doi.org/XXXXXXX.XXXXXXX

*Corresponding author.

1 INTRODUCTION

Along with the ubiquity of GPS-equipped, networked devices and the accompanying deployment of sensing technologies, Spatial Crowdsourcing (SC) has gained increasing popularity, where task requesters can issue spatial tasks (e.g., taking a scenic photo or reporting a hot spot) to SC servers and a crowd of mobile workers are engaged to provide pervasive and cost-effective services to finish these spatial tasks by physically moving to the specified locations. To ensure the quality of SC services, SC servers can recommend spatial tasks to workers based on their context information (e.g., spatio-temporal information) extracted from their interactions with tasks and smartphone sensors, where spatio-temporal information (e.g., location and mobility) plays an essential role in SC. Compared with mandatory task assignment, i.e., assigning a task to each worker at a time and the worker is forced to perform the assigned task, a task recommendation system provides a list of potential and available tasks that are more likely to be accepted by the worker. The worker can select the most interested one from the recommended task list, which can ensure continuous and high worker participation and satisfaction to some extent. Due to its natural connection to the physical world, SC is relevant to a wide spectrum of daily applications, which need specialized algorithms to accomplish effective task recommendation.

Existing studies on SC [2, 4, 13, 14] have contributed many techniques for task recommendation in different application scenarios. They have explored approaches to provide workers with better opportunities to obtain information when choosing tasks. For example, leveraging the designed privacy-preserving location matching mechanism, Alamer et al. propose a location privacy-aware task recommendation framework, achieving secure task recommendation while protecting location privacy for workers [2]. Chen et al. propose a stochastic task recommendation framework, which considers workers' historical trajectories, desired time budgets, and the inherent probabilistic uncertainty about their future trajectories [4]. Gao et al. study a top-k team recommendation problem and design a two-level-based framework that recommends some suitable teams to crowd workers to satisfy the skill requirements of complex tasks [13, 14]. However, existing studies focus mainly on task recommendation in an area (called hometown) where workers perform daily activities or on traditional top-1 and top-k recommendation methods, and thus leave challenges related to effective task recommendation largely unaddressed. We face three main challenges.

Challenge I: How to model workers' preferences for spatial tasks adaptively when workers are in different areas? Unlike traditional recommendation systems, workers do not provide their ratings on tasks in SC systems, and thus we need to transform workers' travel or task-performing behavior into ratings (e.g., preference scores). Most SC studies put their focuses on modeling workers' preferences in a daily activity area (i.e., hometown) [2, 4, 57, 62]. Inevitably, however, a worker may travel to a new place (i.e., an out-of-town area) since human mobility has a high degree of freedom and variation [7, 40], where the worker has little or no knowledge about the location-based tasks in this area. This leads to a new problem, namely out-of-town task recommendation, which aims to find tasks that a worker may be interested in when the worker travels out of the hometown. To be specific, out-of-town task recommendation is designed for those workers who travel from their hometown areas to out-of-town areas they have seldom been to before. Individual workers' hometown preferences cannot be directly used for the task recommendation when workers travel in unfamiliar out-of-town areas due to the gap between hometown preferences and out-of-town behavior (i.e., travel

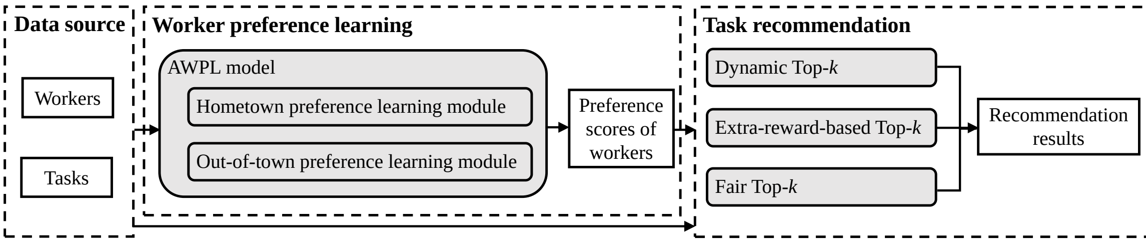


Fig. 1. Framework Overview

intention drifts). Therefore, the crucial question that needs to be answered is how to consider worker's premises and still keep flexible.

Challenge II: How to recommend tasks to workers dynamically to achieve high task coverage rate? Considering that the main common goals in SC are to achieve the maximal number of completed tasks and to satisfy workers, a good task recommendation system needs to maximize the coverage rate of the recommended tasks with a limited k value while keeping high preference-based utility of workers, where k is the number of recommended tasks for each worker. k should be limited since a high k value increases difficulty of task selection for workers. Existing studies in SC generally ignore the dynamic spatio-temporal distributions of workers and tasks, which can often lead to poor recommendation results. For instance, if we use a traditional top-k recommendation method in which k is fixed and the recommendation failure of some tasks is ignored, some tasks are unlikely to be recommended/exposed to workers, which results in a low task coverage rate.

Challenge III: How to take fairness into consideration when recommending tasks, ensuring that task recommendation does not discriminate against certain tasks or task requesters? Traditional recommendation methods have focused on maximizing worker/user satisfaction by tailoring the results according to the personalized preferences of individual workers/users, largely ignoring the interest of task requesters. Several item recommendation studies that recommend suitable items to users have shown that such user-centric designs may undermine the well-being of item providers [1, 3, 11, 17, 20]. In SC, the competitive relationship between tasks requires a fair way to allocate the exposure of tasks to workers. Unfair allocation-of-exposure of tasks can cause the Matthew effect [17, 37], which means that the high-ranked tasks and their associated points of interest (POIs) are more likely to gain additional attentions to influence future rankings, while low-ranked tasks and their associated POIs will be marginalized gradually. Typically, an SC platform recommends the k most relevant/interested tasks to the corresponding workers [4, 13, 14]. While such top-k recommendation methods achieve high worker utility, they may affect the fairness of tasks being recommended/exposed to workers negatively, which leads to the turnover of task requesters.

Observing these unmet challenges, this paper will go beyond the state of the art and develop an SC framework, called Adaptive Task Recommendation (*AdaTaskRec*), for effective task recommendation by considering workers' travel-intention-based preferences, the dynamic numbers of workers and tasks, and the exposure-based fairness among tasks. The *AdaTaskRec* framework consists of two phases, i.e., a worker preference learning and a task recommendation phase, as shown in Figure 1. In the first phase, we design an Adaptive Worker Preference Learning (AWPL) model that learns worker preferences for tasks adaptively based on workers' current locations, where each task is associated with a POI. Considering that tasks (that are often micro-tasks [41]) are highly dependent on locations and workers tend to perform tasks around the

POIs that will be visited, learning workers' preferences for various tasks becomes identical with learning their preferences for the corresponding POIs that tasks are located at. Therefore, AWPL aims to learn worker preferences for POIs that tasks are associated with. To be specific, AWPL consists of two modules, a hometown and an out-of-town preference learning module. The first module hybrids different machine learning techniques to model workers' hometown preferences for different POIs. It adopts gated Graph Neural Network (GNN) and Gated Recurrent Unit (GRU) to model the transition and the sequence patterns among POIs, respectively, and then calculate workers' hometown preferences with travel intentions that are captured by a neural topic model. The second module learns workers' out-of-town preferences by combining workers' preference embeddings obtained by an attention mechanism, travel intentions captured by an improved neural topic model, and geographical distance embeddings of POIs, and meanwhile transferring workers' hometown preferences into their out-of-town behavior.

In the task recommendation phase, to maximize the coverage rate of recommended tasks with a limited k value as well as keep high preference-based utility of workers, we propose a dynamic top-k and an extra-reward-based top-k method, which consider the dynamic number distributions of workers and tasks, and the round-based extra reward, respectively. To enable fair recommendation, we consider a notion of fairness, task exposure-based fairness, and design a fair top-k method for achieving the long-term sustainability of SC platforms. It is worth mentioning that each recommendation method has its own superiority in terms of the coverage rate of recommended tasks, the average k value, the average preference-based utility of workers, and the exposure-based fairness of tasks, as demonstrated in the experimental evaluation (cf. Section 5). Therefore, these methods can be chosen according to different application requirements.

Our contributions can be summarized as follows:

1) We propose a task recommendation framework for SC, called Adaptive Task Recommendation (*AdaTaskRec*), that considers workers' travel-intention-based preferences in task recommendation.

2) We propose an adaptive worker preference learning model, which calculates workers' hometown and out-of-town preferences adaptively according to workers' current locations (solving *Challenge I*).

3) Three strategies are given that consider different task recommendation concerns, i.e., the coverage rate of recommended tasks, the average k value, the preference-based utility of workers, and the exposure-based fairness of tasks, solving *Challenges II* and *III*.

4) We report on experiments using real data, offering evidence of the effectiveness of the proposal.

The remainder of the paper is organized as follows. Section 2 covers the problem statement, and Section 3 details the worker preference learning architecture. We present the task recommendation algorithms in Section 4, followed by a coverage of experimental results in Section 5. Section 6 surveys related work, and Section 7 concludes the paper.

2 PROBLEM STATEMENT

We proceed to give necessary preliminaries and then define the problem addressed. Table 1 shows the notation used throughout the paper.

DEFINITION 1 (WORKER). *A worker, denoted as $w = (l, d)$, is able to perform spatial tasks. A worker can be in 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*

Table 1. Summary of Notation

Symbol	Definition
w	Worker
$w.l$	Location of worker w
$w.d$	Reachable distance of worker w
p	POI
$p.l$	Location of POI p
$p.S$	The set of tasks associated with p
s	Spatial task
$s.p$	POI where s is located
$s.e$	Expiration time of s
$s.r$	Reward of s
$s.l$	Location of s
$RS(w)$	Reachable task set of worker w
$t(a,b)$	Travel time from location a to location b
$d(a,b)$	Travel distance from location a to location b
$RTS_k(w)$	k-recommended-task-set of worker w
$U(RTS_k(w))$	Preference-based utility of worker w
$c_w(s)$	Worker w's preference score for task s
R	A spatial task recommendation
$R.C$	Task coverage rate for task recommendation R
$\mathbb{R}$	A spatial task recommendation set
R_{opt}	Optimal task recommendation

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 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 $s = (p, e, r)$, encompasses a POI $s.p$, a task expiration deadline $s.e$, and a reward $s.r$ that the worker completing s will obtain.*

A spatial task s can be finished only if a worker physically moves to its location (i.e., the location of $s.p$) before the expiration time. For simplicity, we use $s.l$ to denote the location of $s.p$, i.e., the location of task s. Next, a task s can be recommended to a worker only if the worker arrives at its location before its expiration time $s.e$. Although an SC server can recommend multiple tasks to a worker, a worker can only choose one task at a time according to the single task assignment mode [24]. Once a worker chooses a task to perform, the worker is offline until the task is finished.

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) The 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) The 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 task 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 worker w obtained from a recommended task set $RTS_k(w)$ is proportional to the sum of preference scores of workers for the tasks in $RTS_k(w)$. Therefore, the preferences of worker w for $RTS_k(w)$ can be used to derive the preference-based utility that worker w gains from the recommendation. Recommending the k most interested tasks will give the maximum possible utility.

Definition 6 (Worker Preference-based Utility). *The preference-based utility of worker w can be defined as the ranking metric, Normalized Discounted Cumulative Gain (NDCG) [23], over the recommended task set $RTS_k(w)$:*

$$U\left(RTS_k(w)\right) = \frac{DCG\left(RTS_k(w)\right)}{DCG\left(RTS_k^*(w)\right)}, \tag{1}$$

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

where $DCG(RTS_k(w))$ denotes the discounted cumulative gain of worker w over $RTS_k(w)$ that can be computed by Equation 2, $RTS_k^*(w)$ 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 $s \in RTS_k(w)$, and $rank(w, s \mid RTS_k(w))$ is the position that task s is placed at in the ranking task set $RTS_k(w)$ for w.

We use the terms "preference-based utility" and "utility" interchangeably when the context is clear.

Definition 7 (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: $(w_1, RTS_k(w_1))$, $(w_2, RTS_k(w_2))$,...,$(w_{|W|}, RTS_k(w_{|W|}))$, where $|W|$ denotes the number of the worker set.*

Let $R.C$ denote the task coverage rate for task recommendation R, which is the ratio between the number of recommended tasks and the total number of tasks, i.e., $R.C = \frac{|\cup_{w \in W} RTS_k(w)|}{|S|}$, and $\mathbb{R}$ denote all possible recommendations. The problem investigated can be stated as follows.

Problem Statement. Given a worker set W and a task set S, the task recommendation problem is to find an optimal task recommendation R_{opt} that achieves the following goals:

1) primary optimization goal: maximize the task coverage rate, i.e., $\forall\ R_i \in \mathbb{R}\ (R_i.C \leq R_{opt}.C)$, with a limited k ($k \ll |S|$) value; and

2) secondary optimization goal: maximize the average worker preference-based utility.

