CCS-TA: Quality-Guaranteed Online Task Allocation in Compressive Crowdsensing

Leye Wang[1], Daqing Zhang[1,5], Animesh Pathak[2], Chao Chen[3], Haoyi Xiong[1], Dingqi Yang[1,4], Yasha Wang[5]
[1]CNRS UMR 5157 SAMOVAR, Institut Mines-Télécom/Télécom SudParis, France
[2]INRIA Paris-Rocquencourt, France; [3]Chongqing University, China
[4]University of Fribourg, Switzerland; [5]Peking University, China
{leye.wang, daqing.zhang}@telecom-sudparis.eu, animesh.pathak@inria.fr, cschaochen@cqu.edu.cn,
haoyi.xiong@telecom-sudparis.eu, dingqi.yang@unifr.ch, wangyasha@pku.edu.cn

ABSTRACT
Data quality and budget are two primary concerns in urban-scale mobile crowdsensing applications. In this paper, we leverage the spatial and temporal correlation among the data sensed in different sub-areas to significantly reduce the required number of sensing tasks allocated (corresponding to budget), yet ensuring the data quality. Specifically, we propose a novel framework called *CCS-TA*, combining the state-of-the-art compressive sensing, Bayesian inference, and active learning techniques, to dynamically select a minimum number of sub-areas for sensing task allocation in each sensing cycle, while deducing the missing data of unallocated sub-areas under a probabilistic data accuracy guarantee. Evaluations on real-life temperature and air quality monitoring datasets show the effectiveness of CCS-TA. In the case of temperature monitoring, CCS-TA allocates 18.0-26.5% fewer tasks than baseline approaches, allocating tasks to only 15.5% of the sub-areas on average while keeping overall sensing error below 0.25°C in 95% of the cycles.

Author Keywords
Crowdsensing; Task Allocation; Data Quality

ACM Classification Keywords
H.4.m Information Systems Applications: Miscellaneous.

INTRODUCTION
With the pervasiveness of the sensor-rich mobile phones and wireless infrastructure, urban-scale monitoring applications (e.g., noise monitoring [11, 28] and air pollution monitoring [10, 18]) increasingly resort to mobile crowdsensing (MCS) [15, 37] rather than installing static and dedicated sensor networks. *Data quality* and *budget* are two primary concerns of the MCS organizers when planning urban-scale monitoring tasks — while MCS applications require the high-quality sensing data that can well represent the actual value of the target sensing area, the organizers also endeavor to minimize the cost of recruiting the participants.

To address the data quality concern for a location-centric MCS task, existing work often uses the *coverage ratio* as a major metric. Specifically, both *full coverage* [33, 35] and *probabilistic coverage* [2, 7, 17, 38] metrics have been introduced and studied. By ensuring coverage of each sensing sub-area or cell (full coverage) or most of the cells (probabilistic coverage) by mobile participants, the MCS applications attempt to obtain accurate and reliable sensing values throughout the target area, in order to compile a full sensing picture which meets the needs of the MCS organizers. The underlying connection between full coverage (or high probabilistic coverage) and data quality is that the full coverage (or high probabilistic coverage) can ensure the MCS organizers get representative sensing values for the target area. In essence, for the MCS organizers, the data quality requirement is to *obtain a reasonably accurate sensing value for each sub-area (cell)*, either through direct sensing or deduction.

In order to obtain the representative data for each cell, the most straight-forward way is to let MCS organizers recruit participants in all cells and allocate sensing tasks to them. The down side of this approach is that the MCS organizers need a high incentive budget for recruiting participants from each cell. Based on this idea, a lot of existing work tries to reduce the incentive budget by minimizing the number of redundant allocated sensing tasks or selected participants [2, 17, 33, 35, 38]. However, the number of required participants or tasks is still significant since most of the cells in the target area have to be covered physically by recruited participants.

In order to further reduce the number of recruited participants, an alternative approach is to select only a few cells for physical data sensing, while deducing the data of the rest of cells of the target area and ensuring the data accuracy. This alternative method is actually feasible as many kinds of environmental data, such as temperature [26] and noise [28, 41], have been shown to be able to be inferred with good quality due to the strong temporal and spatial correlations among the data.

With this insight in mind, we propose to use the overall sensing data accuracy, rather than the sensing area coverage, as the data quality metric. We further exploit the temporal and

UbiComp '15, September 7–11, 2015, Osaka, Japan.

http://dx.doi.org/10.1145/2750858.2807513

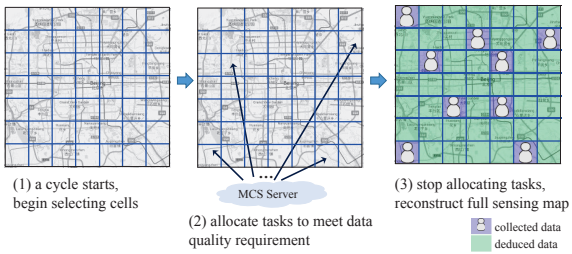

Figure 1: Use case: temperature monitoring in an urban area.

spatial correlations among the sensing data to deduce the missing data of unsensed cells from the sensing data in selected cells. By actively selecting a minimal number of cells for task allocation, we try to minimize the number of recruited participants, while ensuring the overall data accuracy meets a predefined bound.

The basic idea can be illustrated by the following use case (Figure 1): an MCS organizer launches an environment temperature monitoring task in a target urban area, which has already been divided into cells according to the organizer's requirement. The organizer needs to update the full temperature sensing map once every hour (sensing cycle), and in each sensing cycle, the data quality requirement is that the mean absolute error for the whole area should be less than *0.25°C*. In each sensing cycle, to meet the data quality requirement while minimizing the number of the allocated tasks, the organizer actively selects a subset of the cells to sense physically, i.e., allocating tasks to the participants in those selected cells. Based on the sensed temperature values of those selected cells, the temperature values of the remaining cells are deduced.

To accomplish the MCS task described in the use case above, we need to address the following two research challenges:

1) How many and which cells should be chosen for task allocation? In each cycle, to minimize the number of allocated tasks while ensuring the data quality, the organizer needs to select a minimum subset of the cells to allocate tasks. In order to find this minimum cell combination, we need to identify the *salient* cells whose sensing values, if collected, can help deduce the values of other cells to the maximum extent. However, how to identify the salient cells in an online manner is not trivial, because without foreknowing the true sensing value of a cell, it is hard to predict how much that value can help increase the data accuracy if collected.

2) How to quantitatively measure and estimate the data quality online throughout an MCS task without knowing the true sensing values of the unsensed cells? Since the true sensing values of the unsensed cells are unknown, we cannot measure the data accuracy directly by comparing the deduced data with the unknown ground truth. We thus need a method to efficiently estimate such sensing data accuracy online in each sensing cycle. Furthermore, as the estimated data accuracy intrinsically has certain discrepancy from the actual accuracy, it is practically hard to strictly guarantee *all* the sensing cycles to meet a predefined error bound. Therefore, we need to find a practical way to define the data quality requirement for an MCS task instead of merely setting the error bound for each sensing cycle.

With the above-mentioned research objective and challenges, the **main contributions** of the paper are:

1) We propose a *novel sensing data quality metric* to assess the quality of an MCS task, called (ϵ, p)*-quality*, which considers not only the required error bound ϵ but also what fraction p of sensing cycles should meet the bound ϵ. We further propose to exploit the temporal and spatial correlations among the sensing values of different cells to significantly reduce the number of allocated tasks. To the best of our knowledge, this is the first work in MCS that attempts to reduce the number of cells being sensed by intelligently selecting which cells are best for sensing while deducing the missing values of the remaining cells, to ensure that the overall data accuracy meets a predefined quality requirement.

2) In order to reduce the number of sensing cells required for task allocation in each sensing cycle while achieving the predefined (ϵ, p)-quality in MCS, we propose a two-phase online task allocation framework, called *CCS-TA (Compressive CrowdSensing Task Allocation)*. With CCS-TA, in phase one we decide which cell is the best to add for sensing in each cycle according to *active learning* techniques. After collecting the sensing data from the selected cells, in phase two we propose a *Bayesian inference* based method to estimate the overall data accuracy online after deducing the data of the remaining cells using *compressive sensing*. Based on the estimated overall data accuracy, CCS-TA decides whether more cells should be selected for task allocation to ensure the predefined (ϵ, p)-quality.

