UniTask: A Unified Task Assignment Design for Mobile Crowdsourcing-Based Urban Sensing

Zhidan Liu, *Member, IEEE*, Zhenjiang Li, *Member, IEEE*, and Kaishun Wu, *Member, IEEE*

***Abstract*—Mobile crowdsourcing (MCS) becomes an emerging paradigm for various useful urban sensing application designs by assigning the crowdsourcing tasks to the participants with rich-sensor equipped mobile devices. To effectively assign MCS tasks, many research efforts have been made in the literature. However, most prior schemes mainly optimize certain performance metrics in the assignment, yet overlooking other metrics, which thus cannot guarantee the overall system performance. This also limits the applicability of the proposed solution dedicated to the targeted performance metrics only. In this paper, we present *UniTask*, a unified task assignment design to address these issues. *UniTask* jointly considers the representative MCS performance metrics, including coverage, latency, and accuracy, to optimize the overall system utility. We mathematically formulate this problem and prove its NP-hardness. To efficiently schedule tasks, we also propose a utility-aware heuristic algorithm in *UniTask*. Moreover, a set of optimization techniques are further designed to enhance *UniTask*. Extensive evaluations are performed on two real-world datasets. Experimental results demonstrate that utility is an effective indicator of the system's overall performance. With an optimization on the system utility, *UniTask* can outperform the baseline methods on these individual performance metrics.**

***Index Terms*—Mobile crowdsourcing (MCS), task assignment, urban sensing.**

I. Introduction

RELYING on the high penetration and substantial availability of mobile devices (e.g., smartphones) that are equipped with powerful CPUs and rich sensors, mobile crowdsourcing (MCS) has become a promising paradigm to sense and collect data for smart urban applications, such as detecting traffic conditions [22], [46], monitoring air qualities [28], [42], building urban noise maps [29], etc. Such applications usually contain plenty of crowdsourcing *tasks* to be assigned to a set of *workers* (i.e., individuals having mobile devices), who will participate and complete the tasks by traveling to certain locations for collecting the desired data [6]. Such tasks are usually associated with rewards as an incentive to encourage participants to provide a better quality of service [40].

To assign the crowdsourcing tasks to the workers, excessive research efforts have been made so far, by mainly optimizing the following performance metrics [14], [33].

- *Coverage:* The spatial sensing coverage of the collected data in the target area [7], [38], [44].
- *Latency:* The execution time for all tasks to be accomplished for timely data gathering [3], [32], [41].
- *Accuracy:* Guaranteeing that the collected data do not differ from the true values too much [15], [18], [27].

In the literature, prior studies mainly focus on above metrics separately, e.g., optimizing one subset of these metrics while overlooking the rest [15], [38], [41]. However, there are two major limitations when we lack a holistic view on these metrics in the design. First, the system's overall performance gain cannot be fully obtained, since the performance from the omitted metric perspective could be highly sacrificed [14]. Second, the applicability of the proposed solution is limited to the targeted performance metric only [12]. When a similar application is considered yet by optimizing a different metric, it is unknown how to adjust previous designs for producing a new solution desired in this new application context.

To overcome above two limitations, in this paper, we present a unified task assignment design—*UniTask*, which jointly considers the task assignment metrics (i.e., coverage, latency, and accuracy) to optimize the overall system utility. *UniTask* is also flexible for adjusting the importance of different metrics to cater for the requirements of different applications. Therefore, *UniTask* could serve as a unified platform to support various crowdsourcing applications, instead of developing separate solutions for different smart urban applications. In addition to the unified task assignment framework, we further propose the following dedicated designs in *UniTask*.

First, as it is hardly to let the workers' collected data directly cover the entire target area, to further enhance the coverage, *UniTask* also supports an advanced working modality—enabling the inferred global data coverage. In this mode, we can estimate the global data distribution from the sparsely

Manuscript received December 23, 2018; revised March 5, 2019; accepted March 29, 2019. Date of publication April 4, 2019; date of current version July 31, 2019. This work was supported in part by China NSFC Grant (No. 61802261), NSF Grant of Shenzhen University (No. 2018061), Tencent "Rhinoceros Birds" - Scientific Research Foundation for Young Teachers of Shenzhen University, and ECS Grant from Research Grants Council of Hong Kong (No. CityU21203516). This work was also partially supported by China NSFC Grant (No. 61872248, No. U1736207), Guangdong NSF Grant No. 2017A030312008, Shenzhen Science and Technology Foundation (JCYJ20170302140946299, JCYJ20170412110753954, GRCK2017082111300325), Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China (No. 161064), GDUPS (2015), Tianjin Key Laboratory of Advanced Networking (TANK), and the project "PCL Future Regional Network Facilities for Large-scale Experiments and Applications (PCL2018KP001)". *(Corresponding author: Kaishun Wu.)*

Z. Liu is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: liuzhidan@szu.edu.cn).

Z. Li is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: zhenjiang.li@cityu.edu.hk).

K. Wu is with the PCL Research Center of Networks and Communications, Peng Cheng Laboratory, Shenzhen 518040, China, and also with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: wu@szu.edu.cn).

Digital Object Identifier 10.1109/JIOT.2019.2909296

2327-4662 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

collected data, by using some techniques like compressive sensing [4], [9]. If this modality is not selected, *UniTask* can also work with the original modality to use the raw data coverage—we can achieve a high data accuracy at these locations with assigned tasks, but with a limited coverage for the target area only [35]. In *UniTask*, we organically integrate the selection of these two different modalities into the system utility definition, so that different options will lead to different task assignment results for ensuring the good system performance. For example, we intentionally prefer to derive the complete traffic conditions of the whole road network in many traffic monitoring applications [23], [46], while for the investigation of point of interests (POIs) [17], [18], e.g., reporting crowdedness of POIs or labeling attractions, we can rapidly switch to the original working modality.

Second, we formally formulate this unified task assignment problem as an optimization problem, which aims to maximize the overall utility given the limited reward budget and available workers. This problem can be reduced to the well-known orienteering problem, which is NP-hard [11]. Therefore, we propose a utility-aware heuristic algorithm that can maximize the total utilities of assigned tasks given the spatial distribution of available workers. In addition, a set of optimization techniques are further proposed to accelerate the task assignments for the large scale urban sensing.

In summary, the contributions of this paper are as follows.

- We consider the task assignments of MCS for comprehensive sensing quality, and optimize the system's overall utility in the task assignments. We formulate this problem as an optimization problem and prove its NP-hardness.
- We propose a utility-aware heuristic algorithm to address the task assignment problem, which can maximize the overall utility with adequate computation overhead. We further propose a set of optimization techniques to enhance the *UniTask* design.
- We conduct extensive experiments with two real-world datasets to evaluate the performances of *UniTask*. Experimental results show that *UniTask* can achieve comprehensive sensing quality on the coverage, latency, and accuracy, and significantly outperform baseline methods.

The rest of this paper is organized as follows. We present the preliminary and problem formulation in Section II. The design of *UniTask* is detailed in Section III. In Section IV, we evaluate *UniTask* on two real-world datasets. We review related works in Section V, and conclude this paper in Section VI.

II. Preliminary and Problem Formulation

In this section, we introduce the MCS system model, present our proposed utility metric, and then mathematically formulate this utility optimization problem for the MCS task assignment.

A. System Model

In general, when a sensing request on certain type of urban-scale data (e.g., the traffic condition or the air quality) is submitted to an MCS system, it is first decomposed into a series of micro tasks. The urban area can be then virtually divided into grids to facilitate the task assignments [35], and crowdsourcing data will be collected from different grids [34]. In particular, crowdsourcing tasks will be assigned to a set of workers, who will travel along a planned path (cross the grids) linking the associated tasks to sense the urban data and upload it to the system. If the global data coverage is desired, a global data recovery algorithm f_{inf} (e.g., compressive sensing) will be further applied to the data collected from these workers.

TABLE I
Summary of Key Notations

Notation	Description
t_i	The i-th crowdsourcing task
loc_{t_i}	Location of task t_i
$\mathbb{T}$	The set of all crowdsourcing tasks with size of N
$\mathcal{B}_{all}$	Total distance budget
$\mathbb{W}$	The set of available workers with size of M
w_j	The j-th worker
loc_{w_j}	Location of worker w_j
b_{w_j}	Total distance budget for worker w_j
ε_{w_j}	Sensing time of worker w_j to complete a task
$\mathcal{T}_{w_j}$	The set of tasks assigned to worker w_j
$\mathcal{P}_{w_j}$	The sensing path of worker w_j connecting all tasks in $\mathcal{T}_{w_j}$
f_*	The recovery algorithm, set as f_{dir} or f_{inf}
$\mathbb{V}$	The set that contains the tasks assigned already
$u_{t_i}^{w_j}$	The utility for assigning task t_i to worker w_j

Fig. 1(a) illustrates an example of MCS for urban sensing, where 3 out of 5 workers are assigned with 13 tasks. Before we formally formulate this process, we first introduce the task and worker models in the following, and all the key notations used in this section are summarized in Table I.

1) Task Model: In *UniTask*, we define the occurrence of each task $t_i \in \mathbb{T}$ in one grid and denote the location of this grid as loc_{t_i}, which can be easily extend, so that one task will occupy multiple grids. To complete task t_i, a worker needs to travel to loc_{t_i} and conducts the task by sensing the required urban data via her mobile device. In this paper, we also consider the "freshness" of the collected data, implying that the tasks should be completed as soon as possible. For instance, the traffic condition is time-dependent, and only the timely collected traffic information is useful to the public.

On the other hand, to encourage workers to join in the crowdsourcing, the MCS system usually provides certain incentives, e.g., monetary rewards, for the participating workers [40], [45]. For instance, the incentive can be associated with the traveling distance each worker travels for completing the tasks. Therefore, the total budget for an urban sensing request can be transformed as a total distance budget, denoted as $\mathcal{B}_{\text{all}}$.

2) Worker Model: We assume the workers register themselves in the MCS system and will randomly appear in the physical area of interest to accept tasks. In the MCS system, each worker $w_j \in \mathbb{W}$ is associated with a set of attributes, denoted as $<\text{id}_{w_j}, \text{loc}_{w_j}, b_{w_j}, \varepsilon_{w_j}>$, where id_{w_j} represents the unique identifier of worker w_j, loc_{w_j} is w_j's current location, b_{w_j} is the distance budget for w_j to accept tasks, and ε_{w_j} is the sensing time for worker w_j to complete a task.

