Expertise-Aware Truth Analysis and Task Allocation in Mobile Crowdsourcing

Xiaomei Zhang*, Yibo Wu†, Lifu Huang‡, Heng Ji‡ and Guohong Cao†
*Department of Mathematics and Computational Science, University of South Carolina Beaufort
†Department of Computer Science and Engineering, The Pennsylvania State University
‡Computer Science Department, Rensselaer Polytechnic Institute
*xiaomei@uscb.edu, †{yxw185,gcao}@cse.psu.edu, ‡{huangl7,jih}@rpi.edu

***Abstract*—Mobile crowdsourcing has received considerable attention as it enables people to collect and share large volume of data through their mobile devices. Since the accuracy of the collected data is usually hard to ensure, researchers have proposed techniques to identify truth from noisy data by inferring and utilizing the reliability of users, and allocate tasks to users with higher reliability. However, they neglect the fact that a user may only have expertise on some problems (in some domains), but not others. Neglecting this expertise diversity may cause two problems: low estimation accuracy in truth analysis and ineffective task allocation. To address these problems, we propose an *Expertise-aware Truth Analysis and Task Allocation (ETA^2)* approach, which can effectively infer user expertise and then allocate tasks and estimate truth based on the inferred expertise. ETA^2 relies on a novel semantic analysis method to identify the expertise domains of the tasks and user expertise, an expertise-aware truth analysis solution to estimate truth and learn user expertise, and an expertise-aware task allocation method to maximize the probability that tasks are allocated to users with the right expertise while ensuring the work load does not exceed the processing capability at each user. Experimental results based on two real-world datasets demonstrate that ETA^2 significantly outperforms existing solutions.**

I. Introduction

Today's mobile devices, such as smartphones, smart watches and tablets, possess excellent capabilities in sensing and communication. Thanks to the prevalence of these mobile devices, recent years have seen the emergence of many mobile crowdsourcing applications [1][2][3]. With mobile crowdsourcing, people are able to collect and share observations and measurements about themselves and their surroundings. For example, commercial mobile apps such as *Gigwalk* and *Field Agent* recruit mobile users to collect data for various tasks, such as reporting the latest price of some specific product in supermarkets. In some other apps, such as map based service, through crowdsourcing, real-time traffic conditions are monitored and shared with drivers.

The quality of crowdsourcing depends on the accuracy of the collected data. However, in the real world, data accuracy is hard to ensure due to many reasons, such as the limited sensing capability, the unreliable data source, and some uncontrollable subjective factors. For example, accurate noise level may not be obtained if the reporting user does not have the right sensor installed in the mobile device. A user may intentionally generate data instead of performing the task at the specified location to save time, effort, or resources such as battery. To address this problem, it is crucial to estimate the truth (i.e., the accurate data) based on the collected data by applying appropriate truth analysis techniques. In addition to truth analysis, it is equally important to allocate tasks to appropriate users so that high-quality data can be collected.

Existing techniques on truth analysis and task allocation focus on studying the reliability of users, based on the assumption that high-reliability users tend to provide high-quality data. By inferring and utilizing user reliability, the truth can be better identified by assigning higher weights to users with higher reliability [4][5][6]. Similarly, by considering user reliability, tasks can be allocated to users with higher reliability, in order to collect high-quality data [7][8][9].

However, these existing techniques are based on the assumption that the reliability of a user does not change with tasks, while neglecting the fact that a user may only have expertise on some problems (in some domains), but not others. Neglecting this expertise diversity may cause two problems: low estimation accuracy in truth analysis and ineffective task allocation. This is because a user inferred to have high reliability may actually have low expertise in some domains, and provide low-quality data for tasks in these domains. A more severe problem is related to unfair task allocation, i.e., all tasks are assigned to a few users with "high-reliability" while the remaining majority of users are not assigned with any tasks.

Considering these problems, it is important to design expertise-aware solutions for truth analysis and task allocation. However, it is a challenge to design expertise-aware solutions due to the following two reasons. First, it is hard to identify user expertises and the expertise domains of the tasks without any prior knowledge on user behaviors and the ground truth of the tasks. Second, allocating tasks to users with the highest expertise is hard to achieve in many cases. This is because some users may have high expertise in multiple domains, but only have limited processing capability, and therefore can not finish all the assigned tasks within the time limit.