3 WORKER PREFERENCE MODELING

With the rapid growth of Location-Based Social Networks (LBSNs), it is now available to study and analyze workers' mobility behavior in real world, which helps to explore workers' preferences for tasks and task-performing behavior in SC. Due to the fact that SC is highly dependent on spatial information like mobility and spatial tasks are often micro-tasks (e.g., taking a scenic photo or reporting a hot spot), workers tend to perform tasks according to their locations, i.e., workers accept tasks according to whether the tasks are located in the interested POIs. In other words, workers often perform tasks around various POIs when

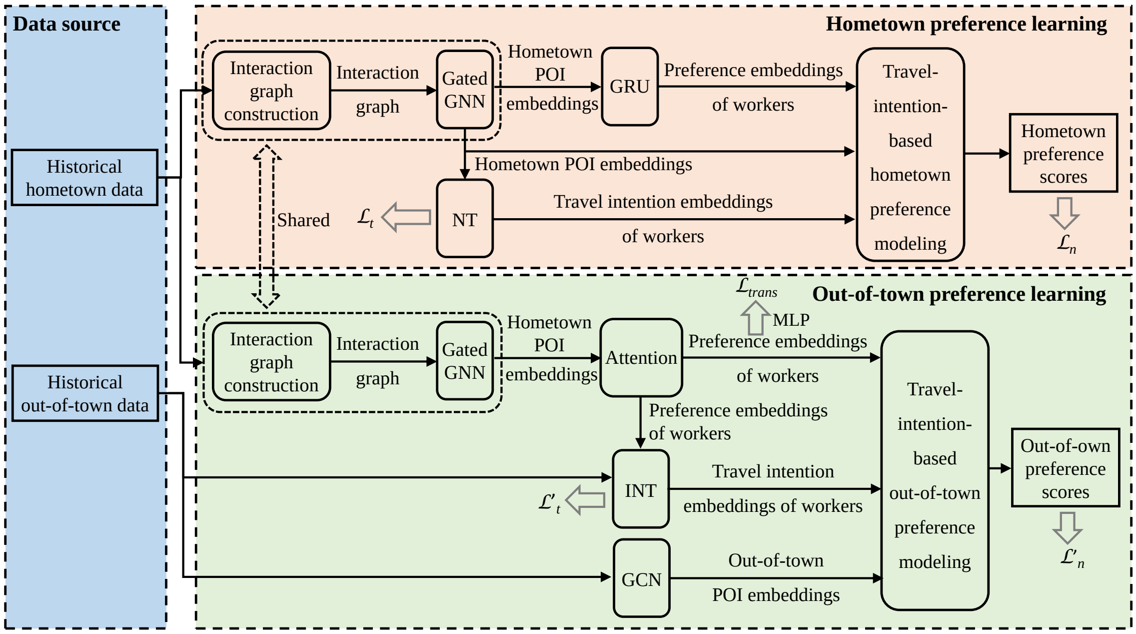


Fig. 2. AWPL Model Overview

they visit these POIs. Therefore, the preferences of workers for tasks can be regarded as those for POIs. When the context is clear, we use workers' preferences for POIs to denote workers' preferences for tasks that are associated with these POIs.

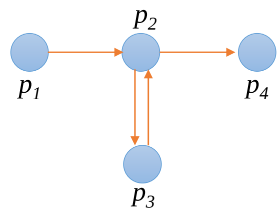
Studies [35, 50] show that people usually visit nearby POIs that are located in small regions, called hometown. However, due to the strong mobility characteristic of people, it is more likely for them to travel out of their hometown areas. Previous task recommendation systems [4, 13, 14] mainly focus on recommending tasks that may reside in a worker's hometown where the worker performs daily activities, which makes the recommendation results less useful. As a result, we propose an Adaptive Worker Preference Learning (AWPL) model, which aims to learn worker preferences for POIs based on whether workers travel in their hometown or out-of-town areas in order to recommend suitable tasks to workers. We first give an overview of the AWPL model and then provide specifics on each module in the model.

3.1 Solution Overview

The AWPL model overview is shown in Figure 2. It is a two-module architecture, where one is to learn workers' hometown preferences and the other is to learn their out-of-town preferences. We give a set of historical POIs where workers performed tasks, which include hometown and out-of-town POIs. In the hometown preference learning module, we first construct a worker-specific interaction graph based on the historical hometown POIs, which is fed into the gate Graph Neural Network (GNN) to get the POI embeddings by modeling the transition patterns among these POIs. Then we use the Gated Recurrent Units (GRU) to model the sequential patterns among POIs and learn workers' preference embeddings. Meanwhile, a Neural Topic (NT) model is adopted to capture the travel intention embeddings of workers. Based on workers' preference and travel intention embeddings as well as POI embeddings, we can get the

$$P_w = (p_1, p_2, p_3, p_2, p_4)$$

(a) POI Sequence of Worker w



(b) Interaction Graph

	Outgoing p_1	Outgoing p_2	Outgoing p_3	Outgoing p_4	Incoming p_1	Incoming p_2	Incoming p_3	Incoming p_4
p_1	0	1	0	0	0	0	0	0
p_2	0	0	1/2	1/2	1/2	0	1/2	0
p_3	0	1	0	0	0	1	0	0
p_4	0	0	0	0	0	1	0	0

(c) Matrix Concatenation

Fig. 3. Interaction Graph Construction

hometown preference scores of workers through the travel-intention-based hometown preference modeling. We combine two loss functions, i.e., the travel-intention-based loss $\mathcal{L}_t$ and the preference score estimation loss $\mathcal{L}_n$ generated from the NT model and the preference scores, respectively, to train this module.

The out-of-town preference module shares the same interaction graph and gate GNN, which aim to generate the embeddings of hometown POIs. Then we adopt an attention mechanism to aggregate the embeddings of these POIs to get the hometown preference embeddings of workers. An Improved NT (INT) model is designed to model the out-of-town travel intention embeddings of workers, and the Graph Convolutional Network (GCN) is introduced to model the geographical distance between out-of-town POIs, generating the out-of-town POI embeddings. Finally, the travel-intention-based out-of-town preference modeling part learns workers' out-of-town preferences by taking their preference and travel intention embeddings and out-of-town POI embeddings into account. This module considers three losses, i.e., the transfer loss $\mathcal{L}_{trans}$ (that is to transfer workers' hometown preferences to their out-of-town behavior by Multi-Layer Perceptron (MLP)), the travel-intention-based loss $\mathcal{L}'_t$, and the preference score estimation loss $\mathcal{L}'_n$, for training.

Note that the hometown and the out-of-town preference learning modules are trained separately, which do not affect each other. After training, the trained model is transferred to a preference learner. When a worker arrives, it first detects the current location of the worker and then feeds the worker into different modules based on whether the worker travels in hometown or out of town to get the worker's preference scores for different tasks.

3.2 Hometown Preference Learning

3.2.1 Interaction Graph Construction. A sequence of POIs where a worker performs tasks can be represented by a directed graph, where each node denotes a POI that tasks are associated with. The edge from node p_i (denoting POI p_i) to node p_j means that the worker performs tasks in p_j after completing tasks in p_i. Supposing that worker w visits a sequence of POIs, denoted as $\mathcal{P}_w = (p_1, p_2, p_3, p_2, p_4)$, the interaction graph G_w of w can be constructed, as shown in Figures 3(a)–3(b). Following the weight normalization principle [47], where the normalized weight of each edge is calculated as the occurrence of the edge divided by the out-degree/in-degree of that edge's start node, the outgoing (denoted as $\boldsymbol{A}_w^{out}$) and the incoming (denoted as $\boldsymbol{A}_w^{in}$) adjacent matrices are normalized around rows, respectively. Taking the outgoing matrix in Figure 3(c) as an example, the number of edges from p_2 to p_3 is 1 and the out-degree of p_2 is 2, so the normalized weight of the edge is $1/2$. Then we use the concatenation of the two matrices to represent the directed graph, i.e., $\boldsymbol{A}_w = [\boldsymbol{A}_w^{out}, \boldsymbol{A}_w^{in}]$. We use bold letters, e.g., $\boldsymbol{A}$ and $\boldsymbol{a}$, to denote matrices and vectors.

3.2.2 Transition Pattern Modeling with Gated GNN. First, we adopt a randomly initialized embedding matrix (i.e., $\boldsymbol{P}^0 = [\boldsymbol{p}_1^0, \boldsymbol{p}_2^0, \cdots, \boldsymbol{p}_N^0] \in \mathbb{R}^{N\times d}$) to convert the hometown POIs into d-dimensional embeddings, where N denotes the whole number of hometown POIs, and $\boldsymbol{p}_i^0 \in \mathbb{R}^d$ represents the embedding of POI p_i. Next, we feed the embeddings of POIs visited by worker w (i.e., $\boldsymbol{P}_w^0 = [\boldsymbol{p}_1^0, \boldsymbol{p}_2^0, \cdots, \boldsymbol{p}_n^0]$, where n is the number of POIs visited by w) and the concatenated matrix $\boldsymbol{A}_w$ into the gated GNN to model the complex transition patterns among the POIs, which can be described as follows.

$$
\begin{aligned}
\boldsymbol{a}_i^t &= \boldsymbol{A}_{p_i:}^T [\boldsymbol{p}_1^{t-1}, \boldsymbol{p}_2^{t-1}, \cdots, \boldsymbol{p}_n^{t-1}]^T \boldsymbol{H} + \boldsymbol{b}_a, \\
\boldsymbol{z}_i^t &= \sigma(\boldsymbol{\mathcal{W}}_z \boldsymbol{a}_i^t + \boldsymbol{U}_z \boldsymbol{p}_i^{t-1}), \\
\boldsymbol{r}_i^t &= \sigma(\boldsymbol{\mathcal{W}}_r \boldsymbol{a}_i^t + \boldsymbol{U}_r \boldsymbol{p}_i^{t-1}), \\
\tilde{\boldsymbol{p}}_i^t &= tanh(\boldsymbol{\mathcal{W}}_o \boldsymbol{a}_i^t + \boldsymbol{U}_o(\boldsymbol{r}_i^t \odot \boldsymbol{p}_i^{t-1})), \\
\boldsymbol{p}_i^t &= (1 - \boldsymbol{z}_i^t) \odot \boldsymbol{p}_i^{t-1} + \boldsymbol{z}_i^t \odot \tilde{\boldsymbol{p}}_i^t,
\end{aligned} \tag{3}
$$

where $\boldsymbol{a}_i^t$ denotes an aggregated embedding of p_i's neighbors based on G_w from the previous step, $\boldsymbol{A}_{p_i:}^T$ denotes the two columns of blocks in $\boldsymbol{A}_w^{out}$ and $\boldsymbol{A}_w^{in}$ corresponding to p_i, $[\boldsymbol{p}_1^{t-1}, \boldsymbol{p}_2^{t-1}, \cdots, \boldsymbol{p}_n^{t-1}]$ is a sequence of POI embeddings in the historical data of worker w, $\boldsymbol{p}_j^{t-1}$ $(1 \leq j \leq n)$ denotes the embedding of p_j in the $(t-1)$th step $(t \geq 1)$, and $\boldsymbol{H}$, $\boldsymbol{b}_a$, $\boldsymbol{\mathcal{W}}_*$ and $\boldsymbol{U}_*$ $(* = z, r, o)$ are trainable parameters. Next, $\boldsymbol{z}_i^t$ and $\boldsymbol{r}_i^t$ are the update and reset gates, respectively, which are used for controlling the information flow process, $\sigma(\cdot)$ is a sigmoid activation function, and $\odot$ represents the element-wise multiplication operator. The embedding $\boldsymbol{p}_i^t$ can be calculated through the previous embedding (i.e., $\boldsymbol{p}_i^{t-1}$) and the fusion of the current aggregated embedding (i.e., $\tilde{\boldsymbol{p}}_i^t$). Finally, the learned hometown POI embeddings of worker w are denoted by $\boldsymbol{P}_w = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_n]$.

3.2.3 Preference Summarization with GRU. Although gated GNN is proven to be helpful for capturing transition patterns among locations [47, 50], the sequential patterns among locations cannot be well modelled. Moreover, since we focus on the preference abstracted by historical visited POIs, we need to adopt a method to summarize the historical POI embeddings into a worker's hometown preference embedding. Bearing these in mind, we adopt GRU [8], a well-known sequential model that considers temporal features using memory cell units, to learn worker w's hometown preference embedding, $\boldsymbol{w}$.

$$\boldsymbol{w} = GRU([\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_n]), \tag{4}$$

where $[\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_n]$ is the input hometown POI embeddings learned from gated GNN.

3.2.4 Travel Intention Modeling with Neural Topic (NT) Model. For better understanding a worker's mobility patterns, we give a Neural Topic (NT) model to discover travel intention distribution of the worker. We assume that each POI visit is generated by a latent topic mixture $\Theta \in \mathbb{R}^{M\times d}$, where M stands for the number of travel latent intentions and each row in Θ is a to-be-learnt vector representing the features of each travel intention. Given the randomly initialized hometown POI embeddings $\boldsymbol{P}^0 \in \mathbb{R}^{N\times d}$, the i-th travel intention distribution on the POIs, denoted as Φ_i, can be described as follows.

$$\Phi_i = softmax(\boldsymbol{P}^0 \Theta_i), \tag{5}$$

where $\Phi_i \in \mathbb{R}^N$, Θ_i is the i-th row of Θ, and $softmax(\cdot)$ is a function to convert a vector of N real values into another vector of N values that sum to 1.

Then, to capture the travel intentions of worker w, we convert the POIs into a bag-of-words vector $\boldsymbol{g}_w \in \mathbb{R}^N$, where each entry in $\boldsymbol{g}_w$ denotes the number of visits. For instance, the entry $\boldsymbol{g}_w[p_j]$ equals to 3 if and only if w visited p_j three times before. Following the study [50], we hypothesize that the latent topic mixture can be generated by Gaussian softmax construction, which means that a worker's latent travel intentions (i.e., $\boldsymbol{x}_w$) are drawn from the standard Gaussian distribution (i.e., $\boldsymbol{x}_w \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$). To make $\boldsymbol{x}_w$ traceable, the variational posterior distribution is adopted, as shown in Equation 6.

$$q(\boldsymbol{x}_w|\boldsymbol{g}_w) = N(\mu_w, \sigma_w^2), \tag{6}$$

where $\mu_w \in \mathbb{R}^M$ and $\sigma_w^2 \in \mathbb{R}^M$ are the mean and variance of the Gaussian distribution, respectively, which are induced by the worker's bag-of-words vector (e.g., $\boldsymbol{g}_w$), as shown in the following equations.

$$\begin{aligned} \tilde{\boldsymbol{g}}_w &= F_{en}(\boldsymbol{g}_w), \\ \mu_w &= F_\mu(\tilde{\boldsymbol{g}}_w), \\ \log \sigma_w^2 &= F_\sigma(\tilde{\boldsymbol{g}}_w), \end{aligned} \tag{7}$$

where $\tilde{\boldsymbol{g}}_w \in \mathbb{R}^d$ is the encoded embedding of $\boldsymbol{g}_w$, and $F_{en}(\cdot)$, $F_\mu(\cdot)$, and $F_\sigma(\cdot)$ represent the encoder, the mean, and the variance layer, respectively, all of which are two-layer Multi-Layer Perceptrons (MLPs) with ReLU activation. After obtaining the posterior distribution $q(\boldsymbol{x}_w|\boldsymbol{g}_w)$, we adopt the reparameterization trick to resample $\boldsymbol{x}_w \in \mathbb{R}^M$ from this distribution. Then, we calculate worker w's latent topic distribution (i.e., the travel intention embedding), $\boldsymbol{u}_w \in \mathbb{R}^M$, as follows.

$$\boldsymbol{u}_w = softmax(\boldsymbol{\mathcal{W}}_u \boldsymbol{x}_w + \boldsymbol{b}_u), \tag{8}$$

where $\boldsymbol{\mathcal{W}}_u \in \mathbb{R}^{M \times M}$ and $\boldsymbol{b}_u \in \mathbb{R}^M$ are the trainable parameters.