3) We conduct extensive evaluations on real-life temperature [20] and air quality [40] monitoring datasets to show the effectiveness of CCS-TA. In the case of temperature monitoring, CCS-TA needs to assign tasks to only 15.5% of the cells on average which can ensure the overall sensing error below 0.25°C in 95% of the cycles, i.e., satisfying $(0.25°C, 0.95)$-quality. As a comparison, the baseline approaches need to allocate 18.0-26.5% more tasks to ensure the same data quality.

PROBLEM STATEMENT

In this section, we first define the key concepts used throughout this paper. Then, we clarify our assumptions, followed by the problem formulation.

Definitions

To formally define the sensing data quality metric used in this paper, we first define the concepts about the *sensing and selection matrices* (see Figure 3 for an example).

Definition 1. Full Sensing Matrix. For a location-centric MCS task involving m cells and n sensing cycles, its *full sensing matrix* is denoted as F_{m*n}, where each entry $F[i, j]$ denotes the true sensing data of cell i in cycle j.

Definition 2. Cell-Selection Matrix. In a *cell-selection matrix* S_{m*n}, each entry $S[i, j]$ denotes whether or not the corresponding entry in the full sensing matrix $F[i, j]$ is selected for sensing: if cell i is selected for sensing in cycle j, $S[i, j] = 1$;

otherwise, $S[i,j] = 0$.

Definition 3. Collected Sensing Matrix. A *collected sensing matrix* C_{m*n} records the actual collected sensing data:

$$C = F \circ S$$

where $\circ$ denotes the element-wise product of two matrices.

Definition 4. Sensing Matrix Reconstruction Algorithm. A *sensing matrix reconstruction algorithm* $\mathcal{R}$ attempts to reconstruct a *full sensing matrix* $\hat{F}_{m*n}$ from the *collected sensing matrix* C_{m*n}:

$$\mathcal{R}(C_{m*n}) = \hat{F}_{m*n} \approx F_{m*n}$$

Now, we define the *overall sensing error*, which represents the data quality.

Definition 5. Overall Sensing Error. It quantifies the difference between the reconstructed full sensing matrix $\hat{F}$ and the true full sensing matrix F. In this paper, we focus on the *overall sensing error of each sensing cycle separately*. For sensing cycle k, the overall sensing error is defined as:

$$\mathcal{E}_k = error(\hat{F}[:,k], F[:,k])$$

where $F[:,k]$ is the kth column of F, i.e., the true sensing values of all the m cells in cycle k, and $\hat{F}[:,k]$ contains the corresponding deduced sensing values by using the reconstruction algorithm $\mathcal{R}$.

Note that the specific technique to calculate the overall sensing error (i.e., the $error()$ function) depends on the type of sensing data. In this paper, we focus on two widely-used metrics: *mean absolute error* (for continuous values, e.g., temperature [3]), and *classification error* (for classification labels, e.g., PM2.5 air quality index (AQI) descriptors [40]).

- *Mean Absolute Error*

$$error(\hat{F}[:,k], F[:,k]) = \frac{\sum_{i=1}^{m} |\hat{F}[i,k] - F[i,k]|}{m} \quad (1)$$

- *Classification Error*

$$error(\hat{F}[:,k], F[:,k]) = 1 - \frac{\sum_{i=1}^{m} I(\psi(\hat{F}[i,k]), \psi(F[i,k]))}{m} \quad (2)$$

  where $\psi()$ is the function to map a value to its classification label. $I(x,y) = 1$ if $x = y$; otherwise, 0.

With the overall sensing error, we define the data quality for an MCS task including n sensing cycles:

Definition 6. (ϵ, p)-quality. For an MCS task lasting for n sensing cycles, it satisfies (ϵ, p)-*quality*, iff

$$|\{k|\mathcal{E}_k \leq \epsilon, 1 \leq k \leq n\}| \geq n \cdot p$$

where ϵ is a predefined error bound for the overall sensing error of each cycle, and p is a predefined probability threshold to quantify the minimum fraction of the cycles whose errors should satisfy the error bound ϵ.

Ideally, for a predefined error bound ϵ, we expect that an MCS task keeps the overall sensing error lower than ϵ in *all* ($p = 1$) the cycles. However, it is intractable for a real-life MCS task to satisfy $(\epsilon, 1)$-quality because we cannot know the accurate overall sensing error $\mathcal{E}_k$ but have to estimate it (as the ground truth F cannot be foreknown). Thus we focus on the cases where p is large (e.g., 0.9 or 0.95), to guarantee the overall sensing error bounded by ϵ in *most* (e.g., 90% or 95%) cycles. Later we show how this allows us to use some techniques from probability theory and Bayesian statistics to handle the problem.

Assumptions

We make the following assumptions in this paper.

Assumption 1. Massive Candidate Participants. There are sufficient number of candidate participants across the target sensing area, so for any cell in any sensing cycle, the organizer can always find a participant to allocate a sensing task.

We believe this assumption is realistic in many current and future MCS applications. For example, the crowdsourcing traffic monitoring application *WAZE* (`https://www.waze.com`) has already appealed to more than 50 million users. With this massive user base, this assumption can be easily satisfied, especially in the user-intensive urban areas.

Assumption 2. High Quality Sensing. Every participant returns the accurate sensing value if a task is allocated to her.

The *assumption 2* also appears in other existing research work [42]. We note that in real life, it is not always true due to possible issues such as sensor error or varying conditions. But with an attractive incentive scheme in place, this assumption is also reasonable.

Assumption 3. Not Moving Out During Sensing. After a participant receives a sensing task in a cell, she will not move out of the cell before she finishes sensing.

Assumption 3 ensures that if we allocate a sensing task to a participant in cell *i*, her returned sensing value will actually represent cell *i*. This assumption can usually be satisfied if the sensing task does not consume much time. For example, with an embedded ambient temperature sensor, a smartphone can obtain the temperature reading in a few seconds;[1] for air quality monitoring, usually the sensor needs 30-60 seconds to be prepared to start sensing and then the sampling cycle is 2-10 seconds [10, 18].

Assumption 4. Cycle-Length-Satisfying Sequential Sensing. Each sensing cycle is sufficiently long to collect enough sensing values by *sequentially* allocating tasks — the next task is allocated after the sensing value of the previous task is returned.

Like *assumption 3*, if the sensing task is quick (e.g., a few seconds for temperature sensing), then *assumption 4* can also be satisfied for most MCS tasks with reasonable-length cycles (e.g., 30-minute cycle).

[1]The response time of the temperature/humidity sensor SHTC1 of Galaxy S4 is about 8 seconds [1].

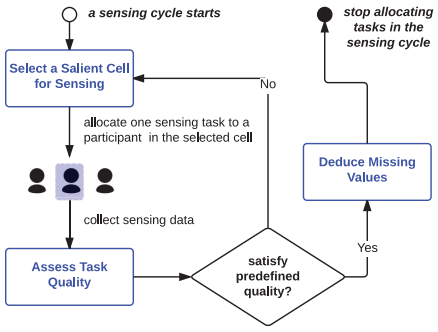

Figure 2: Workflow of CCS-TA.

In summary, the above assumptions are made for the following reasons:

- *Assumption 1* allows us to reduce the task allocation problem to a *cell selection* problem.
- Combining *assumptions 2 and 3*, we only need to allocate one task to one participant in cell i during cycle j in order to get the true sensing value from cell i in cycle j.
- *Assumption 4* allows us to use an iterative method, i.e., progressively selecting the cells for sensing, in each cycle, for solving the *cell selection* problem.