In this paper, we propose an *Expertise-aware Truth Analysis and Task Allocation (ETA^2)* approach to address the aforementioned challenges. ETA^2 can effectively infer user

This work was supported in part by the National Science Foundation (NSF) under grant CNS-1421578, and by Network Science CTA under grant W911NF-09-2-0053.

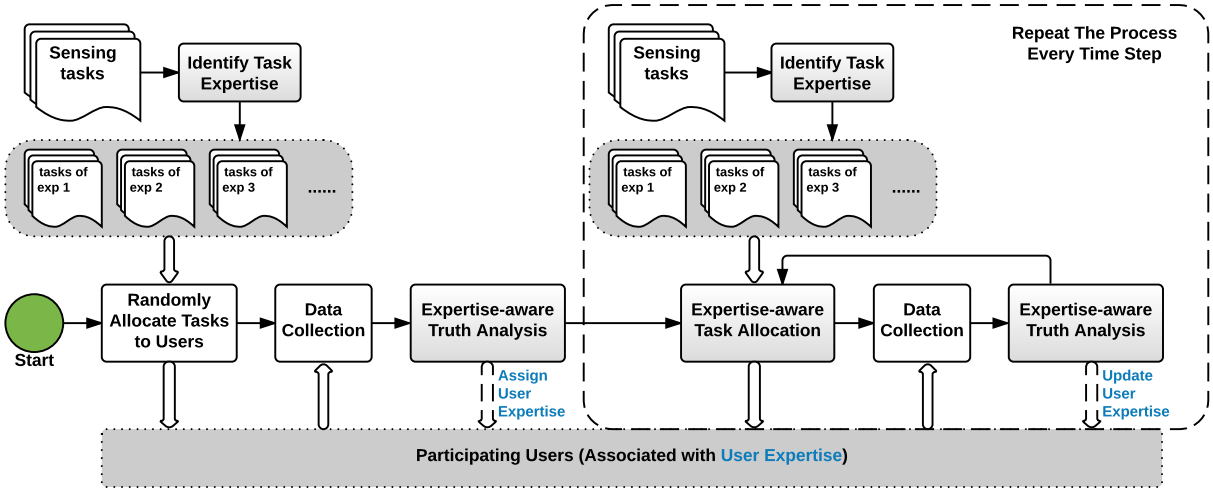


Fig. 1: An overview of the ETA^2 approach

expertise and then allocate tasks and estimate truth based on the inferred expertise. More specifically, we have the following contributions:

- We first propose techniques to identify the expertise domains of the tasks. Specifically, we design a novel semantic analysis method to extract and quantify the semantic information of tasks based on the task descriptions. Then, we propose a *dynamic hierarchical clustering* approach to cluster tasks based on their extracted semantic information, so that each cluster corresponds to one expertise domain and the tasks inside the cluster belong to the corresponding expertise domain.
- For *expertise-aware truth analysis*, we build expertise-aware statistical models by combining the expertise models of users and tasks, and apply *maximum likelihood estimation (MLE)* to estimate the truth and learn user expertise.
- For *expertise-aware task allocation*, we formalize an optimization problem which maximizes the probability that tasks are allocated to users with the right expertise, while ensuring that the workload at each user does not exceed its maximum processing capability.

The rest of the paper is organized as follows. Section II introduces some background and gives an overview of ETA^2. Section III presents how to identify the expertise domains of the tasks. Section IV and Section V present expertise-aware truth analysis and expertise-aware task allocation, respectively. Evaluation results are discussed in Section VI. Section VII reviews the related work and Section VIII concludes the paper.

II. Preliminary

A. Background

The system we consider includes a crowdsourcing server and multiple users who are able to communicate with the server using their mobile devices. A group of tasks are created at the server at each time step (e.g., each day). After a task is created, the server allocates it to a set of selected users specifying the required data, the task deadline and the estimated processing time required for completing the task. Multiple users may be queried for each task considering that the data collected from a single user may be inaccurate. Then, the selected users collect data as specified by the task descriptions and send data back to the server. The collected data item is accepted by the server only if it is sent back within the time limit. After receiving all provided data for a task, the server estimates the truth using a truth-estimation technique.

Without loss of generality, we assume the processing capability is limited at each user, i.e., only limited time can be used for collecting data during each time step. Therefore, only limited number of tasks can be performed at each user.