3.2.5 Travel-intention-based Hometown Preference Modeling. To obtain the preference of a worker w on howntown POIs with travel intentions, we concatenate the travel intention embedding (i.e., $\boldsymbol{u}_w$) and the worker's preference embedding (i.e., $\boldsymbol{w}$), as shown in Equation 9.

$$\boldsymbol{r}_w = \boldsymbol{\mathcal{W}}_r[\boldsymbol{u}_w, \boldsymbol{w}] + \boldsymbol{b}_r, \tag{9}$$

where $\boldsymbol{\mathcal{W}}_r \in \mathbb{R}^{M+d}$ and $\boldsymbol{b}_r \in \mathbb{R}^d$ are the trainable parameters, which are used to map the concatenated embedding into a d-dimensional space, and $\boldsymbol{r}_w$ is the travel-intention-based preference embedding of worker w.

We calculate the similarity between the travel-intention-based preference embedding of worker w and the embeddings of POIs to obtain the worker's hometown preference scores $\boldsymbol{c}_w$ for POIs.

$$\boldsymbol{c}_w = softmax(\boldsymbol{P}^0 \boldsymbol{r}_w), \tag{10}$$

where $\boldsymbol{c}_w \in \mathbb{R}^N$ is a vector denoting w's preference scores for POIs, the entry $\boldsymbol{c}_w[p_j]$ represents the preference score of w for POI p_j (i.e., the preference score of w for tasks located in p_j), and $\boldsymbol{P}^0$ denotes the whole hometown POI embeddings, which are same with the input of gated GNN. The higher the preference score $\boldsymbol{c}_w[p_j]$ is, the more likely it is that w is willing to perform tasks in the location of p_j.

3.2.6 Training. For training the model, we linearly combine two different loss functions (i.e., travel-intention-based loss $\mathcal{L}_t$ and preference score estimation loss $\mathcal{L}_n$) with a predefined balance weight α as follows.

$$\mathcal{L} = \alpha \mathcal{L}_t + \mathcal{L}_n, \tag{11}$$

where $\mathcal{L}_t$ is to reveal the latent travel intentions of workers from their historical visited POIs, and $\mathcal{L}_n$ is to model workers' preferences through fitting the predicted preference scores into the ground-truths. We formally define $\mathcal{L}_t$ and $\mathcal{L}_n$ in Equations 12 and 13, respectively.

$$\mathcal{L}_t = -\sum_{w \in W} \Big(\mathbb{E}_{q(\boldsymbol{x}_w|\boldsymbol{g}_w)} \big(\boldsymbol{g}_w^T \log(\boldsymbol{u}_w \Phi) \big) + \mathbb{D}_{KL} \big(q(\boldsymbol{x}_w|\boldsymbol{g}_w) || p(\boldsymbol{x}_{\boldsymbol{w}}) \big) \Big), \tag{12}$$

$$\mathcal{L}_n = -\sum_{w \in W} \sum_{p \in P} \big(\mathcal{G}_w^p \log(\boldsymbol{c}_w[p]) \big), \tag{13}$$

where W and P denote the total workers and POIs, respectively. The first term in Equation 12 is the reconstruction error of historical POIs by travel intention modeling with the NT model. Next, $\mathbb{D}_{KL}(\cdot)$ is the Kullback-Leibler divergence, which is adopted to minimize the difference between the posterior distribution and the predefined prior standard Gaussian distribution, i.e., $p(\boldsymbol{x}_w) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Equation 13 shows the cross-entropy loss, in which the ground truth $\mathcal{G}_w^p$ equals to 1 if the next POI where worker w performs tasks is p; otherwise, it is 0. Next, $\boldsymbol{c}_w[p]$ is the estimated preference score of worker w for POI p.

3.3 Out-of-town Preference Learning

Recommending a task to a worker in the out-of-town area is more difficult than in the hometown area due to the data sparsity problem (i.e., a worker has seldom been to the out-of-town area) and travel intention drifts (i.e., the behavior of a worker may be different when the worker travels in an unfamiliar area). To alleviate these problems, we need to use the abundant visited POIs of workers in the hometown area to enhance the accuracy of out-of-town preference learning, i.e., transferring the learned hometown preferences into out-of-town preferences. Inspired by the success of TrainOR for out-of-town recommendation [50], we adopt it to learn the preferences of workers for out-of-town POIs. Specifically, this module also involves the interaction graph construction for the hometown POIs of workers and transition pattern modeling with gated GNN, which are same with Sections 3.2.1–3.2.2. One difference with the hometown preference learning module is that workers' preference embeddings are obtained by concatenating the summarized POI embeddings through an attention mechanism (instead of GRU), as shown in Figure 2. Then, an Improved Neural Topic (INT) model is applied on the visited out-of-town POIs to model the travel intention embeddings of workers. Moreover, we encode the geographical distances between out-of-town POIs into embeddings, and the preference scores can be calculated using the dot-product operator between the out-of-town POI embeddings and workers' preference embddings.

3.3.1 Preference Summarization with Attention Mechanism. Considering the effectiveness of the attention mechanism for summarizing the worker's embeddings from POIs [50], we adopt a vanilla attention method to aggregate the embeddings of hometown POIs of worker w as follows.

$$\begin{aligned} \beta_i &= \boldsymbol{q}^T \sigma(\boldsymbol{\mathcal{W}}_q \boldsymbol{p}_i + \boldsymbol{b}_q), \\ \boldsymbol{w} &= \sum_{p_i \in P_w} \beta_i \boldsymbol{p}_i, \end{aligned} \tag{14}$$

where $\boldsymbol{q}^T$, $\boldsymbol{\mathcal{W}}_q$, and $\boldsymbol{b}_q$ are used to compute the attention weight, $\boldsymbol{p}_i$ denotes the embedding of hometown POI p_i where worker w performed tasks, P_w denotes the set of hometown POIs that w visited, and $\boldsymbol{w}$ is the preference embedding of w. We should note that the training of hometown and out-of-town preference learning are separate. Thus, when the context is clear, we do not distinguish the symbols used in the two modules, which means that $\boldsymbol{w}$ is used to represent the preference embedding of worker w in both modules.

3.3.2 Out-of-town Travel Intention Modeling with Improved Neural Topic (INT) Model. For better modeling the out-of-town travel intentions of workers, we improve the original NT model proposed in Section 3.2.4 by two aspects, i.e., the construction of bag-of-words and the calculation of the travel intention embeddings. To be specific, the bag-of-words vector (i.e., $\boldsymbol{g}_w$) is obtained by the visited out-of-town POIs of worker w. For the calculation of travel intention embeddings, we do not directly use the latent topic distribution (i.e., $\boldsymbol{u}_w$ in Section 3.2.4) to denote worker w's travel intention embedding since it may result in the label leakage problem. This is because the visited POIs in the out-of-town area will be used as labels to train the preference learning model. Thus, we generate the travel intention embedding of worker w from the worker's preference embedding (i.e., $\boldsymbol{w}$) and the latent topic mixture (i.e., Θ) by using a cross attention mechanism, as shown in Equation 15.

$$\boldsymbol{u}_w = \mathit{softmax}(\Theta \boldsymbol{w})^T \Theta \tag{15}$$

We can see from the equation that the similarity between each latent travel intention and the workers' preferences (i.e., $\Theta\boldsymbol{w}$) is first computed, and then the similarity vector is converted into a distribution, where the i-th entry in this distribution represents the probability that the worker travels a POI with the ith latent travel intention. After that, we treat this distribution as a weighted vector to weight the whole latent travel intentions to generate the worker-specific travel intention embedding (i.e., $\boldsymbol{u}_w$).

3.3.3 Geographical Distance Modeling with Graph Convolutional Network (GCN). Since geographical distance between POIs is a key factor for boosting POI embeddings, we adopt GCN to model the geographical distance between POIs due to its effectiveness in modeling spatial data [53]. To achieve this, we first build an undirected weighted geometric graph, in which each node represents an out-of-town POI, and the edge weight is related to the distance of the connecting nodes. Specifically, we use an adjacent matrix $\boldsymbol{A}_g \in \mathbb{R}^{N' \times N'}$ to denote this graph, where N' denotes the number of distinct POIs in the out-of-town area. An entry of $\boldsymbol{A}_g$, $\boldsymbol{A}_g[i,j]$, can be computed as $\exp(-d(p_i.l, p_j.l))$, where $d(p_i.l, p_j.l)$ denotes the travel distance between the locations of POIs p_i and p_j. Based on $\boldsymbol{A}_g$, we use GCN to boost the POI embeddings in the following.

$$\boldsymbol{P}_{out} = ReLU(\boldsymbol{A}_g \boldsymbol{P}^0_{out} \boldsymbol{\mathcal{W}}_g + \boldsymbol{b}_g), \tag{16}$$

where $ReLU(x) = \max\{0, x\}$, $\boldsymbol{P}^0_{out} \in \mathbb{R}^{N' \times d}$ and $\boldsymbol{P}_{out} \in \mathbb{R}^{N' \times d}$ are the randomly initialized and the updated POI embeddings, respectively, and $\boldsymbol{\mathcal{W}}_g \in \mathbb{R}^{d \times d}$ and $\boldsymbol{b}_g \in \mathbb{R}^d$ are the trainable parameters.

3.3.4 Travel-intention-based Out-of-town Preference Modeling. Similar to the hometown preference modeling, we use a linear transform to convert the concatenation of the latent travel intention embedding of worker w (i.e., $\boldsymbol{u}_w$) and the worker's preference embedding (e.g., $\boldsymbol{w}$) into a d-dimensional space. Then the preference scores $\boldsymbol{c}_w$ between workers and out-of-town POIs can be obtained through the dot-product operator, shown in Equation 17.

$$\begin{aligned} \boldsymbol{r}_w &= \boldsymbol{\mathcal{W}}_{rr}[\boldsymbol{u}_w, \boldsymbol{w}] + \boldsymbol{b}_{rr}, \\ \boldsymbol{c}_w &= softmax(\boldsymbol{P}_{out}\boldsymbol{r}_w), \end{aligned} \tag{17}$$

where $\boldsymbol{r}_w$ is the travel-intention-based preference embedding of worker w, $\boldsymbol{\mathcal{W}}_{rr}$ and $\boldsymbol{b}_{rr}$ are the trainable parameters, and $\boldsymbol{P}_{out}$ is the out-of-town POI embeddings learned by GCN.

3.3.5 Preference Transfer and Training. Considering travel intention drifts, we leverage a Multi-Layer Perceptron (MLP) to transfer the hometown preferences into out-of-town behavior, which is inspired by the cross-domain recommendation [32]. The transfer loss is defined as follows.

$$\mathcal{L}_{trans} = \sum_{w \in W} ||MLP(\boldsymbol{w}) - \boldsymbol{w}^0||, \tag{18}$$

where $MLP(\cdot)$ is an MLP-based mapping function [32], and $\boldsymbol{w}^0$ is a randomly initialized out-of-town preference embedding of w.

The whole training objective is described as follows.

$$\mathcal{L} = \gamma_1 \mathcal{L}_{trans} + \gamma_2 \mathcal{L}'_n + \gamma_3 \mathcal{L}'_t, \tag{19}$$

where γ_1, γ_2, and γ_3 are parameters controlling the contributions of different parts, $\mathcal{L}'_n$ denotes the loss of out-of-town preference estimation based on the Bayesian Personalized Ranking (BPR) function [36], and $\mathcal{L}'_t$ can be computed by Equation 12 based on the embeddings generated by INT in Section 3.3.2. To improve the efficiency of the BPR calculation, we randomly select a fixed size of positive samples (i.e., the visited POIs) and their counterparts (i.e., the unvisited POIs) at each training iteration.

4 TASK RECOMMENDATION

Once workers' preferences for tasks are obtained, a typical recommendation method is top-k recommendation, which recommends a list of the top-k most interested tasks to a worker to choose. The worker can choose any task from the task list only if the task is available, i.e., the task is not selected by other workers currently. Traditional top-k recommendation methods have focused on maximizing worker satisfaction by tailoring the results only according to the personalized preferences of individual workers. However, such worker-centric design may lead to low coverage rate of the recommended tasks, which impacts SC platforms adversely. Considering the goals of our problem, we propose three methods, i.e., dynamic top-k recommendation, extra-reward-based top-k recommendation, and fair top-k recommendation. Before introducing the three methods, we detail how to generate reachable task sets for workers, which will be used throughout the task recommendation process.

4.1 Reachable Task Set Generation

Due to the constraints of workers' reachable distance and tasks' expiration time, each worker can only access a small subset of tasks, call *Reachable Task Set*, which is defined in Definition 4. The reachable task set for a worker w, denoted by $RS(w)$, should satisfy the following conditions: $\forall s \in RS(w)$

1) $t_{now} + t(w.l, s.l) < s.e$, and

2) $d(w.l, s.l) \leq w.d$,

where t_{now} is the current time, $t(w.l, s.l)$ is the travel time from worker w's location $w.l$ to task s's location $s.l$, and $d(w.l, s.l)$ is the travel distance from location $w.l$ to location $s.l$. The above two conditions guarantee that the worker can travel from the origin to the location of any reachable task s in a reachable task set before it expires. Accordingly, we can get an available worker set for each task s, denoted by $AW(s)$.