Problem Formulation

Based on the previous definitions and assumptions, we formulate the research problem as follows: *Given an MCS task with m cells and n cycles, and a sensing matrix reconstruction algorithm $\mathcal{R}$, select a minimal subset of sensing cells during the whole MCS task process (i.e., minimize the number of non-zero entries in the cell-selection matrix S), while ensuring that the overall sensing errors of at least $n \cdot p$ cycles are below the predefined error bound ϵ (i.e., satisfying (ϵ, p)-quality):*

$$\begin{aligned} \min \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} S[i,j] \\ s.t., \quad & |\{k|\mathcal{E}_k \le \epsilon, 1 \le k \le n\}| \ge n \cdot p \\ where \quad & \mathcal{E}_k = error(\hat{F}[:,k], F[:,k]) \\ & \hat{F} = \mathcal{R}(C),\ C = F \circ S \end{aligned}$$

If we know F in advance, it is possible for us to get the optimal S by enumerating all the possibilities (given sufficient computation time). However, in real life, we cannot foreknow F, which makes the problem challenging: (1) $\mathcal{E}_k$ cannot be directly obtained, and (2) the cell selection process is monotonic (i.e., only after we set $S[i,j] = 1$ can we get $F[i,j]$, and this selection cannot be retracted to save incentive costs). To overcome these difficulties, we propose CCS-TA, which leverages an iterative process to select cells for sensing in each cycle, with details elaborated in the following sections.

OVERVIEW OF CCS-TA

In this section, we introduce the design of CCS-TA. Figure 2 shows the workflow of CCS-TA. In each cycle, CCS-TA iteratively *selects the next salient cell for sensing* and waits for allocating a sensing task to a participant present in that cell, until the *estimated data quality* satisfies the predefined (ϵ, p)-*quality* requirement. Then, the task allocation stops and *missing data values of the unsensed cells are deduced*.

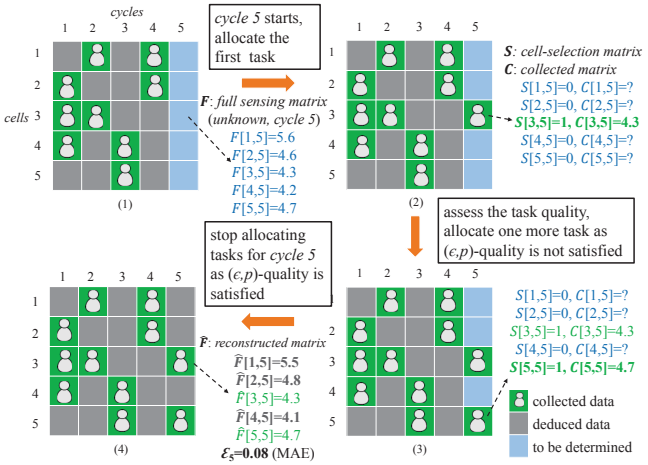

Figure 3: An example of CCS-TA process (5 cells and 5 cycles, temperature).

Figure 3 shows an example to illustrate the task allocation process of CCS-TA in one sensing cycle. Suppose the target sensing area contains five cells and the fifth sensing cycle starts currently; in the beginning, no sensing data is collected in *cycle 5* (Figure 3-1, ground truth F is unknown). CCS-TA works as follows:

1) CCS-TA selects the first salient cell (*cell 3*, $S[3,5]$=1) and allocates a sensing task to one participant appearing in *cell 3*. This participant performs the sensing task and returns the sensing data to CCS-TA (Figure 3-2, $C[3,5]$=$F[3,5]$=4.3°C).

2) After CCS-TA gets the sensing data of *cell 3*, it assesses whether the data quality satisfies the predefined (ϵ, p)-quality. Assume the assessment result is *no*, CCS-TA continues selecting the next salient cell (*cell 5*, $S[5,5]$=1) to allocate another sensing task (Figure 3-3, $C[5,5]$=$F[5,5]$=4.7°C).

3) After collecting the sensing data from *cells 3 and 5* in *cycle 5*, CCS-TA assesses whether the data quality satisfies the predefined (ϵ, p)-quality again. If the obtained result is *yes*, CCS-TA stops further task allocations for *cycle 5* and deduces the missing data of the unsensed cells (Figure 3-4, $\hat{F}[1,5]$, $\hat{F}[2,5]$, and $\hat{F}[4,5]$ are deduced).

DETAILED DESIGN OF CCS-TA

In this section, we describe the algorithms used in CCS-TA: *deducing missing values*, *determining task allocation stopping criterion*, and *selecting salient cell for sensing*.

Deducing Missing Values

To reconstruct the full sensing matrix from the partically collected sensing values, *Compressive Sensing* (*CS*) is commonly used in the literature [26, 42]. In this section, we first introduce the basic idea of CS, and then illustrate an enhanced version of CS, called *Spatio-Temporal Compressive Sensing* (*STCS*), which considers the spatial and temporal correlations among the environmental data explicitly to further improve

Algorithm 1 LOO-BI task allocation stopping criterion

Input:
C_{m*k}: collected sensing matrix with m locations and k cycles, non-collected entry is $null$ (current cycle is k).
$\mathcal{R}()$: a sensing matrix reconstruction algorithm (e.g., *STCS*).
$error$: an error metric (e.g., *mean absolute error*).
ϵ, p: predefined (ϵ, p)-quality requirement.
Output:
$stop$: *true*/*false* – stop/continue task allocation.

$\mathbf{x} \leftarrow [\,]$ ▷ vector storing collected data
$\mathbf{y} \leftarrow [\,]$ ▷ vector storing re-deduced data
for $i \leftarrow 1$ to m **do**
  if $C[i,k] \neq null$ **then** ▷ if data is collected in $C[i,k]$
    $\mathbf{x}.append(C[i,k])$
    $C' \leftarrow C$
    $C'[i,k] \leftarrow null$ ▷ remove one collected data
    $\hat{F}' \leftarrow \mathcal{R}(C')$ ▷ re-deduce the removed data
    $\mathbf{y}.append(\hat{F}'[i,k])$
  end if
end for
$P(\mathcal{E}_k \leq \epsilon) \leftarrow BayesianInference(error, \mathbf{x}, \mathbf{y}, \epsilon)$
if $P(\mathcal{E}_k \leq \epsilon) \geq p$ **then**
  $stop \leftarrow true$
else
  $stop \leftarrow false$
end if
return $stop$

the reconstruction performance [26, 39]. We use STCS to deduce missing values in CCS-TA, given its improved reconstruction accuracy over normal CS and the other methods [26].

CS: Compressive Sensing

Given a partially collected sensing matrix C, *compressive sensing* reconstructs the full sensing matrix $\hat{F}$ based on the low-rank property:

$$\begin{aligned} &\min\ rank(\hat{F}) \\ &s.t.,\ \hat{F} \circ S = C \end{aligned} \tag{3}$$

Directly solving this problem is hard because it is nonconvex. Based on the *singular value decomposition*, i.e., $\hat{F} = LR^T$, and compressive sensing theory [5, 12, 29], a more practical optimization problem is formulated [26, 39, 42], which changes rank minimization to minimizing the sum of L and R's Frobenius norms:

$$\min\ \lambda(||L||_F^2 + ||R||_F^2) + ||LR^T \circ S - C||_F^2 \tag{4}$$

where λ is used to make the trade-off between rank minimization and accuracy fitness. To get the optimal $\hat{F}$, we use an alternating least squares [26, 39, 42] procedure to estimate L and R iteratively ($\hat{F} = LR^T$).

STCS: Spatio-Temporal Compressive Sensing

As environment data such as temperature usually exhibits strong spatial and temporal correlations, explicit spatio-temporal correlations are introduced into compressive sensing in recent work [26, 39], called *spatio-temporal compressive sensing*, which focuses on the optimization function below:

$$\begin{aligned} \min\ &\lambda_r(||L||_F^2 + ||R||_F^2) + ||LR^T \circ S - C||_F^2 \\ &+ \lambda_s||\mathbb{S}(LR^T)||_F^2 + \lambda_t||(LR^T)\mathbb{T}^T||_F^2 \end{aligned} \tag{5}$$

where $\mathbb{S}$ and $\mathbb{T}$ are spatial and temporal constraint matrices respectively; λ_r, λ_s, and λ_t are chosen to balance the weights of different elements in the optimization problem.

Similar to the *CS* optimization problem (4), the above *STCS* optimization problem (5) could be solved by using alternating least squares [26, 39]. We elaborate below our strategies of choosing the temporal and spatial constraint matrices.

Temporal constraint matrix ($\mathbb{T}$): Like [26, 39], we choose the temporal constraint matrix $\mathbb{T}$ as $Toeplitz(0, 1, -1)_{n*n}$ (total n sensing cycles), which considers the temporal correlation in the following manner — for a specific cell, its sensing values in two continuous sensing cycles should be similar.