A worker w_j can accept multiple tasks, denoted as a task set $\mathcal{T}_{w_j}$, and she needs to travel from her original location loc_{w_j}

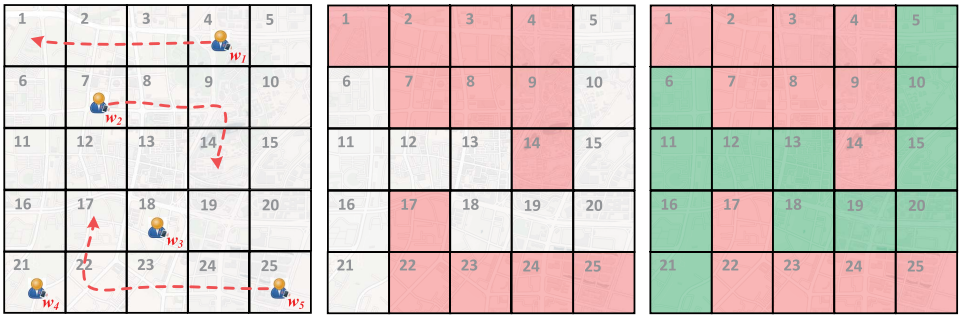

Fig. 1. Illustration of the task assignments. (a) Area of interest is divided into grids and each grid contains one task. Three selected workers perform tasks following their sensing paths. (b) Derived urban information through recovery algorithm f_{dir} with 52% coverage and 100% accuracy. (c) Derived urban information through recovery algorithm f_{inf} with 100% coverage yet not 100% accuracy.

to sequentially visit the location loc_{t_i} of each task $t_i \in \mathcal{T}_{w_j}$, which finally forms a sensing path $\mathcal{P}_{w_j}$ connecting all the assigned tasks. After the tasks are completed, we assume that workers will immediately upload the sensed data to the MCS system [34], [36]. By finishing all the tasks in $\mathcal{T}_{w_j}$, worker w_j will receive the reward that is proportional to the overall time spent on the traveling and sensing. Similar as prior works for the simplicity of the system design [18], we also omit other costs in *UniTask*, e.g., energy consumption of mobile devices.

As the maximum length of all sensing paths approximately determines the latency of an urban sensing request, path $\mathcal{P}_{w_j}$ should be well planned (i.e., the specific tasks and their orders in $\mathcal{T}_{w_j}$), so that the total traveling distance will not violate the budget b_{w_j} and the overall latency can be reduced.

B. Unify Different Metrics

As introduced in Section I, previous MCS task assignment designs mainly consider three performance metrics: 1) coverage; 2) latency; and 3) accuracy. In addition, two working modalities can be further selected: 1) global data coverage inference (f_{inf}) and 2) coverage by the data directly collected by workers (f_{dir}). Fig. 1(b) and (c) illustrates the derived urban information through f_{dir} and f_{inf}, respectively, based on the collected data by the selected workers in Fig. 1(a). However, to unify the above factors for the task assignment, we need to carefully consider the following mutual impacts between different metrics.

- *Coverage Versus Accuracy:* When we adopt f_{dir}, we could achieve a limited coverage while a high accuracy (100% in this case) at these grids with assigned tasks. On the other hand, if f_{inf} is used, although we can have 100% coverage through sparse samples-based data recovery, the accuracy is sacrificed since recovered data at the un-sensed grids may not be fully accurate.
- *Coverage Versus Latency:* To increase the coverage (using f_{dir}), we can plan walking paths for workers to visit as more grids as possible (within the limit of both individual worker's budget and the total budget). Such a strategy, however, will raise the traveling time and thus potentially improve the latency. Hence, we should carefully plan the path for each worker to balance coverage and latency (i.e., minimizing the maximum path length).
- *Accuracy Versus Latency:* There is no direct relation between them when we adopt f_{dir}, while it becomes subtle if f_{inf} is utilized. For an inference algorithm f_{inf}, the data from each grid may contribute differently to the global data recovery. The grids with the most informative values should be thus sensed with a higher priority, which can be realized by analyzing historical data [10]. However, such grids could be far away from the workers. If we enforce to select such grids, the latency could be prolonged.

1) Observations: By understanding different metrics, we observe that the coverage metric can be directly indicated by the number of sensed grids, while latency can be calculated as the maximum of all workers' task execution time (including both traveling time and sensing time). For accuracy, if we could collect data from the most informative grids, the accuracy for the both directly sensed grids and the recovered global data distribution is also high [23], [34], [46]. Therefore, we propose to use the *entropy* as the indicator to measure the informative importance of a task on improving the recovery accuracy. For each task t_i that corresponds to collect data at the ith grid, we treat its state (e.g., traffic condition or air quality) as a random variable X_i and thus its entropy is

$$H(X_i) = -\sum_{x \in X_i} P(x) \log(P(x))$$

where x is a specific value of X_i, and $P(x)$ is the probability of x over all possible values of X_i. Due to the underlying spatial correlation of urban data, the informative value to conduct task t_i will be a conditional entropy $H(X_i|\mathbb{V})$ when a set of tasks, denoted as $\mathbb{V}$, have been already assigned. Thus, when task t_i is conducted given $\mathbb{V}$, its benefit is reduced.

2) Utility Definition: With above observations, we can define the system utility in (1), which jointly considers all three performance metrics and two working modalities to evaluate the "value" of a task-worker assignment on improving the overall performance gain. For one task t_i assigned to worker w_j, we define the utility $u_{t_i}^{w_j}$ of this assignment as

$$u_{t_i}^{w_j} = \frac{H(t_i|\mathbb{V})^{\mathbf{I}_f}}{\text{delay}(t_i,\ w_j)} \times \left[1 - \frac{\text{len}(\mathcal{P}_{w_j})}{\bar{L}}\right]^{+} \tag{1}$$

where $H(t_i|\mathbb{V})$ is the conditional entropy of t_i given current assigned task set $\mathbb{V}$, and $\text{delay}(t_i,\ w_j)$ measures the time cost

introduced by assigning task t_i to worker w_j. Specifically, we define delay(t_i, w_j) as

$$\text{delay}(t_i,\ w_j) = \frac{\text{dist}(\text{loc}_{t_i}, \text{loc}_{w_j})}{\varphi} + \varepsilon_{w_j} \tag{2}$$

where dist($\text{loc}_{t_i}, \text{loc}_{w_j}$) measures the distance between task t_i and worker w_j's current location, and φ is a typical walking speed. Usually, a crowdsourcing task can be finished within minutes or even seconds, while the distance to the target location to conduce the sensing could cause a longer traveling time [34], e.g., hundreds or even thousands of meters for the worker to walk. The term ($[\text{dist}(\text{loc}_{t_i}, \text{loc}_{w_j})]/\varphi$) in (2) is thus normally much larger than the term ε_{w_j}. In this case, which is a typical setting in previous works [18], [34], [36], we can mainly focus on the workers traveling time. Hence, by default, we omit the sensing time term for the utility calculation, while we also experimentally examine the impact of omitting sensing time on the sensing performance in Section IV-C.

Besides, $\mathbf{I}_f$ in (1) is a binary value to indicate the working modality f_*, which is set as

$$\mathbf{I}_f = \begin{cases} 0, & \text{when } f_* = f_{\text{dir}} \\ 1, & \text{when } f_* = f_{\text{inf}}. \end{cases}$$

$\mathbf{I}_f$ implicitly manages the relation of coverage and accuracy.

- When $\mathbf{I}_f = 0$, $u_{t_i}^{w_j}$ changes to $([1 - (\text{len}(\mathcal{P}_{w_j})/\bar{L})]^+/[\text{delay}(t_i, w_j)])$ that omits the accuracy metric and each task contributes equally on the metric of coverage.
- When $\mathbf{I}_f = 1$, $u_{t_i}^{w_j}$ will ignore the coverage metric but pursue the pairs of task and worker with high informative value and low latency.

In addition, the function $[x]^+$ in (1) is defined as

$$[x]^+ = \max(0, x)$$

which returns a non-negative value. $\text{len}(\mathcal{P}_{w_j})$ measures the length of worker w_j's sensing path $\mathcal{P}_{w_j}$, and $\bar{L}$ is the average length of all sensing paths. Hence, the term $[1-(\text{len}(\mathcal{P}_{w_j})/\bar{L})]^+$ implicitly prevents assigning tasks to the workers who are already assigned with long sensing paths, and thus can restrict the overall latency of the whole crowdsourcing procedure.

C. Problem Formulation

Given the definition of utility, we formally formulate the problem of the task assignment for MCS systems with unified performance metric in this section.

Formulation: Assume an urban sensing job is decomposed into a set of N tasks $\mathbb{T} = \{t_1, t_2, \ldots, t_N\}$ by dividing the area of interest into grids. Meanwhile, workers are online ready with random locations in the area. Each worker w_j will be assigned with multiple tasks, and she needs to travel along a sensing path $\mathcal{P}_{w_j}$ planned by the MCS system to visit each task location and conduct the data collection. Both tasks and workers are location bounded, and each worker w_j has a distance budget b_{w_j}. Due to the limit of total budget $\mathcal{B}_{\text{all}}$, only partial tasks are assigned to some well-selected workers. The task assignment scheme should maximize the overall system utility by jointly considering the performance from coverage, latency, and accuracy three crucial aspects. The problem is thus modeled as follows:

$$\begin{aligned} \max \quad & \sum_{t_i \in \mathbb{T}, w_j \in \mathbb{W}} u_{t_i}^{w_j} \\ \text{s.t.} \quad & \text{len}(\mathcal{P}_{w_j}) \le b_{w_j} \quad \forall w_j \in \mathbb{W} \\ & \sum_{w_j \in \mathbb{W}} \text{len}(\mathcal{P}_{w_j}) \le \mathcal{B}_{\text{all}}. \end{aligned} \tag{3}$$

The objective of the optimization problem aims to maximize the utility of all task assignments, which indirectly pursues both high revenue of coverage and accuracy and low latency. The constraints in (3) represent that each worker can only accept tasks within a budget b_{w_j}, and the total traveling distance of all selected workers is bounded by budget $\mathcal{B}_{\text{all}}$.

