SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

January 2026, Vol. 69, Iss. 1, 112106:1–112106:17
https://doi.org/10.1007/s11432-024-4473-3

Profit-aware online crowdsensing task assignment for intelligent transportation services

Guanglei ZHU[1], Yafei LI[1,2,3*], Jiawen ZHANG[1], Ke WANG[1,2,3],
Lei CHEN[4] & Mingliang XU[1,2,3*]

[1] *School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China*
[2] *National Supercomputing Center in Zhengzhou, Zhengzhou 450001, China*
[3] *Engineering Research Center of Intelligent Swarm Systems, Ministry of Education, Zhengzhou 450001, China*
[4] *Huawei Noah's Ark Lab, Hong Kong 999077, China*

Received 1 October 2024/Revised 24 February 2025/Accepted 4 June 2025/Published online 4 January 2026

Abstract With the rapid advancement of sensing and wireless technologies, intelligent transportation services (ITSs) have significantly enhanced the quality of public travel by capturing real-time traffic conditions. In practice, ITS platforms face bottlenecks in comprehensively acquiring urban traffic dynamics due to occasional sensor failures and coverage deficiencies, which, in turn, affect the quality of ITSs. To address these issues, crowdsensing, as an emerging computing paradigm, assigns traffic status sensing tasks (e.g., taking photos) to vehicle-based mobile participants (a.k.a. workers), improving the timeliness and spatial coverage of traffic monitoring. The above process raises a hot topic, i.e., online crowdsensing task assignment. Most existing methods primarily consider travel costs as the basic factor for pricing models to incentivize workers to complete crowdsensing tasks. However, due to the neglect of supply-demand dynamics in task pricing, these methods still suffer from imbalanced distributions of workers and tasks. In this paper, we propose a novel profit-aware online crowdsensing task assignment (POCTA) problem, which aims to maximize overall revenue by incorporating a supply-demand-aware pricing model. This model dynamically adjusts task prices based on current and predicted supply-demand conditions. We develop an efficient two-stage framework, predict-then-assign, to solve the POCTA problem. In the prediction stage, we build the end-to-end multi-view spatio-temporal attention network to predict the distributions of future crowdsensing tasks. In the matching stage, we propose the break-and-rematch online task assignment algorithm, which iteratively invokes a packing-aware matching operator and an adaptive assignment-breaking operator to optimize task assignment. Extensive experiments on two real-world datasets validate the efficiency and effectiveness of our proposed solutions.

Keywords location-based services, sequence prediction, task assignment, crowdsensing, intelligent transportation

Citation Zhu G L, Li Y F, Zhang J W, et al. Profit-aware online crowdsensing task assignment for intelligent transportation services. Sci China Inf Sci, 2026, 69(1): 112106, https://doi.org/10.1007/s11432-024-4473-3

1 Introduction

With the development of sensing and wireless technologies, intelligent transportation services (ITSs) in smart cities continue to improve the quality of life for urban residents. However, despite the widespread deployment of costly devices (e.g., sensors and cameras) and tracking infrastructure (e.g., monitors and radars), ITSs still face bottlenecks in comprehensively collecting ITS-related information (e.g., traffic conditions and accident reports). These bottlenecks arise from issues such as occasional sensor failures and blind spots caused by insufficient coverage, especially during emergencies like large-scale power outages [1–3]. To address these limitations, as illustrated in Figure 1, crowdsensing (a manual way) has emerged as a promising service paradigm [1, 4, 5]. It leverages a large number of transportation participants equipped with smart devices to collect traffic data, thereby effectively compensating for the lack of comprehensive traffic information. A typical crowdsensing service involves three key parties: crowdsensing tasks, vehicle-based mobile workers (workers in short), and the platform, where mobile workers carrying smart devices travel to designated locations to complete crowdsensing tasks assigned by the platform. This process raises a central problem in ITSs, online crowdsensing task assignment [5–7], which aims to determine the optimal matching between mobile workers and crowdsensing tasks.

In the literature, current studies related to online crowdsensing task assignment can be divided into two categories based on optimization objectives: maximizing the number of completed tasks and maximizing overall profit. Methods in the first category (e.g., [3,8]) prioritize improving the task completion ratio. However, these approaches may

[*] Corresponding author (email: ieyfli@zzu.edu.cn, iexumingliang@zzu.edu.cn)

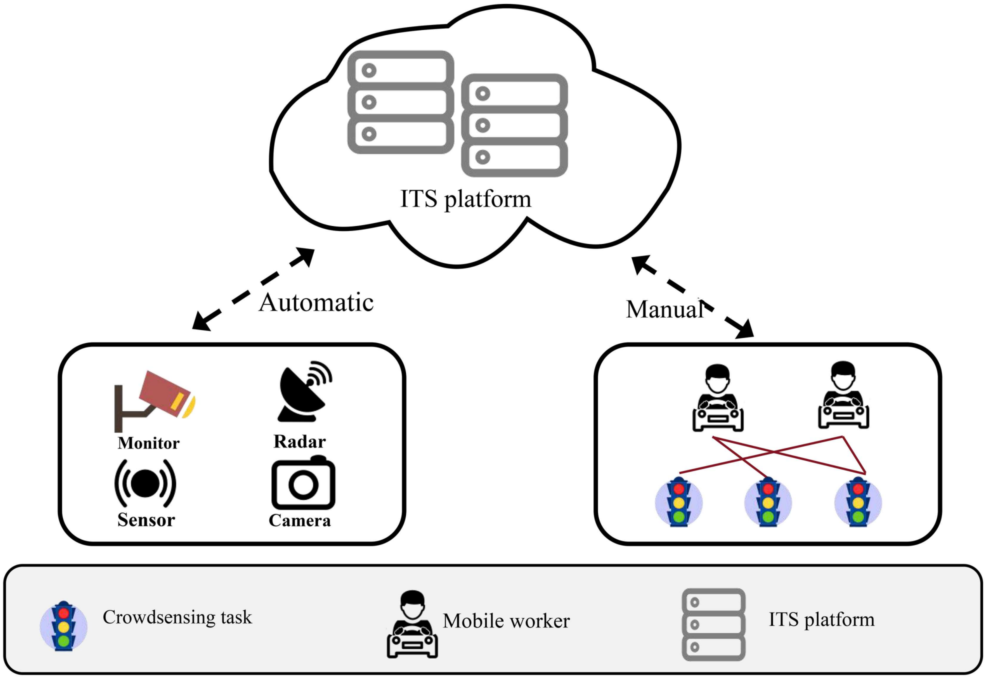

Figure 1 (Color online) Illustration of two ways to collect ITS-based information.

neglect economic sustainability, as completing more tasks does not necessarily lead to higher profitability for the platform. In practice, commercial platforms often emphasize maximizing profits when assigning tasks. Therefore, profit-oriented strategies (e.g., [5, 9, 10]) explicitly aim to optimize platform revenue. Several studies [5, 10–15] explore flexible pricing models that incentivize both platforms and mobile workers by taking into account the real-time states of workers and tasks. Zhao et al. [10] proposed a task pricing model that incorporates temporal constraints to encourage the completion of near-expiry tasks. Li et al. [11] introduced a fair pricing mechanism in which each task receives a discount from its original distance-based cost. These profit-driven mechanisms undoubtedly motivate workers and improve platform revenues to some extent. However, the supply-demand dynamics of crowdsensing tasks critically influence service pricing. Adaptive task pricing based on this relationship can guide mobile workers toward areas with a surplus of tasks but a shortage of workers, thereby alleviating service delays caused by insufficient workforce availability. Despite the significance of supply-demand dynamics, recent studies [10, 11, 14] on ITS-based task assignment have seldom incorporated this factor into their pricing strategies in an effective manner. Besides, given a batch of workers and tasks, existing studies [5, 9, 10, 14] typically derive the current optimal matching plan in a one-shot fashion based on the optimization objective. However, such strategies often overlook long-term benefits, which may lead to suboptimal assignment quality. Since prior assignments may influence future matching pairs, adaptively breaking low-quality assignments allows workers to be reassigned to more suitable tasks in subsequent rounds.

In this paper, we investigate the profit-aware online crowdsensing task assignment (POCTA) problem, which involves incentivizing workers to move toward worker-undersupplied (or high-demand) areas through demand-supply-aware pricing. Given a set of vehicle-based mobile workers and a set of crowdsensing tasks, the goal of the POCTA problem is to find the optimal task assignment by maximizing overall profit. However, solving the POCTA problem with supply-demand-aware pricing faces two main challenges. First, an effective pricing model relies on predicting the demand for future tasks, but accurately forecasting task distributions from complex spatiotemporal dependencies is non-trivial. Second, crowdsensing tasks and mobile workers arrive at the platform dynamically in the form of streams, making it difficult to find an optimal solution in an online scenario. Thus, designing efficient matching algorithms to assign tasks to mobile workers is another challenge. To address these challenges, we propose an efficient two-stage framework, predict-then-assign (PTA), which integrates task prediction and task assignment. Specifically, in the task prediction stage, we construct an end-to-end, deep learning-based network, the multi-view spatio-temporal attention network (MVSTAnet), to predict the arrival patterns of future crowdsensing tasks. The pricing model then dynamically adjusts task prices based on supply-demand conditions to incentivize mobile workers to complete the assigned tasks. In the task assignment stage, we devise the deep reinforcement learning (DRL)-based break-and-rematch online task assignment (BROTA) algorithm, which iteratively uses the matching and breaking operators to adaptively assign tasks to the most suitable workers. Our main contributions are summarized as follows.

- We study the POCTA problem based on a supply-demand-aware pricing model, which aims to find the best task assignment with the goal of maximizing the platform's overall profit. We also theoretically analyze the hardness of the POCTA problem.
- We develop a powerful multi-view task prediction model, MVSTAnet, to predict the arrival regions of potential tasks. In MVSTAnet, we introduce the dual-path residual connection GCN module to capture both similarity and geographic dependencies among spatial regions. We also design the dual-channel spatio-temporal attention module to effectively extract short-term local and long-term global spatiotemporal dependencies.
- We propose an efficient DRL-based task assignment algorithm, BROTA, which consists of matching and breaking operators. BROTA iteratively breaks assignments with lower profit and higher incurred supply-demand gaps, then rematches the broken assignments to optimize task assignment from a long-term perspective.
- Extensive experiments on two real-world datasets validate the desirable performance of our solutions across a range of parameter settings.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 formally defines the POCTA problem. Section 4 presents the proposed solutions. Extensive experiments are presented in Section 5, followed by the conclusion and future work in Section 6.

2 Literature review

In this section, we survey the related studies on spatiotemporal task prediction and online task assignment.