Spatial constraint matrix ($\mathbb{S}$): The spatial constraint matrix $\mathbb{S}$ is used to express the correlations between the sensing data from different cells. Generally, if two cells are close to each other, their sensing data might be similar. In [26], due to the lack of the GPS information, the authors have to use sensor network topology to construct $\mathbb{S}$. Here, since we can get the GPS positions of the cells, we directly use the distance to model the correlation between cell i and j, $c_{i,j}$, as $1/distance(cell_i, cell_j)$. Then, we get $\mathbb{S}$ as follows:

$$\mathbb{S}_{i,i} = -1;\ \mathbb{S}_{i,j} = c_{i,j} / \sum_{k \neq i} c_{i,k},\ \text{if } i \neq j$$

Determining Task Allocation Stopping Criterion

In CCS-TA, for each sensing cycle, deciding when to stop task allocation is a key issue: if we stop too early, the server might not collect enough data to achieve the predefined (ϵ, p)-quality for the MCS task; if we stop too late, then the server might collect redundant data, which would lead to waste in the organizer's budget. A well-designed stopping criterion should guarantee the data quality to satisfy the predefined (ϵ, p)-quality requirement, while allocating sensing tasks to as few cells as possible in each sensing cycle.

To this end, we propose a *Leave-One-Out Bayesian-Inference* (*LOO-BI*) based method to decide the stopping criterion for each sensing cycle, as shown in Algorithm 1. First, LOO-BI uses *leave-one-out* re-sampling method to obtain a set of re-deduced sensing data with the corresponding true collected data. Then, comparing the re-deduced data to the true collected data, *Bayesian inference* is leveraged to assess whether the current data quality can satsify the predefined (ϵ, p)-quality requirement or not.

Leave-One-Out Re-Sampling

In statistics, *leave-one-out* is a popular re-sampling method to measure the performance for many prediction and classification algorithms [22]. Suppose we have m true observations, the basic idea of leave-one-out is for each time, we leave one observation out and using the other $m-1$ observations (as training data set) to make a prediction for the excluded observation. By running this process on all m observations, we get m predictions accompanying with the m true observations, which can be used to estimate the prediction error.

We adopt the basic idea of leave-one-out in the LOO-BI (Algorithm 1), where we actually run leave-one-out on the col-

lected data of the current cycle k (line 3-11). In each iteration, LOO-BI attempts to temporarily remove one piece of collected data of the current cycle k and then run the reconstruction algorithm $\mathcal{R}$ to re-deduce the removed data (line 6-9). After enumerating all the collected data in cycle k, we finally get two vectors $\mathbf{x}$ and $\mathbf{y}$: $\mathbf{x}$ stores the true collected data for the current cycle k, while $\mathbf{y}$ stores the corresponding re-deduced data by using leave-one-out. Suppose we have already collected data from m' cells for the current cycle, then both $\mathbf{x}$ and $\mathbf{y}$ have m' elements:

$$\mathbf{x} = \langle x_1, x_2, \cdots, x_{m'} \rangle, \quad \mathbf{y} = \langle y_1, y_2, \cdots, y_{m'} \rangle$$

where x_i is the i *th* ground truth data collected in cycle k, and y_i is the corresponding re-deduced data by leaving x_i out of the collected data.

Based on the ground truth set $\mathbf{x}$ and the leave-one-out re-deduced set $\mathbf{y}$, in the next section, we will describe how to assess whether the task could satisfy (ϵ, p)-quality or not.

Assessing Task Quality by Bayesian Inference

According to the *law of large numbers* in probability theory [14], ensuring a task satisfies (ϵ, p)-quality can be achieved by making sure that the probability of the error of each sensing cycle being at most ϵ be at least p, formally:

$$\forall k, 1 \leq k \leq n : P(\mathcal{E}_k \leq \epsilon) \geq p \tag{6}$$

Thus, the problem of assessing whether a task can satisfy (ϵ, p)-quality is converted to calculate $P(\mathcal{E}_k \leq \epsilon)$. To this end, we need to estimate the probability distribution for the cycle k's overall sensing error $\mathcal{E}_k$, which can be done with Bayesian inference [16].

In Bayesian inference, we see $\mathcal{E}_k$ as an unknown parameter with a *prior* probability distribution $g(\mathcal{E}_k)$;[2] based on our observation $\boldsymbol{\theta}$ (obtained from the leave-one-out re-deduced data, will be explained later), we update the probability distribution of $\mathcal{E}_k$, getting the *posterior* probability distribution $g(\mathcal{E}_k|\boldsymbol{\theta})$ according to the *Bayes' Theorem*:

$$g(\mathcal{E}_k|\boldsymbol{\theta}) = \frac{f(\boldsymbol{\theta}|\mathcal{E}_k) g(\mathcal{E}_k)}{\int_{-\infty}^{\infty} f(\boldsymbol{\theta}|\mathcal{E}_k) g(\mathcal{E}_k) d\mathcal{E}_k} \tag{7}$$

where $f(\boldsymbol{\theta}|\mathcal{E}_k)$ is the *likelihood* function that represents the conditional probability of observing $\boldsymbol{\theta}$ given $\mathcal{E}_k$.

The posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ is thus the estimated probability distribution of $\mathcal{E}_k$, based on which we can approximate $P(\mathcal{E}_k \leq \epsilon)$:

$$P(\mathcal{E}_k \leq \epsilon) \approx \int_{-\infty}^{\epsilon} g(\mathcal{E}_k|\boldsymbol{\theta}) d\mathcal{E}_k \tag{8}$$

If $P(\mathcal{E}_k \leq \epsilon) \geq p$, then CCS-TA stops the task allocation for current cycle k and waits for the start of the next cycle; otherwise, CCS-TA continues selecting a new cell to collect sensing data (see Algorithm 1, line 12-18).

[2]The prior distribution is often selected as a non-informative probability distribution (such as uniform distribution) if we do not have specific prior knowledge about $\mathcal{E}_k$.

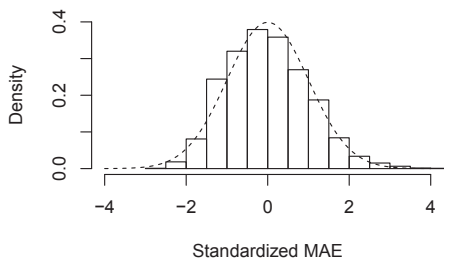

Figure 4: Histogram of mean absolute error with fitted normal curve.

Note that for different error metrics, the calculation processes for the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ are different (as the likelihood functions $f(\boldsymbol{\theta}|\mathcal{E}_k)$ are usually different). In the next two subsections, we introduce how to compute the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ for two-widely used error metrics, *mean absolute error* (for continuous value, e.g., temperature [3]) and *classification error* (for classification label, e.g., AQI descriptor [40]).

Bayesian Inference for Mean Absolute Error

When $\mathcal{E}_k$ is defined as mean absolute error (MAE, Eq. (1)), we use the absolute difference of $\mathbf{y}$ (leave-one-out re-deduced data set) and $\mathbf{x}$ (true collected data set) as the observation $\boldsymbol{\theta}$ (suppose m' sensed data has already been collected in the current cycle):

$$\begin{aligned} \boldsymbol{\theta} &= abs(\mathbf{y} - \mathbf{x}) \\ &= \langle |y_1 - x_1|, |y_2 - x_2|, \ldots, |y_{m'} - x_{m'}| \rangle \end{aligned}$$

After inspecting our evaluation temperature dataset (which will be described in detail later), we find that the MAE in each sensing cycle follows the normal distribution. Figure 4 shows the histogram of the standardized MAE (i.e., MAE divided by the standard deviation of MAE in each cycle) when we keep the data of 10% of the cells and infer that of the remaining 90%. Thus, by supposing that the sampled absolute errors satisfy the normal distribution around mean $\mathcal{E}_k$ and variance σ^2, we get the likelihood function (the probability of observing $\boldsymbol{\theta}$ given a specific $\mathcal{E}_k$):

$$f(\boldsymbol{\theta}|\mathcal{E}_k) : \ \theta_i = |y_i - x_i| \sim \mathcal{N}(\mathcal{E}_k, \sigma^2)$$

Given the above, calculating the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ from the above likelihood function and observation is a classic Bayesian statistics problem: *inferring normal mean with unknown variance*, which can be solved by fixing the variance σ^2 to the sample variance s^2 and then directly calculating the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ by t-distribution [4]. For the prior $g(\mathcal{E}_k)$, we select the *Jeffreys' flat prior* [23]: $g(\mathcal{E}_k) = 1, \ \forall \mathcal{E}_k$. Then, the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ satisfies the following (m'-1) degree-of-freedom t-distribution:

$$g(\mathcal{E}_k|\boldsymbol{\theta}) \sim t_{m'-1}(\bar{\theta}, s^2) \tag{9}$$

where $\bar{\theta}$ is the sample mean of the values in $\boldsymbol{\theta}$. Based on the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ obtained from Eq. (9), we can then use Eq. (8) to calculate the $P(\mathcal{E}_k \leq \epsilon)$ and decide whether more cells should be selected in the current cycle k.