Problem Hardness: We study the hardness of our problem.

Theorem: The optimization problem in (3) is NP-hard.

Proof: We prove the theorem by reducing from the NP-hard orienteering problem [11]. An orienteering problem can be described as follows: given a set of vertices, each with a score, and any two vertices are connected with an edge. The goal is to find a path, limited in length, which visits some vertices to maximize the sum of collected scores [11].

For a given orienteering problem, we can transform it to an instance of our problem. We consider a special setting of our problem, where $f_* = f_{\text{dir}}$ and only one worker is available. For those tasks $\mathbb{T}$ and worker w_1, we build a weighted graph, where vertices are tasks or worker w_1 and edges are formed between any two vertices. Each task vertex v_i has a utility value with respect to worker vertex v_1, which is calculated as $(1/[\text{dist}(\text{loc}_{v_i}, \text{loc}_{v_1})])$ [when $I_f = 0$, sensing time ε_{w_1}, φ, and $[x]^+$ are omitted in (1)], and edge weight is set as the distance between the corresponding two vertices. Given a budget $\mathcal{B}_{\text{all}}$ ($= b_{w_1}$), our problem aims to find a path $\mathcal{P}_{w_1}$ originated at v_1 with total length no more than $\mathcal{B}_{\text{all}}$ to maximize the total utility value of the path. We find that the special case of our problem is an orienteering problem, which is known NP-hard [11]. Therefore, our problem is also NP-hard. ■

Although our problem can be reduced to the orienteering problem, it cannot be directly solved by the existing algorithms proposed for the orienteering problem or its variants [13]. This is because we need to select a subset of workers to maximize the total utility values of all sensing paths. In addition, rather than assuming static values for vertices in the orienteering problem, vertex values in our problem are varying when $I_f = 1$ and different vertices are visited by different workers. Therefore, we will propose an efficient and practical solution to solve this optimization problem in the next section.

III. Design of *UniTask*

A. Overview

Fig. 2 illustrates the system architecture of *UniTask*, which consists of three major modules: *Task Generation*, *Task Assigner*, and *Data Processing*.

1) Task Generation: Given an urban sensing job, the *Task Generation* module divides the area of interest into grids according to the sensing granularity specified by the urban application. Each grid is regarded as one task, which is then initialized with an identifier and a position as the center of

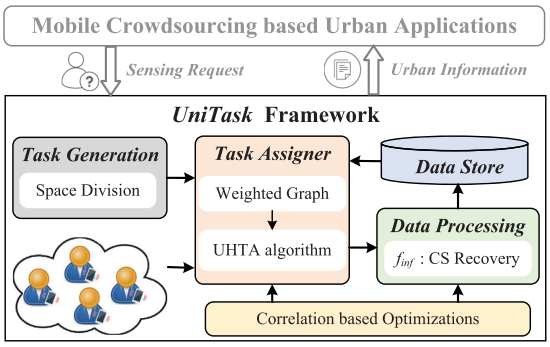

Fig. 2. System architecture of *UniTask*.

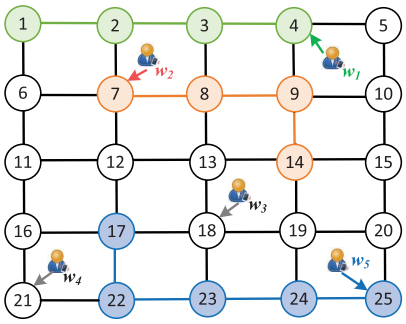

Fig. 3. Weighted graph for Fig. 1(a), where tasks are represented as vertices and edges are formed between any two immediately neighboring tasks. The weights of edges are the same as one distance unit.

the grid. The *Task Generation* module can be customized by urban applications to form specific tasks. For example, a traffic monitoring application can divide roads of a city into segments and each road segment is initialized as a task for workers to collect traveling speed. Air quality monitoring application can divide a city into grids of the same size (e.g., 1 km × 1 km), and each grid is formed as a task for measuring the air quality.

2) Task Assigner: Taking the tasks and available workers as input, *Task Assigner* module (Section III-B) first generates a weighted graph based on the information of tasks, and then heuristically produces a sensing path for each worker by iteratively assigning tasks based on their utilities.

3) Data Processing: By gathering all sensed data from the selected workers, the *Data Processing* module (Section III-C) conducts global data recovery (when $f_* = f_{\text{inf}}$) to derive the complete urban information. Specifically, since the compressive sensing theory [9] has been widely adopted for missing value inferences [15], [35], [39], we adopt compressive sensing as a concrete example to instrument the design in *UniTask*. Finally, urban data are saved to *Data Store*, which can be used to assist task assignments along with the historical data and support various urban applications as well.

In addition to the basic design, we also propose a set of enhancements (Section III-D) that exploit the temporal–spatial correlation of urban data to further improve the performance of *UniTask* on computation efficiency and data recovery accuracy.

B. Task Assigner

In this core system module, we first construct a weighted graph from the tasks and workers to facilitate the task assignments in *UniTask*.

1) Weighted Graph: Given the space division of an urban area, we will build a weighted graph $\mathtt{G}(\mathtt{V}, \mathtt{E})$ to represent the tasks and their associated costs, where tasks are viewed as *vertices* and *edges* are formed between any two immediately neighboring grids.

Each vertex inherits the attributes (i.e., identity and location) of the corresponding task and has the utility as its *value*, which is calculated using (1) and will be dynamically updated according to the tasks that have been assigned already. Each edge is associated with a *weight*, which is set as the geographic distance between two corresponding grids. Since workers randomly appear in the area of interest and must be in some grids. Hence, we also place the available workers into the graph and associate each worker to her nearest vertex. To complete the assigned tasks $\mathcal{T}_{w_j}$, worker w_j should traverse the corresponding vertices along the connecting edges, which will derive the sensing reward as the summation of utilities of all visited vertices and the traveling cost as the summation of weights of all traversed edges.

The construction of a weighted graph could be customized according to the adopted space division method as well, where the definition of an edge is highly related to specific urban applications. For example, Fig. 3 shows the weighted graph for Fig. 1(a), where edges are formed only between horizontal and vertical neighboring grids and their weights are set as the one distance unit between two grids. Besides, the five workers are associated with their nearest vertices, as shown by Fig. 3.

2) Heuristic Algorithm: To derive the optimal solution for the formulation in (3), we need to enumerate all possible combinations of assigned tasks given the available workers and choose the one with the largest utilities as the final task assignment decisions. It is, however, computationally prohibited due to the scale of urban sensing, which usually involves thousands (or even more) of tasks, and the dynamics of utility given different assigned task-worker pairs. Therefore, we propose a utility-aware heuristic task assignment (UHTA) algorithm that continuously updates the possible assignments and their utility values given workers' current locations, and then heuristically select the assignment with the maximum utility in an iterative manner. The pseudocode of UHTA is listed in Algorithm 1, and we detail each step in the following.

3) Initialization: We initialize the assigned task set $\mathbb{V}$ and candidate task assignment set $\mathbb{C}$ as the empty sets. Specifically, $\mathbb{V}$ stores the already assigned tasks, and $\mathbb{C}$ records the candidate task assignment decision x_{ij}, which will assign task t_i to worker w_j. Since each worker w_j originally stays at some grid, the corresponding task of that grid can be viewed as an *initial task* (but not assigned yet) for worker w_j. Therefore, we can initialize some candidate task assignment decisions given all workers' original locations and calculate the corresponding utility $u_{t_i}^{w_j}$ of each candidate decision x_{ij} using (1).

Step 1: After calculating the utilities for all candidate assignments x_{ij} in $\mathbb{C}$, we select the decision with the maximum utility as the best task assignment in the current iteration.

Step 2: Once x_{ij} is selected, task t_i is assigned to worker w_j. Thus, all candidate decisions related to t_i are removed out from $\mathbb{C}$, and t_i is included into both $\mathbb{V}$ and $\mathcal{P}_{w_j}$. Since worker w_j will move to the location of task t_i, all prior candidate decisions related to w_j should also be removed from $\mathbb{C}$.

Algorithm 1: UHTA

Input: Task set $\mathbb{T}$, worker set $\mathbb{W}$, worker budget b_{w_j}, total budget $\mathcal{B}_{all}$
Output: Sensing path $\mathcal{P}_{w_j}$ and $\mathbb{V}$

Assigned task set $\mathbb{V} \leftarrow \emptyset$
Candidate task assignment set $\mathbb{C} \leftarrow \emptyset$
Sensing path $\mathcal{P}_{w_j} \leftarrow \emptyset$ for worker $w_j \in \mathbb{W}$
Initialize $\mathbb{C}$ using all workers' initial tasks
while $(\mathcal{B}_{all} > 0)$ **do**
  Step ① : Select task assignment decision x_{ij} with the maximum $u_{t_i}^{w_j}$ from $\mathbb{C}$;
  Step ② : Remove x_{i*} and x_{*j} from $\mathbb{C}$, add task t_i into $\mathbb{V}$ and $\mathcal{P}_{w_j}$;
  Step ③ : Check and update w_j's budget b_{w_j} and total budget $\mathcal{B}_{all}$ by deducting sensing cost of t_i;
  Step ④ : Find t_i's 1-hop neighbors $\mathcal{N}_i$, and for $t_z \in \mathcal{N}_i \,\&\, t_z \notin \mathbb{V}$ add x_{zj} into $\mathbb{C}$;
  Step ⑤ : Update $u_{t_i}^{w_j}$ of x_{ij} in $\mathbb{C}$ using Eq. (1).

Step 3: We will check whether there are sufficient budget b_{w_j} and $\mathcal{B}_{\text{all}}$ for this task assignment. If b_{w_j} is insufficient, we will not assign t_i to w_j, and restart a new assignment. If $\mathcal{B}_{\text{all}}$ is insufficient, we will terminate the whole task assignment procedure and output the results. Otherwise, we update w_j's budget b_{w_j} and total budget $\mathcal{B}_{\text{all}}$ by deducting the edge weight, which is the traveling cost between w_j's current location and the location of t_i.