In this paper, we consider the collected sensing data to be numerical values. Then, the magnitude of the data may vary tremendously for different tasks. To ensure different tasks can be processed using the same statistical model, the data value for a task is normalized.

B. An Overview of ETA^2

Figure 1 shows an overview of ETA^2, which has the following three main modules:

- *Identifying Task Expertise*: This module is used to find the expertise domains of the tasks.
- *Expertise-aware Truth Analysis*: This module is used to infer the truth after data for the tasks have been collected from users. In the process of truth analysis, the expertise of users can also be identified.
- *Expertise-aware Task Allocation*: This module is used to assign the newly created tasks to users according to their expertise.

The process as shown in Figure 1 starts with a warm-up period, after the `Start` button. When the system first starts, there is no prior knowledge about user expertise. Thus, in the warm-up period, the tasks are allocated to users randomly.

After data have been collected from users, user expertise can be learned. The learned user expertise can be further utilized for task allocation.

After the warm-up period, ETA^2 starts an iterative process, as shown inside the dotted square in Figure 1. When new tasks are created in each iteration (time step), the server first finds the expertise of each task. Then, the server applies the expertise-aware task allocation technique to assign tasks to users with the right expertise. Then, users collect data as specified by the task description and send the data back to the server. After receiving the data, the server estimates the truth by applying expertise-aware truth analysis techniques and updates user expertise by incorporating the newly collected data.

C. Distribution of Random Observation

In this paper, we assume the random observations of users for a task follow normal distribution. To validate this assumption, we conduct an experiment to find the distribution of the observation error based on two real-world datasets. More information about the two datasets can be found in Section VI. The observation error is simply computed as the difference between the observation and the ground truth divided by the standard deviation among all observations. Figure 2 shows the result. The figure also shows the probability density function of the standard normal distribution. As can be seen, the error of the observed data follows the standard normal distribution very well. This result also implies that the random observations of users can be approximated by normal distribution, with the mean to be the ground truth of the tasks.

D. Expertise Model

In our expertise model, there are $\mathcal{D}$ expertise domains, where $\mathcal{D}$ is not fixed and may be increased when new tasks are added to the system. Each sensing task j belongs to one expertise domain, denoted as d_j. The expertise profile of a mobile user i is represented by a $\mathcal{D} \times 1$ vector:

$$U^i = [u_1^i, u_2^i, ..., u_{\mathcal{D}}^i]^T \quad (1)$$

where u_k^i is user i's expertise in domain k ($u_k^i \geq 0$). A higher u_k^i means more expertise. $u_k^i = 0$ means i has no expertise in domain k. A user may have expertise in multiple domains.

The expertise of user i for task j, represented as $u_{d_j}^i$, determines the quality of the data provided by the user. Having higher expertise means that the user is likely to provide higher-quality data for the task. A user with lower expertise is likely to provide data deviating from the ground truth. Specifically, we assume that the observation of user i for task j follows normal distribution $N(\mu_j, (\sigma_j/u_{d_j}^i)^2)$, where μ_j is the ground truth of task j, and $\sigma_j/u_{d_j}^i$ is the standard deviation. Here, σ_j is the *base number* of task j, which is used to normalize the data value of task j. σ_j is unknown but can be learned as presented in Section IV.

III. Task Expertise Identification

A. Basic Idea

To design expertise-aware truth analysis, we need to first identify the expertise domains of the tasks. Since there is no existing expertise information, we cannot simply classify tasks to existing expertise domains. The only information available for a task is the task description. There are many existing techniques [10][11] to identify the expertise domains or topics of documents by statistically analyzing the appearance of words in the documents. However, these techniques cannot be directly applied to the task descriptions, because they require the documents to be long enough for effective statistical results, but the task descriptions are usually short in crowdsourcing.

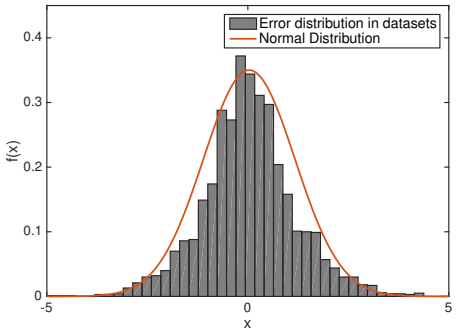


Fig. 2: The observation error follows normal distribution.