2.1 Spatiotemporal task prediction

In the field of ITSs, spatiotemporal task prediction [3, 16] involves predicting future task arrival patterns based on intricate spatial and temporal dependencies. Deep learning methods are powerful in learning complex feature representations from large-scale data and have been widely used to address the above challenges [17–19]. Recurrent neural network (RNN) and its variations, such as long short-term memory (LSTM) and gated recurrent unit (GRU), have presented their powerful capabilities in capturing temporal dependencies [17, 20, 21]. Additionally, graph convolutional network (GCN) [17,18] has been applied to handle graph-structured data, effectively capturing spatial dependence. While these methods have shown promising results, they often oversimplify the correlations between spatial and temporal features as separate components rather than interdependent factors. To this end, recent advances [17,21] focus on developing integrated frameworks that merge multiple modules to capture diverse features, including spatial, temporal, and contextual. Zhao et al. [17] introduced an integration network, the temporal graph convolutional network (T-GCN), which combines GCN and GRU to achieve remarkable performance in traffic forecasting. However, these models tend to overlook the differential contributions of temporal and spatial features in the prediction process. To address this gap, attention mechanisms have been increasingly employed to assign varying weights to spatial and temporal features. Zheng et al. [18] proposed the graph multi-attention network (GMAN), comprising multiple spatiotemporal attention blocks to address distinct influences of spatiotemporal factors. In light of these studies, we aim to develop an integrated DL-based model that accurately predicts the arrival regions of future tasks. Different from existing models [3, 17, 18], our proposed MVSTAnet emphasizes the dynamic correlations between spatial and temporal features, further improving prediction accuracy.

2.2 Online task assignment

Online task assignment refers to finding suitable workers for sequentially arriving tasks. In the literature, some approaches focus on maximizing the number of completed tasks [3], while others prioritize maximizing overall profit [5, 9, 10]. However, mainstream methods often yield sub-optimal solutions by focusing primarily on local task assignment and overlooking the potential benefits of long-term predictions [22, 23]. For commercial platforms, profit optimization is a top priority, prompting many studies [3, 12, 20, 24] to incorporate long-term predictive information to improve short-term task assignment. Tong et al. [15] proposed dynamically adjusting task prices based on future demand and supply distributions, and Ren et al. [25] designed a hybrid batch processing framework to optimize task assignment through worker payment adjustments aligned with supply-demand dynamics. While these methods enhance task assignment, they fail to develop a supply-demand-aware pricing model that incentivizes workers to move to high-demand areas. Such guidance could improve task matching quality and increase overall profit. In this paper, our POCTA framework introduces a demand-supply-guided pricing model that encourages workers to

Table 1 Notation and description.

Notation	Description
$\mathcal{T}$	A set of crowdsensing tasks
W	A set of mobile workers
$\mathcal{G}$	A graph
δ	The packing threshold of packing policy
P/F	The number of past/future time steps
$\pi(\cdot,\cdot)$	The shortest distance of two locations
$\mathrm{price}(\tau)$	The supply-demand-aware price of a crowdsensing task
$u(\cdot,\cdot)$	The matching revenue of an assignment
$\mathcal{U}(M)$	The total profit of the platform
$X_t/\hat{Y}_{t+1}$	The input/output sequence
$\Delta g(\cdot)$	The supply-demand gap value

relocate, thereby maximizing long-term profit. This distinguishes existing studies [10, 11, 13] from the goal of our POCTA problem.

3 Models and problem formulation

In this section, we first present several preliminary definitions, then formally define the POCTA problem and prove its hardness. Table 1 lists the main notations of this paper.

3.1 System model

In ITSs, the general system model for the POCTA problem involves crowdsensing tasks, mobile workers, and the ITS platform, operating as follows: crowdsensing tasks dynamically arrive at the platform with essential information, such as location and fare. Mobile workers periodically update their status, e.g., current location, to the ITS platform. The platform then calculates the matching values (e.g., matching revenue) for candidate matching pairs, and finds suitable vehicle-based mobile workers for crowdsensing tasks while maximizing the overall profit of the ITS platform.

In our system model, we establish the POCTA problem on a road network represented by the undirected graph $\mathcal{G}_n = (V_n, E_n)$, where V_n denotes the set of road intersections and E_n denotes the set of road segments. Each road segment $e_{ij} \in E_n$ connects two road intersections l_i and l_j $(l_i, l_j \in V_n)$. We next introduce the definitions of crowdsensing task and vehicle-based mobile worker.

Definition 1 (Crowdsensing task)**.** A crowdsensing task is denoted by a four-entry tuple $\tau = (l_\tau, t_\tau, t_\tau^d,\ f_\tau)$, indicating that τ is published on the platform at time t_τ, and locates at the road intersection $l_\tau \in V_n$ with a published fare f_τ and expire time t_τ^d.

Note that each crowdsensing task (task in short) τ must be assigned to an available worker before its deadline t_τ^d or cannot be allocated thereafter. Following [5, 25], we omit the task processing time, i.e., once the assigned worker arrives at l_τ, the task τ is considered to be completed.

Definition 2 (Vehicle-based mobile worker)**.** A vehicle-based mobile worker $w \in W$ is denoted as a five-entry tuple $w = (l_w, c_w, t_w, t_w^d, \mathcal{T}_w)$, where $l_w \in V_n$ is the current location, t_w is the arriving time, t_w^d is the leaving time, c_w is the maximum number of tasks that w expects to complete and $\mathcal{T}_w$ represents the set of tasks assigned to w but not yet completed.

Mobile workers equipped with vehicles (workers in short) can travel freely on the road network $\mathcal{G}_n$. For ease of presentation, we denote $\bar{c}_w = c_w - |\mathcal{C}_w|$ as the remaining maximum expected number of tasks that w could perform, where $\mathcal{C}_w$ is a set recording completed tasks. Following the settings in [5, 22, 23, 26], a worker goes offline either after completing c_w tasks (i.e., $\bar{c}_w = 0$) or after deadline time t_w^d.

Definition 3 (Schedule)**.** Given a worker w with m assigned tasks $\mathcal{T}_w = \{\tau_1, \ldots, \tau_m\}$, the schedule S_w of w is denoted by $S_w = (l_{\tau_1}, \ldots, l_{\tau_{|\mathcal{T}_w|}})$, where $l_\tau \in V_n$ is a location that will be reached by w.

Following existing studies [5, 9, 14], we assume that workers always follow the shortest path between two locations in their schedules. In addition, we do not reorder tasks in the current schedule S_w when inserting a new task, as reordering all locations in S_w incurs significant computational overhead and is unsuitable for real-time scenarios where task insertions occur frequently [5, 9, 14]. We consider S_w is valid if it satisfies the following constraints for all $\tau \in \mathcal{T}_w$.

- Deadline constraint. The worker w must arrive at l_τ before task deadline t_τ^d and worker deadline t_w^d.

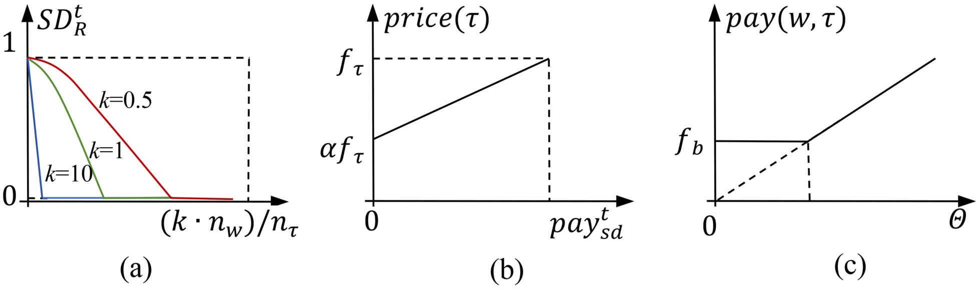

Figure 2 (Color online) Curves of SD_R^t, price(τ) and pay(w,τ). (a) Supply-demand degree; (b) price model; (c) worker payment.

- Maximum expected number constraint. The cumulative assigned tasks $\mathcal{C}_w$ of w cannot exceed the maximum expected number c_w, i.e., $\bar{c}_w \geqslant 0$.
- Invariable constraint. Once a task τ is assigned to the worker w, it cannot be changed.

3.2 Price model

A key objective of ITS platforms is to incentivize more workers to complete tasks while maximizing the platform's profit [5,9,15,27]. Intuitively, when a region is undersupplied with workers, the platform tends to raise task prices to attract available workers. To this end, we design a supply-demand-aware pricing model based on two core insights: (i) task prices should accurately reflect changes in supply and demand; and (ii) pricing should effectively incentivize workers to relocate to undersupplied regions, thereby reducing supply-demand gaps and increasing platform profit.

Specifically, we divide the road network $\mathcal{G}_n$ into N non-overlapping spatial regions $\mathcal{R} = \{R_1, \ldots, R_N\}$, where each region is a subgraph of $\mathcal{G}_n$ (i.e., $R = (V_R, E_R)$ with $V_R \subseteq V_n$ and $E_R \subseteq E_n$), and is responsible for reporting its supply and demand conditions. We assume that all tasks within the same region and time share the same supply-demand status. To quantify these conditions in region R at time t, we employ the supply-demand degree SD_R^t [25], which is calculated as

$$\mathrm{SD}_R^t = \begin{cases} 0, & \text{if } k \cdot n_w \geqslant n_\tau \text{ or } n_\tau = 0, \\ 1, & \text{if } n_w = 0 \text{ and } n_\tau \neq 0, \\ -\tanh\left(\ln \dfrac{k \cdot n_w}{n_\tau}\right), & \text{otherwise}, \end{cases} \tag{1}$$

where n_w and n_τ denote the numbers of workers and tasks in region R, respectively; k is a parameter that adjusts for the relative numbers of workers and tasks; and t refers to the t-th time step (i.e., a time interval). Figure 2(a) illustrates how the value of SD_R^t varies with different values of k. We approximate k using the average capacity $\bar{c}_w$ of workers in region R, calculated as $k = \sum_{i=1}^{n_w} \bar{c}_{w_i}/n_w$, which can be estimated from historical data. Based on the supply-demand degree, we formally introduce our supply-demand-aware pricing model.