4.2 Task Recommendation Methods

4.2.1 Dynamic Top-k Method. One way to maximize the recommended task coverage rate is to increase the variability of the number (i.e., k) of recommended tasks among workers to adapt to the dynamic numbers of workers and tasks. Therefore, we design a dynamic top-k method that recommends task lists with different k values for different workers based on their neighbouring workers and tasks. The intuition is that a worker is willing to have more choices when being in a task-intensive area.

Specifically, the k value for worker w is set according to the distribution of neighbouring workers and tasks, as shown in Equation 20.

$$k = \left\lceil \frac{\left|\cup_{w \in NW_m(w)} RS(w)\right|}{m} \right\rceil, \tag{20}$$

where $NW_m(w)$ denotes the set of m neighbouring workers of worker w, and m is the number of the neighbouring workers that is application-specific.

4.2.2 Extra-reward-based Top-k Method. Inevitably, if we only consider worker preferences in recommendation, some tasks may always be ignored and never be selected by workers, which affects the task coverage rate negatively. To solve this issue, we introduce the concept of extra reward to give priority to the ignored tasks for improving the task coverage rate. More specifically, we introduce the extra-reward-based preference of worker w for task s, denoted by $RP(w, s)$, which is computed in Equation 21.

$$\begin{aligned} RP(w, s) &= \alpha c_w(s) + (1-\alpha) Re(s), \\ Re(s) &= \frac{N_{ig}(s)}{N_{exp}(s)}, \end{aligned} \tag{21}$$

where $\alpha \in [0, 1]$ is a parameter controlling the contribution of w's preference for s (i.e., $c_w(s)$) and s's extra reward (i.e., $Re(s)$). Next, $Re(s)$ is the ratio between the number ($N_{ig}(s)$) of task recommendation rounds where task s is ignored, i.e., task s is not selected by workers for N_{ig} task recommendation rounds, and the expected number ($N_{exp}(s)$) of the task recommendation rounds for s, where $N_{exp}(s)$ can be specified by the task requester or the SC platform.

4.2.3 Fair Top-k Method. Although the above methods can achieve high task coverage rate, they generate great disparity in the exposure of tasks (cf. Section 5.2.2), which is unfair for tasks and may also hurt SC platforms in the long term. If only a few tasks get most of the exposure, the requesters of other tasks would struggle on the SC platform, which will force them to either quit or switch to other platforms. This, in turn, may limit the choices for workers, degrading the overall experience on the SC platform. Thus, it is important to reduce exposure inequalities in task recommendation. To achieve this, we consider the exposure-based fairness across tasks and ensure the task exposure-based fairness while giving personalized recommendations. Following the previous studies [15, 39], we define the exposure of task s over the task recommendation R in

Equation 22.

$$Exp\,(s \mid R) = \frac{1}{|W|}\sum_{w\in W} b_s^w, \qquad b_s^w = \begin{cases} 1 & \text{if } s \in R.RTS_k(w) \\ 0 & \text{otherwise,} \end{cases} \tag{22}$$

where $Exp\,(s \mid R)$ denotes the exposure of s over R, $|W|$ denotes the number of workers, b_s^w is an indicator function, and $R.RTS_k(w)$ is the k-recommended-task-set of worker w in task recommendation R.

Based on Equation 22, an intuitive approach is to make the exposure of all tasks nearly equal. However, such recommendation may decrease the overall worker utilities [34]. To solve this issue and achieve a good trade-off between the overall worker utilities and task exposure-based fairness, we propose a fair task recommendation with coarse-to-fine tuning. Specifically, we transform the task recommendation problem into a non-shareable and discrete task allocation problem, where each task contains several copies and each copy is non-shareable (i.e., no task copy can be allocated to multiple workers) and discrete (i.e., no task copy can be broken into pieces). The number of task copies, $X(s)$ ($X(s)$ is an integer), also called task exposure limit, can be computed in Equation 23.

$$X(s) = \begin{cases} \left[\left\lfloor \frac{k|W|}{|S|} \right\rfloor, \left\lceil \frac{k|W|}{|S|} \right\rceil\right] & |AW(s)| \geq \left\lceil \frac{k|W|}{|S|} \right\rceil \\ |AW(s)| & \text{otherwise,} \end{cases} \tag{23}$$

where $|S|$ denotes the number of tasks, and $|AW(s)|$ denotes the number of available workers for task s. Since the total exposure of tasks remains limited $k|W|$, the average exposure for task can be approximated to be $\left[\left\lfloor \frac{k|W|}{|S|} \right\rfloor, \left\lceil \frac{k|W|}{|S|} \right\rceil\right]$ when $|AW(s)| \geq \left\lceil \frac{k|W|}{|S|} \right\rceil$, ensuring fairness to some extent. Otherwise (i.e., $|AW(s)| < \left\lceil \frac{k|W|}{|S|} \right\rceil$), $X(s)$ should be set to the number of available workers, i.e., $|AW(s)|$. Intuitively, when each task s is recommended to $X(s)$ workers, we can get the optimal or the near-optimal fair task recommendation, where the exposure of all tasks nearly equal. Dividing each task into $X(s)$ copies, we use the coarse and fine tuning to achieve fair task recommendation.

Coarse Tuning (CT). Algorithm 1 shows the coarse tuning process, which takes the reachable task sets RS for all workers, the available worker sets AW for all tasks, a task set S, and a k value as input. After initialization (lines 1–2), for each worker, if the number of the recommended tasks is less than k (i.e., $|R(w)| < k$), we allocate/recommend a task to the worker (lines 4–15). Specifically, we first compute a set of copy-aware preference scores (i.e., $c'_w(RS(w))$) for reachable tasks (i.e., tasks in $RS(w)$) according to whether there exist available task copies to be allocated (lines 6–9). If available copies of task s exist (i.e., $|s.cps| > 0$), the copy-aware preference score is same with the preference score of w for s (i.e., $c'_w(s) \leftarrow c_w(s)$); otherwise, $c'_w(s) \leftarrow 0$, where $s.cps$ denotes the copies of s. Next, we can get the task s with the highest copy-aware preference score (line 10). If the copies of s are available (i.e., $|s.cps| \neq 0$), we allocate it to worker w (lines 11–12). Then one copy of task s is removed from the copies $S.cps$ of S, all the copies of s are removed from $RS(w)$, and worker w is removed from $AW(s)$ (lines 13–15). The coarse tuning procedure iteratively allocates each worker the most interested task from the current available tasks until no allocations are given (lines 16–17).

Fining Tuning (FT). Algorithm 2 shows the fining tuning process, which takes a worker set W, a task set S, and a k value as input and outputs a task recommendation R. We first compute the available worker

Algorithm 1: Coarse Tuning (CT)

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

1 $R \leftarrow \emptyset$;
2 $r \leftarrow 0$;
3 **while** *True* **do**
4   **for** *each* $w \in W$ **do**
5   **if** $|R(w)| < k$ **then**
6             $c'_w(RS(w)) \leftarrow \emptyset$; /*$c'_w(RS(w))$ denotes the copy-aware preference scores of w for reachable tasks*/
7   **for** *each* $s \in RS(w)$ **do**
8   **if** $|s.cps| > 0$ **then** $c'_w(s) \leftarrow c_w(s)$;/*$s.cps$ denotes the copies of task s*/
9   **else** $c'_w(s) \leftarrow 0$;
10             $s \leftarrow \arg\max_{s \in RS(w)} c'_w(s)$; /*Find the task with the highest copies-aware preference score*/
11   **if** $|s.cps| \neq 0$ **then**
12                 $R \leftarrow R \cup (w, s)$;
13                 $S.cps \leftarrow S.cps - s$; /*$S.cps$ denotes the copies of the task set S*/
14                 $RS(w) \leftarrow RS(w) - s.cps$; /*Remove all the copies of s from the reachable task set of w*/
15                 $AW(s) \leftarrow AW(s) - w$; /*Remove w from the available set of s*/
16   **if** $|R| == r$ **then**
17   break;
18     $r \leftarrow |R|$;
19 **Return** R;

set $AW(s)$ for each task s and make X copies for s, where $X = \min\{\left\lfloor \frac{k \cdot |W'|}{|S'|} \right\rfloor, |AW(s)|\}$ (lines 3–5). Then we sort the workers in W according to the number (i.e., $|RS(w)|$) of their reachable tasks ascendingly, where $RS(w)$ is w's reachable tasks computed based on $AW(s)$ (line 7). The intuition is that workers with fewer reachable tasks are more likely to be allocated unsuccessfully when they are allocated later, so we give priority to them. Next, we call the Coarse Tuning (CT) algorithm to allocate task copies to workers (line 9). After the first allocation, for each allocated task $s \in S - S'$ whose number of copies is 0, we relax the task exposure limit by adding a copy to the task (i.e., $s.cps \leftarrow s.cps \cup \{s^{X+1}\}$) (lines 10–11). Getting a new unallocated task set S', we reallocate them to workers by CT to get a new task recommendation R (lines 13–14). Finally, for each worker w whose number of recommended tasks is less than k' ($k' = \min\{k, |RS(w)|\}$), we further relax the fairness condition and recommend tasks to w according to the preference scores if the task is not allocated to w before (lines 15–20).

4.3 Limitation and Discussion of Task Recommendation Methods

The proposed task recommendation methods can be applied in real-world scenarios such as real-time ride-hailing services (e.g., Uber[1]), on-wheel meal-ordering services (e.g., GrubHub[2]), etc., where suitable tasks can be recommended to workers especially when the workers are unfamiliar with the areas they are

[1] https://www.uber.com/
[2] https://get.grubhub.com/

Algorithm 2: Fining Tuning (FT)

Input: All workers W, all tasks S, a specified value k
Output: Task recommendation R

$S' \leftarrow S$;
$S'.cps \leftarrow \emptyset$;
for *each* $s \in S$ **do**
  Compute the available worker set $AW(s)$;
  $s.cps \leftarrow \{s^1, s^2, ..., s^X\}$, where $X = \min\{\left\lfloor \frac{k \cdot |W|}{|S'|} \right\rfloor, |AW(s)|\}$; /*Each task s is divided into X copies*/
  $S'.cps \leftarrow S'.cps \cup s.cps$;
Sort workers in W according to $|RS(w)|$ ascendingly, where $RS(w)$ is w's reachable tasks computed based on $AW(s)$;
$RS' \leftarrow RS$;
$R' \leftarrow CT(RS', AW, S', k)$;
for *each* $s \in S - S'$ **do**
  $s.cps \leftarrow s.cps \cup \{s^{X+1}\}$;
  $S'.cps \leftarrow S'.cps \cup s.cps$;
$R'' \leftarrow CT(RS', AW, S', k)$;
$R \leftarrow R' \cup R''$;
for *each* $w \in W$ **do**
  $k' \leftarrow \min\{k, |RS(w)|\}$;
  if $|R(w)| < k'$ **then**
    Sort tasks in $RS'(w)$ according to the preference scores of w desendingly;
    for *each* $s \in RS'(w)$ *and* $|R(w)| < k'$ **do**
      $R \leftarrow R \cup (w, s)$;
Return R;

located. The dynamic top-k method uses a dynamic k value to recommend tasks considering the distribution of neighbouring workers and tasks, while the extra-reward-based top-k method combines the extra reward with workers' preferences that gives priority to the ignored tasks for improving the task coverage rate.

Although these two methods can achieve high coverage rate of the recommended tasks, leading more tasks being recommended, the task recommendations show great disparity in the exposure of tasks (cf. Section 5.2.2), where only a few tasks get most of the exposure. To solve the unfair task recommendation, we propose the fair top-k method, which reduces exposure inequalities in task recommendation by taking the exposure-based fairness across tasks and gives personalized recommendations.

Compared with the original top-k method that recommends each worker k most interested tasks (i.e., k tasks with the highest preference scores) from reachable tasks of the worker, the above methods affect the preference-based utilities of workers negatively to some extent. However, our proposed methods have considerable performance in terms of the recommended task coverage rate, the average k value, and the individual exposure disparity of tasks, which can be applied in task recommendation scenarios with different needs. The recommendation performance of these methods is affected by different parameters, e.g., the numbers of tasks, workers, and POIs, the expiration time of tasks, the reachable distance of workers, and the number of recommended tasks, and we study the effects of these parameters in our experimental part in Section 5.2.2.

5 EXPERIMENTAL EVALUATION

We evaluate the performance of the worker preference learning and the task recommendation on real data. The experimental setup is presented in Section 5.1, followed by a coverage of key experimental results in Section 5.2. We conduct the experiments on an Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz with 128 GB RAM and an GeForce GTX 1080 GPU.

5.1 Experimental Setup

Due to the lack of benchmark for Spatial Crowsourcing (SC) task recommendation algorithms, we use two real check-in datasets from Foursquare-Global[3], **IS→IZ** (Istanbul → Izmir) and **JA→BA** (Jakarta → Bandung), to simulate the task recommendation scenario, where **IS→IZ** stands for traveling from Istanbul (hometown city) to Izmir (out-of-town city), and **JA→BA** stands for traveling from Jakarta (hometown city) to Bandung (out-of-town city). The travel records of the above datasets are generated from April 2012 to September 2013. For ensuring the data quality, we filter out the POIs that are visited less than five times in both datasets. Besides, the users whose hometown check-ins are less than five times or out-of-town check-ins are less than three times are removed. After being filtered, the statistics of the datasets are given in Table 2. It is expected that the numbers of out-of-town check-ins and POIs are far less than those of hometown check-ins and POIs since most of the footprints of users are left in their hometown areas, leading to the insufficiency of out-of-town check-ins [12, 45]. In particular, a previous study shows that the ratio of the hometown and the out-of-town check-ins of a user is, on average, 1 : 0.0047 [38].