Step 4: We view t_i's immediately neighboring vertices as w_j's possible next task along the sensing path $\mathcal{P}_{w_j}$. Therefore, we first find t_i's 1-hop neighbors as $\mathcal{N}_i$, and check whether w_j has enough budget to complete a task t_z, where $t_z \in \mathcal{N}_i$ and $t_z \notin \mathbb{V}$. If yes, we add x_{zj} into $\mathbb{C}$; otherwise, this task assignment decision is considered as infeasible.

Step 5: Since we have updated both $\mathbb{V}$ and $\mathbb{C}$, the informative value of conducting each remaining task t_i will also be influenced. Therefore, we need to update the utility $u_{t_i}^{w_j}$ of each candidate assignment decision x_{ij} in $\mathbb{C}$ using (1) to continue the task assignments.

We repeat above five steps until total budget $\mathcal{B}_{\text{all}}$ is consumed. The UHTA algorithm will finally output the sensing path $\mathcal{P}_{w_j}$ for the selected worker w_j and the assigned task set $\mathbb{V}$.

4) Complexity Analysis: The computation overhead of UHTA algorithm mainly comes from the updates of $u_{t_i}^{w_j}$. The iterative task assignments will not stop until we consume all available budgets to derive the assigned task set $\mathbb{V}$. Given total budget $\mathcal{B}_{\text{all}}$, worker set $\mathbb{W}$ with M workers, and worker budget b_{w_j}, we derive the remaining available budget as $\mathcal{B} = \min(\mathcal{B}_{\text{all}}, b_{w_j} \times M)$. Assume the average edge weight as c, the size of $\mathbb{V}$ can be estimated as $v = (\mathcal{B}/c)$, which means the UHTA algorithm will run v iterations. In each iteration, we need to update the utility $u_{t_i}^{w_j}$ for each candidate task assignment decision x_{ij} in $\mathbb{C}$. Set $\mathbb{C}$ is initialized and updated according to the movements of all workers, and thus its size can be estimated as $|\mathbb{C}| = n \times M$, where n is the average size of 1-hop neighbor set $\mathcal{N}_i$ for all vertices in the weighted graph. In addition, the computation of conditional entropy is related with the assigned task set size $|\mathbb{V}|$ and the training data size $|\mathbb{D}|$. The computation complexity of each iteration is thus about $\mathcal{O}(v \times n \times M \times |\mathbb{D}|)$. Therefore, the overall computation complexity of the UHTA algorithm is about $\mathcal{O}(v^2 \times n \times M \times |\mathbb{D}|)$.

C. Data Processing

Once UHTA algorithm determines the task assignments, each selected worker will travel along the sensing path $\mathcal{P}_{w_j}$ to perform tasks and upload the collected data to the MCS system for data processing. If f_{dir} is adopted as the recovery algorithm, we only need to associate the collected data to the corresponding grids and save them into the data store.

In practice, urban applications usually require the complete information covering the whole target area of interest. Thus, we propose and implement a recovery algorithm f_{inf} by exploiting the compressive sensing technique [9] to reconstruct a full picture of the requested urban data from sparse samples.

1) Compressive Sensing-Based Data Recovery: Urban data (e.g., traffic conditions and air quality data) usually exist redundancy and such a property enables the technique, like compressive sensing, to precisely recover the original signal from much fewer samples [9]. Specifically, the compressive sensing theory states that a *k-sparse* signal $\mathbf{d}$ of size N (i.e., k nonzero elements in $\mathbf{d}$, and $k \ll N$) can be reconstructed from m measurements (denoted as $\mathbf{y}$, and $m \ll N$), which are acquired by applying a linear transform Φ on the signal $\mathbf{d}$, i.e., $\mathbf{y} = \Phi\mathbf{d}$. The natural signal $\mathbf{d}$ may not be *k-sparse* while it can still be *compressible* in some transform domain Ψ [e.g., discrete Fourier transform base and discrete cosine transform (DCT) base], i.e., $\mathbf{d} = \Psi\mathbf{s}$, where $\mathbf{s}$ is a *k-sparse* signal and Ψ is called the representation base.

For our MCS-based urban sensing, we can also build a compressive sensing-based data recovery system $\mathbf{y} = \Phi\Psi\mathbf{s}$ to derive the complete urban information as follows.

- *Measurement Vector* $\mathbf{y}$*:* We organize the m collected data from the workers according to the order of their corresponding grids to form $\mathbf{y}$.
- *Transform Base* Φ*:* The task assignments have implicitly determined the linear transform base Φ of size $m \times N$. For the ith collected data from the jth task (grid), we set $\Phi(i, j) = 1$. The other elements in Φ are all set as 0.
- *Representation Base* Ψ*:* For most urban data, the DCT base has been proved to be a suitable base for achieving a good sparse representation of the original signal [15], and thus is usually chosen as the representation base by default.

According to the compressive sensing theory, the sparse signal $\mathbf{s}$ can be accurately reconstructed with high probability (even the obtained measurements $\mathbf{y}$ are polluted with errors) by solving an l_1-minimization problem

$$\bar{\mathbf{s}} = \arg\min_{\mathbf{s} \in R^N} \|\mathbf{s}\|_{l_1} \quad \text{s.t.} \quad \|\mathbf{y} - \Phi\Psi\mathbf{s}\|_{l_2}^2 \leq \epsilon \tag{4}$$

when $m \geq ak \log(N/k)$, where a is a small positive constant and ϵ is the error tolerance [4]. We can solve the optimization problem in (4) using some standard compressive sensing solvers, e.g., the Basis Pursuit solver from SparseLab [1], and recover the desired urban data as $\mathbf{d} = \Psi\bar{\mathbf{s}}$.

D. Enhancements

Due to the prevalent temporal–spatial correlation among urban data [36], [44], there are still some promising opportunities for us to further improve the performances of *UniTask* on both computation efficiency and sensing quality.

1) Accelerating the Computation of Utility: When we run the UHTA algorithm, *UniTask* needs to continuously update the utility $u_{t_i}^{w_j}$ of each candidate decision x_{ij} in $\mathbb{C}$. By inspecting (1), we find that the calculation of conditional entropy $H(t_i|\mathbb{V})$ (when $\mathbf{I}_f = 1$) is computationally intensive, since the set $\mathbb{V}$ will include more and more assigned tasks along with the execution of task assignments.

Intuitively, for a given grid t_i, the states of its neighboring grids are more informatively valuable to infer t_i's state than the ones far away. To examine this intuition, we calculate the information gain of different pairs of grids with various hop distances using two real-world datasets (see more details about the datasets in Section IV-A). For a given grid t_i, the information gain of knowing t_j to infer t_i is calculated as

$$\mathrm{IG}(t_i|t_j) = H(t_i) - H(t_i|t_j)$$

where $H(t_i)$ is the entropy of task t_i's state and $H(t_i|t_j)$ is the conditional entropy once t_j's state is known. In principle, a larger $\mathrm{IG}(t_i|t_j)$ indicates knowing t_j's state can significantly reduce the uncertainty on inferring t_i's state. We plot the statistical results in Fig. 4, where we indeed find that distant grids have smaller information gain.

Therefore, we can obtain a shrunk set $\mathbb{V}_{t_i}^h$ by only keeping h-hop neighbors of t_i in $\mathbb{V}$. Formally, we have

$$\mathbb{V}_{t_i}^h = \mathbb{V} \cap \mathcal{N}_i^h$$

where $\mathcal{N}_i^h$ contains the h-hop neighbors of t_i, and h can be empirically set according to the property of specific urban data. For the air quality and traffic monitoring applications, we set h as 7 and 6, respectively, according to the results in Fig. 4, as further grids have no extra gain for the inference.

In addition, since the utility calculations of candidate decisions x_{ij} are independent, we could adopt multithreading technique to parallelize these utility calculations to further improve the computation efficiency.

2) Building Domain-Specific Representation Base: Although DCT has been widely used to sparsely represent various data [15], we can still build some representation base by leveraging domain-specific knowledge to achieve better performance.

For urban sensing applications, we could explicitly learn the spatial correlation among urban data using the multiple linear regression (MLR) model [2], and use the coefficients of these models to construct a representation base Ψ, which would compress urban data into a much sparser vector [23]. Specifically, the state s_i of the ith grid can be represented as a linear combination of the states of other grids, which can be expressed through the MLR model as

$$s_i = \lambda_{i,0} + \sum_{r=1, r\neq i}^{N} \lambda_{i,r} \times s_r$$

where λ_* is the model coefficient and N is the number of grids. With sufficient historical data, we can train the MLR model using the least-square technique.

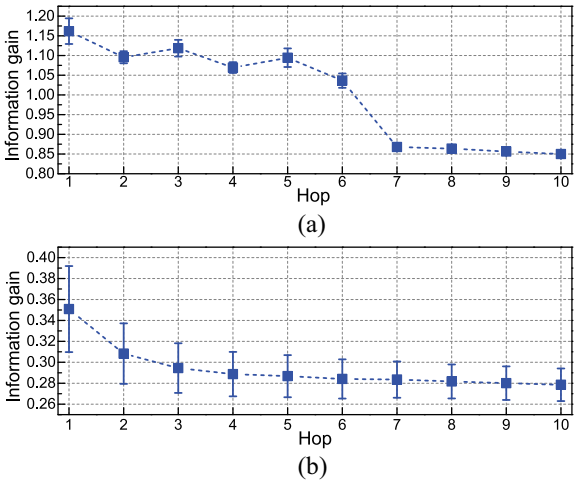

Fig. 4. Information gain for inferring the state of a grid from neighboring grids of different hops, evaluated using (a) air quality data and (b) traffic data.