Definition 4 (Supply-demand-aware price)**.** Given a task τ with fare f_τ, the supply-demand-aware price price(τ) at the time step t comprises basic price pay_f and supply-demand-based price $\mathrm{pay}_{\mathrm{sd}}^t$, which is calculated as

$$\mathrm{price}(\tau) = \alpha \cdot \mathrm{pay}_f + (1-\alpha) \cdot \mathrm{pay}_{\mathrm{sd}}^t. \tag{2}$$

The curve of price(τ) is shown in Figure 2(b). Clearly, price(τ) is anchored by the base price pay_f and primarily influenced by the supply-demand-based price $\mathrm{pay}_{\mathrm{sd}}^t$. In practice, the platform can proactively adjust the preference coefficient α to control the relative contributions of these two components [5]. The base price pay_f may account for multiple factors such as task completion time [10,14]. For simplicity, we set it equal to the task fare, i.e., $\mathrm{pay}_f = f_\tau$. The supply-demand-based price $\mathrm{pay}_{\mathrm{sd}}^t$ is calculated based on both current and predicted future supply-demand conditions, i.e.,

$$\mathrm{pay}_{\mathrm{sd}}^t = f_\tau . \sum_{i=0}^{F} \lambda^i \Phi_{\mathrm{sd}}(\mathrm{SD}_{R_\tau}^{t+i}), \tag{3}$$

where $F \in \mathbb{Z}^+$ denotes the number of future time steps, and $\Phi_{\mathrm{sd}}(\mathrm{SD}_{R_\tau}^t) = \mathrm{SD}_{R_\tau}^{t+i} / \sum_{j=0}^{F} \mathrm{SD}_{R_\tau}^{t+j}$ represents the supply-demand-based contribution ratio ($i, j \in \mathbb{Z}^+$). According to (3), there are two cases for determining $\mathrm{pay}_{\mathrm{sd}}^t$. The first case where at most one SD_R^{t+j} is non-zero, i.e., $\sum_{i=0}^{F} \mathrm{SD}_{R_\tau}^{t+i} = \mathrm{SD}_{R_\tau}^{t+j}$, also includes two conditions. (i) All supply-demand levels are zero. Since no supply-demand level affects the price, the supply-demand-based price pay_{sd}^t is 0. (ii) Only one supply-demand level is non-zero. The pay_{sd}^t is determined by the only non-zero value SD_R^{t+j}. In the common case, where more than one SD_R^{t+i} are non-zero, we proportionally weight their contributions to pay_{sd}^t using $\Phi_{\text{sd}}(\cdot)$. In addition, since the influence of future predictions should decrease with time, we incorporate an attenuation coefficient λ to discount the impact of SD_R^{t+i} at later time steps [9]. To illustrate the operation of supply-demand-aware pricing more intuitively, we provide a running example as follows.

Example 1. Consider a matching pair (w, τ), where task τ is located in the region R with a fare $f_\tau = 10$. We set $\alpha = 0.5$, $\lambda = 0.8$, and $F = 2$. Suppose that the values of SD_R^{t+i} varying $i \in [0, F]$ are 1, 0 and 0.5. According to (3), the SD-based contribution ratios $\Phi_{\text{sd}}(\cdot)$ varying time steps $i \in [0, F]$ are 0.67, 0, and 0.33. The supply-demand-aware price for the worker w is $\text{price}(\tau) = 0.5 \times 10 + 0.5 \times 10 \times (0.8^0 \times 0.67 + 0.8^1 \times 0 + 0.8^2 \times 0.33) = 5 + 5 \times 0.88 = 9.4$.

3.3 Problem formulation

Based on the aforementioned system and pricing models, in what follows, we first define the objective function (i.e., matching revenue) of the POCTA problem, which is used to evaluate the quality of each worker-task assignment. We then formally define the POCTA problem.

Definition 5 (Matching revenue)**.** Given a mobile worker w and a crowdsensing task τ, the matching revenue of matching pair $(w, \tau) \in W \times \mathcal{T}$ can be calculated as $u(w, \tau) = f_\tau - \text{pay}(w, \tau)$, where $\text{pay}(w, \tau)$ is the payment of w.

Since workers exhibit varying levels of efficiency and quality in task execution, we consider worker payment $\text{pay}(w, \tau)$ to be determined by multiple factors, including supply-demand conditions, detour distance, and task remaining time, which is calculated by

$$\text{pay}(w, \tau) = \text{price}(\tau)\Big(\beta\Delta\pi + (1 - \beta)\Delta t_w\Big), \tag{4}$$

where $\text{price}(\tau)$ denotes the supply-demand-aware price, and $\Theta = \beta\Delta\pi + (1 - \beta)\Delta t_w$ is a discount function with range $[0, 1]$. Here, β is a parameter that balances the influence of detour distance and task urgency. Specifically, $\Delta\pi(w, \tau)$ represents the detour ratio incurred by inserting task τ into the current schedule S_w of w, and is computed as $\Delta\pi(w, \tau) = \min_{1 \leqslant i \leqslant |S_w|-1} 1 - \frac{\pi(l_i, l_{i+1})}{\pi(l_i, l_\tau) + \pi(l_\tau, l_{i+1})}$, where $\pi(\cdot, \cdot)$ is the shortest distance between two locations; and Δt_w is the task response ratio, which is calculated as $\Delta t_w = 1 - \frac{t_\tau^d - t_c}{t_\tau^d - t_\tau}$, where t_c is the current time. Obviously, in our worker payment model, the platform's profit is positively influenced under the following three conditions: (i) the supply-demand condition is globally balanced or oversupplied; (ii) the detour distance between workers and tasks is minimized; and (iii) tasks are assigned as early as possible.

As stated above, the platform's profit is inversely related to worker payments; thus, maximizing overall profit entails minimizing total payments to workers. According to (4), the worker payment lies within the range $[0, \text{price}(\tau)]$. However, intuitively, workers should not be unpaid or underpaid. To address this, we introduce a guaranteed payment f_b for completing task τ to ensure a minimum benefit for each worker. For simplicity, we assume f_b is positively correlated with the task fare, i.e., $f_b = \epsilon f_\tau$, where ϵ is a tunable scaling factor. As illustrated in Figure 2(c), when $\text{pay}(w, \tau) < f_b$, the platform pays the worker w the guaranteed amount f_b; otherwise, the worker receives $\text{pay}(w, \tau)$ and thereby gains additional rewards. We next provide the formal definition of task assignment.

Definition 6 (Task assignment)**.** Given a set of tasks $\mathcal{T}$ and a set of workers W, task assignment refers to the platform finding a matching plan $M = \{(w, \tau) | w \in W, \tau \in \mathcal{T}\}$ that satisfying both workers' and tasks' constraints.

In the POCTA problem, tasks arrive at the platform dynamically in the form of streams [14, 28]. Following the method proposed in [5, 9, 14, 26], we divide the stream of tasks into batches $(b_1, \ldots, b_n)$ and assign tasks at the end of each batch. Any unassigned tasks in the current batch b_i will be handled in the subsequent batch b_{i+1}. Based on the system and price models defined above, the POCTA problem can now be formally defined as follows.

Definition 7 (POCTA problem)**.** Given a set of crowdsensing tasks $\mathcal{T}$ and a set of vehicle-based mobile workers W, the POCTA problem aims to find the best task assignment M with platform profit

$$\mathcal{U}(M) = \sum_{(w,\tau) \in M} u(w, \tau), \tag{5}$$

s.t., (i) $\mathcal{U}(M) \geqslant \max_{M_i \in \mathcal{M}} \mathcal{U}(M_i)$, (ii) $\sum_{w_i \in W} \mathbb{I}(w_i, \tau) \leqslant 1$, (iii) $\sum_{\tau_i \in \mathcal{T}_w} \mathbb{I}(w, \tau_i) \leqslant \bar{c}_w$, where $\mathcal{M}$ records all task assignment plans, $\mathcal{T}_w \subseteq \mathcal{T}$ denotes the tasks assigned to worker w, and $\mathbb{I}(\cdot, \cdot)$ is an indicator function that returns 1 if task τ is assigned to worker w and 0 otherwise. Eq. (i) indicates that the task assignment plan M is optimal in $\mathcal{M}$, maximizing the platform's profit $\mathcal{U}(M)$; Eq. (ii) ensures that each task τ is assigned to at most one worker [5, 9, 14];

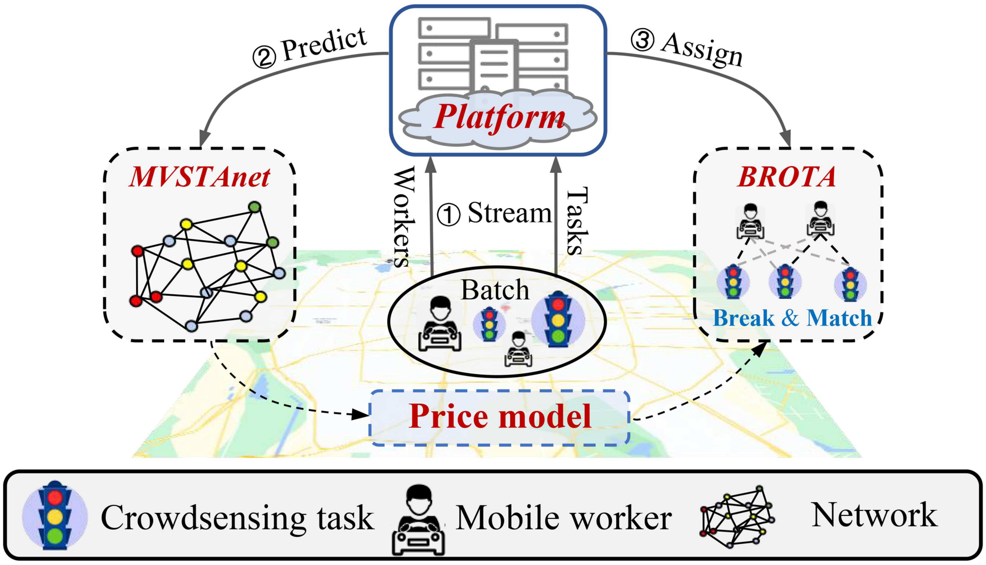

Figure 3 (Color online) Overview of the PTA framework for the POCTA problem.

Eq. (iii) states that a worker w can be assigned one or multiple tasks, as long as their schedule S_w remains valid after inserting the assigned tasks.

We finally theoretically analyze the complexity of the POCTA problem in Theorem 1.

Theorem 1. The POCTA problem is NP-hard.

Proof. The proof is shown in Appendix A.

4 Solutions for POCTA problem

In this section, we first present the PTA framework and then detail our proposed solutions.

4.1 Framework overview

As shown in Figure 3, the PTA framework is designed to effectively address the POCTA problem by maximizing the platform's profit. It comprises two main stages: task prediction and task assignment. Specifically, ① given a stream of crowdsensing tasks and vehicle-based mobile workers, ② PTA first employs the DL-based MVSTAnet to predict the future spatiotemporal distribution of tasks at the regional level. Then the price model adaptively adjusts the task price based on the predicted supply-demand conditions to incentivize workers to move with demand orientation. In the second stage, ③ PTA utilizes the BROTA algorithm, which iteratively applies a packing-aware matching operator and an adaptive assignment-breaking operator to identify the optimal task assignment.