Table 2. Statistics of Datasets

Dataset		# Users	# POIs	# Check-ins
IS→IZ	Istanbul	1,362	14,231	152,910
	Izmir		1,317	28,865
JA→BA	Jakarta	1,405	15,208	160,673
	Bandung		1,948	31,490

In our experiments, we assume that users are workers of the SC system since users who check in to different spots are good candidates to perform spatial tasks in the vicinity of those spots. Each worker has both hometown and out-of-town check-ins. When the historical check-ins are used to train the hometown preference learning module, the out-of-town check-ins are removed. The check-in POIs are the locations where workers perform tasks. We randomly split workers following the proportions: 80%, 10%, and 10% to form a training set, a validation set, and a testing set. To train and evaluate the hometown preference learning module, we follow the session-based recommendation paradigm [47] to augment the datasets, which have been proven to be effective for improving the recommendation accuracy. For example, if the original POI sequence of a worker is (p_0, p_1, p_2, p_3), we can divide it as three sequences, i.e., $((p_0), (p_1))$, $((p_0, p_1), (p_2))$, and $((p_0, p_1, p_2), (p_3))$, where the first part denotes the historical records, and the second denotes the location that the worker will visit next and is used as a label to train the hometown preference learning module.

[3] https://sites.google.com/site/yangdingqi/home/foursquare-dataset

To train the out-of-town preference learning module, we follow the experimental settings in TrainOR [50], where the hometown and out-of-town check-ins of the same user are used as training or testing sample. Then we merge the two testing sets for the overall testing evaluation and report the results. In the whole Adaptive Worker Preference Learning (AWPL) model that includes the hometown and the out-of-town modules, we use the hometown module to calculate the preference scores of workers for POIs when we detect a worker is in the hometown; otherwise, we adopt the out-of-town module to calculate workers' preference scores.

For the task recommendation experiments, tasks are generated randomly on POIs, which means that each POI has several tasks. It is common practice in experimental studies of SC platforms to use uniformly (and randomly) distributed attribute values [41], the argument being that this captures the effects of the attributes on a more fair basis. The reward of each task is set to 1, and the speed of workers in both datasets is set to 5km/h. Since the numbers of users in both datasets are insufficient, we generate workers based on the long-term check-in POIs. Specifically, for hometown workers, since we adopt a session-based recommendation mechanism [47], we take each session to simulate a travel record of a worker. For example, $((p_0, p_1, p_2), (p_3))$ is one of the hometown sessions of a worker, which is generated from (p_0, p_1, p_2, p_3) by augmentation. The second part (i.e., (p_3) used as the label for training) is discarded in the recommendation experiments, and the last POI (i.e., p_2) in the first part represents the current location of a worker, which is to determine which modules (i.e., the hometown or the out-of-town modules) should be used. It should be noteworthy that different sessions generated from the same sequence of check-ins correspond to travel records of different workers. Thus, we can 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. In the experiments of task recommendation, 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.

5.2 Experimental Results

5.2.1 Performance of Worker Preference Learning. In this set of experiments, we evaluate the performance of the worker preference learning phase.

Evaluation Methods. We study the following methods.

1) TOP: A naive method, which recommends the top-N frequently visited POIs in the target city.

2) SR-GNN [47]: A GNN-based model, which utilizes GNNs to model the complex transitions of POIs. To make this method applicable to out-of-town recommendation, we take hometown check-ins as input and make predictions on the out-of-town POIs trained with Bayesian Personalized Ranking (BPR) [36], where preference embeddings of workers can be obtained by gated GNN.

3) TrainOR [50]: An out-of-town POI recommendation method considering travel intentions, which is adapted to enabling hometown recommendation by aggregating each hometown worker's preferences and travel intentions to obtain the hometown preference embedding of the worker. As a result, a worker's hometown preference scores over POIs are the inner product of the worker's hometown preference embedding and the POIs' embeddings.

4) AWPL-I: A variant of the proposed AWPL model, which removes the travel intention modeling part. As a result, it recommends only based on workers' preferences.

5) AWPL: Our proposed model, which includes a hometown and an out-of-town module.

Table 3. Accuracy on Two Datasets

Methods	IS→IZ				JA→BA			
	Rec@10	Rec@20	Rec@30	MAP	Rec@10	Rec@20	Rec@30	MAP
TOP	17.730%	25.577%	29.050%	17.153%	8.868%	14.790%	17.094%	11.293%
SR-GNN	14.806%	19.484%	21.667%	23.349%	7.852%	12.366%	15.283%	12.348%
TrainOR	18.021%	22.592%	27.131%	23.946%	12.119%	16.319%	18.954%	17.142%
AWPL-I	20.316%	26.751%	31.286%	25.299%	14.145%	19.693%	22.750%	17.551%
AWPL	**22.576%**	**28.335%**	**33.569%**	**26.127%**	**14.788%**	**19.882%**	**23.464%**	**18.562%**

Metrics. We use Recall@N (Rec@N, where $N = 10, 20, 30$) and Mean Average Precision (MAP) to evaluate accuracy of the above methods. The larger the values of the above metrics are, the more accurate the model is. We also evaluate the efficiency of the methods, including training and testing time.

Parameter Settings. For all latent representations, the number of the hidden size is fixed to 128. In the travel intention modeling, we set the topic number M as 15. In the training stage, for simplification of tuning hyper-parameters, we set γ_1=γ_2=$(1-\gamma_3)/2$ (shown in Equation 19), where γ_3 is set to 0.9. We used Adam optimizer to train our model with an initial learning rate 0.001 and an $L2$ regularization with weight 10^{-5}. The testing is repeated over five times using different data splits, and the average result is reported.

Accuracy. We report the Rec@N ($N = 10, 20, 30$) and the MAP values in Table 3. The best performance by an existing method (TOP, SR-GNN, and TrainOR) is underlined, and the overall best performance is marked in bold. For both datasets, APWL achieves the highest Rec@N, which outperforms the best among the baseline methods by up to 25.276% and 23.794% in IS→IZ and JA→BA, respectively. In terms of MAP, APWL performs best among all methods, followed by its variant APWL-I and other methods in both datasets, showing the superiority of the two-module architecture for worker preference learning. APWL always achieves better accuracy than APWL-I regardless of metrics, which demonstrates the necessity of travel intention modeling in worker preference learning.

Efficiency. We study the training time (of each epoch) and the testing time (of each worker) for all methods on two datasets, as shown in Figure 4. We can see that our model and its variant, i.e., AWPL and AWPL-I, take much more time for training compared with SR-GNN anf TRAINOR. This is because our methods have to train a large amount of the augmented hometown data while others (i.e., SR-GNN and TRAINOR) only need to train a small amount of out-of-town data. Although SR-GNN and TRAINOR are more efficient for training and testing, they perform worse than AWPL and AWPL-I in terms of accuracy, shown in Table 3. Figure 4 also shows that AWPL and AWPL-I run in less than six millisecond when computing worker preferences, which indicates their feasibility in real task recommendation scenarios.

5.2.2 Performance of Task Recommendation. Next, we evaluate the performance of task recommendation.

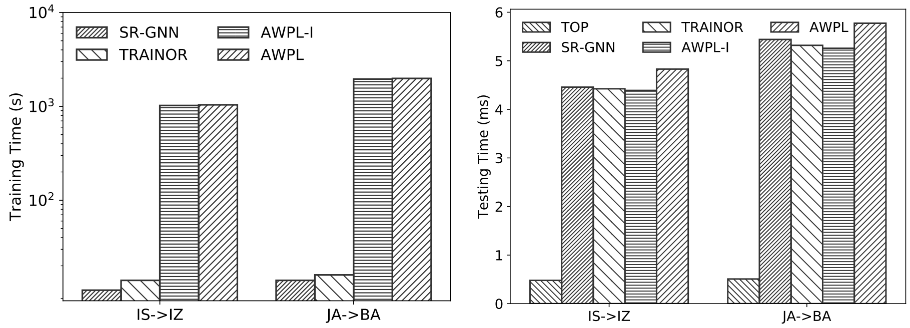
Evaluation Methods. We study the following methods.

1) Top-k: The traditional Top-k method, which recommends each worker k most interested tasks (i.e., k tasks with the highest preference scores) from reachable tasks of the worker.

2) DyTop-k: The proposed Dynamic Top-k method, where the number of the neighbouring workers is set to 5, i.e., $m = 5$.

3) ERTop-k: The proposed Extra-Reward-based Top-k method, where $\alpha = 0.5$.

4) FairRec [34]: A two-sided fair recommendation method that considers fairness for customers and products, where customers are regarded as workers and products are regarded as tasks.



(a) Training Time (b) Testing Time

Fig. 4. Training Time and Testing Time

Table 4. Experiment Parameters

Parameter	Value
Number of tasks, $\vert S\vert$	2K, 3K, 4K, 5K, 6K
Number of workers, $\vert W\vert$	2K, 4K, 6K, 8K, 10K
Number of POIs, $\vert P\vert$	1K, 2K, 3K, 4K, 5K
Expiration time of tasks (h), e	0.6, 0.9, 1.2, 1.5, 1.8
Reachable distance of workers (km), d	1, 2, 3, 4, 5
Number of recommended tasks (except for DyTop-k), k	4, 6, 8, 10, 12, 14

5) FairTop-k: The proposed Fair Top-k method.

Metrics. Four main metrics are compared for the above methods.

1) TCR: Task Coverage Rate that is the ratio between the number of recommended tasks and the total number of tasks.

2) k_{mean}: The average k value, i.e., $k_{mean} = \frac{\sum_{w \in W} w.k}{|W|}$, where $w.k$ denotes the k value of worker w.

3) APU: The Average Preference-based Utility of workers, i.e., $APU = \frac{\sum_{w \in W} U(RTS_k(w))}{|W|}$, where $U(RTS_k(w))$ is the preference-based utility of worker w that can be calculated by Equation 1.

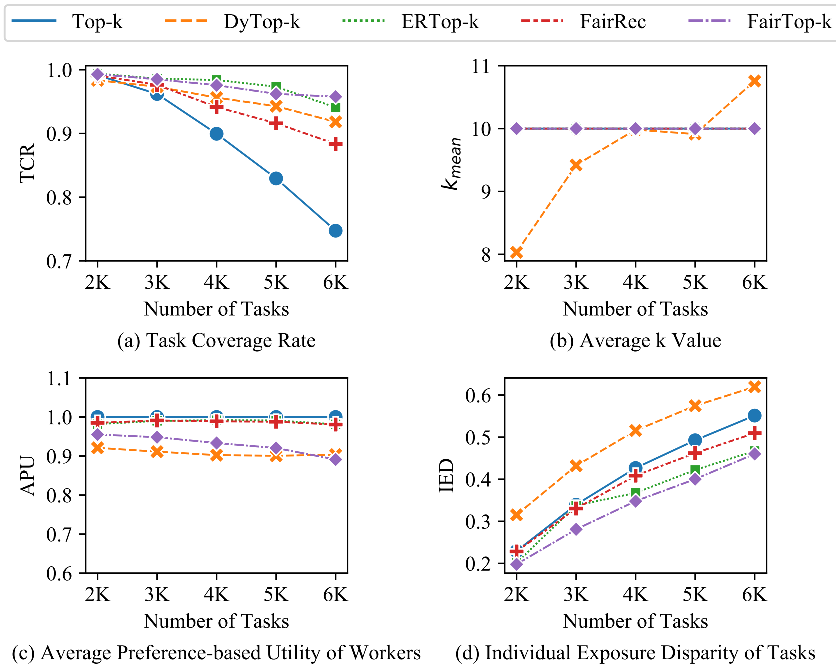
4) IED: The Individual Exposure Disparity [26] of tasks, which is measured by the Gini coefficient [16] in the following.

$$IED = \frac{\sum_{s,s' \in S} |Exp\,(s \mid R) - Exp\,(s' \mid R)|}{2|S| \sum_{s'' \in S} Exp\,(s'' \mid R)}, \tag{24}$$

where $Exp\,(s \mid R)$ denotes the exposure of task s over task recommendation R. It is easy to see that IED, the range of which is $[0, 1]$, measures the pairwise exposure disparity. When $IED = 0$, the task recommendation R achieves the perfect equality (i.e., the best task exposure-based fairness) where all individual tasks have the same exposure, while $IED = 1$ represents the maximal inequality in terms of individual task exposure. The smaller the IED is, the more fair the method is.

Parameter Settings. Table 4 shows our experimental settings, where the default values of all parameters are underlined. Note that we evaluate different k values for all methods except for DyTop-k since its k value is set adaptively based on the numbers of neighboring workers and tasks.