We build an MLR model for each grid, and then extract the coefficients of all models to construct a matrix A to compress the urban data vector $\vec{\mathbf{s}} = [1, s_1, s_2, \ldots, s_N]^T$, where element 1 added in $\vec{\mathbf{s}}$ takes over the constant item in the MLR model, into a sparse vector. The matrix A is expressed as follows:

$$\mathsf{A} = \begin{bmatrix} \gamma & 0 & 0 & \cdots & 0 \\ \lambda_{1,0} & -1 & \lambda_{1,2} & \cdots & \lambda_{1,N} \\ \lambda_{2,0} & \lambda_{2,1} & -1 & \cdots & \lambda_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \lambda_{N,0} & \lambda_{N,1} & \lambda_{N,2} & \cdots & -1 \end{bmatrix} \tag{5}$$

where γ is a small constant (set as 0.0001 by default) and the ith row corresponds to t_i's MLR model. In theory, if all MLR models are well trained and indeed capture the spatial correlation among urban data, the projection of $\vec{\mathbf{s}}$ on A would be a sparse vector containing many zeros, i.e., $\mathsf{A} \times \vec{\mathbf{s}} \sim \vec{\mathbf{0}}$. Therefore, we will derive a domain-specific base $\Psi = \mathsf{A}^{-1}$.

3) Handling the Case of Insufficient Measurements: According to the compressive sensing theory [9], at least $ak\log(N/k)$ measurements are needed for an accurate data reconstruction. In practice, we may not have enough measurements when the total budget $\mathcal{B}_{\text{all}}$ is significantly limited.

To address this issue, we could "borrow" some measurements from previous round of data collection to supplement current measurement vector $\mathbf{y}$ by exploiting temporal correlation of urban data. In general, most urban data are temporally stable and thus the states of a grid in two consecutive rounds are quite similar. Therefore, we can select some samples from task assignments $\mathbb{V}_{\text{pre}}$ of previous round to enable an accurate data recovery. We can derive a candidate sample set $\mathbb{S}$ as

$$\mathbb{S} = \mathbb{V}_{\text{pre}} - \left(\mathbb{V}_{\text{pre}} \cap \mathbb{V}\right) \tag{6}$$

by filtering the grids that have been sensed in both consecutive rounds. For each sample $t_i \in \mathbb{S}$, we calculate its conditional entropy $H(t_i|\mathbb{V})$. We select the one with the largest entropy value, and add this task t_i and its sensed data into $\mathbb{V}$ and the measurement vector $\mathbf{y}$, respectively. We repeat the selections until we accumulate $\lceil ak\log(N/k)\rceil$ measurements in $\mathbf{y}$

if possible. With sufficient data, we can build our compressive sensing recovery system to derive the complete urban information.

IV. Performance Evaluation

A. Experiment Setup

By utilizing two real-world datasets, we consider two practical MCS-based urban applications, i.e., traffic monitoring and air quality monitoring, to evaluate *UniTask*.

Datasets: We use the traffic dataset from Singapore and air quality index (AQI) dataset from Beijing to examine *UniTask*'s performance for these two applications.

1) Traffic Dataset: This dataset contains traffic samplings collected from more than 12 000 taxis over Singapore in July and August of 2015. Each taxi will report a traffic sampling every 30 s, including a timestamp, a GPS location, and the traveling speed. There are about 10 millions traffic samplings each day.

A speed profile for each road segment, which includes 15-min average traffic speeds of a day in the week, is aggregated from the raw traffic samplings [25]. We use the speed profiles from *Monday* to *Thursday* to train the parameters of *UniTask*, and the speed profiles on *Friday* for the evaluation. In our traffic monitoring application, we focus on the primary road network of the downtown area in Singapore, which includes more than 2000 road segments. We regard sensing the traffic condition of each primary road segment as a task, and calculate the traveling distance between two tasks as the distance between the middle points of their corresponding road segments. For this application, we set the default number of available workers $|\mathbb{W}| = 50$, and configure the worker's traveling distance budget as $b_{w_j} = 2\,\text{km}$. To complete a task, a worker will travel to the road and take two (or more) consecutive photographs, so that recent computer vision techniques [16] can be adopted to estimate the traffic speed from the captured photographs.

2) AQI Dataset: This dataset contains hourly snapshots of the AQI values of PM2.5, PM10, and NO2, which are reported by 36 air quality monitoring stations in the Beijing city for several months in 2013 [42]. We use 11 days (November 9, 2013—November 19, 2013) of PM2.5 AQI values for the experiments, which contain complete reports for all the 36 stations. Specifically, data of the first 8 days are used for the parameter training and the rest data are used for evaluation. Similarly, a worker needs to take some photographs at some specific location to complete a task, and then image analytic and deep learning techniques are used to produce the accurate air quality estimation [28].

We divide the urban area of Beijing city into $1\,\text{km} \times 1\,\text{km}$ grids and only keep the grids having stations, just as done like [34]. Furthermore, we rearrange them into 6×6 equally spaced grids according to their originally geographic locations, and assume the distance between any two immediately neighboring grids as one distance unit. For the air quality monitoring application, we set $|\mathbb{W}| = 10$ for the available workers, with the worker budget configured as $b_{w_j} = 5$ distance units.

Performance Metrics: We use the following metrics to evaluate the performances of different task assignment schemes.

1) Number of assigned tasks indicates the grid coverage with collected data. Given the total budget $\mathcal{B}_{\text{all}}$, we prefer a scheme with more assigned tasks, especially when $\mathbf{I}_f = 0$.

2) Latency is decided by the total time for completing all the assigned tasks. As the sensing time is much less than the traveling time (see the explanations in Section II-B), we mainly utilize the maximum path length of all workers to approximate the latency, i.e.,

$$\text{Latency} = \max \text{len}\left(\mathcal{P}_{w_j}\right), \quad \forall w_j \in \mathbb{W}.$$

3) Normalized mean squared error (NMSE) measures the estimation accuracy in l_2-norm for traffic monitoring application when $\mathbf{I}_f = 1$. Assuming $\hat{x}$ is the estimation of data x, the NMSE is defined as

$$\text{NMSE} = \frac{||x - \hat{x}||_{l_2}}{||x||_{l_2}}.$$

4) Classification accuracy measures the estimation accuracy for the air quality monitoring application when $\mathbf{I}_f = 1$. Instead of providing the specific AQI value, we translate this value into 6 air quality levels, similar as the U-Air project [42]. These levels are defined as: *Good* ($0 \sim 50$), *Moderate* ($51 \sim 100$), *Unhealthy for Sensitive Groups* ($101 \sim 150$), *Unhealthy* ($151 \sim 200$), *Very Unhealthy* ($201 \sim 300$), and *Hazardous* (≥ 301). Therefore, assuming $\hat{x}$ is the estimation of data x, the classification accuracy is calculated as

$$\text{Classification accuracy} = \frac{\pi\left(\psi\left(\hat{x}\right), \psi\left(x\right)\right)}{N}$$

where function $\psi(x)$ maps an AQI value x to a specific level, and $\pi(x, y) = 1$ if $x = y$; otherwise, 0. N is number of grids.

Baselines: Since there are no existing works to directly solve our formulation (in Section II-C), we thus compare *UniTask* with the following baseline methods.

- *wKNN* assigns the tasks using the same UHTA algorithm as *UniTask*, while it recovers the missing data using the classic weighted K-nearest neighbors algorithm [8]. We use the inverse of distance as the weight and set $K = 5$. *wKNN* is mainly used for comparing the data recovery accuracy with *UniTask* when $\mathbf{I}_f = 1$.
- *Random* selects a worker in a random manner and assigns the worker with a task until total budget $\mathcal{B}_{\text{all}}$ is consumed. It reconstructs the complete data using the compressive sensing technique with DCT as the representation base. *UniTask* compares *Random* on both the task assignments and data recovery accuracy.
- *MinCost* always selects the worker having the minimum used budget and assigns her with a task with the minimum cost (i.e., taking $(1/[\text{dist}(\text{loc}_{t_i}, \text{loc}_{w_j})])$ as the selection metric). It reconstructs the complete data using compressive sensing technique with DCT as the representation base. *MinCost* tries to minimize the cost of each task assignment decision, and thus could approach the optimal performance on coverage and latency. We compare *MinCost* and *UniTask* on both task assignments and data recovery accuracy.
- *UniTask-DCT* is almost the same as *UniTask*, except it adopts DCT as the representation base. Thus, it is mainly

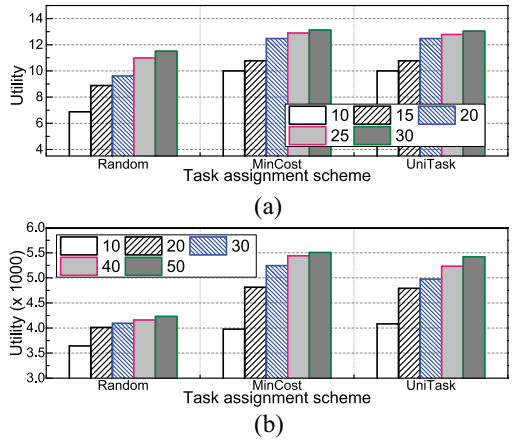

Fig. 5. Total utility of different task assignment schemes given varying budget $\mathcal{B}_{\text{all}}$, when $\mathbf{I}_f = 0$.

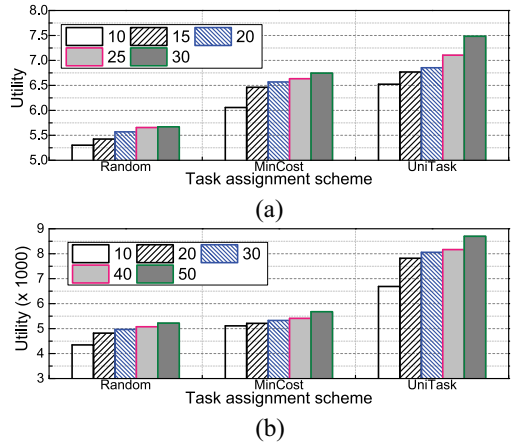

Fig. 6. Total utility of different task assignment schemes given varying budget $\mathcal{B}_{\text{all}}$, when $\mathbf{I}_f = 1$.

used to examine the data recovery accuracy of different representation bases, i.e., DCT and domain-specific base.

We implement and run all schemes in an HP Z440 Workstation that has 12 3.5-GHz Intel Xeon CPU cores and 32-GB memory. For the compressive sensing-based data recovery, we set the constant $a = 1.5$, the sparsity k as 2 and 35 (thus the required number of measurements are 13 and 307) for air quality monitoring and traffic monitoring, respectively.