4.2 Multi-view spatio-temporal attention network

The supply-demand-aware price model suggests that we need to predict both supply and demand conditions at the regional level. For worker supply prediction, we adopt the arrival time method introduced in [12], which has been shown to be effective because workers are traceable and their mobility patterns are predictable. For task demand prediction, we propose the DL-based MVSTAnet, which effectively captures three types of correlations. (i) Temporal correlation: the demand in a given time step is often influenced by previous time steps. For example, in the context of crowdsensing, if demand (e.g., detecting congestion) increases in previous time steps, it is likely to continue increasing in subsequent steps. (ii) Spatial correlation: the demand conditions of adjacent or similar regions may affect each other. For instance, when demand (e.g., taking pictures) surges in a city's central business district, it often triggers increased demand in nearby regions due to spatial and functional connectivity. (iii) Spatiotemporal correlation: the demand conditions of a region at time step t may be influenced by its adjacent regions in earlier time steps. For instance, if traffic conditions worsen in a region, commuters may seek alternative routes in surrounding areas, thereby increasing demand (e.g., detecting congestion) in neighboring regions during subsequent time steps.

The MVSTAnet structure, shown in Figure 4, comprises three main components: the temporal correlation view $\mathcal{V}_T$, spatial correlation view $\mathcal{V}_S$, and spatiotemporal correlation view $\mathcal{V}_{ST}$. MVSTAnet takes as input historical

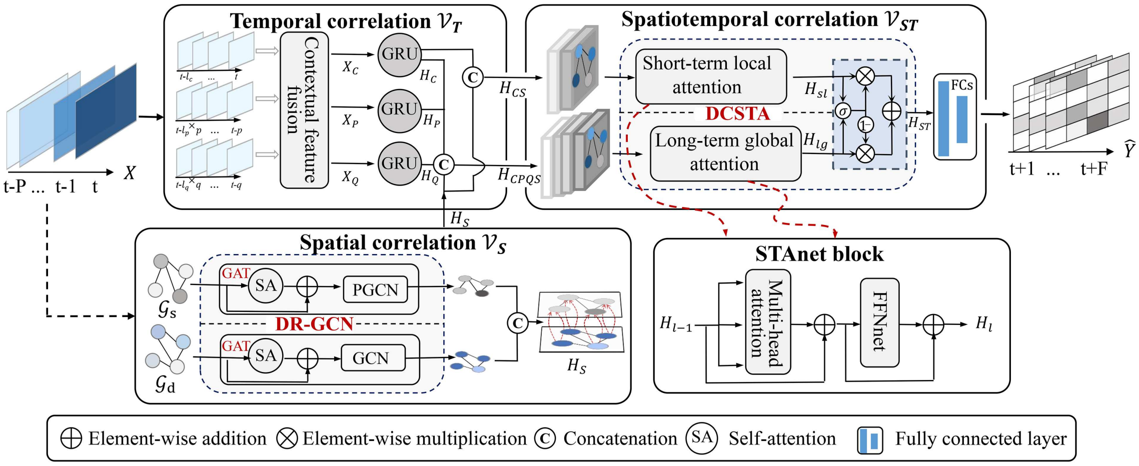
Temporal correlation $\mathcal{V}_T$
t-l_c ... t
t-l_p×p ... t-p
t-l_q×q ... t-q
Contextual feature fusion
X_C
X_P
X_Q
GRU
GRU
GRU
H_C
H_P
H_Q
C
C
H_{CS}
H_{CPQS}
t-P ... t-1 t
X
H_S
Spatiotemporal correlation $\mathcal{V}_{ST}$
Short-term local attention
Long-term global attention
DCSTA
H_{sl}
H_{lg}
σ
1−
H_{ST}
FCs
t+1 ... t+F
$\hat{Y}$
Spatial correlation $\mathcal{V}_S$
$\mathcal{G}_s$
$\mathcal{G}_d$
GAT
SA
PGCN
DR-GCN
GAT
SA
GCN
C
H_S
STAnet block
H_{l-1}
Multi-head attention
FFNnet
H_l
⊕ Element-wise addition ⊗ Element-wise multiplication © Concatenation (SA) Self-attention Fully connected layer

Figure 4 (Color online) Overview of the MVSTAnet.

task observations $[X_{t-P},\ldots,X_t]\in\mathbb{R}^{P\times N}$, the road network $\mathcal{G}_n=(V_n,E_n)$, and auxiliary factors $[\psi_{t-P},\ldots,\psi_t]$ (e.g., weather and events), to predict future task distributions $[\hat{Y}_{t+1},\ldots,\hat{Y}_{t+F}]\in\mathbb{R}^{F\times N}$, where N is the number of regions, and P and F denote the numbers of past and future time steps, respectively. Specifically, $\mathcal{V}_T$ takes three types of temporal sequences as input, closeness (X_C), period (X_P), and trend (X_Q), and captures their hidden temporal features (H_C, H_P, and H_Q) using GRUs. In $\mathcal{V}_S$, we construct the similarity-aware graph $\mathcal{G}_s$ and distance-aware graph $\mathcal{G}_d$, and propose the dual-path residual connection GCN (DR-GCN) module to capture the similarity and geographic correlations of regions in the road network. In $\mathcal{V}_{ST}$, we integrate temporal and spatial features (H_{CS} and H_{CPQS}), and feed them into the dual-channel spatio-temporal attention (DCSTA) module to extract both short-term local ($H_{\rm sl}$) and long-term global ($H_{\rm lg}$) spatiotemporal dependencies. Finally, the spatiotemporal features H_{ST} are projected into the prediction results via fully connected layers (FCs). For model training, we use the loss function ${\rm loss}=||Y-\hat{Y}||_2+\omega L_{\rm reg}$ to measure the error between predicted results $[\hat{Y}_{t+1},\ldots,\hat{Y}_{t+F}]$ and ground truth $[Y_{t+1},\ldots,Y_{t+F}]$, where ω is a hyperparameter and $L_{\rm reg}$ is the L_2 regularization term used to alleviate overfitting.

Temporal correlation modeling view $\mathcal{V}_T$. Given a sequence of historical data, we divide it into three segments along the temporal dimension [21]: recent, daily, and weekly periodic time steps. These segments generate three types of temporal sequences: closeness $X_C=[X_{t-l_c},X_{t-(l_c-1)},\ldots,X_t]$; period $X_P=[X_{t-l_p\times p},X_{t-(l_p-1)\times p},\ldots,X_{t-p}]$; and trend $X_Q=[X_{t-l_q\times q},X_{t-(l_q-1)\times q},\ldots,X_{t-q}]$. Here, l_c, l_p, and l_q denote the number of time steps; p is the daily time interval (e.g., 24 h per day); and q is the weekly interval (e.g., 24 × 7 h per week).

Since task release is significantly influenced by external factors, we use the contextual factor fusion (CFF) module to integrate three external factors, i.e., weather, date (day of the week), and time (time of day), into the historical data using feature embedding techniques, followed by FCs. We then introduce GRU models to capture the temporal dependencies hidden in X_C, X_P, and X_Q. GRU is a commonly used model for processing time series data, employing multiple gates to regulate the retention and removal of features within a sequence.

Spatial correlation modeling view $\mathcal{V}_S$. Accurate prediction of task demand requires effective modeling of intricate spatial dependencies hidden in input sequences. According to the functional characteristics and physical proximity of regions [16, 20, 21], there are two main types of spatial correlations: semantic relationships and geographical distribution. Specifically, semantic relationships describe that similar functional regions often exhibit similar demand changes; geographical distribution points that the proximity of regions affects task distribution due to the mobility of tasks and workers. To this end, we construct two graphs, namely, the similarity-aware graph and the distance-aware graph, to model the above two types of spatial correlations.

- The similarity-aware graph is denoted as $\mathcal{G}_s=(\mathcal{R},A_s)$, where $\mathcal{R}\in\mathbb{R}^N$ is the region set, $A_s\in\mathbb{R}^{N\times N}$ is similarity matrix of regions. We calculate each edge weight $A^s_{i,j}\in A_s$ using the cosine similarity between feature vectors $\boldsymbol{R}_i$ and $\boldsymbol{R}_j$, which incorporate properties such as points of interests' (POIs in short) types, the number of POIs of each type, and historical demand observations.
- The distance-aware graph is denoted as $\mathcal{G}_d=(\mathcal{R},A_d)$, where $A_d\in\mathbb{R}^{N\times N}$ is proximity matrix of N regions $\mathcal{R}$. The weight $A^d_{i,j}\in A_d$ of each edge is determined by the Euclidean distance between regions R_i and R_j.

It is important to note that $\mathcal{G}_s$ and $\mathcal{G}_d$ can also be extended to other graphs according to specific prediction targets, such as the trajectory similarity graph and the connectivity-aware graph [16, 17, 20, 21]. Since $\mathcal{G}_s$ and $\mathcal{G}_d$ are non-Euclidean graph structures that lack translation invariance, traditional discrete convolution, which relies on translation invariance, cannot be directly applied to non-Euclidean data. GCNs consisting of multiple convolutional layers are effective at processing such non-Euclidean structures [16–18]. They operate directly on the structural information of graphs and capture local patterns by aggregating features from neighboring nodes. However, traditional GCNs mainly capture local spatial correlations, which may limit their ability to model global dependencies. This is because even if some nodes are far apart in the graph structure, there may still be functional similarities.

We hence propose the DR-GCN module, which mainly exploits the graph attention network (GAT) to effectively model semantic and geographical spatial correlations. GAT usually comprises a set of self-attention (SA) layers and GCN layers [16, 29]. We also apply residual connection after SA layers to alleviate gradient vanishing and stabilize network training. Moreover, for modeling spatial dependency in $\mathcal{G}_d$, we use a standard GCN that performs a 1-hop diffusion process in each layer, enriching a node's feature vector by linearly combining the features of its neighbors. The GCN layer is defined as

$$h_{l+1} = \sigma\left(D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}h_l W_l\right), \tag{6}$$

where W_l is a trainable weight matrix, and $D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}$ is the symmetric normalized laplacian with $\hat{A} = D - A$. For modeling spatial dependency in $\mathcal{G}_s$, we adopt a globally consistent convolution approach, termed PGCN, which uses point-wise mutual information (PPMI) to encode semantic information and capture global correlations between nodes [30, 31]. The PGCN layer is formulated as

$$h_{l+1} = \sigma\left(D^{-\frac{1}{2}}PD^{-\frac{1}{2}}h_l W_l\right), \tag{7}$$

where P is a PPMI matrix calculated from random walks in the graph [30]. After capturing both semantic and geographical spatial features using GCN and PGCN, respectively, we finally concatenate them to obtain the fused spatial representation H_S.