Effect of $|S|$. We first study the effect of the number of tasks $|S|$ on two datasets, IS→IZ and JA→BA. From Figures 5(a) and 6(a), we can see that ERTop-k and FairTop-k always achieve higher Task Coverage Rate (TCR) compared with Top-k by up to 25.3% and 28.0%, respectively, and DyTop-k outperforms better



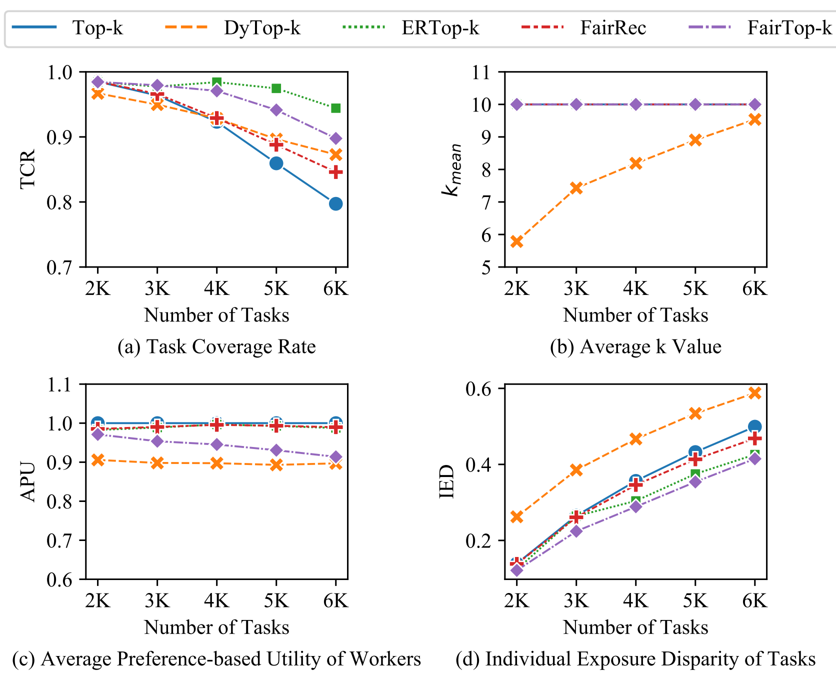
(a) Task Coverage Rate

(b) Average k Value

(c) Average Preference-based Utility of Workers

(d) Individual Exposure Disparity of Tasks

Fig. 5. Effect of $|S|$ on IS→IZ



(a) Task Coverage Rate

(b) Average k Value

(c) Average Preference-based Utility of Workers

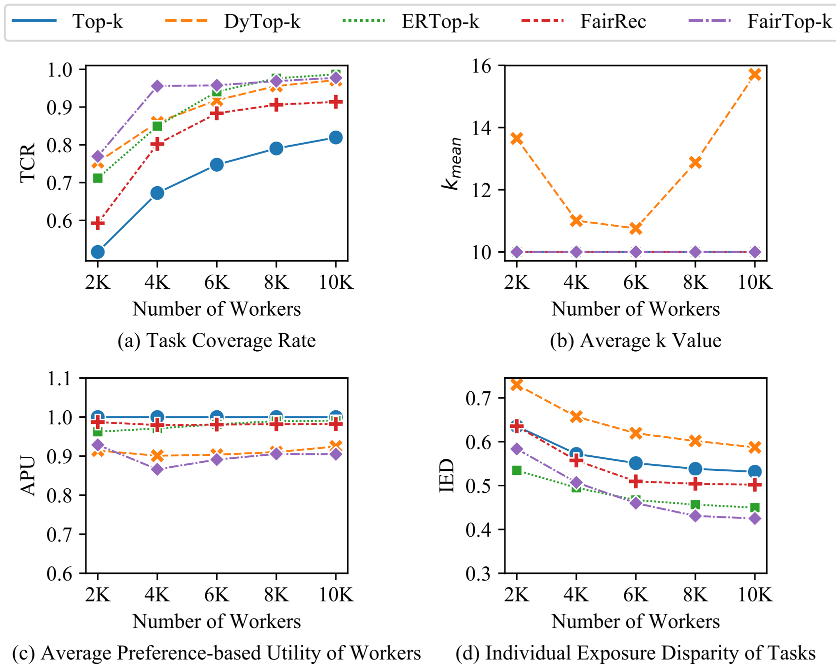
(d) Individual Exposure Disparity of Tasks

Fig. 6. Effect of $|S|$ on JA→BA

than Top-k when $|S| \geq 3K$ on IS$\rightarrow$IZ and when $|S| \geq 4K$ on JA$\rightarrow$BA, which shows the superiority of our methods in terms of TCR. In most cases, our methods perform better than FairRec in terms of TCR. Besides, the superiority is more prominent when the number of tasks increases, i.e., the performance gaps of our methods and Top-k are increasing with $|S|$ grows. Apparently, the TCR of all methods decline with increasing $|S|$, but the TCR of our methods decline much more slowly, showing their good scalability, i.e., it is able to adapt easily to the increasing number of tasks. Figures 5(b) and 6(b) show the average k value among workers, where Top-k, ERTop-k, FairRec, and FairTop-k have fixed k values, i.e., $k = 10$. It means they recommend k tasks to each worker at each task recommendation. We observe that when $3K \leq |S| \leq 5K$ on IS$\rightarrow$IZ and when $|S| \geq 4K$ on JA$\rightarrow$BA, DyTop-k has smaller average k values but higher TCR values than Top-k, which demonstrates the advantage of using dynamic k values. When it comes to the Average Preference-based Utility (APU) of workers in Figures 5(c) and 6(c), since Top-k recommends the k most interested tasks for each worker, it achieves the highest APU (i.e., the optimal APU), followed by ERTop-k, FairRec, FairTop-k, and DyTop-k in most cases on both datasets. ERTop-k can obtain 98.1%–99.2% of the optimal APU, and its APU is consistently higher than those of FairTop-k (by up to 10.1%) and DyTop-k (by up to 11.0%), which demonstrates the advantage of the extra-reward-based strategy. FairTop-k and DyTop-k can achieve up to 97.1% and 92.1% of the optimal APU, respectively. Although FairTop-k performs worse than Top-k and ERTop-k in terms of APU, it performs best in the task exposure-based fairness, i.e., its Individual Exposure Disparity (IED) is the smallest on both datasets, as shown in Figures 5(d) and 6(d). The performance of ERTop-k in exposure-based fairness is second only to FairTop-k. This is so because ERTop-k gives a higher priority to the tasks that are ignored in the previous recommendations by setting the extra reward, which improves the average exposure of tasks. FairRec performs worse than our FairTop-k in terms of the task exposure-based fairness, which demonstrates the superiority of FairTop-k.

Effect of $|W|$. Next, we study the effect of $|W|$, the number of workers to be recommended. As shown in Figures 7(a) and 8(a), our proposed methods, i.e., DyTop-k, ERTop-k, and FairTop-k, can always achieve higher TCR than the traditional Top-k method and FairRec, which can improve the TCR by up to 57.1% and 35.4%, respectively. In Figure 7(b), the k values of DyTop-k are higher than those of others regardless of $|W|$ on IS$\rightarrow$IZ. This may be due to the fact that tasks in this dataset are intensive around workers, leading to high dynamic k values for achieving high TCP. Figure 8(b) shows that DyTop-k is able to obtain higher TCR with smaller k values compared to Top-k when $2K \leq |W| \leq 6K$ on JA$\rightarrow$BA. The APU of Top-k is the highest, but it cannot achieve good task exposure-based fairness, as shown in Figures 7(c), 7(d), 8(c), and 8(c). FairTop-k is still the most fair method in most cases on both datasets, and always outperforms FairRec in terms of fair task recommendation. We also observe that ERTop-k is able to achieve a good trade-off between APU (second to Top-k) and IED (that is even smaller than that of FairTop-k in some cases.).

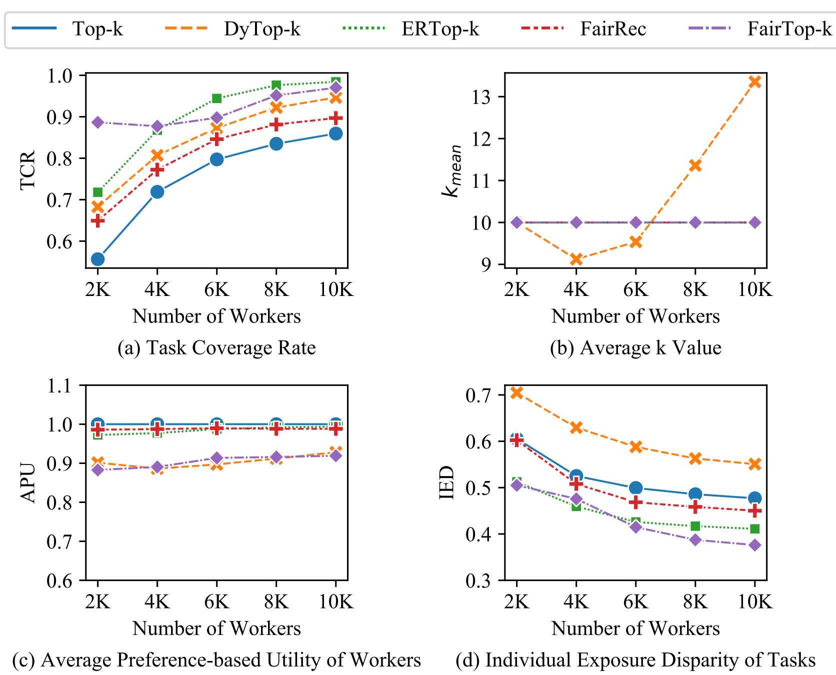
Effect of $|P|$. Figures 9 and 10 show the effect of the number of POIs, $|P|$, on the performance of all methods. When the number of POIs increases, the TCR of all methods are stable, as shown in Figures 9(a) and 10(a). DyTop-k, ERTop-k, FairRec, and FairTop-k can obtain higher task coverage rate than Top-k while sacrificing some utility of workers, as shown in Figures 9(c) and 10(c). It is worth mentioning that the average k values of DyTop-k are always smaller than that of Top-k on JA$\rightarrow$BA (cf. Figure 10(b)) while it outperforms Top-k in terms of the task coverage rate by 9.1%–11.1%, which shows its superiority. From Figures 9(d) and 10(d), we can see that FairTop-k still outperforms other methods in terms of fairness.



(a) Task Coverage Rate (b) Average k Value

(c) Average Preference-based Utility of Workers (d) Individual Exposure Disparity of Tasks

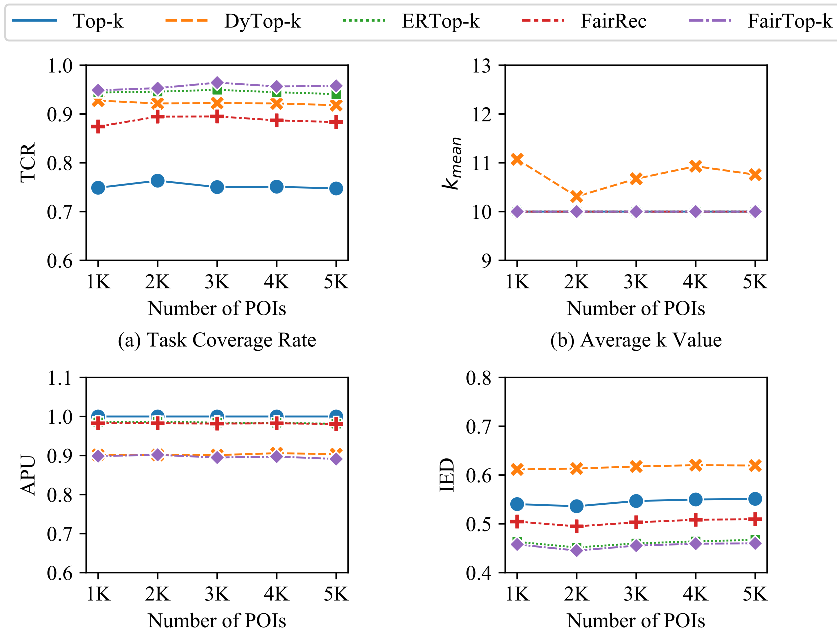
Fig. 7. Effect of $|W|$ on IS→IZ



(a) Task Coverage Rate (b) Average k Value

(c) Average Preference-based Utility of Workers (d) Individual Exposure Disparity of Tasks

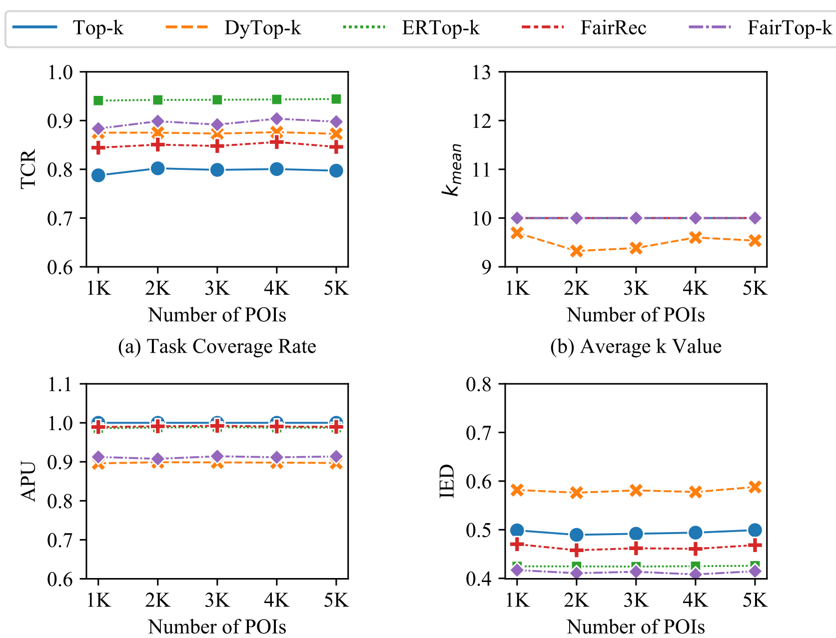
Fig. 8. Effect of $|W|$ on JA→BA



(a) Task Coverage Rate (b) Average k Value

(c) Average Preference-based Utility of Workers (d) Individual Exposure Disparity of Tasks

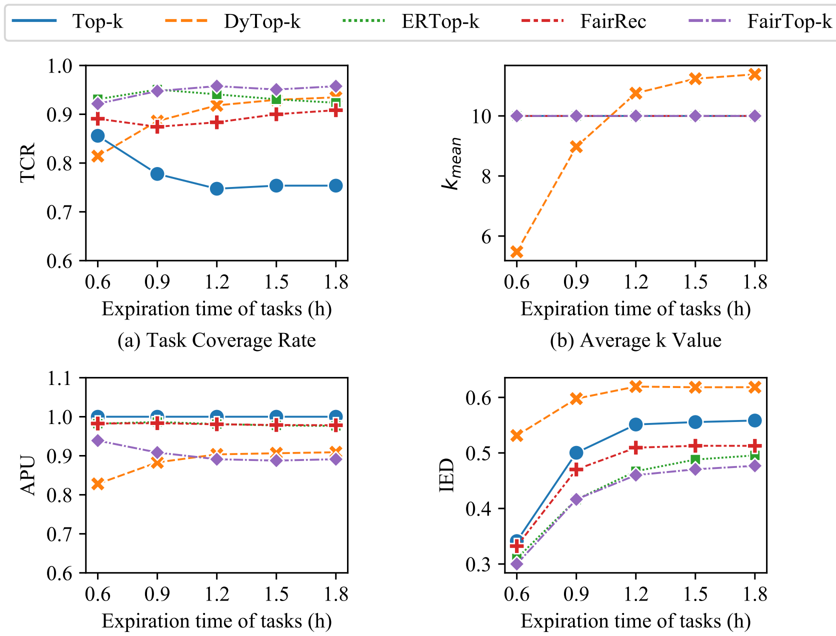
Fig. 9. Effect of $|P|$ on IS→IZ



(a) Task Coverage Rate (b) Average k Value

(c) Average Preference-based Utility of Workers (d) Individual Exposure Disparity of Tasks