Bayesian Inference for Classification Error

For classification problems, the classification error (Eq. (2)) measures the percentage of the test data that is classified into a wrong label. We now show how we use Bayesian inference to estimate the posterior distribution for the classification error $\mathcal{E}_k$. First, as our reconstruction algorithm $\mathcal{R}$ deals with continuous values, we map $\mathbf{x}$ and $\mathbf{y}$ to their corresponding classification labels using the mapping function $\psi()$ in Eq. (2), e.g., for the PM2.5 AQI value between 0 and 50, we map it into the AQI descriptor label "Good". Afterward, we use the $I()$ function on $\psi(\mathbf{x})$ and $\psi(\mathbf{y})$ to get our observation $\boldsymbol{\theta}$:

$$\begin{aligned}\boldsymbol{\theta} &= I(\psi(\mathbf{x}), \psi(\mathbf{y})) \\ &= \langle I(\psi(x_1), \psi(y_1)), I(\psi(x_2), \psi(y_2)), \ldots, I(\psi(x_{m'}), \psi(y_{m'}))\rangle\end{aligned}$$

Each θ_i is either 1 (success, $\psi(x_i) = \psi(y_i)$) or 0 (failure, $\psi(x_i) \neq \psi(y_i)$), and the classification error $\mathcal{E}_k$ is exactly the failure ratio. Suppose each θ_i is independent, then it satisfies the *Bernoulli* distribution with the probability of $1 - \mathcal{E}_k$:

$$f(\boldsymbol{\theta}|\mathcal{E}_k) : \ \theta_i = I(\psi(x_i), \psi(y_i)) \sim Bernoulli(1 - \mathcal{E}_k)$$

Based on this likelihood function $f(\boldsymbol{\theta}|\mathcal{E}_k)$, the problem to infer the posterior $g(\mathcal{E}_k|\boldsymbol{\theta})$ is converted to a classic Bayesian statistics problem, *Coin Flipping* [4, 16]. We choose the *uniform prior* for $\mathcal{E}_k$: $g(\mathcal{E}_k) = 1 \ \text{ for } \ 0 \leq \mathcal{E}_k \leq 1$. Then the posterior for $\mathcal{E}_k$ follows *Beta* distribution [4, 16]:

$$g(\mathcal{E}_k|\boldsymbol{\theta}) \sim Beta(m' - z + 1, z + 1) \tag{10}$$

where $z = \sum_{i=1}^{m'} \theta_i$, i.e., the number of successes. Then, for the classification error $\mathcal{E}_k$, combing Eq. (10) and Eq. (8), we can calculate the $P(\mathcal{E}_k \leq \epsilon)$ to decide whether we need to continue the task allocation.

Computation Complexity of LOO-BI

As there are two phases for the computation of LOO-BI, we discuss the computation complexity of both phases respectively. First, to use leave-one-out to estimate the sensing error, LOO-BI needs to run the reconstruction algorithm $\mathcal{R}$ for m' times, where m' is the number of the already collected sensing values in the current cycle. This time consumption dominates the running time so that the computation complexity is $O(m' \cdot T_{\mathcal{R}})$ for the leave-one-out part, where $T_{\mathcal{R}}$ is the complexity of the reconstruction algorithm $\mathcal{R}$. For the second part, Bayesian inference, recalling Eq. (9) and Eq. (10), we can simply use two distributions, *t*-distribution and *Beta* distribution respectively, to calculate the posterior for mean absolution error and classification error, which makes the computation process of Bayesian inference run much faster than the leave-one-out part. In summary, the computation complexity of LOO-BI is dominated by the leave-one-out part, which is $O(m' \cdot T_{\mathcal{R}})$. If m' is large, sequentially executing LOO-BI might consume much time. Fortunately, though we need to run $\mathcal{R}$ for m' times, each run is independent, so LOO-BI can be easily parallelized to accelerate as needed.

Selecting Salient Cell for Sensing

When the output of the stopping criterion LOO-BI is *false*, CCS-TA will continue selecting more cells for sensing. During this process, selecting some salient cells for sensing may reduce the overall sensing error more significantly, e.g., the missing values of some cells might incur more inference errors and are thus more uncertain. If CCS-TA can identify these salient cells, the number of the allocated tasks can possibly be reduced to make the data quality satisfy the predefined (ϵ, p)-quality requirement earlier, compared to other simple cell selection methods such as random selection.

Based on the recent research advances in *active learning* on matrix completion, we use a method proposed in [6], called *Query by Committee (QBC)*, to select the salient cell to allocate the next task (*committee* here refers to a set of various matrix reconstruction algorithms). QBC attempts to use each algorithm in the committee to reconstruct the full sensing matrix. Then it allocates the next task to the cell with the largest variance among the deduced values of different algorithms [6].

In CCS-TA, the committee includes *CS*, *STCS*, *KNN-S*, and *KNN-T*. *CS* and *STCS* are described previously, and *KNN-S* and *KNN-T* use the classic *K-Nearest Neighbors (KNN)* [8] method to interpolate missing values. For a missing value, KNN uses a weighted average of the values of the k nearest neighbors. In sensing matrix reconstruction, we can perform KNN on columns or rows, i.e., using spatial (KNN-S) or temporal (KNN-T) *K* nearest neighbors. Specifically, for a missing value $F[i, j]$ (cell i in cycle j), KNN-S attempts to find *K* nearest spatial neighbors $F[i', j]$ (weight $\propto 1/distance(cell_i, cell_{i'})$), while KNN-T attempts to find *K* nearest temporal neighbors $F[i, j']$ (weight $\propto 1/|j - j'|$).

Computation Complexity of QBC

The running time of the QBC method is primarily spent on using all the algorithms in the committee to reconstruct the sensing matrix. Suppose for each reconstruction algorithm $\mathcal{R}_i$ in the committee, the computation complexity is $T_{\mathcal{R}_i}$, then the complexity of QBC is $O(\sum_i T_{\mathcal{R}_i})$. If the committee contains more algorithms, then running QBC sequentially will cost more time. However, like LOO-BI, since the executions of different reconstruction algorithms are independent, QBC can also be parallelized to improve runtime performance.

Summary of the Design of CCS-TA

In CCS-TA, we leverage *STCS* to deduce the missing values of unsensed cells based on the sensing values of the selected salient cells, which are actively chosen by *QBC* to reduce the overall sensing error more effectively. Furthermore, we propose *LOO-BI* to determine when the task allocation should be stopped in each sensing cycle to achieve the (ϵ, p)-*quality* predefined by the MCS organizer.

EVALUATION

In this section, we use two sensing datasets — temperature and PM 2.5 air quality — to evaluate CCS-TA.

Experiment Setup

To evaluate the real-world applicability of our work, we use two real-life sensing datasets, *temperature* (*TEMP*) and *PM 2.5 air quality* (*PM25*), to create two experimental scenarios. Note that though the two datasets are collected by

sensor networks or static stations, we assume the MCS participants can obtain them using smartphone applications (as in [10, 18]).