Spatiotemporal correlation modeling view $\mathcal{V}_{ST}$. In the context of our POCTA framework, task prediction involves two primary types of spatiotemporal correlations: short-term local and long-term global correlations. Short-term correlations capture the influence of adjacent times and regions on future task prediction, while long-term correlations reflect the impact of semantically related but distant times and regions. To this end, we design the DCSTA module to assign dynamic weights to the spatiotemporal features associated with both types of correlations.

For short-term local spatiotemporal correlation, we concatenate the closeness temporal feature H_C and spatial feature H_S into short-term spatiotemporal feature H_{CS}. For long-term global spatiotemporal correlation, we concatenate closeness, period, and trend features H_C, H_P, H_Q and spatial feature H_S into long-term spatiotemporal feature H_{CPQS}. We feed both H_{CS} and H_{CPQS} into the DCSTA module, which primarily consists of the spatiotemporal network (STAnet) block and a gated fusion technique. The STAnet block is employed to assign adaptive attention weights to the spatiotemporal features across various spatial regions and temporal steps. Specifically, as shown in Figure 4, given the input feature H_{l-1} of the $(l-1)$-th block, the STAnet block first uses the multi-head attention mechanism to learn the feature representation of multiple feature spaces. Then, a two-layer feed forward neural network (FFNnet) is used to further learn the interactions among latent features. The process of an STAnet block can be formulated as

$$\text{head}_j^l = \text{Attention}(W_j^{Q_l}H_l + W_j^{K_l}H_l + W_j^{V_l}H_l), \tag{8a}$$

$$\text{MultiHead}(H_l) = \text{Concat}(\text{head}_1^l, \dots, \text{head}_K^l)W_l, \tag{8b}$$

$$\hat{H}_l = \left(\text{ReLu}(\hat{H}_l^a W_1^l)\right) W_2^l, \tag{8c}$$

where head_j^l denotes the result of the j-th self-attention layer of the STAnet block; $W_j^{Q_l}, W_j^{K_l}$ and $W_j^{V_l}$ are weight matrices of query, key, and value subspaces, respectively; $\text{MultiHead}(H_l)$ denotes the result of the multi-head attention layer; $\hat{H}_l^a$ is the result of the residual connection; ReLu is a nonlinear activation function; $\hat{H}_l$ is the result of the FFNnet layer.

The gated fusion technique enhances the performance and robustness of the models by filtering out the noise and redundancy in the feature fusion process. Thus, in the DCSTA module, we use a gated fusion to adaptively fuse H_{sl} and H_{lg} to output the dual attention feature H_a, which is formulated as

$$H = z \otimes H_{\text{sl}} + (1 - z) \otimes H_{\text{lg}}, \tag{9a}$$

$$z = \sigma\left(H_{\text{sl}}W_{z1} + H_{\text{lg}}W_{z2} + b_z\right), \tag{9b}$$

where W_{z1}, W_{z2} and b_z are learnable parameters, and z is the gate. After passing through the FC layers, we finally obtain the predicted results $[\hat{Y}_{t+1}, \dots, \hat{Y}_{t+F}]$.

4.3 Dynamic price-aware crowdsensing task assignment

In this subsection, we present the proposed BROTA algorithm for task assignment and detail its main components, i.e., matching operator and breaking operator.

4.3.1 *Break-and-rematch online task assignment*

In the POCTA problem, maximizing the platform's profit means minimizing worker payment $\mathrm{pay}(w, \tau)$ (referring to (4)), which is dominated by three aspects: detour distance, response time, and supply-demand price. A higher matching revenue of the assignment pair (w, τ) usually suggests that (i) the detour distance between w and τ is lower; (ii) the task τ is completed soon; and (iii) the region where task τ is located enjoys worker oversupplied or supply-demand balance conditions. Evidently, if condition (iii) is satisfied, then conditions (i) and (ii) are more likely to be achieved. However, in practice, the number of workers is limited, usually much smaller than the number of tasks. Hence, achieving global worker oversupply or balanced supply-demand conditions is a stringent requirement. To this end, we propose the BROTA algorithm, which iteratively uses breaking and matching operators to optimize task assignment, i.e., breaking the matching pairs with lower profit and higher supply-demand imbalance, and rematching the broken sides of workers and tasks. We adopt a batch-based processing mode that accumulates dynamically arriving tasks and matches them with suitable workers within each batch. Compared with the instance processing mode (i.e., tasks are processed sequentially in the order of arrival), the batch-based mode typically results in higher matching quality [3, 9, 10, 14, 32]. This is because the matching batch can include more candidate workers and tasks, thereby increasing the likelihood of optimal task assignment.

The pseudo-code of BROTA is presented in Algorithm 1. Given a stream of tasks $\mathcal{T}$, a set of workers W and a batch size Δb, BROTA first accumulates a batch of tasks in $\mathcal{T}_t$ (lines 4–6). Then, when the cumulative timeline t_{cum} reaches the batch size Δb, BROTA performs supply-demand prediction (lines 8–11) and break-and-rematch assignment (lines 12–27), respectively. Specifically, in the supply-demand prediction step, BROTA first retrieves available workers $W_t \in W$ that satisfy worker constraints. It then encodes historical task sequences $X = [X_{t-P}, \ldots, X_t]$, auxiliary factors $\psi = [\psi_{t-P}, \ldots, \psi_t]$ and road network $\mathcal{G}_n = [\mathcal{G}_s, \mathcal{G}_d]$. After that BROTA obtains the future spatiotemporal patterns of tasks $\hat{Y} = M_p(\mathcal{G}_n; X, \psi)$ with the proposed MVSTAnet, and the future spatiotemporal patterns of workers $\hat{Z} = [\hat{Z}_{t+1}, \ldots, \hat{Z}_{t+F}]$ with the arrival time method [12]. In the break-and-rematch matching step, BROTA first calls the matching operator to generate the initial task assignment M_t^I. It then filters out matching pairs M_t^B that incur a large supply-demand gap, while assignments M_t^G that help balance the supply-demand gap (e.g., workers in oversupplied regions are guided to tasks in undersupplied regions) are stored in M_t. BROTA finally iteratively invokes the breaking and matching operators κ times to break and repair the worker-task matching and output the optimal result M_t^B. Here κ is the maximum number of tasks assigned to a single worker in the current batch.

4.3.2 *Packing-aware matching operator*

For the matching operator, a typical solution is to model the relationship between workers W_t and tasks $\mathcal{T}_t$ in b_t as a weighted bipartite graph $\mathcal{G}_b$, and to find the optimal matching plan M_t based on $\mathcal{G}_b$. We first introduce the multi-round bipartite graph matching algorithm (MBiGM), which invokes the Kuhn-Munkres (KM) algorithm [33] multiple times to derive the optimal task assignment in $\mathcal{G}_b$. However, MBiGM incurs significant computational overhead due to the repeated construction and updating of the bipartite graph. To address this limitation, we next propose the breaking-based task packing assignment (BTPA) algorithm, which achieves a balance between efficiency and effectiveness by incorporating task-packing and package-breaking strategies, along with several acceleration techniques. BTPA models the relationships between workers and task packages (i.e., grouped tasks) within a batch as a dual-weighted bipartite graph $\mathcal{G}_b = (W_t, \Gamma^{\mathcal{T}}, \mathcal{E}_t)$, where each edge $e_{ij} \in \mathcal{E}_t$ is associated with two weights $[u(w, \Gamma^{\tau}), \Delta g(w, \Gamma^{\tau})]$, representing the matching revenue and supply-demand gap increment, respectively. The weight Δg reflects the impact of assigning a task package Γ^{τ} to worker w on the overall supply-demand balance. $\Delta g \geqslant 0$ indicates that the gap is either unchanged (e.g., assignments performing in the same region) or worsened (e.g., assignment leads to oversupply or undersupply); conversely, $\Delta g < 0$ indicates that the supply and demand gap is alleviated.

In the BTPA algorithm, we group spatially adjacent tasks (e.g., located on the same street or in the same neighborhood) into a single package Γ^{τ}. The pairwise distance between any two tasks satisfies $\pi(l_{\tau_i}, l_{\tau_j}) \leqslant \delta$, where δ is a predefined threshold that represents the maximum allowable intra-package distance. After task packing, we identify reachable workers $W^{\Gamma^{\tau}}$ for each package Γ^{τ}. We also adopt time-bounded inverted lists (TBIL, our proposed method in [5]) for worker indexing and status updates to accelerate the search for candidate workers. It

Algorithm 1 Break-and-rematch online task assignment.

```
Input: A stream of tasks $\mathcal{T}$, a set of workers $W$, and a batch size $\Delta b$.
Output: A matching plan $M$.
$M \leftarrow \emptyset$, $t_{\rm cum} \leftarrow 0$, and $\mathcal{T}_t \leftarrow \emptyset$;
Initialize the task prediction model $M_p$, matching operator $M_r$, and breaking operator $M_b$;
while time line $t$ is not terminal do
    for each new arriving task $\tau \in \mathcal{T}$ do
        $\mathcal{T}_t \leftarrow \mathcal{T}_t \bigcup \tau$;
        Remove $\tau$ from $\mathcal{T}$;
        /*    Step 1: supply-demand prediction    */
        if $\Delta b == t_{\rm cum}$ then
            Retrieve available workers $W_t \in W$ that satisfy worker constraints;
            Encoding historical task sequences $X$, auxiliary factors $\psi$ and road network $\mathcal{G}_n$;
            Obtain the future spatiotemporal patterns of tasks $\hat{Y} = M_p(\mathcal{G}_n; X, \psi)$;
            Obtain the future spatiotemporal patterns of workers $\hat{Z} = [\hat{Z}_{t+1}, \ldots, \hat{Z}_{t+F}]$ with the arrival time method [12];
            /*    Step 2: break-and-rematch assignment    */
            $M_t^I \leftarrow$ initialize task assignment with matching operator $M_r(W_t, \mathcal{T}_t)$;
            $M_t^G, M_t^B \leftarrow$ Split $M_t^I$ considering incurred supply-demand gap;
            $M_t \leftarrow M_t \bigcup M_t^G$;
            for $k \leftarrow 0$ to $\kappa$ do
                $M_t^B \leftarrow$ Invoke breaking operator $M_b(\hat{Y}, \hat{Z}, M_t^B)$;
                Update $W_t$ and $\mathcal{T}_t$ based on $M_t^B$;
                $M_t^R \leftarrow$ Invoke matching operator $M_r(W_t, \mathcal{T}_t)$;
                if $R_t(M_t^R) > R_t(M_t^B)$ then
                    $M_t^B \leftarrow M_t^R$;
                end if
                $k \leftarrow k + 1$;
            end for
            $M_t \leftarrow M_t \bigcup M_t^B$ and $M \leftarrow M \bigcup M_t$;
            $t_{\rm cum} \leftarrow 0$, $\mathcal{T}_t \leftarrow \emptyset$ and $M_t \leftarrow \emptyset$;
        end if
    end for
    $t_{\rm cum} \leftarrow t_{\rm cum} + 1$;
end while
Return: $M$.
```