Fig. 10. Effect of $|P|$ on JA→BA



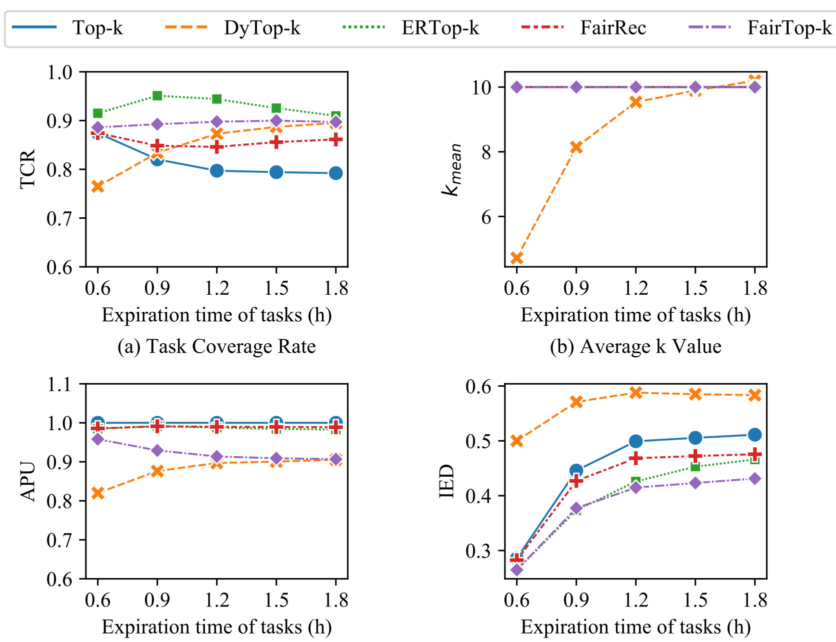
(a) Task Coverage Rate

(b) Average k Value

(c) Average Preference-based Utility of Workers

(d) Individual Exposure Disparity of Tasks

Fig. 11. Effect of e on IS→IZ



(a) Task Coverage Rate

(b) Average k Value

(c) Average Preference-based Utility of Workers

(d) Individual Exposure Disparity of Tasks

Fig. 12. Effect of e on JA→BA

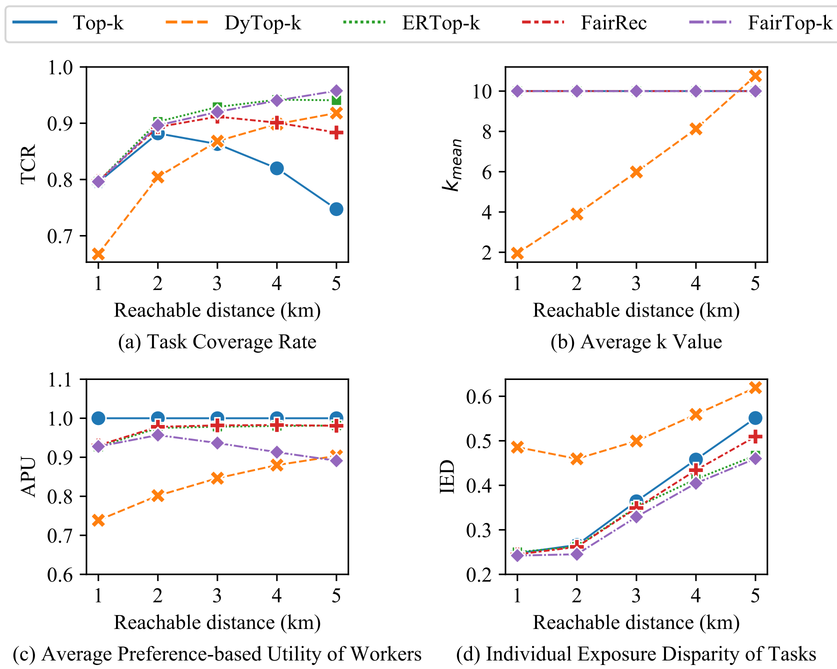
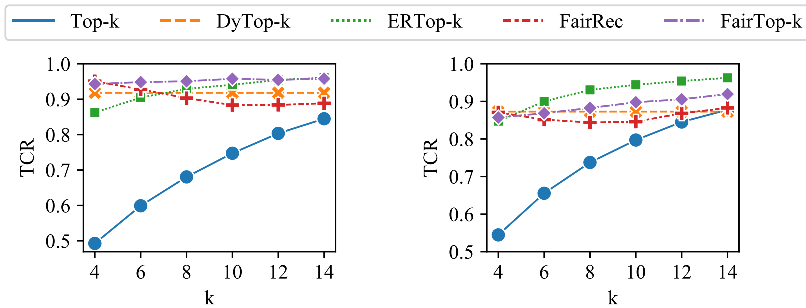


Fig. 13. Effect of d on IS→IZ

Effect of e. Next, we study how the expiration time (e) of tasks affects the recommendation performance. Figures 11(a) and 12(a) show that Top-k performs worse than ERTop-k and FairTop-k and shows a downward trend with increasing e. When e gets larger, the average k values of DyTop-k show an upward trend (see Figures 11(b) and 12(b)) since each worker has more reachable tasks with more relaxed valid time, increasing the dynamic k value for the worker. Although Top-k obtains the highest APU among all methods, which is shown in Figures 11(c) and 12(c), it performs worst in terms of TCR. As expected, FairTop-k is still the most fair task recommendation method with the lowest IED, as shown in Figures 11(d) and 12(d). To save space, in the following experiments, we omit results on JA→BA as these are similar to those on IS→IZ.

Effect of d. We study the effect of d, reachable distance of workers. Figure 13(a) shows that the TCR values of all methods (except for Top-k) increase gradually with increasing d since a larger d implies that more tasks are reachable for workers and can be recommended to them. Top-k deteriorates at a significantly faster pace in terms of TCR when $d \geq 2$. In Figure 13(b), when $3 \leq d < 5$, the benefits of DyTop-k become significant. It achieves higher task coverage rate than Top-k with a smaller k, which can effectively reduce the difficulty of task selection for workers while guaranteeing more tasks being recommended. However, DyTop-k performs the worst in most cases in terms of APU and IED, as shown in Figures 13(c) and 13(d). Therefore, it is not suitable for the scenarios that pursuit high APU and task exposure-based fairness.

Effect of k. We also study the effect of k on the performance of all methods by reporting the task coverage rate results on the two datasets in Figure 14. When the k value is limited, e.g., $k < 14$, our proposed methods including DyTop-k, ERTop-k, and FairTop-k can obtain higher task coverage rate than Top-k on IS→IZ and JA→BA. We also observe that FairRec performs worse than our methods when $k \geq 8$ on IS→IZ



(a) Task Coverage Rate on IS->IZ (b) Task Coverage Rate on JA->BA

Fig. 14. Effect of k

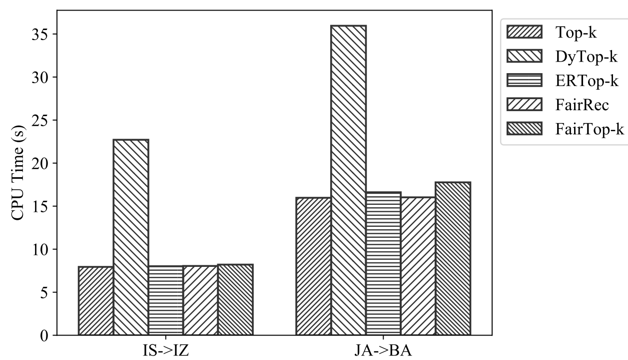


Fig. 15. Task Recommendation Efficiency

and when $k \geq 6$ on JA$\rightarrow$BA. The performance of DyTop-k, ERTop-k, and FairTop-k stay stable or are improved gradually when k grows, showing that they scale well to different k values when k is limited.

Task Recommendation Efficiency. In the final set of experiments, we study the task recommendation efficiency, i.e., the CPU time for find a task recommendation. As illustrated in Figure 15, DyTop-k is the most time-consuming method since it has to compute the dynamic k value for each worker by finding the neighboring workers. The other methods, i.e., Top-k, ERTop-k, FairRec, and FairTop-k, are neck-to-neck in terms of CPU time, which demonstrates that it is feasible to apply ERTop-k, FairRec, and FairTop-k in real SC task recommendation applications.

Summary of our empirical study. The findings of the empirical study can be summarized as follows:

1) Our APWL model achieves the highest accuracy in worker preference learning.

2) For task recommendation, ERTop-k and FairTop-k always achieve higher task coverage rate compared with Top-k. In most cases, DyTop-k has smaller average k values but higher task coverage rate than Top-k, which means that DyTop-k can achieve considerable task coverage rate by recommending fewer tasks for each worker. ERTop-k can obtain 80.9%–99.7% of the optimal APU (average preference-based utility of workers), and FairTop-k achieves the best task exposure-based fairness. Our proposed methods can be applied to different applications according to their requirements.

6 RELATED WORK

Spatial Crowdsourcing (SC) is outsourcing location-dependent tasks to a large group of mobile workers in the form of an open call to reduce the production cost, where workers perform spatial tasks that involve traveling to specified locations [5, 6, 9, 18, 21, 25, 28–30, 44, 46, 49, 52, 55, 56, 59–61]. Depending on how tasks are assigned to workers, SC can be classified into Server Assigned Tasks (SAT) mode and Worker Selected Tasks (WST) mode [24]. Most studies [27, 42–44, 51, 54, 57, 58] assume the SAT mode, where an SC server takes charge of the task assignment. For example, Li et al. assign a group of workers for each task and take social impact into workers' preference [27]. Zhao et al. design a preference-based algorithm to assign tasks considering workers' preference for different tasks [57]. Tong et al. propose a two-sided online micro-task assignment problem in SC and design different task assignment methods to solve it [42]. Xu et al. systematically study the generic insertion operator for dynamic ridesharing and propose a partition-based framework to achieve efficient route planning in SC [51]. Further, an insertion-based framework is proposed to solve the unified route planning problem for shared mobility in SC, where a novel dynamic programming algorithm is designed to achieve linear time complexity [43]. However, one drawback of this mode is that the SC server assigns tasks to workers compulsively without considering the willingness of workers. In practice, a worker is unlikely to honestly and promptly complete the assigned task when the worker is not interested in it, which cannot guarantee the quality of the task result. Under such a situation, workers are usually provided with very limited support throughout the task selection and completion processes. Therefore, we adopt the WST mode in this work, with which the SC server publishes the spatial tasks publicly and online workers can choose any spatial tasks in their vicinity autonomously without the need to coordinate with the SC server. However, a worker has to select a task from a large number of tasks to perform in order to earn the associated reward. In particular, most workers must browse through a long list of open tasks manually before they determine the suitable ones, which is time-consuming and tends to be sub-optimal due to subjective, ad hoc worker behavior. Therefore, it is important to investigate on how to support workers to select tasks on SC platforms easily and effectively. Task recommendation comes into being, which can help workers to choose their satisfied tasks faster as well as help requesters to receive high-quality output quicker.

Recommendation systems are a kind of tools that provide suggestions of potentially useful items based on individual preferences. They have achieved enormous success in many applications, such as product recommendation in e-commerce [10, 34, 48] and POI recommendation in location-based services [50]. Recent studies have explored the use of recommendation systems in SC, which can help workers to find their appropriate tasks for achieving high-quality task results [2, 4, 13, 14, 31]. For example, Chen et al. [4] formulate a task recommendation problem as a stochastic integer linear programming model and propose a multi-agent task recommendation framework considering stochastic spatiotemporal uncertainty. Gao et. al. [14] study a top-k team recommendation problem for complex tasks and propose a two-level-based framework including an approximation algorithm with the provable approximation ratio and an exact algorithm with pruning techniques. However, the above studies ignore workers' travel intentions and their drifts from their hometown areas to out-of-town areas during task recommendation. Travel intention is a key context associated with spatio-temporal information, which plays a crucial role in SC. Besides, most of these studies put their focuses on traditional top-k recommendation methods without considering the dynamic numbers of workers and tasks, resulting in a low coverage rate of recommended tasks. Moreover, while the above approaches are followed to maximize the satisfaction of individual workers, they fail to address fairness during task recommendation, which affects tasks and their requesters adversely at an SC platform. Recent studies focus on the fairness of recommendation systems from the perspective of customers and/or from the perspective of product providers [34, 48], where the customers can be regarded as workers and the product providers can be regarded as task requesters. For example, Wu et al. propose a two-sided fairness-aware recommendation model for both customers and providers [48]. However, they consider the group fairness among providers by introducing a fair exposure baseline for each provider, while we put more focus on the individual fairness among tasks in SC. Thus, this model cannot solve our problem well. The closest related study to our fair task recommendation is the study [34], which develops a FairRec framework for two-sided fair recommendation and considers the fairness among products. However, it differs from our work in terms of problem setting and objectives. First, each item can be recommended to all the customers in FairRec, while in our work, each task can only be recommended to its available workers (i.e., the recommended task can be reachable to these workers) due to spatio-temporal constraints. Second, the goal of FairRec is to maximize the customer utility (i.e., the satisfaction of individual customers) while minimizing the inequality in product exposures. Nevertheless, we aim to maximize the coverage rate of recommended tasks and the worker preference-based utility while reducing the individual exposure disparity of tasks. It is particularly noticeable that we compare FairRec with our work in the experiments.