To address this problem, we design a semantic analysis method, called *pair-word*, to extract and quantify the semantic information of the tasks based on the task descriptions. With the extracted semantic information, we are able to measure the distance (or similarity) between tasks. Then, we can cluster these tasks based on their distance so that each cluster represents one expertise domain and the tasks inside this cluster belong to the corresponding expertise domain. Specifically, a *dynamic hierarchical clustering* approach is proposed to cluster tasks and identify expertise. As new tasks arrive, the dynamic hierarchical clustering approach can dynamically identify their expertise domains by creating new clusters or merging to existing clusters. In the rest of the section, we first discuss how to extract the semantic information with our pair-word method, and then present the dynamic hierarchical clustering approach.

B. Semantic Information Extraction

To effectively extract the semantic information from the task description, we design a pair-word based method to identify two types of important terms within each description sentence: $Query$ term, which refers to the words or phrases that describe the requirement of a specific task, and $Target$ term, which contains the desired information. For example, the following shows two tasks and their identified $Query$ and $Target$ terms ($Query$ and $Target$ terms are manually identified).

- *Task 1*: What is the noise level around the municipal building?
  Query: noise level; *Target*: municipal building
- *Task 2*: How many students have attended the seminar today?
  Query: students; *Target*: seminar

We utilize distributed semantics of $< Query, Target >$ to capture the meaning of each task description. Word embedding is an efficient technique to map each word or phrase to a low-dimensional vector based on their global contexts. We use the Continuous Skip-gram model [12] to learn lexical representation for each single word from the entire Wikipedia dump (August 11, 2014). For multi-word terms, a simple element-wise additive model ($V = x_1 + x_2 + ... + x_i$) [12] is exploited, where V represents a phrase embedding and $x_1, x_2, ..., x_i$ represent the individual embeddings of the words in V. We concatenate the vector representation of $Query$ term V_Q and $Target$ term V_T for each task description and use Euclidean distance metric to measure the distance between two tasks i and j based on their semantic vectors:

$$E(i,j) = \frac{1}{2}||[V_Q^i, V_T^i] - [V_Q^j, V_T^j]||^2 \quad (2)$$

where $[V_Q, V_T]$ denotes the concatenation of two vectors V_Q and V_T for each task. With this pair-word extraction method, we can efficiently capture the semantic information that two tasks shared.

C. Dynamic Hierarchical Clustering

1) Hierarchical Clustering: Based on the distance metric between tasks, tasks are clustered together. Although there are many clustering techniques in the literature [13], we select hierarchical clustering based on the following two reasons. First, with hierarchical clustering, the number of clusters is not fixed. As a result, clusters can be updated and new clusters can be added when new tasks are added. Second, hierarchical clustering is effective and simple with only one parameter γ, which is used to quantify the *minimum allowed distance* between clusters. Here, the cluster distance is calculated as the average distance between tasks in the two clusters. After the hierarchical clustering process, the distance between any two clusters should be equal or larger than the *minimum allowed distance*. Assume the longest distance between all existing tasks is d^*. The *minimum allowed distance* can be represented as $\gamma \cdot d^*$, where d^* is a fixed value and $\gamma \in [0, 1]$ is the parameter, which can be flexibly set according to specific requirements.

With a set of m tasks from the warm-up period, the basic hierarchical clustering works as follows:

1) *Initialization:* Each of the m tasks starts its own cluster.
2) *Merging clusters:* Pick two clusters that are closest and merge them. This step repeats until the termination criterion is satisfied, as defined in the next step.
3) *Termination:* The algorithm terminates if the distance between the closest clusters in one round is equal to or larger than the *minimum allowed distance* between clusters, $\gamma \cdot d^*$.

2) Dynamic Hierarchical Clustering: The above algorithm can be directly applied to identify the expertise domains of the tasks in the warm-up period. As new tasks are created, they should be classified to some existing clusters, and the dynamic hierarchical clustering method is proposed to achieve this goal.

Dynamic hierarchical clustering only differs from hierarchical clustering in the initialization step. Assume m' new tasks are created. For each new task, a new cluster is created, and then m' new clusters are created. If there are $\mathcal{M}$ existing clusters before new tasks are created, there will be $\mathcal{M} + m'$ clusters in the initialization step. Then, the $\mathcal{M} + m'$ clusters are merged following the same "*Merging clusters*" process, and it terminates when the termination criterion is satisfied.