partitions the road network into grid cells and constructs an inverted list for each cell based on workers' reachable areas and time intervals. When searching for candidate workers for a task, TBIL first identifies adjacent grid cells and then further filters workers within those cells. This approach avoids searching all workers globally, thereby improving search efficiency. There are also cases where the package cannot find a valid worker. Hence, we design a package-breaking policy, i.e., tasks with the minimum remaining time $\overline{t}_{\min}$ are successively removed to generate a new package Γ_s^τ until candidate workers $W_1^{\Gamma^\tau}$ are found or greedily selecting workers who can perform the maximum number of tasks in the package. Finally, we model the bipartite graph $\mathcal{G}_b$, where candidate matching pair with dual weights $(w, \Gamma^\tau, [u(w, \Gamma^\tau), \Delta g(w, \Gamma^\tau)])$ and find the best task assignment M_t in the multiple rounds manner. Note that we normalize $u(w, \Gamma^\tau)$ and $\Delta g(w, \Gamma^\tau)$ to pose equal impact. Due to the space limitation, we detail the MBiGM algorithm and BTPA algorithm in Appendixes B.1 and B.2.

4.3.3 *Adaptive assignment breaking operator*

Intuitively, for the breaking operator, reducing matching pairs that break the supply-demand equilibrium and worsen worker oversupply could further improve the long-term profits of the platform. However, we should note that in our POCTA problem, breaking matching pairs is a dynamic and complex decision-making process affected by multiple factors (e.g., supply-demand conditions and distance). Traditional heuristic methods, e.g., greedy methods, cannot make long-term optimal decisions. To this end, we adopt a DRL-based method as the breaking operator to learn breaking strategies. We define the breaking decision process as the model of multi-agent Markov decision process (MMDP) $\mathcal{DP}$=$(W, \mathbb{S}, \mathbb{A}, P, R, \gamma)$, which is detailed below.

Agent $w_i \in W$. In our MMDP, each worker is treated as an agent. We assume that all agents are homogeneous due to their similar capacity to complete spatial tasks. It is worth noting that the number of agents may differ in each matching round.

State $s_t \in \mathbb{S}$. We denote the joint state of all agents as s_t. Then, the state $s_t^i \in s_t$ of an agent w_i is presented as a six-entry tuple $s_t^i = (\Delta\pi_i, \Delta t_{w_i}, \eta_i^g, \eta_i^p, \eta_i^{wg}, \eta_i^{sd})$. All six entries are normalized to range $[0, 1]$. Specifically, $\Delta\pi_i$ and Δt_{w_i} represent the detour ratio and response time ratio, respectively (as detailed in (4)). The entry η_i^g denotes the proportion of assignments whose supply-demand gap is greater than the matching pair (w_i, τ_i). It is computed as $\eta_i^g = \frac{|\{w \in W_\tau | \Delta g(w, \tau_i) > \Delta g(w_i, \tau_i)\}|}{|W_\tau|}$, where W_τ is a set of candidate workers of τ, and $\Delta g(w_i, \tau_i)$ represents the potential increase in the global supply-demand gap after virtually completing (w_i, τ_i). Similarly, η_i^p denotes the proportion of matching pairs with a profit lower than that of (w_i, τ_i). The entry η_i^{wg} represents the proportion of assignments assigned to w_i in the current batch b_t that have a supply-demand gap lower than that of (w_i, τ_i). Clearly, η_i^{wg} reflects the potential loss in supply-demand balance if the assignment (w_i, τ_i) is removed. More specifically, if $|\mathcal{T}^t_{w_i}| > 0$, $\eta_i^{wg} = \frac{|\{\tau_j \in \mathcal{T}^t_{w_i} | \Delta g(w_i,\tau_j) > \Delta g(w_i,\tau_i)\}|}{|\mathcal{T}^t_{w_i}|}$; if $|\mathcal{T}^t_{w_i}| = 0$, $\eta_i^{wg} = 0$, where $\mathcal{T}^t_{w_i}$ is a set of tasks assigned to the worker w_i in the t-th batch. The entry $\eta_i^{sd} = \frac{|\sum_{p=0}^{F} \mathrm{SD}^{tp}_{R_{w_i}} - \sum_{q=0}^{F} \mathrm{SD}^{tq}_{R_{\tau_i}}|}{F}$ specifies the supply-demand gap between the region where w_i is located and that of τ_i over the next F time steps. It reflects the future supply-demand impact on the agent's break-or-not decision.

Action $a_t \in \mathbb{A}$. At each time step t, the action of agent w_i is defined as $a_t^i \in \{0, 1\}$, where $a_t^i = 1$ or 0 represents the break-or-not decision for a matching edge of w_i. The joint action a_t of all agents thus represents a sequence of break-or-not decisions for the entire task assignment at t.

Immediate reward $R = \mathbb{S} \times \mathbb{A} \to \mathbb{R}$. We evaluate the effectiveness of the breaking operations in task assignment based on the current profit $R_t(M_t) = u_t(M_t) - \mathrm{gap}_t^{sd}(M_t)$ from the rematching operation. As such, the cumulative reward is typically defined as the sum of rewards obtained by agents over a series of future time steps starting from a certain time step, formulated as

$$R_t(s_t, a_t) = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n}, \tag{10}$$

where R_t is the cumulative reward starting from time step t; $\gamma^{n-1} R_{t+n}$ is the immediate reward obtained at time step $t + n$; γ is the discount factor, which ranges between 0 and 1 and determines the relative importance of future rewards compared to immediate rewards.

As mentioned above, since the number of agents can vary in each matching round, the dimensions of the action a_t and state s_t often differ from those of a_{t+1} and s_{t+1}. To address this, we adopt the centralized learning framework proposed in [28, 34], which uses a single neural network to model the decisions of all agents, with shared weights across agents. We leverage double deep Q-network (DDQN) to solve the decision problem in the above MMDP model. DDQN mitigates overestimation bias in Q-learning by using a target network for action selection, which enhances stability and convergence. We train the networks using the mini-batch backpropagation algorithm to minimize the loss:

$$\mathcal{L}(\theta) = \left[Q_t(s_t, a_t; \theta) - \left(r_t + \gamma \max_{a_{t+1} \in \mathcal{A}} Q\left(s_{t+1}, a_{t+1}; \theta'\right) \right) \right]^2, \tag{11}$$

where θ and θ' are learnable parameters of behavior and target Q-networks [35], respectively.

5 Evaluation

In this section, we first detail the experimental settings and then evaluate the performance of the proposed methods on two real-world datasets.

5.1 Experiment settings

Datasets and parameters. We conduct experiments on two real-world datasets [5, 9, 11, 14]: Chengdu (CD for short) dataset comprising 7065937 orders in November 2016, and Haikou (HK) dataset including 14160170 orders from May 1 to October 31, 2017. Each instance records five properties: index, origin, destination, departure time, and arrival time. These two datasets are sourced from Didi Chuxing[1]. The road networks for Chengdu (with 290517 nodes and 20343 edges) and Haikou (with 135521 nodes and 6377 edges) are extracted from OpenStreetMap[2]. The experimental parameter settings, including their default values (highlighted in bold), are configured in Table 2. Due to space limitations, the detailed experimental settings and several extensive experiments are provided in Appendix C.

Metrics and implementation. We adopt three commonly used evaluation metrics in the prediction research community [16–18, 20] for task prediction comparison, i.e., mean absolute error (MAE), root mean squared error (RMSE), and accuracy (Acc). For task assignment, we utilize two commonly used metrics in the spatial crowdsourcing community [5, 16], i.e., the total profit of the platform (an effectiveness metric, TP) and batch processing

[1] https://outreach.didichuxing.com.

[2] http://www.openstreetmap.org.

Table 2 Parameter settings. The default values are highlighted in bold.

Parameters	Values
# of past time steps P	6, **12**, 18, 24
# of future time steps F	1, **2**(HK), **3**(CD), 4
# of tasks $\|\mathcal{T}\|$ (HK) (k)	4, 6, **8**, 10, 15
# of workers $\|W\|$ (HK) (k)	0.4, 0.5, **0.6**, 0.7, 0.8
# of tasks $\|\mathcal{T}\|$ (CD) (k)	20, 25, **30**, 35, 50
# of workers $\|W\|$ (CD) (k)	0.9, 1.2, **1.5**, 1.8, 2.0
Maximum expected tasks c_w	20, **25**(HK), **30**(CD), 35, 50

time (an efficiency metric, BPT). All methods are implemented in Python and PyTorch, and evaluated on a PC equipped with an Intel i9-11900@2.50 GHz CPU, an NVIDIA GeForce RTX 3060 GPU, and 32 GB of memory.

Compared approaches. For task prediction, we compare MVSTAnet with four advanced solutions. (i) T-GCN [17], an integrated network that effectively captures spatial and temporal correlations. (ii) GMAN [18], extracts spatiotemporal correlations and assigns different weights using several well-designed attention modules. (iii) STGCSL [20], an integrated network that captures spatial, temporal, and short-term spatiotemporal dependencies. (iv) BSTGCNet [16], the latest task prediction network that explicitly models the correlations between zones from the perspectives of similarity, proximity, and distance, and enables the extraction of deep non-Euclidean spatial features. For task matching, we compare the proposed algorithms, including: (i) BROTA+BTPA (BRBT in short), where BTPA serves as the match operator; (ii) BTPA; (iii) BROTA+MBiGM (referred to as BRMB), where MBiGM serves as the match operator. We also compare these with the following state-of-the-art solutions. (i) PBO [5], the latest packing-aware approach that packs suitable tasks into packages, while it lacks a package-breaking mechanism. (ii) OTA [3], a predictive solution that applies a depth-first search procedure to identify a valid task set (including real-time and future tasks) for each worker. (iii) SIDF* [6], a distance-based greedy method that selects candidate workers for tasks with the shortest travel distances.

5.2 Analysis of experimental results

In this subsection, we examine the performance of MVSTAnet and BROTA.