To enable the task recommendation quality, it is attractive to integrate mechanisms that can learn workers' preferences adaptively based on their current locations and different factors (e.g., the coverage rate of recommended tasks, the k value, the utility of workers, and the exposure-based fairness of tasks) into account in task recommendation. Complementing existing studies, our study aims to learn workers' hometown and out-of-town preferences and design a variety of recommendation methods to enable more effective task recommendation in SC.

It is worth mentioning that food delivery [19, 22, 33] is one of the most common applications in SC, where food orders can be regarded as spatial tasks, delivery vehicles can be regarded as workers, and food delivery can be regarded as task assignment. Food delivery differs from our work in terms of the problem definition and settings, as well as the objectives. First, most of the food delivery studies adopt the SAT mode and define a task assignment and scheduling problem, in which a set of food orders are assigned to each vehicle and a route plan is scheduled for each vehicle by the food delivery platform. In the task assignment and scheduling setting, each vehicle must deliver (perform) the assigned food orders (tasks). In contrast, we adopt the WST mode and define a task recommendation problem that recommends k tasks to each worker. In our problem setting, a task can be recommended to several workers, and each worker can choose a satisfied task from the recommended tasks. Second, most of the food delivery studies aim to minimize the (expected) delivery time, while we aim to maximize the coverage rate of recommended tasks and the average worker preference-based utility. Due to the different problem definition, settings, and objectives, the food order assignment methods cannot solve our problem.

7 CONCLUSION AND FUTURE WORK

We propose an <u>Ada</u>ptive <u>Task</u> <u>Rec</u>ommendation (*AdaTaskRec*) framework that is capable of learning workers' preferences for tasks in different areas adaptively and recommending tasks to workers for particular goals. To model worker preferences in different areas (including hometown and out-of-town areas), we design a two-module architecture that involves workers' travel intentions and their drifts, thus capturing the hometown and the out-of-town preferences of workers more accurately. For task recommendation, we design three methods, i.e., dynamic top-k, extra-reward-based top-k, and fair top-k, to meet different application needs, i.e., maximizing the task coverage rate, limiting the k values, maximizing workers' utility, and achieving task exposure-based fairness. An extensive 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, task coverage rate of recommended tasks (with limited k), and task exposure-based fairness. One interesting research direction is to consider worker fairness into spatial crowdsourcing task recommendation since maintaining task fairness can cause an overall loss in worker utility and the utility loss is likely to be distributed unfairly among workers. Therefore, we need to recommend tasks in a way such that the utility loss is allocated among workers fairly. The other direction is to explore the factors, such as the time that a worker performs a task, task features (e.g., task popularity, task complexity, task difficulty, task risk level, skill requirement, etc.), and social impact, when modeling workers' preference in spatial crowdsourcing.

ACKNOWLEDGMENTS

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

REFERENCES

[1] Himan Abdollahpouri and Robin Burke. Multi-stakeholder recommendation and its connection to multi-sided fairness. *CoRR*, abs/1907.13158, 2019.

[2] Abdulrahman Alamer, Jianbing Ni, Xiaodong Lin, and Xuemin Shen. Location privacy-aware task recommendation for spatial crowdsourcing. In *WCSP*, pages 1–6, 2017.

[3] Robin Burke. Multisided fairness for recommendation. *CoRR*, abs/1707.00093, 2017.

[4] Cen Chen, Shih-Fen Cheng, Hoong Chuin Lau, and Archan Misra. Towards city-scale mobile crowdsourcing: Task recommendations under trajectory uncertainties. In *IJCAI*, pages 1113–1119, 2015.

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

[6] Y. Cheng, B. Li, X. Zhou, Y. Yuan, and L. Chen. Real-time cross online matching in spatial crowdsourcing. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, 2020.

[7] Eunjoon Cho, Seth A Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *SIGKDD*, pages 1082–1090, 2011.

[8] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *SSST@EMNLP*, pages 103–111, 2014.

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

[10] Abhisek Dash, Abhijnan Chakraborty, Saptarshi Ghosh, Animesh Mukherjee, and Krishna P. Gummadi. When the umpire is also a player: Bias in private label product recommendations on e-commerce marketplaces. In *FAccT*, pages 873–884, 2021.

[11] Benjamin Edelman, Michael Luca, and Dan Svirsky. Racial discrimination in the sharing economy: Evidence from a field experiment. *American Economic Journal: Applied Economics*, 9(2):1–22, 2017.

[12] Gregory Ference, Mao Ye, and Wang-Chien Lee. Location recommendation for out-of-town users in location-based social networks. In *CIKM*, pages 721–726, 2013.

[13] Dawei Gao, Yongxin Tong, Jieying She, Tianshu Song, Lei Chen, and Ke Xu. Top-k team recommendation in spatial crowdsourcing. In *WAIM*, pages 191–204, 2016.

[14] Dawei Gao, Yongxin Tong, Jieying She, Tianshu Song, Lei Chen, and Ke Xu. Top-k team recommendation and its variants in spatial crowdsourcing. *DSE*, 2(2):136–150, 2017.

[15] Yingqiang Ge, Shuchang Liu, Ruoyuan Gao, Yikun Xian, Yunqi Li, Xiangyu Zhao, Changhua Pei, Fei Sun, Junfeng Ge, Wenwu Ou, and Yongfeng Zhang. Towards long-term fairness in recommendation. In *WSDM*, pages 445–453, 2021.

[16] Corrado Gini. Measurement of inequality and incomes. *General Information*, 75(1):163–169, 1972.

[17] M. Graham, I. Hjorth, and V. Lehdonvirta. Digital labour and development: impacts of global digital labour platforms and the gig economy on worker livelihoods. *Transfer*, 23(2):135–162, 2017.

[18] Bin Guo, Yan Liu, Leye Wang, Victor OK Li, Jacqueline CK Lam, and Zhiwen Yu. Task allocation in spatial crowdsourcing: Current state and future directions. *Internet of Things Journal*, 5(3):1749–1764, 2018.

[19] Anjali Gupta, Rahul Yadav, Ashish Nair, Abhijnan Chakraborty, Sayan Ranu, and Amitabha Bagchi. Fairfoody: Bringing in fairness in food delivery. In *AAAI*, pages 11900–11907, 2022.

[20] Aniko Hannak, Claudia Wagner, David García, Alan Mislove, Markus Strohmaier, and Christo Wilson. Bias in online freelance marketplaces: Evidence from taskrabbit and fiverr. In *CSCW*, pages 1914–1933, 2017.

[21] Danula Hettiachchi, Vassilis Kostakos, and Jorge Gonçalves. A survey on task assignment in crowdsourcing. *CSUR*, 55(3):49:1–49:35, 2023.

[22] Manas Joshi, Arshdeep Singh, Sayan Ranu, Amitabha Bagchi, Priyank Karia, and Puneet Kala. Batching and matching for food delivery in dynamic road networks. In *ICDE*, pages 2099–2104, 2021.

[23] K. Jrvelin and J. Keklinen. Cumulated gain-based evaluation of ir techniques. *TOIS*, 20(4):422–446, 2002.

[24] Leyla Kazemi and Cyrus Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In *SIGSPATIAL*, pages 189–198, 2012.

[25] Tinghao Lai, Yan Zhao, Weizhu Qian, and Kai Zheng. Loyalty-based task assignment in spatial crowdsourcing. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1014–1023, 2022.

[26] Jie Li, Yongli Ren, and Ke Deng. Fairgan: Gans-based fairness-aware learning for recommendations with implicit feedback. In *WWW*, pages 297–307, 2022.

[27] Xiang Li, Yan Zhao, Jiannan Guo, and Kai Zheng. Group task assignment with social impact-based preference in spatial crowdsourcing. In *DASFAA*, pages 677–693, 2020.

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

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

[30] Jiaxin Liu, Liwei Deng, Hao Miao, Yan Zhao, and Kai Zheng. Task assignment with federated preference learning in spatial crowdsourcing. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1279–1288, 2022.

[31] Dan Lu, Qilong Han, Hongbin Zhao, and Kejia Zhang. Optimal task recommendation for spatial crowdsourcing with privacy control. In *ICPCSEE*, pages 412–424, 2017.

[32] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. Cross-domain recommendation: An embedding and mapping approach. In *IJCAI*, pages 2464–2470, 2017.

[33] Ashish Nair, Rahul Yadav, Anjali Gupta, Abhijnan Chakraborty, Sayan Ranu, and Amitabha Bagchi. Gigs with guarantees: Achieving fair wage for food delivery workers. In *IJCAI*, pages 5122–5128, 2022.

[34] Gourab K. Patro, Arpita Biswas, Niloy Ganguly, Krishna P. Gummadi, and Abhijnan Chakraborty. Fairrec: Two-sided fairness for personalized recommendations in two-sided platforms. In *WWW*, pages 1194–1204, 2020.

[35] Tuan-Anh Nguyen Pham, Xutao Li, and Gao Cong. A general model for out-of-town region recommendation. In *WWW*, pages 401–410, 2017.

[36] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[37] A. Roegiest, A. Lipani, A. Beutel, A. Olteanu, and T. Kamishima. Facts-ir: Fairness, accountability, confidentiality, transparency, and safety in information retrieval. *SIGIR*, 53(2):20–43, 2019.

[38] Salvatore Scellato, Anastasios Noulas, Renaud Lambiotte, and Cecilia Mascolo. Socio-spatial properties of online location-based social networks. In *AAAI*, volume 5, pages 329–336, 2011.

[39] Ashudeep Singh and Thorsten Joachims. Fairness of exposure in rankings. In *SIGKDD*, pages 2219–2228, 2018.

[40] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-László Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[41] Yongxin Tong, Jieying She, Bolin Ding, Libin Wang, and Lei Chen. Online mobile micro-task allocation in spatial crowdsourcing. In *ICDE*, pages 49–60, 2016.

[42] Yongxin Tong, Yuxiang Zeng, Bolin Ding, Libin Wang, and Lei Chen. Two-sided online micro-task assignment in spatial crowdsourcing. *TKDE*, 33(5):2295–2309, 2019.

[43] Yongxin Tong, Yuxiang Zeng, Zimu Zhou, Lei Chen, and Ke Xu. Unified route planning for shared mobility: An insertion-based framework. *TODS*, 47(1):1–48, 2022.

[44] Yongxin Tong, Zimu Zhou, Yuxiang Zeng, Lei Chen, and Cyrus Shahabi. Spatial crowdsourcing: a survey. *VLDBJ*, 29(1):217–250, 2020.

[45] Hao Wang, Yanmei Fu, Qinyong Wang, Hongzhi Yin, Changying Du, and Hui Xiong. A location-sentiment-aware recommender system for both home-town and out-of-town users. In *SIGKDD*, pages 1135–1143, 2017.

[46] Ziwei Wang, Yan Zhao, Xuanhao Chen, and Kai Zheng. Task assignment with worker churn prediction in spatial crowdsourcing. In *CIKM*, pages 2070–2079, 2021.

[47] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *AAAI*, pages 346–353, 2019.

[48] Yao Wu, Jian Cao, Guandong Xu, and Yudong Tan. TFROM: A two-sided fairness-aware recommendation model for both customers and providers. In *SIGIR*, pages 1013–1022, 2021.

[49] Jinfu Xia, Yan Zhao, Guanfeng Liu, Jiajie Xu, Min Zhang, and Kai Zheng. Profit-driven task assignment in spatial crowdsourcing. In *IJCAI*, pages 1914–1920, 2019.

[50] Haoran Xin, Xinjiang Lu, Tong Xu, Hao Liu, Jingjing Gu, Dejing Dou, and Hui Xiong. Out-of-town recommendation with travel intention modeling. In *AAAI*, pages 4529–4536, 2021.

[51] Yi Xu, Yongxin Tong, Yexuan Shi, Qian Tao, Ke Xu, and Wei Li. An efficient insertion operator in dynamic ridesharing services. *TKDE*, 34(8):3583–3596, 2020.

[52] Guanyu Ye, Yan Zhao, Xuanhao Chen, and Kai Zheng. Task allocation with geographic partition in spatial crowdsourcing. In *CIKM*, pages 2404–2413, 2021.

[53] Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, and Hui Xiong. Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction. In *AAAI*, pages 1186–1193, 2020.

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

[55] Yan Zhao, Jiannan Guo, Xuanhao Chen, Jianye Hao, Xiaofang Zhou, and Kai Zheng. Coalition-based task assignment in spatial crowdsourcing. In *ICDE*, pages 241–252, 2021.

[56] Yan Zhao, Yang Li, Yu Wang, Han Su, and Kai Zheng. Destination-aware task assignment in spatial crowdsourcing. In *CIKM*, pages 297–306, 2017.

[57] Yan Zhao, Jinfu Xia, Guanfeng Liu, Han Su, Defu Lian, Shuo Shang, and Kai Zheng. Preference-aware task assignment in spatial crowdsourcing. In *AAAI*, pages 2629–2636, 2019.

[58] Yan Zhao, Kai Zheng, Yue Cui, Han Su, Feida Zhu, and Xiaofang Zhou. Predictive task assignment in spatial crowdsourcing: A data-driven approach. In *ICDE*, pages 13–24, 2020.

[59] Yan Zhao, Kai Zheng, Jiannan Guo, Bin Yang, Torben Bach Pedersen, and Christian S Jensen. Fairness-aware task assignment in spatial crowdsourcing: Game-theoretic approaches. In *ICDE*, pages 265–276, 2021.

[60] Yan Zhao, Kai Zheng, Yang Li, Han Su, Jiajun Liu, and Xiaofang Zhou. Destination-aware task assignment in spatial crowdsourcing: A worker decomposition approach. *TKDE*, pages 2336–2350, 2019.

[61] Yan Zhao, Kai Zheng, Yunchuan Li, Jinfu Xia, Bin Yang, Torben Bach Pedersen, Rui Mao, Christian S. Jensen, and Xiaofang Zhou. Profit optimization in spatial crowdsourcing: Effectiveness and efficiency. *TKDE*, 2022.

[62] Yan Zhao, Kai Zheng, Hongzhi Yin, Guanfeng Liu, Junhua Fang, and Xiaofang Zhou. Preference-aware task assignment in spatial crowdsourcing: from individuals to groups. *TKDE*, 34(7):3461–3477, 2020.