B. Overall Performance

Performance Comparison on Utility: UniTask adopts utility as a comprehensive metric for task assignments, wherein a larger utility implies the higher overall gains from coverage, latency, and accuracy these three aspects (we will also examine these aspects individually below). Therefore, we compare total utility of different task assignment schemes given a varying budget $\mathcal{B}_{\text{all}}$. For *Random* and *MinCost* schemes, we calculate their total utility using (1) according to the orders of their task assignments. We vary total budget $\mathcal{B}_{\text{all}}$ from 10 to 30 (with a step of 5) and 10 to 50 (with a step of 10) for the air quality monitoring and traffic monitoring applications, respectively.

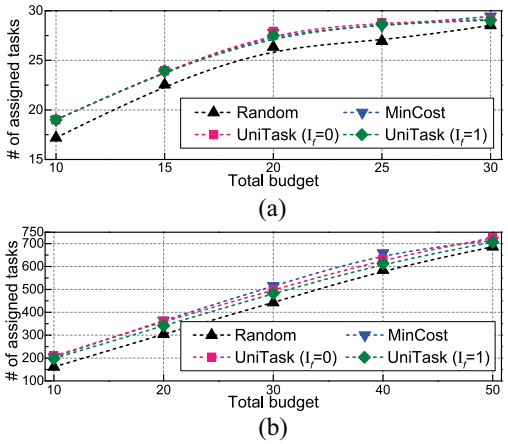

Fig. 7. Comparisons of different schemes on the number of assigned tasks with varying budget $\mathcal{B}_{\text{all}}$.

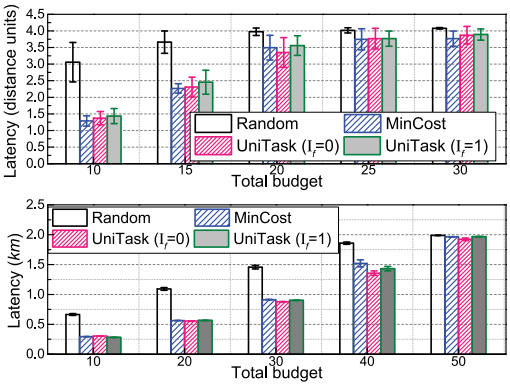

Fig. 8. Performance comparison of different schemes on the latency metric with varying budget $\mathcal{B}_{\text{all}}$.

Figs. 5 and 6 show the comparison results when we adopt different working modalities.

Fig. 5 plots the results for the direct data collection strategy (i.e., $\mathbf{I}_f = 0$). In general, more budgets allow each scheme to assign more tasks and thus more utilities are accumulated, as demonstrated in both Figs. 5 and 6. In the $\mathbf{I}_f = 0$ case, *UniTask* has a similar total utility as *MinCost* for air quality monitoring application, and only a bit smaller total utility than *MinCost* scheme for traffic monitoring application. In both applications, *UniTask* performs better than *Random* scheme. These results indicate that *UniTask* would have comparable performance with *MinCost* on the coverage and latency, and better performance than *Random*.

Fig. 6 presents the results for the $\mathbf{I}_f = 1$ case, where utility not only considers the cost of each task assignment, but also takes its informative value into consideration. From Fig. 6, we find that *Random* has the smallest total utility, while *UniTask* performs the best on total utility. The performance gap is even larger in the traffic monitoring application, which implies that *UniTask* is able to perform well in the large scale applications. Compared to the $\mathbf{I}_f = 0$ case, *UniTask* has more significant advantages for MCS in the $\mathbf{I}_f = 1$ case.

In order to verify above conclusions, we will compare the performances of different task assignment schemes on each individual metric, coverage, latency, and accuracy, in detail.

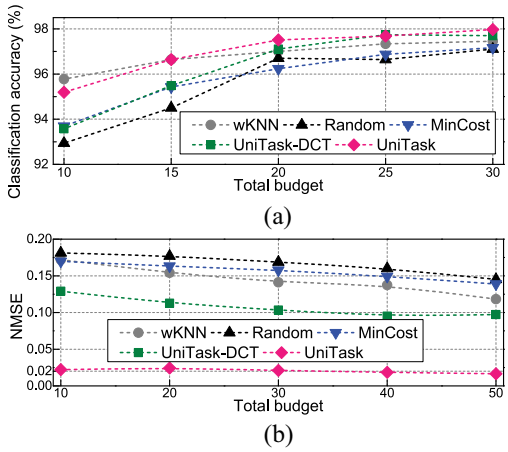

Fig. 9. Performance comparison of different schemes on (a) classification accuracy for air quality monitoring and (b) NMSE for traffic monitoring.

1) Comparison on Coverage: We compare *Random*, *MinCost*, and *UniTask* (with different $\mathbf{I}_f$ settings) on the number of assigned tasks (i.e., coverage), and plot the results in Fig. 7. Typically, the number of assigned tasks increases linearly with the total budget $\mathcal{B}_{\text{all}}$ in both applications. Among the four schemes, *Random* has the worst performance while *MinCost* can allocate the most tasks given the available budget $\mathcal{B}_{\text{all}}$. From Fig. 7, we see that *UniTask* has similar performance with *MinCost* on the two applications for both $\mathbf{I}_f$ settings. For the traffic monitoring application, *UniTask* performs slighter better when $\mathbf{I}_f = 0$, as shown in Fig. 7(b). Because when $\mathbf{I}_f = 0$, utility favors task assignment decisions with the minimum traveling costs and thus can allocates more tasks.

2) Comparison on Latency: We compare the latency performances of different schemes in Fig. 8. In principle, when we have more budget, we could assign more tasks to workers and the completion time (i.e., latency) will prolong as well, just as described in Fig. 8. Since *Random* assigns tasks in a random manner, thus it may assign a worker with too many tasks, resulting in a longer sensing path and a large latency. Fig. 8 shows that *Random* has the largest latency among all schemes. *MinCost* aims to reduce the total costs and thus performs well on the latency. With the utility metric, *UniTask* can also limit the maximum path length of all workers during the task assignment procedure. For the small-scale air quality monitoring application, *UniTask* achieves comparable latency as *MinCost*. For the traffic monitoring, *UniTask* performs even slightly better than *MinCost* when $\mathcal{B}_{\text{all}} \geq 30$. For example, the latencies (approximated as the maximum path length) for *Random*, *MinCost*, *UniTask* ($\mathbf{I}_f = 0$), and *UniTask* ($\mathbf{I}_f = 0$) are 1.86 km, 1.52 km, 1.36 km, and 1.43 km, respectively, when $\mathcal{B}_{\text{all}} = 40$ in Fig. 8(b). When we have sufficient budget, e.g., $\mathcal{B}_{\text{all}} = 50$, the four schemes assign workers with the most tasks that almost consume all worker budget $b_{w_j} = 2$ km, as shown in Fig. 8(b). From Fig. 8, we find that *UniTask* also performs slightly better for $\mathbf{I}_f = 0$ than $\mathbf{I}_f = 1$, as utility emphasizes on minimizing the traveling cost when $\mathbf{I}_f = 0$.

3) Comparison on Accuracy: We compare the accuracy performances for *wKNN*, *Random*, *MinCost*, *UniTask-DCT*, and *UniTask* in Fig. 9, where we set $\mathbf{I}_f = 1$ for *UniTask-DCT* and *UniTask*. From Fig. 9(a), we see that both *UniTask-DCT* and *UniTask* outperform *Random* and *MinCost* with much higher classification accuracy, with the maximum gap as 2%. Interesting, *wKNN* performs well in this application and has better classification accuracy than *Random* and *MinCost*.

TABLE II
COMPARISON ON EXECUTION TIME (IN SECONDS) WITH DIFFERENT TOTAL BUDGET FOR THE TRAFFIC MONITORING APPLICATION

$\mathcal{B}_{all}$	wKNN	Random	MinCost	UniTask-DCT	UniTask
10	1.12	0.65	0.72	**6.88**	**6.80**
20	2.34	0.97	1.12	3.47	3.20
30	3.46	1.31	1.53	5.16	4.93
40	4.65	1.75	1.96	6.76	6.34
50	5.31	2.10	2.22	7.70	7.34

Fig. 9(b) also demonstrates the superiority of *UniTask* on the metric of NMSE than other four schemes, with much more remarkable improvements. For traffic monitoring application, *wKNN* achieves slightly higher accuracy than *Random* and *MinCost* by exploiting spatial correlation among urban data. *UniTask-DCT* further improves the accuracy of *wKNN* through the compressive sensing-based data recovery and outperforms both *Random* and *MinCost* through wise task assignments. By comparing the performances of *UniTask-DCT* and *UniTask*, we can find that explicit spatial correlation modeling via MLR models indeed helps the data reconstruction. Compared to the default DCT base, our domain-specific representation base can improve the accuracy by 3.6 times at least. For example, when $\mathcal{B}_{\text{all}} = 30$, the NMSE of *wKNN*, *Random*, *MinCost*, *UniTask-DCT*, and *UniTask* are 0.14, 0.17, 0.16, 0.10, and 0.02, respectively. In this setting, *UniTask* outperforms these schemes by 6 times, 7.5 times, 7 times, and 4 times, respectively.

Performance Comparison on Execution Time: In addition to above comparisons, we also examine the computation efficiency of each scheme on the traffic monitoring application,[1] and present the average execution time under different $\mathcal{B}_{\text{all}}$ settings in Table II. Compared to the schemes adopting utility-based task assignments, *Random* and *MinCost* run much faster due to their simple task assignment mechanisms. Compared to *wKNN*, *UniTask* runs a bit longer (i.e., 0.86 to 5.68 s more time) due to the extra operations of supplement sample selections and compressive sensing-based data recovery. Although *wKNN*-based data recovery is more computationally efficient, it cannot always guarantee high accuracy (as shown in Fig. 9). It is interesting to see *UniTask* runs a bit faster than *UniTask-DCT*, which implies that MLR-based representation base can not only improve the data recovery accuracy but also the computation efficiency, with about 5% improvement on the execution time. From Table II, we observe that the execution times of both *UniTask-DCT* and *UniTask* with $\mathcal{B}_{\text{all}} = 10$ are even larger than the other settings (e.g., $\mathcal{B}_{\text{all}} = 20 \sim 40$). This is because a few budget cannot allocate sufficient number of tasks to enable the accurate compressive sensing-based data recovery, thus *UniTask* will selectively pick

[1]Since all schemes can finish the task assignments and data recovery in milliseconds for the air quality monitoring application due to its small scale, we thus omit these results in the evaluation.