5.2.1 *Evaluation of task prediction*

We first compare MVSTAnet with other state-of-the-art networks.

Exp-1: effect of P. Table 3 shows the performance of MVSTAnet and other models in terms of RMSE, MAE, and ACC for different P values (ranging from 6 to 24). For both the CD and HK datasets, the performance of all models generally improves as P increases, since the inclusion of more historical data allows for richer feature learning. However, we observed that the improvement in prediction performance is not linear with increasing P, but rather gradually diminishes. We attribute this phenomenon to the following reasons: (i) the inherent limitations of model capacity; (ii) the presence of outliers or irregular historical patterns in the data; and (iii) an excessive amount of input may lead to overfitting. We explain the superiority of MVSTAnet as follows. Compared to T-GCN, MVSTAnet captures richer spatial features and employs attention mechanisms to enhance robustness. In contrast to GMAN and STGCSL, it can simultaneously model three types of correlations, i.e., spatial, temporal, and spatiotemporal. Moreover, MVSTAnet outperforms BSTGCNet due to its DCSTA module, which captures both short-term local and long-term global spatiotemporal correlations. This significantly enriches the model's understanding of historical patterns and further improves predictive accuracy.

Exp-2: effect of F. Table 4 presents the performance of models with different F values. It is observed that as the F value increases, the performance of all models gradually declines. This trend is attributed to the fact that, with a fixed input sequence length, a larger output prediction horizon increases the difficulty of the prediction task. Similar to the effect observed with increasing P, the performance initially decreases slowly with increasing F, but then declines sharply beyond a certain point. For example, on the HK dataset, MVSTAnet's accuracy drops by only 3.36% when F increases from 1 to 2, but declines by 16.70% when F reaches 4. Likewise, MVSTAnet outperforms other models in almost all metrics. For instance, on the CD dataset, MVSTAnet's accuracy exceeds that of T-GCN, GMAN, and BSTGCNet by a margin ranging from 1.85% to 3.99%. These results on varying F values further confirm the strong performance of MVSTAnet in task prediction.

Table 3 Prediction results on CD and HK datasets with different P values. Bold values represent the best performance of all prediction networks under the relevant metrics.

P	Metric	Haikou					Chengdu				
		T-GCN	GMAN	STGCSL	BSTGCNet	MVSTAnet	T-GCN	GMAN	STGCSL	BSTGCNet	MVSTAnet
6	RMSE	6.1054	5.55	5.7236	**5.3054**	5.3257	2.9961	2.8821	3.0275	2.7225	**2.6525**
	MAE	4.5096	4.1594	4.2945	**3.8275**	3.9135	2.2235	2.1901	2.2972	2.0372	**1.9434**
	ACC	0.7836	0.8151	0.8115	**0.8241**	0.8223	0.6431	0.6565	0.6431	0.6735	**0.6834**
12	RMSE	5.4888	5.3758	5.3872	5.2738	**5.1351**	2.8534	2.8524	2.8551	2.6241	**2.5561**
	MAE	4.2199	4.0015	4.0472	3.7537	**3.541**	2.1440	2.1435	2.1965	1.9372	**1.8734**
	ACC	0.8005	0.8208	0.8187	0.8356	**0.8407**	0.6601	0.6606	0.6593	0.6861	**0.6927**
18	RMSE	5.3677	5.3492	5.3532	5.2531	**5.1257**	2.8334	2.8143	2.8291	2.5376	**2.4882**
	MAE	4.0131	3.9867	3.8421	3.6218	**3.5072**	2.1601	2.1527	2.1653	1.8379	**1.7797**
	ACC	0.8182	0.8213	0.825	0.8395	**0.8457**	0.6624	0.6634	0.6621	0.6981	**0.7021**
24	RMSE	5.3526	5.3154	5.3351	5.1033	**5.0145**	2.7721	2.7697	2.7721	2.4734	**2.3718**
	MAE	3.9951	3.9886	4.1162	3.5492	**3.3381**	2.0961	2.0887	2.1219	1.7581	**1.6729**
	ACC	0.8217	0.8273	0.8271	0.8419	**0.8430**	0.6698	0.6712	0.6651	0.7037	**0.7045**

Table 4 Prediction results on CD and HK datasets by varying F values. Bold values represent the best performance of all prediction networks under the relevant metrics.

Dataset	F	T-GCN	GMAN	STGCSL	BSTGCNet	MVSTAnet
HK	1	0.8436	0.8551	0.8465	0.8641	**0.8743**
	2	0.8005	0.8208	0.8187	0.8356	**0.8407**
	3	0.7482	0.7613	0.7525	**0.7895**	0.7857
	4	0.6417	0.6773	0.6721	0.6919	**0.7073**
CD	1	0.8135	0.8265	0.8331	0.8435	**0.8534**
	2	0.7335	0.7541	0.7435	0.7657	**0.7726**
	3	0.6601	0.6606	0.6593	0.6861	**0.6927**
	4	0.6198	0.6212	0.6251	0.6437	**0.6450**

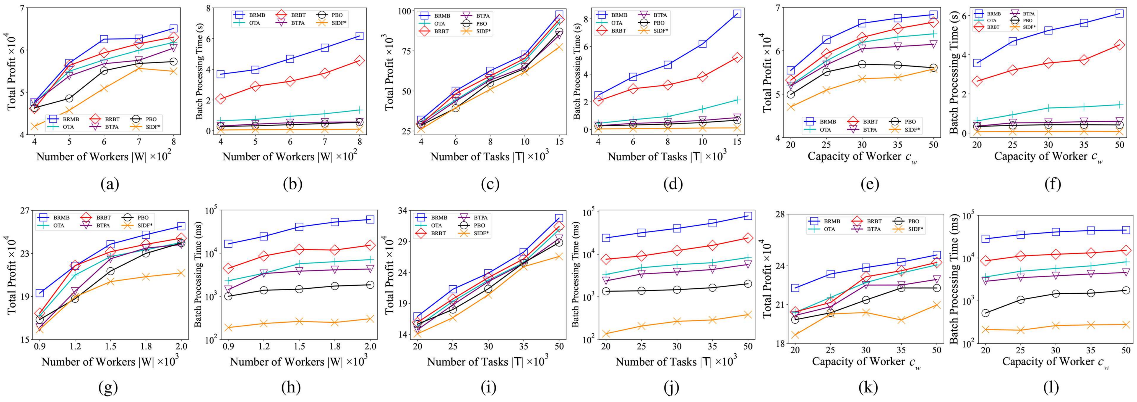

Figure 5 (Color online) Matching results on CD and HK datasets by varying the values of $|W|$, $|\mathcal{T}|$ and c_w. (a) $|W|$ vs. TP (HK); (b) $|W|$ vs. BPT (HK); (c) $|\mathcal{T}|$ vs. TP (HK); (d) $|\mathcal{T}|$ vs. BPT (HK); (e) c_w vs. TP (HK); (f) c_w vs. BPT (HK); (g) $|W|$ vs. TP (CD); (h) $|W|$ vs. BPT (CD); (i) $|\mathcal{T}|$ vs. TP (CD); (j) $|\mathcal{T}|$ vs. BPT (CD); (k) c_w vs. TP (CD); (l) c_w vs. BPT (CD).

5.2.2 *Evaluation of task assignment*

We next examine the performance of our proposed task assignment algorithms.

Exp-3: effect of $|W|$**.** Figures 5(a), (b), (g), (h) illustrate the efficiency (i.e., BPT) and effectiveness (i.e., TP) of all algorithms as varying the values of $|W|$. While increasing $|W|$ initially leads to higher TP and BPT values, the growth rate gradually slows down owing to the diminishing marginal utility of additional workers and the constraints imposed by task and worker deadlines. As expected, although SIDF* and PBO excel in efficiency, they are local optimization methods. Thus, they are unable to achieve higher-quality task assignments. In contrast, the break-and-rematch algorithms (BRMB and BRBT) incur higher computational costs due to graph breaking and rematching. However, they deliberately trade off some efficiency to achieve greater effectiveness. For instance, under the default settings in the HK dataset, BRBT incurs approximately 2.5 s of additional matching time to

Table 5 Significance testing of the proposed methods.

Dataset	Prediction (MVSTAnet VS T-GCN)					Matching (proposed methods VS SIDF*)				
	Metric	$\overline{D}$	S_D	t-statistic	Significant?	Method	$\overline{D}$	S_D	t-statistic	Significant?
CD	RMSE	0.3215	0.04133	19.05	√	BRBT	28013.82	3461.27	19.82	√
	MAE	0.3157	0.0319	24.24	√	BTPA	21535.55	1684.28	31.32	√
	ACC	0.0378	0.0063	14.70	√	BRMB	34611.44	4747.57	17.86	√
HK	RMSE	0.3667	0.0269	33.39	√	BRBT	7432.58	1775.5	10.25	√
	MAE	0.6630	0.0451	36.01	√	BTPA	6117.63	605.53	24.78	√
	ACC	0.0463	0.0088	12.89	√	BRMB	11583.93	907.09	31.28	√

achieve a 4.54% increase in TP compared to BTPA. In fact, the increase in BPT is generally acceptable in most ITS applications, given the corresponding gain in profit. Compared to OTA, which assigns both real-time and future tasks, our BTPA algorithm may not always achieve higher TP, but consistently outperforms in terms of BPT. This difference arises because OTA directly guides workers to future task locations, while our method adjusts real-time task prices based on anticipated supply and demand, indirectly guiding workers and influencing supply-demand dynamics. However, both BRMB and BRBT achieve higher TP than OTA, which illustrates that our BROTA can provide a higher quality of task assignment. Similar performance trends are observed across both datasets.

Exp-4: effect of $|\mathcal{T}|$. Figures 5(c), (d), (i), (j) display the effect of varying $|\mathcal{T}|$ on the performance of all algorithms. The results indicate that both BPT and TP steadily increase with the growth of $|\mathcal{T}|$ across both datasets. This is because a larger $|\mathcal{T}|$ provides workers with more opportunities to complete real-time tasks. Across all values of $|\mathcal{T}|$, BRMB and BRBT consistently achieve the highest TP, outperforming MBiGM and BTPA; this demonstrates the effectiveness of the assignment-breaking strategy. Among them, BRMB yields the highest profit but incurs more batch processing time, followed by BRBT, OTA, BTPA, PBO, and SIDF*.