The *TEMP* dataset has been collected by the *SensorScope* project [20]. The project maintains nearly one hundred sensors across the EPFL campus (500m×300m) to collect various environment readings, e.g., temperature, solar radiation, and humidity. For our evaluation, we divide the target area into 100 cells (each cell is 50m×30m), and find that 57 cells are deployed with the temperature sensors. If a cell has more than one sensor, we select the sensor collecting the most number of the temperature readings. In summary, *TEMP* dataset includes the temperature readings in 57 cells from 2007-07-01 to 2007-07-07; each sensing cycle lasts for 30 minutes. To measure the data quality for temperature, referring to [3], we use the *mean absolute error* (Eq. (1)) as the metric.

PM25 dataset has been collected by the *U-Air* project [40], which includes PM2.5, PM10, and NO2 AQI (air quality index) values reported by 36 air quality monitoring stations in Beijing. For our evaluation, like [40], we also split the Beijing urban area to 1km×1km cells and only use the cells where the stations are situated. In summary, *PM25* dataset includes the PM2.5 AQI values on 36 station-situated cells from 2013-11-10 to 2013-11-20; each sensing cycle lasts for one hour. To measure the data quality for PM2.5 AQI, we follow the methods used in [40] — each AQI value is classified to a range called *descriptor*, which is used as the basis for computing the *classification error* discussed in Eq. (2). Six levels of descriptors are defined: *Good* (0∼50), *Moderate* (51∼100), *Unhealthy for Sensitive Groups* (101∼150), *Unhealthy* (150∼200), *Very Unhealthy* (201∼300), and *Hazardous* (301∼500).

Performance Analysis: Deducing Missing Values

First, we aim to verify the effectiveness of *STCS* in deducing missing values for temperature and PM 2.5 compared to the other state-of-the-art matrix reconstruction algorithms described before, including CS, KNN-S, and KNN-T. Note that for STCS and CS, optimization parameters are selected as in [26]; for KNN-S and KNN-T, we set K to 3 because by comparing different values of K, we find that K = 3 achieves the best performance for KNN-S and KNN-T.

Figure 5 and 6 show the overall sensing error (with standard deviation) of different reconstruction algorithms on TEMP and PM25 datasets, respectively. In this experiment, we iteratively consider each sensing cycle *k* as the latest cycle, reconstruct the full sensing matrix based on the collected sensing matrix from cycle *1* to *k*, and calculate the overall sensing error for the cycle *k*. The x axis, i.e., *sparsity ratio*, denotes the fraction of unsensed entries in the collected sensing matrix. Consistent with the literature [26], our evaluation results also show the improved accuracy of STCS over the other methods, verifying that compressive sensing is effective in deducing the missing environmental data such as temperature and air quality, especially when the explicit spatio-temporal correlations are incorporated. Therefore, for both TEMP and PM25 datasets, we use STCS to deduce the missing values.

	TEMP		*PM25*	
ϵ	0.25°C	0.30°C	$6/36$	$9/36$
$p = 0.90$	0.915	0.919	0.904	0.912
$p = 0.95$	0.943	0.949	0.944	0.965

Table 1: Fraction of the cycles whose errors are lower than the error bound ϵ.

Performance Analysis: Stopping Criterion

Then, we evaluate the effectiveness of the stopping criterion *LOO-BI*. We use various settings of (ϵ, p)-quality to see what fraction of sensing cycles can actually keep the overall sensing error less than ϵ. Table 1 shows the results for both TEMP and PM25 datasets. For p, we purposely set it to a large value as 0.90 and 0.95, i.e., ensuring *most* (90% or 95%) sensing cycles' error to be less than the predefined error bound ϵ, which we think is a more reasonable and realistic scenario than small p for MCS organizers. For ϵ, we vary it from 0.25°C to 0.30°C for TEMP and $6/36$ to $9/36$ for PM25. Note that for PM25, the error bound $X/36$ represents that to satisfy this error bound, more than $36 - X$ cells have the correct AQI level.

From Table 1, we see that for any predefined error bound ϵ, the actual fraction of the cycles whose errors are less than ϵ is quite similar to the p in the predefined (ϵ, p)-quality. Specifically, for $p = 0.90$, all the actual fractions are larger than 0.90; for $p = 0.95$, even though the the actual fractions sometimes are slightly less than 0.95, the values are still quite near 0.95 (in our settings, the smallest actual fraction is 0.943, only 0.007 smaller than 0.95). Based on these results, we verify that, by using LOO-BI as the stopping criterion, CCS-TA can well satisfy the predefined (ϵ, p)-quality.

Performance Analysis: Number of Allocated Tasks

After selecting the best sensing data reconstruction algorithm and verifying the effectiveness of the stopping criterion, now we focus on analyzing the research objective — *how many allocated tasks could CCS-TA reduce while ensuring a certain data quality?*

We first compare CCS-TA with baseline approaches to see the advantage of CCS-TA. Afterward, we compare CCS-TA with the *OPTIMAL* solution, i.e., if the sensing data of all the cells for each cycle is foreknown, what is the minimal number of allocated tasks that we can achieve under a predefined error bound ϵ. Although the OPTIMAL solution is unrealistic from a performance perspective, we want to show the performance gap between CCS-TA and the ideal condition to identify the potential improvement space for CCS-TA.

CCS-TA vs. Baselines

To compare with CCS-TA, we use the following baselines:

- **ICDM-FIX-k**: An alternative way to extend the QBC active learning method [6] from ICDM'13 to the MCS task allocation mechanism is fixing the task number *k* in each sensing cycle, while still using QBC to actively select cells to allocate tasks; we call this modified algorithm ICDM-FIX-k. Compared to ICDM-FIX-k, CCS-TA shows the benefit brought by LOO-BI, which helps the organizer decide when to stop the task allocation, thus adaptively ad-

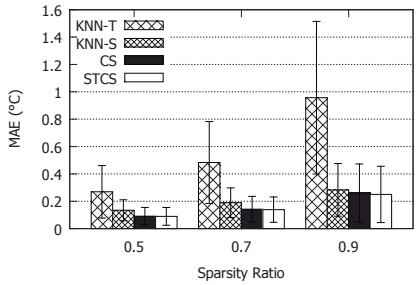

Figure 5: Error of different reconstruction algorithms (TEMP).

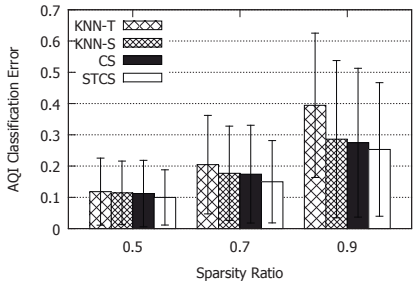

Figure 6: Error of different reconstruction algorithms (PM25).

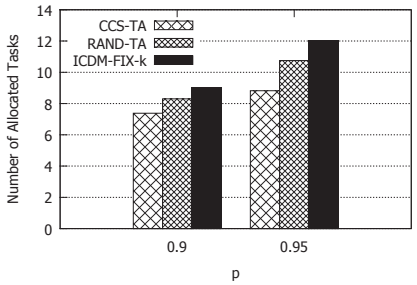

Figure 7: Number of allocated tasks (TEMP, $\epsilon = 0.25°$C, varying p).

TEMP	*CCS-TA*	*RAND-TA*	*ICDM-FIX-k*
$p = 0.9$	0.915	0.900	0.911 ($k = 9$)
$p = 0.95$	0.943	0.943	0.949 ($k = 12$)

Table 2: Fraction of the cycles whose errors are lower than 0.25°C (TEMP).

PM25	*CCS-TA*	*RAND-TA*	*ICDM-FIX-k*
$p = 0.9$	0.912	0.916	0.884 ($k = 16$)
$p = 0.95$	0.965	0.969	0.961 ($k = 18$)

Table 3: Fraction of the cycles whose errors are lower than 9/36 (PM25).

justing the number of the sensed cells for different cycles.

- **RAND-TA**: In this baseline, we *randomly* select the next cell for sensing, but still leverage LOO-BI as the task allocation stopping criterion. Compared to RAND-TA, CCS-TA shows the advantage of applying QBC to select the salient cells for sensing.

On the TEMP dataset, for the predefined (ϵ, p)-quality, we set the error bound ϵ as 0.25°C and p as 0.9 or 0.95. Before comparing the number of allocated tasks, we need to ensure that all the methods can achieve the similar task quality. While CCS-TA has already been verified to be able to satisfy (ϵ, p)-quality in the previous section, now we need to check the two baselines. Table 2 shows the results. We can see that RAND-TA can also satisfy (ϵ, p)-quality well, as it adopts LOO-BI as the stopping criterion like CCS-TA. For ICDM-FIX-k, we tune k to achieve the similar task quality, which leads to $k = 9$ for $p = 0.9$ and $k = 12$ for $p = 0.95$.