TABLE III
COMPARISON ON EXECUTION TIME (IN SECONDS) WITH DIFFERENT ENHANCEMENT TECHNIQUES APPLIED FOR UTILITY COMPUTATION

$\mathcal{B}_{all}$	Basic design	Shrunk set	Shrunk set + Multi-threading
10	105.5	10.6	6.8
20	320.8	10.4	3.2
30	586.6	16.1	4.9
40	709.8	20.5	6.3
50	812.5	23.3	7.3

some informative samples from previous round to supplement the measurements of current round. Such sample selections consume time.

In summary, utility can indeed indicate the comprehensive sensing quality of task assignments, and utility-based *UniTask* achieves much better performance than the alternatives. Furthermore, *UniTask* can finish the task assignments and data recovery within seconds for large-scale urban applications.

C. Evaluation on UniTask Design

In this section, we will study the impacts of the enhancement designs on *UniTask*'s performance with the traffic monitoring application, where we set $\mathbf{I}_f = 1$ for the experiments.

1) Impact of Enhancement for Utility Computation: In Section III-D, we propose to use a shrunk set $\mathbb{V}_{t_i}^h$ instead of the assigned task set $\mathbb{V}$ to accelerate the computation of conditional entropy $H(t_i|\mathbb{V})$. To examine the effectiveness, we conduct experiment with varying budget $\mathcal{B}_{\text{all}}$. The execution times of different enhancement techniques applied are presented in Table III. For the basic design, *UniTask* needs hundreds of seconds (i.e., from 105.5 to 812.5 s) to assign the tasks due to the computation overhead caused by increasing size of assigned task set $\mathbb{V}$. If we use the shrunk set $\mathbb{V}_{t_i}^h$ instead of the original set $\mathbb{V}$, we can see the execution time is reduced to 23.3 s at most, with obvious speedup from 9 times to 34 times. If multithreading technique is applied, the overall execution time can be further reduced to <10 s, with a maximum speedup of 120 times than the basic design.

We also conduct an experiment to check whether a shrunk set $H(t_i|\mathbb{V})$ would harm the data recovery accuracy. We plot the results in Fig. 10, where we set $\mathcal{B}_{\text{all}} = 30$. Surprisingly, we find the accuracy using a shrunk set is even slightly better than the result using the original set. For example, the 50-percentile NMSEs of basic design and optimized design are 0.0186 and 0.0176, respectively. This may be that for a given grid t_i, set $\mathbb{V}$ contains too many irrelevant items, which possibly results in misleading hint on the task assignment decisions.

2) Impact of the Supplement Sample Selections: In Section III-D, we propose to "borrow" samples from previous round to supplement measurements of current round for more accurate compressive sensing-based data recovery by exploiting temporal correlation among urban data. We vary the budget $\mathcal{B}_{\text{all}}$ from 5 to 25, and compare the NMSE results on the scenarios, where this enhancement is on and off. The results are shown in Fig. 11. In principle, we need $ak\log(N/k) \approx 307$ measurements for the traffic monitoring application. Fig. 11(a) shows that when $\mathcal{B}_{\text{all}} < 20$, *UniTask* will select supplemental samples from previous round. Specifically, when $\mathcal{B}_{\text{all}}$ is

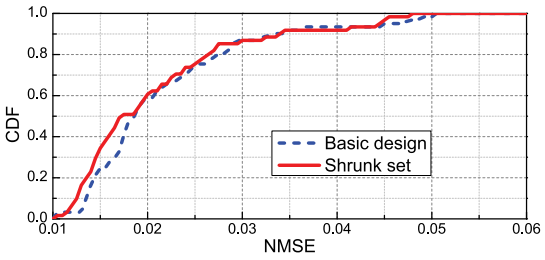

Fig. 10. Performance comparison on accuracy for the basic design and the enhancement technique of the shrunk set applied.

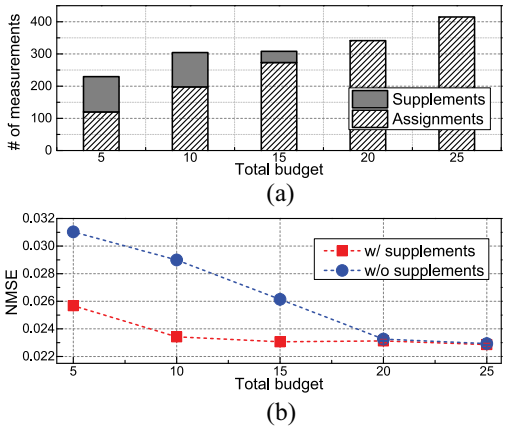

Fig. 11. Performance evaluation of the supplement sample selections.

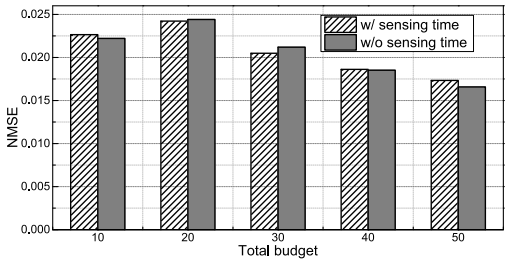

Fig. 12. Performance comparison when the sensing time is considered.

smaller, more supplemental samples are selected. If we disable this enhancement, we observe a reduced NMSE performance in Fig. 11(b) when $\mathcal{B}_{\text{all}} < 20$ (i.e., the results of "w/o supplement"). On average, this enhancement can improve the accuracy by 20%.

3) Impact of Sensing Time: We finally conduct experiments to understand the impact on the performance due to the omission of the sensing time in the utility calculation. We assume the typical walking speed $\varphi = 5$ km/h and the sensing time for all workers follows a uniform distribution in [1, 10] s, since a worker can quickly take two consecutive photographs within seconds for the traffic speed estimation [16]. To examine the quality of the completed tasks achieved in these two settings, measured by the NMSE, Fig. 12 shows that the performance of omitting the sensing time is quite similar as the performance when we include the sensing time in the task assignment, e.g., the difference is within 0.001. Therefore, the impact of omitting the sensing time on *UniTask*'s performance is small.

V. Related Work

A. Mobile Crowdsourcing

The dramatic proliferation of rich-sensor equipped mobile devices (e.g., smartphones) has enabled a novel computing paradigm named MCS [6], which exploits the power of crowds having smart mobile devices to perform location-dependent tasks at large scale. Recently, MCS has been widely adopted to build various novel urban sensing applications, such as traffic monitoring [22], [46], air quality monitoring [44], urban noise mapping [29], transit services [5], [24], [43], etc. In addition to the specific application designs, incentive mechanism [40], [45] and security and privacy protection [20], [21], [26], [31] have also been studied to guarantee the applicability of MCS in the real world. Specifically, Meng *et al.* [26] analyzed the security and privacy problems in urban sensing. Li *et al.* [20] studied the privacy leakage of location sharing in mobile computing, and Li *et al.* [21] proposed an approach to infer user demographic information by exploiting the metadata of Wi-Fi traffics. Parallel to these existing works, *UniTask* serves as a fundamental component of MCS to improve the crowdsourcing-based urban applications.

B. Task Assignment in Mobile Crowdsourcing

As one of the most crucial issues in the design of MCS systems, tasks are expected to be allocated to proper workers to achieve the best performance, subject to the constrains of total budget and worker's capability [12]. Many task assignment schemes have already been proposed [14], [33], however, most of them aim to maximum the performance on some specific metrics given certain constraints, e.g., budget constrains. For example, [7], [38], and [44] allocate tasks to suitable workers to maximize the spatial coverage; [3], [32], and [41] consider to minimize the completion time of all tasks for gathering timely information; while [15], [18], [27], and [35] prefer to maximize the accuracy (or reliability) of the crowdsourcing-based data collection. Although these works perform well on optimizing partial performance metrics, the system's overall performance gain cannot be fully obtained. Different from prior works, we propose *UniTask* to maximize the total utility of task assignments, which can achieve more comprehensive performances and support various urban sensing applications.

C. Compressive Sensing Applications

With the sparse (or compressible) property of real-world data, advanced techniques, like the compressive sensing theory [4], [9], have been successfully used to reconstruct complete information from a small number of measurements for the networked sensing systems [35], [39]. Typical compressive sensing applications include network traffic reconstruction [30], environmental data recovery [19], [37], road traffic monitoring [23], [46], etc. These works usually focus on the design of sensing matrix while adopting some popular representation bases, e.g., Fourier base or DCT base. Different from these works, we enhance the compressive sensing-based data recovery of *UniTask* with a domain-specific representation base, which explicitly exploits the spatial correlation of urban data and thus can derive higher data recovery accuracy for urban sensing applications.

VI. Conclusion

This paper presents *UniTask* for generic task assignments in MCS-based urban sensing applications. *UniTask* proposes a unified utility metric to jointly consider the system performance from coverage, latency and accuracy three aspects. We formulate this problem and propose a utility-aware heuristic algorithm to solve it, which is further strengthened by a series of enhancement techniques. In addition, two working modalities, i.e., direct data collection and global data recovery, both can be supported by *UniTask*. Rather than developing separate solutions for different smart urban applications, *UniTask* could serve as a flexible and unified platform to support various applications. Experimental results from two real-world datasets also demonstrate that *UniTask* significantly outperforms these alternative schemes and indeed achieves more comprehensive sensing quality.

In the future, we plan to further improve the sensing quality and practicability of *UniTask*. First, in this paper, we assume the sensing data from all the workers are of equal quality, and each task needs to be completed by only one worker. However, workers may provide sensing data of different qualities due to the diversity on workers' sensing devices and expertise. As a result, one task could be assigned to multiple workers for quality-oriented data collection. Therefore, how to assign each task to a sufficient number of suitable workers, given the constraints on the workers capability, tasks requirements, and available resources, is a possible future work. In addition, the current design of *UniTask* ignores the mobility of workers and blindly assigns each worker a traveling path. In fact, we could exploit the historical mobility information of each worker to predict her future mobility trajectory, and thus assign this worker with tasks that locate around her future traveling path. Such a task assignment strategy makes use of workers' mobility information and thus would reduce the traveling overhead and save the budgets. How to accurately and efficiently exploit and integrate the mobility information into task assignments is another possible future work.