Exp-5: effect of c_w. Figures 5(e), (f), (k), (l) report the results as the values of workers' maximum expected number of tasks (i.e., worker capacity) c_w are adjusted. A higher value of c_w intuitively implies that each worker is qualified for a greater number of tasks. As anticipated, the BPT and TP of all algorithms increase accordingly. It is noteworthy that once c_w reaches a certain threshold (e.g., $c_w = 30$ in the HK dataset), the rate of improvement in TP slows down. This phenomenon can be explained by the fact that increasing c_w does not alter the total number of workers, leading to some tasks still struggling to find suitable workers due to constraints such as task deadlines. Notably, algorithms based on packing strategies, i.e., PBO and BTPA, are more susceptible to the effects of small c_w.

5.2.3 *Significance testing*

To validate the effectiveness of our proposed approaches and enhance the credibility of the results, we perform paired t-tests comparing the performance of our methods with the baseline methods (T-GCN [17] and SIDF* [6]). Specifically, we define the null hypothesis as $H_0 : \mu_{\text{proposed}} = \mu_{\text{baseline}}$, indicating no significant difference between the proposed method and the baseline. The alternative hypothesis is denoted as $H_1 : \mu_{\text{proposed}} < \mu_{\text{baseline}}$ (for error metrics RMSE and MAE) or $H_1 : \mu_{\text{proposed}} > \mu_{\text{baseline}}$ (for accuracy Acc and profit TP metrics), indicating that the proposed method performs significantly better than the baseline. We calculated the t-statistic as $t = \frac{\overline{D}}{S_D/\sqrt{n}}$, where $\overline{D}$ is the mean difference between paired observations, S_D is the standard deviation of the differences, and n is the number of paired observations. For task prediction, we select data from 13:00-4:00 over 20 days from the CD and HK datasets, and compute the average RMSE, MAE, and ACC for each day. For task assignment, we conduct 20 rounds of experiments under default settings and record TP for each round. Thus, we obtain 20 statistical observations (i.e., $n = 20$) of RMSE, MAE, ACC, and TP. Hypothesis testing is performed with 19 degrees of freedom and a significance level of $\alpha = 0.025$ (one-tailed). From the t-distribution table, for $df = 19$ and $\alpha = 0.025$ (one-tailed), the t-critical value is $t_{0.025,19} = 2.093$. Thus, if the t-statistic exceeds 2.093, we reject the null hypothesis; otherwise, we fail to reject it.

As shown in Table 5, the significance test for the prediction models reveals that the t-statistic values for all metrics (RMSE, MAE, and ACC) of our proposed MVSTAnet, compared to the baseline T-GCN on both the CD and HK datasets, exceed the critical value of $t_{0.025,19} = 2.093$ (with $df = 19$ and $\alpha = 0.025$). Similarly, for task matching, the t-statistic values of our proposed algorithms (BRBT, BTPA, and BRMB), compared to the baseline SIDF* on TP, also exceed the t-critical value of 2.093. These results show statistically significant improvements at the $\alpha = 0.025$ level (p-value < 0.025), confirming the superiority of our methods over the baselines. Furthermore, the significance tests confirm that our methods consistently outperform the baselines, as evidenced by the t-statistic values exceeding the critical threshold.

6 Conclusion and future work

In this paper, we study the POCTA problem for profit-driven ITSs. We propose the PTA framework to solve the POCTA problem by maximizing the platform's overall profit. We devise MVSTAnet to predict the future distribution of crowdsensing tasks for our task pricing model. Additionally, we develop the online matching algorithm, BROTA, which assigns tasks optimally between vehicle-based mobile workers and crowdsensing tasks. Our comprehensive experiments on real-world datasets validate the effectiveness and efficiency of our proposed methods across diverse parameter settings.

For future work, we aim to extend this research in two directions. First, we plan to leverage the decision-making capabilities of large language models (e.g., DeepSeek and ChatGPT) to further optimize task assignment quality. Second, we plan to investigate the problem of task assignment across multiple platforms to encompass a wider range of workers and tasks, thus further enhancing the overall effectiveness of task assignment.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 62372416, 62325602, 62036010, 61972362), Henan Provincial Natural Science Foundation (Grant No. 242300421215), and China Postdoctoral Science Foundation (Grant No. 2020M682348).

Supporting information Appendixes A–C. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Guo S, Qian X. Optimal drive-by sensing in urban road networks with large-scale ridesourcing vehicles. IEEE Trans Intell Transp Syst, 2024, 25: 14389–14400

2 Fu Y, Qin X, Zhang X, et al. Hybrid recruitment scheme based on deep learning in vehicular crowdsensing. IEEE Trans Intell Transp Syst, 2023, 24: 10735–10748

3 Zhao Y, Zheng K, Cui Y, et al. Predictive task assignment in spatial crowdsourcing: a data-driven approach. In: Proceedings of the IEEE International Conference on Data Engineering, Dallas, 2020. 13–24

4 Chen H H, Guo B, Yu Z W, et al. Mobile crowd photographing: another way to watch our world. Sci China Inf Sci, 2016, 59: 083101

5 Li Y, Li Y, Peng Y, et al. Auction-based crowdsourced first and last mile logistics. IEEE Trans Mobile Comput, 2024, 23: 180–193

6 Zhang S, Qin L, Zheng Y, et al. Effective and efficient: large-scale dynamic city express. IEEE Trans Knowl Data Eng, 2016, 28: 3203–3217

7 Zhou Y P, Fan M J, Ma F F, et al. Solving multi-objective constrained minimum weighted bipartite assignment problem: a case study on energy-aware radio broadcast scheduling. Sci China Inf Sci, 2022, 65: 182101

8 Shi X, Deng F, Lu S, et al. A bi-level optimization approach for joint rack sequencing and storage assignment in robotic mobile fulfillment systems. Sci China Inf Sci, 2023, 66: 212202

9 Li Y, Wu Q, Huang X, et al. Efficient adaptive matching for real-time city express delivery. IEEE Trans Knowl Data Eng, 2022, 35 : 5767–5779

10 Zhao Y, Zheng K, Li Y, et al. Profit optimization in spatial crowdsourcing: effectiveness and efficiency. IEEE Trans Knowl Data Eng, 2022, 35 : 8386–8401

11 Li Y, Wan J, Chen R, et al. Top-k vehicle matching in social ridesharing: a price-aware approach. IEEE Trans Knowl Data Eng, 2019, 33: 1251–1263

12 Liu Z D, Gong Z Y, Li J Z, et al. mT-Share: a mobility-aware dynamic taxi ridesharing. In: Proceedings of the IEEE International Conference on Data Engineering, Dallas, 2020. 961–972

13 Hikima Y, Akagi Y, Kim H, et al. An improved approximation algorithm for wage determination and online task allocation in crowdsourcing. In: Proceedings of the Association for the Advancement of Artificial Intelligence, 2023. 3977–3986

14 Li Y, Li H, Huang X, et al. Utility-aware dynamic ridesharing in spatial crowdsourcing. IEEE Trans Mobile Comput, 2024, 23: 1066–1079

15 Tong Y X, Wang L B, Zhou Z M, et al. Dynamic pricing in spatial crowdsourcing: a matching-based approach. In: Proceedings of the Special Interest Group on Management of Data, Houston, 2018. 773–788

16 Wu Q, Li Y, Zhu G, et al. Prediction-aware adaptive task assignment for spatial crowdsourcing. IEEE Trans Mobile Comput, 2024, 23: 13048–13061

17 Zhao L, Song Y, Zhang C, et al. T-GCN: a temporal graph convolutional network for traffic prediction. IEEE Trans Intell Transp Syst, 2020, 21: 3848–3858

18 Zheng C P, Fan X L, Wang C, et al. GMAN: a graph multi-attention network for traffic prediction. In: Proceedings of the Association for the Advancement of Artificial Intelligence, New York, 2020. 1234–1241

19 Tedjopurnomo D A, Bao Z, Zheng B, et al. A survey on modern deep neural network for traffic prediction: trends, methods and challenges. IEEE Trans Knowl Data Eng, 2022, 34 : 1544–1561

20 Zheng B, Hu Q, Ming L, et al. SOUP: spatial-temporal demand forecasting and competitive supply. IEEE Trans Knowl Data Eng, 2023, 35: 2034–2047

21 Liu Y, Guo B, Meng J, et al. Spatio-temporal memory augmented multi-level attention network for traffic prediction. IEEE Trans Knowl Data Eng, 2024, 36: 2643–2658

22 Tong Y X, She J Y, Ding B L, et al. Online mobile micro-task allocation in spatial crowdsourcing. In: Proceedings of the IEEE International Conference on Data Engineering, Helsinki, 2016. 49–60

23 Tong Y, Zeng Y, Ding B, et al. Two-sided online micro-task assignment in spatial crowdsourcing. IEEE Trans Knowl Data Eng, 2021, 33: 2295–2309

24 Zhang C Z, Guo J, Yang C Y. When the gain of predictive resource allocation for content delivery is large? Sci China Inf Sci, 2023, 66: 222302

25 Ren T Y, Zhou X, L K L, et al. Efficient cross dynamic task assignment in spatial crowdsourcing. In: Proceedings of the IEEE International Conference on Data Engineering, Anaheim, 2023. 1420–1432

26 Zhang M, Zeng Y, Wang K, et al. Learned unmanned vehicle scheduling for large-scale urban logistics. IEEE Trans Intell Transp Syst, 2024, 25: 7933–7944

27 Bai S, Tong S, Feng X, et al. Toward dynamic pricing for city-wide crowdsourced instant delivery services. IEEE Trans Mobile Comput, 2024, 23: 909–924

28 Li Y, Li H, Mei B, et al. Fairness-guaranteed task assignment for crowdsourced mobility services. IEEE Trans Mobile Comput, 2024, 23: 5385–5400

29 Veličković P, Cucurull G, Casanova A, et al. Graph attention networks. In: Proceedings of the International Conference on Machine Learning, Sydney, 2017. 1–12

30 Zhuang C Y, Ma Q. Dual graph convolutional networks for graph-based semi-supervised classification. In: Proceedings of the International World Wide Web Conference, Lyon, 2018. 499–508

31 Wang Y, Zhu R H, Ji P S, et al. Open-set graph domain adaptation via separate domain alignment. In: Proceedings of the Association for the Advancement of Artificial Intelligence, Vancouver, 2024. 9142–9150

32 Yang Y, Cheng Y R, Yang Y R, et al. Batch-based cooperative task assignment in spatial crowdsourcing. In: Proceedings of the IEEE International Conference on Data Engineering, Anaheim, 2023. 1180–1192

33 Kuhn H W. The Hungarian method for the assignment problem. Naval Res Logist, 1955, 2: 83–97

34 Ke J, Xiao F, Yang H, et al. Learning to delay in ride-sourcing systems: a multi-agent deep reinforcement learning framework. IEEE Trans Knowl Data Eng, 2022, 34: 2280–2292

35 Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. Nature, 2015, 518: 529–533