As all the methods can satisfy similar task quality, we compare their numbers of the allocated tasks (i.e., number of selected cells) in Figure 7. When $p = 0.9$, CCS-TA can allocate 11.1% fewer tasks than RAND-TA, and 18.0% fewer tasks than ICDM-FIX-9; when $p = 0.95$, CCS-TA outperforms RAND-TA and ICDM-FIX-12 by assigning 18.0% and 26.5% fewer tasks, respectively. Specifically, CCS-TA allocates tasks to only 12.9% (15.5%) cells on average, while ensuring the overall sensing error below 0.25°C in 90% (95%) of the cycles.

On the PM25 dataset, we get similar observations. See Table 3 and Figure 8 for the results when we set ϵ to 9/36 (i.e., 0.25) and p to 0.90 or 0.95. In general, while achieving similar data quality, CCS-TA allocates 7.5-9.9% and 25.0-31.9% fewer tasks than RAND-TA and ICDM-FIX-k, respectively.

CCS-TA vs. OPTIMAL

Now we compare CCS-TA with the *OPTIMAL* solution. The OPTIMAL solution assumes the sensing data of all the cells

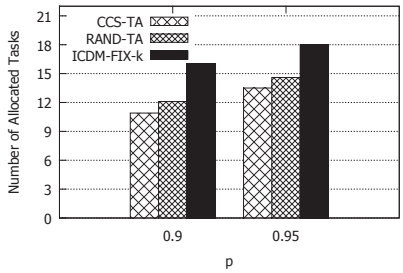

Figure 8: Number of allocated tasks (PM25, ϵ = 9/36, varying p).

	Reconstruction (STCS)	*Stopping Criterion (LOO-BI)*	*Cell Selection (QBC)*
TEMP	0.95s	<9s	1.04s
PM25	0.75s	<7s	0.91s

Table 4: Running time for each sub-step of CCS-TA.

for each cycle is foreknown, so it can choose the optimal (i.e., minimal-size) cell combination to satisfy the error bound ϵ. OPTIMAL is not practical in real life, but it can serve as the bound for CCS-TA. To get the OPTIMAL solution, we enumerate all the possible cell combinations from size 1 to m until we find the smallest-size cell combination that can meet the error bound. The enumeration process is highly time-consuming, so we trim the TEMP dataset to a small number of cells, i.e., 15 cells, to conduct this experiment.

Suppose the error bound ϵ is 0.25°C, for CCS-TA, we set $p = 0.95$ and it actually guarantees the overall sensing error below 0.25°C in 95.4% of the cycles, and the average task number in each cycle is 4.74. In contrast, OPTIMAL allocates only 2.23 tasks per cycle (53.0% fewer tasks than CCS-TA) while ensuring the error below 0.25°C in 100% of the cycles (4.6% more cycles than CCS-TA). Therefore, *a noticeable gap still exists between CCS-TA and OPTIMAL*, which inspires us to to improve CCS-TA to narrow this gap in the future work.

Running Time Analysis

Finally, we study the running time of CCS-TA to see whether it can satisfy the real-life MCS scenario, as well as the speedup it provides vis-a-vis the OPTIMAL solution. We run the experiments on a laptop (Intel core i7-3612QM, 8GB RAM, Windows 7) with Python 2.7. Table 4 shows the running time for different parts of CCS-TA. The most time-consuming part is the LOO-BI stopping criterion, which needs 9 seconds at most. As described previously, LOO-BI is suitable for being parallelized, which can help improve its performance. In summary, on our experimental setup, CCS-TA spends $\sim$10 seconds to allocate one task, i.e., es-

timating the task quality once and, if it cannot meet the predefined (ϵ, p)-quality, finds the next sensing cell. Thus, for the sensing tasks requiring a few seconds, such as temperature sensing, if we can find a participant and receive her sensing data in 10 seconds, CCS-TA can allocate tasks to $\sim$180 cells in an hour; even for the air quality sensing that needs 60 seconds to get a valid reading [10, 18], CCS-TA can allocate tasks to $\sim$50 cells in an hour. We believe this efficiency can meet most real-life MCS scenarios, especially with more powerful servers in CCS-TA deployment environment and more efficient smartphone-equipped sensors in the future. As a comparison, the OPTIMAL solution, even in the trimmed TEMP dataset with 15 cells, needs $>$30 minutes to find an optimal combination containing only 6 cells.

In summary, we have shown that each constituent of the our approach performs better than the baseline. We have shown (in Figures 5 and 6) how STCS is the best choice for reconstruction, and comparison with ICDM-FIX-k shows how LOO-BI as the stopping criterion is better. Since CCS-TA and RAND-TA share the reconstruction and stopping criterion techniques, differing only in the cell-selection process, they are close in performance, although our approach is still marginally better.

RELATED WORK

Task Allocation in Location-centric MCS

Existing work about the task allocation for location-centric MCS applications mainly uses the coverage ratio of the target area as the major quality metric. In early work on this topic, Reddy et al. [30] attempted to recruit a predefined number of participants to maximize the spatial coverage. Later, various work attempts to extend this type of coverage-maximization participant recruitment to different considerations, e.g., participants' reverse auction incentive mechanism [21] and travel time budget [19]. On the other hand, work such as [2, 17, 33, 35, 38] attempts to minimize the incentive budget and/or energy consumption under a full or high probabilistic coverage constraint. Compared to the existing work, we do not use the coverage ratio as the quality metric. Instead, we use a more essential metric, i.e., overall sensing error, to represent data quality, based on which we attempt to reduce the number of the allocated tasks to help the MCS organizers save budget.

Compressive Sensing Applications

Compressive sensing theory [5, 12, 29], is increasingly becoming a powerful tool to reconstruct a sparse vector or matrix based on the sparsity property of vector or low rank property of matrix. Thus a large number of applications based on compressive sensing have appeared, such as network traffic reconstruction [39], environmental data recovery [26, 27, 28], road traffic monitoring [42], and face recognition on smartphones [32]. While the above work primarily focuses on designing the effective algorithms to minimize the reconstruction error under different scenarios, our objective is to minimize the number of the allocated tasks and meet a predefined data quality requirement, thus opening up the possibility of using any suitable deduction techniques. Indeed, in CCS-TA, the compressive sensing algorithms proposed in the above work, e.g., *STCS* [26, 39], is just one possible implementation for deducing the missing values of the unsensed cells. Recently, assuming a fixed number of data need to be collected in each cycle and different sensing data require different costs, Xu et. al [36] study the trade-off between total costs and overall data quality in compressive sensing. Different from [36], instead of fixing the number of data collected in each cycle, we aim to minimize the number of data collected in each cycle while ensuring the data quality.

Besides compressive sensing, there still exist other state-of-the-art methods to deduce missing values for unsensed cells, such as multichannel singular spectrum analysis [25] and expectation maximization [31]. As existing work [26, 42] shows that compressive sensing outperforms these methods, we currently do not explore them in CCS-TA.

Active Learning for Matrix Completion

To solve the problem of choosing the best cells for sensing, the recent techniques on active learning for matrix completion [6, 24, 34], which employ different criteria to actively choose the entry in a matrix, are all applicable. Currently, we use *Query by Committee* in CCS-TA due to its easy implementation and good performance [6].

CONCLUSION AND FUTURE WORK

In this paper, we attempt to reduce the number of the required sensing cells and thus the number of the allocated tasks to participants in location-centric MCS applications by considering the temporal and spatial correlations among the sensing data from different cells. To that end, we propose a novel task allocation framework, called *CCS-TA*, combining the state-of-the-art compressive sensing, Bayesian inference, and active learning mechanisms to actively select a minimum number of sensing cells in each cycle while deducing the missing values of the remaining cells, and ensuring that the overall data accuracy meets a predefined bound. Evaluation results on real-world temperature and air quality monitoring datasets show the effectiveness and applicability of CCS-TA.