References

[1] *SparseLab*. Accessed: Dec. 24, 2018. [Online]. Available: http://sparselab.stanford.edu/

[2] L. S. Aiken, S. G. West, and S. C. Pitts, "Multiple linear regression," in *Handbook of Psychology*. New York, NY, USA: Wiley, 2003, pp. 481–507.

[3] I. Boutsis and V. Kalogeraki, "On task assignment for real-time reliable crowdsourcing," in *Proc. IEEE ICDCS*, 2014, pp. 1–10.

[4] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Commun. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207–1223, 2006.

[5] C. Cao, Z. Liu, M. Li, W. Wang, and Z. Qin, "Walkway discovery from large scale crowdsensing," in *Proc. ACM/IEEE IPSN*, 2018, pp. 13–24.

[6] A. I. Chittilappilly, L. Chen, and S. Amer-Yahia, "A survey of general-purpose crowdsourcing techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2246–2266, Sep. 2016.

[7] Y. Chon, N. D. Lane, Y. Kim, F. Zhao, and H. Cha, "Understanding the coverage and scalability of place-centric crowdsensing," in *Proc. ACM UbiComp*, 2013, pp. 3–12.

[8] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, Jan. 1967.

[9] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[10] W. Du *et al.*, "Optimal sensor placement and measurement of wind for water quality studies in urban reservoirs," in *Proc. ACM/IEEE IPSN*, 2014, pp. 167–178.

[11] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Res. Logist.*, vol. 34, no. 3, pp. 307–318, 1987.

[12] W. Gong, B. Zhang, and C. Li, "Task assignment in mobile crowdsensing: Present and future directions," *IEEE Netw.*, vol. 32, no. 4, pp. 100–107, Jul./Aug. 2018.

[13] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *Eur. J. Oper. Res.*, vol. 255, no. 2, pp. 315–332, 2016.

[14] B. Guo *et al.*, "Task allocation in spatial crowdsourcing: Current state and future directions," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1749–1764, Jun. 2018.

[15] X. Hao, L. Xu, N. D. Lane, X. Liu, and T. Moscibroda, "Density-aware compressive crowdsensing," in *Proc. ACM/IEEE IPSN*, 2017, pp. 29–40.

[16] X. C. He and N. H. Yung, "A novel algorithm for estimating vehicle speed from two consecutive images," in *Proc. IEEE Workshop Appl. Comput. Vis.*, 2007, p. 12.

[17] H. Hu *et al.*, "Crowdsourced POI labelling: Location-aware result inference and task assignment," in *Proc. IEEE ICDE*, 2016, pp. 61–72.

[18] Y. Kang, X. Miao, K. Liu, L. Chen, and Y. Liu, "Quality-aware online task assignment in mobile crowdsourcing," in *Proc. IEEE MASS*, 2015, pp. 127–135.

[19] L. Kong, M. Xia, X.-Y. Liu, M.-Y. Wu, and X. Liu, "Data loss and reconstruction in sensor networks," in *Proc. IEEE INFOCOM*, 2013, pp. 1654–1662.

[20] H. Li, H. Zhu, S. Du, X. Liang, and X. S. Shen, "Privacy leakage of location sharing in mobile social networks: Attacks and defense," *IEEE Trans. Depend. Secure Comput.*, vol. 15, no. 4, pp. 646–660, Jul./Aug. 2018.

[21] H. Li, H. Zhu, and D. Ma, "Demographic information inference through meta-data analysis of Wi-Fi traffic," *IEEE Trans. Mobile Comput.*, vol. 17, no. 5, pp. 1033–1047, May 2018.

[22] Z. Liu, S. Jiang, P. Zhou, and M. Li, "A participatory urban traffic monitoring system: The power of bus riders," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2851–2864, Oct. 2017.

[23] Z. Liu, Z. Li, M. Li, W. Xing, and D. Lu, "Mining road network correlation for traffic estimation via compressive sensing," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1880–1893, Jul. 2016.

[24] Z. Liu, Z. Li, K. Wu, and M. Li, "Urban traffic prediction from mobility data using deep learning," *IEEE Netw.*, vol. 32, no. 4, pp. 40–46, Jul./Aug. 2018.

[25] Z. Liu, P. Zhou, Z. Li, and M. Li, "Think like a graph: Real-time traffic estimation at city-scale," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2018.2873642.

[26] Y. Meng, W. Zhang, H. Zhu, and X. S. Shen, "Securing consumer IoT in the smart home: Architecture, challenges, and countermeasures," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 53–59, Dec. 2018.

[27] C. Miao, L. Su, W. Jiang, Y. Li, and M. Tian, "A lightweight privacy-preserving truth discovery framework for mobile crowd sensing systems," in *Proc. IEEE INFOCOM*, 2017, pp. 1–9.

[28] Z. Pan, H. Yu, C. Miao, and C. Leung, "Crowdsensing air quality with camera-enabled mobile devices," in *Proc. AAAI*, 2017, pp. 1–6.

[29] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: An end-to-end participatory urban noise mapping system," in *Proc. ACM/IEEE IPSN*, 2010, pp. 1–12.

[30] M. Roughan, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and Internet traffic matrices," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 662–676, Jun. 2012.

[31] J. Shu, X. Liu, X. Jia, K. Yang, and R. H. Deng, "Anonymous privacy-preserving task matching in crowdsourcing," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3068–3078, Aug. 2018.

[32] Y. Tong *et al.*, "Flexible online task assignment in real-time spatial data," *Proc. VLDB Endow.*, vol. 10, no. 11, pp. 1334–1345, 2017.

[33] J. Wang, L. Wang, Y. Wang, D. Zhang, and L. Kong, "Task allocation in mobile crowd sensing: State-of-the-art and future opportunities," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3747–3757, Oct. 2018.

[34] L. Wang *et al.*, "CCS-TA: Quality-guaranteed online task allocation in compressive crowdsensing," in *Proc. ACM UbiComp*, 2015, pp. 683–694.

[35] L. Wang *et al.*, "Sparse mobile crowdsensing: Challenges and opportunities," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 161–167, Jul. 2016.

[36] L. Wang *et al.*, "SPACE-TA: Cost-effective task allocation exploiting intradata and interdata correlations in sparse crowdsensing," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 2, p. 20, 2017.

[37] X. Wu and M. Liu, "In-situ soil moisture sensing: Measurement scheduling and estimation using compressive sensing," in *Proc. ACM/IEEE IPSN*, 2012, pp. 1–11.

[38] H. Xiong, D. Zhang, L. Wang, and H. Chaouchi, "EMC3: Energy-efficient data transfer in mobile crowdsensing under full coverage constraint," *IEEE Trans. Mobile Comput.*, vol. 14, no. 7, pp. 1355–1368, Jul. 2015.

[39] L. Xu, X. Hao, N. D. Lane, X. Liu, and T. Moscibroda, "Cost-aware compressive sensing for networked sensing systems," in *Proc. ACM/IEEE IPSN*, 2015, pp. 1–12.

[40] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. ACM MobiCom*, 2012, pp. 1–12.

[41] Y. Zeng, Y. Tong, L. Chen, and Z. Zhou, "Latency-oriented task completion via spatial crowdsourcing," in *Proc. IEEE ICDE*, 2018, pp. 317–328.

[42] Y. Zheng, F. Liu, and H.-P. Hsieh, "U-Air: When urban air quality inference meets big data," in *Proc. ACM SIGKDD*, 2013, pp. 1–9.

[43] P. Zhou, Y. Zheng, and M. Li, "How long to wait? Predicting bus arrival time with mobile phone based participatory sensing," *IEEE Trans. Mobile Comput.*, vol. 13, no. 6, pp. 1228–1241, Jun. 2014.

[44] Q. Zhu, M. Y. S. Uddin, N. Venkatasubramanian, and C.-H. Hsu, "Spatiotemporal scheduling for crowd augmented urban sensing," in *Proc. IEEE INFOCOM*, 2018, pp. 1997–2005.

[45] X. Zhu *et al.*, "A fair incentive mechanism for crowdsourcing in crowd sensing," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1364–1372, Dec. 2016.

[46] Y. Zhu, Z. Li, H. Zhu, M. Li, and Q. Zhang, "A compressive sensing approach to urban traffic estimation with probe vehicles," *IEEE Trans. Mobile Comput.*, vol. 12, no. 11, pp. 2289–2302, Nov. 2013.

Zhidan Liu (M'15) received the B.E. degree in computer science and technology from Northeastern University, Shenyang, China, in 2009, and the Ph.D. degree in computer science and technology from Zhejiang University, Hangzhou, China, in 2014.

He was a Research Fellow with Nanyang Technological University, Singapore. He is currently an Assistant Professor with Shenzhen University, Shenzhen, China. His current research interests include distributed sensing and mobile computing, big data analytics, and urban computing.

Zhenjiang Li (GS'12–M'12) received the B.E. degree in computer science and technology from Xi'an Jiaotong University, Xi'an, China, in 2007, and the M.Phil. degree in electronic and computer engineering and the Ph.D. degree in computer science and engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2009 and 2012, respectively.

He is currently an Assistant Professor with the Computer Science Department, City University of Hong Kong, Hong Kong. His current research interests include wearable and mobile sensing, deep learning and data mining, and distributed and edge computing.

Kaishun Wu (GS'11–M'11) received the Ph.D. degree in computer science and engineering from the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2011.

He was a Research Assistant Professor with HKUST. In 2013, he joined Shenzhen University, Shenzhen, China, as a Distinguish Professor. He also works with the PCL Research Center of Networks and Communications, Peng Cheng Laboratory, Shenzhen. He has coauthored 2 books and has authored or coauthored over 100 high-quality research papers in international leading journals and premier conferences, such as the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, ACM MobiCom, and IEEE INFOCOM. He holds 6 U.S. patents and has over 90 Chinese pending patents.

Dr. Wu was a recipient of the 2012 Hong Kong Young Scientist Award and 2014 Hong Kong ICT awards such as the Best Innovation Award and the IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award. He is a Fellow of the IET.