In the future, we plan to improve this work by re-examining some assumptions made in this work. For example, in real life, the assumption of *Massive Candidate Participants* may not always hold for all the MCS applications. It may happen that, for some cycles, the organizer possibly cannot find any participant to conduct a task in the selected salient cell. In that case, the *task allocation* problem cannot be simplified to the *cell selection* problem and we need to take participants' mobility into account to address the problem. Moreover, in addition to the mean absolute error and classification error, we will further study to use Bayesian inference to assess other kinds of error metrics, e.g., root mean squared error, where some techniques, such as *Markov Chain Monte Carlo* [9] and *Bootstrap* [13] may be needed. Finally, although handling mobility explicitly is not the focus of this paper due to our assumptions, addressing fine-grained and more dynamic mobility would be considered in our future work.

ACKNOWLEDGEMENTS

This research is partly supported by the Microsoft collaborative research grant, EU FP7 MONICA project (PIRSES-GA-2011-295222), and the National High Technology Research and Development Program of China (863) under Grant No. 2013AA01A605.

REFERENCES

1. Shtc1 - digital temperature and humidity sensor. `http://www.sensirion.com/en/products/humidity-temperature/humidity-temperature-sensor-shtc1/`. Accessed: 2015-06-24.
2. Ahmed, A., Yasumoto, K., Yamauchi, Y., and Ito, M. Distance and time based node selection for probabilistic coverage in people-centric sensing. In *SECON* (2011), 134–142.
3. Bolstad, P. V., Swift, L., Collins, F., and Régnière, J. Measured and predicted air temperatures at basin to regional scales in the southern appalachian mountains. *Agricultural and Forest Meteorology 91*, 3 (1998), 161–176.
4. Bolstad, W. M. *Introduction to Bayesian statistics*. John Wiley & Sons, 2007.
5. Candès, E. J., and Recht, B. Exact matrix completion via convex optimization. *Foundations of Computational mathematics 9*, 6 (2009), 717–772.
6. Chakraborty, S., Zhou, J., Balasubramanian, V., Panchanathan, S., Davidson, I., and Ye, J. Active matrix completion. In *ICDM* (2013), 81–90.
7. Chon, Y., Lane, N. D., Kim, Y., Zhao, F., and Cha, H. Understanding the coverage and scalability of place-centric crowdsensing. In *UbiComp* (2013), 3–12.
8. Cover, T., and Hart, P. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory 13*, 1 (1967), 21–27.
9. Dellaportas, P., and Roberts, G. O. An introduction to mcmc. In *Spatial statistics and computational methods*. Springer, 2003, 1–41.
10. Devarakonda, S., Sevusu, P., Liu, H., Liu, R., Iftode, L., and Nath, B. Real-time air quality monitoring through mobile sensing in metropolitan areas. In *UrbComp* (2013), 15:1–15:8.
11. D'Hondt, E., Zaman, J., Philips, E., Boix, E. G., and De Meuter, W. Orchestration support for participatory sensing campaigns. In *UbiComp* (2014), 727–738.
12. Donoho, D. L. Compressed sensing. *IEEE Transactions on Information Theory 52*, 4 (2006), 1289–1306.
13. Efron, B., and Tibshirani, R. J. *An introduction to the bootstrap*. CRC press, 1994.
14. Feller, W. *An introduction to probability theory and its applications*, vol. 2. John Wiley & Sons, 2008.
15. Ganti, R. K., Ye, F., and Lei, H. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine 49*, 11 (2011), 32–39.
16. Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. *Bayesian data analysis*. CRC press, 2013.
17. Hachem, S., Pathak, A., and Issarny, V. Probabilistic registration for large-scale mobile participatory sensing. In *PerCom* (2013), 132–140.
18. Hasenfratz, D., Saukh, O., Sturzenegger, S., and Thiele, L. Participatory air pollution monitoring using smartphones. In *2nd International Workshop on Mobile Sensing* (2012).
19. He, S., Shin, D.-H., Zhang, J., and Chen, J. Toward optimal allocation of location dependent tasks in crowdsensing. In *INFOCOM* (2014), 745–753.
20. Ingelrest, F., Barrenetxea, G., Schaefer, G., Vetterli, M., Couach, O., and Parlange, M. Sensorscope: Application-specific sensor network for environmental monitoring. *ACM Transactions on Sensor Networks 6*, 2 (2010), 17:1–17:32.
21. Jaimes, L. G., Vergara-Laurens, I., and Labrador, M. A. A location-based incentive mechanism for participatory sensing systems with budget constraints. In *PerCom* (2012), 103–108.
22. James, G., Witten, D., Hastie, T., and Tibshirani, R. *An introduction to statistical learning*. Springer, 2013.
23. Jeffreys, H. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences 186*, 1007 (1946), 453–461.
24. Karimi, R., Freudenthaler, C., Nanopoulos, A., and Schmidt-Thieme, L. Non-myopic active learning for recommender systems based on matrix factorization. In *IRI* (2011), 299–303.
25. Kondrashov, D., and Ghil, M. Spatio-temporal filling of missing points in geophysical data sets. *Nonlinear Processes in Geophysics 13*, 2 (2006), 151–159.
26. Kong, L., Xia, M., Liu, X.-Y., Wu, M.-Y., and Liu, X. Data loss and reconstruction in sensor networks. In *INFOCOM* (2013), 1654–1662.
27. Quer, G., Masiero, R., Pillonetto, G., Rossi, M., and Zorzi, M. Sensing, compression, and recovery for wsns: Sparse signal modeling and monitoring framework. *IEEE Transactions on Wireless Communications 11*, 10 (2012), 3447–3461.
28. Rana, R. K., Chou, C. T., Kanhere, S. S., Bulusu, N., and Hu, W. Ear-phone: an end-to-end participatory urban noise mapping system. In *IPSN* (2010), 105–116.
29. Recht, B., Fazel, M., and Parrilo, P. A. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review 52*, 3 (2010), 471–501.
30. Reddy, S., Estrin, D., and Srivastava, M. Recruitment framework for participatory sensing data collections. In *Pervasive*. 2010, 138–155.
31. Schneider, T. Analysis of incomplete climate data: Estimation of mean values and covariance matrices and imputation of missing values. *Journal of Climate 14*, 5 (2001), 853–871.

32. Shen, Y., Hu, W., Yang, M., Wei, B., Lucey, S., and Chou, C. T. Face recognition on smartphones via optimised sparse representation classification. In *IPSN* (2014), 237–248.

33. Sheng, X., Tang, J., and Zhang, W. Energy-efficient collaborative sensing with mobile phones. In *INFOCOM* (2012), 1916–1924.

34. Sutherland, D. J., Póczos, B., and Schneider, J. Active learning and search on low-rank matrices. In *KDD* (2013), 212–220.

35. Xiong, H., Zhang, D., Wang, L., and Chaouchi, H. Emc3: Energy-efficient data transfer in mobile crowdsensing under full coverage constraint. *IEEE Transactions on Mobile Computing 14*, 7 (2015), 1355–1368.

36. Xu, L., Hao, X., Lane, N. D., Liu, X., and Moscibroda, T. Cost-aware compressive sensing for networked sensing systems. In *IPSN* (2015), 130–141.

37. Zhang, D., Wang, L., Xiong, H., and Guo, B. 4w1h in mobile crowd sensing. *IEEE Communications Magazine 52*, 8 (2014), 42–48.

38. Zhang, D., Xiong, H., Wang, L., and Chen, G. Crowdrecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint. In *UbiComp* (2014), 703–714.

39. Zhang, Y., Roughan, M., Willinger, W., and Qiu, L. Spatio-temporal compressive sensing and internet traffic matrices. In *SIGCOMM* (2009), 267–278.

40. Zheng, Y., Liu, F., and Hsieh, H.-P. U-air: when urban air quality inference meets big data. In *KDD* (2013), 1436–1444.

41. Zheng, Y., Liu, T., Wang, Y., Zhu, Y., Liu, Y., and Chang, E. Diagnosing new york city's noises with ubiquitous data. In *UbiComp* (2014), 715–725.

42. Zhu, Y., Li, Z., Zhu, H., Li, M., and Zhang, Q. A compressive sensing approach to urban traffic estimation with probe vehicles. *IEEE Transactions on Mobile Computing 12*, 11 (2013), 2289–2302.