IV. Expertise-Aware Truth Analysis

In this section, we present our expertise-aware truth analysis. Specifically, a statistical model is built based on the expertise models, which treats the truth associated with each task and user expertise as parameters. By applying the technique of *Maximum Likelihood Estimation (MLE)*, both the truth and user expertise in the statistical model can be estimated. We first present the statistical models and the MLE method used to infer the truth and user expertise. Then, we discuss how to dynamically update user expertise when new observations are made from tasks in subsequent time steps.

A. Estimation of Truth and User Expertise

Based on the collected data for the tasks in the warm-up period, we can find the user expertise and the truth associated with each task using MLE. With MLE, the unknown parameters of a statistical model can be estimated given the observed data. In our statistical model, the observed data is the data provided by the users for the sensing tasks. The set of observed data is denoted as

$$X = \{X_1, X_2, ..., X_m\},$$

where m is the number of tasks, and X_j is the set of data provided for task j. The unknown parameters of our statistical model include the expertise of each user i in all of the $\mathcal{D}$ domains, i.e., $u_1^i, u_2^i, ..., u_{\mathcal{D}}^i$, the ground truth for each task j, i.e., μ_j, and the base number for each task j, i.e., σ_j. Overall, the set of unknown parameters is represented as

$$\Theta = \bigcup_{i=1}^{n} \bigcup_{j=1}^{m} \{u_1^i, u_2^i, ..., u_{\mathcal{D}}^i, \mu_j, \sigma_j\}$$

We use $\omega_{ij} \in \{0, 1\}$ to denote whether user i has provided data for task j. If $\omega_{ij} = 1$, user i has provided data for task j, and the data is denoted as x_{ij}; otherwise, $\omega_{ij} = 0$.

If $\omega_{ij} = 1$, the *probability density function (pdf)* of x_{ij} is

$$f(x_{ij}|\Theta) = \frac{1}{\sigma_j / u_{d_j}^i \sqrt{2\pi}} e^{-\frac{(x_{ij}-\mu_j)^2}{2\sigma_j^2 / {u_{d_j}^i}^2}}. \quad (3)$$

From the above equation, we can compute the *pdf* (or likelihood function) that X is observed as

$$\begin{aligned} f(X|\Theta) &= \prod_{j=1}^{m} f(X_j|\Theta) = \prod_{j=1}^{m} \prod_{i=1}^{n} (f(x_{ij}|\Theta))^{\omega_{ij}} \\ &= \prod_{j=1}^{m} \prod_{i=1}^{n} \left(\frac{1}{\sigma_j / u_{d_j}^i \sqrt{2\pi}} e^{-\frac{(x_{ij}-\mu_j)^2}{2\sigma_j^2 / {u_{d_j}^i}^2}}\right)^{\omega_{ij}}. \quad (4) \end{aligned}$$

With ω_{ij} being the exponent of $f(x_{ij}|\Theta)$, $f(X|\Theta)$ only multiplies $f(x_{ij}|\Theta)$ with $\omega_{ij} = 1$.

Given the likelihood function, MLE estimates the parameters by computing the parameter set $\hat{\Theta}$ that maximize the likelihood function. Maximizing the likelihood function is equal to maximizing the log-likelihood function, which is

$$\begin{aligned}\log L(\Theta; X) = &\quad \log f(X|\Theta) \\ = &\quad \sum_{i=1}^{m}\sum_{i=1}^{n} \omega_{ij}[\log(\frac{1}{\sigma_j/u_{d_j}^i\sqrt{2\pi}}) - \frac{(x_{ij}-\mu_j)^2}{2\sigma_j^2/{u_{d_j}^i}^2}]\end{aligned}$$

Then, $\hat{\Theta}$ is computed by setting the derivatives of the log-likelihood function $\log L(\Theta; X)$ over each parameter to be 0. With some derivation, we get

$$\mu_j = \frac{\sum_{i=1}^{n}\omega_{ij}{u_{d_j}^i}^2 x_{ij}}{\sum_{i=1}^{n}\omega_{ij}{u_{d_j}^i}^2}, \sigma_j = (\frac{\sum_{i=1}^{n}\omega_{ij}(x_{ij}-\mu_j)^2{u_{d_j}^i}^2}{\sum_{i=1}^{n}\omega_{ij}})^{\frac{1}{2}}, \tag{5}$$

$$u_k^i = (\frac{\sum_{j=1}^{m} I(d_j=k)\omega_{ij}}{\sum_{j=1}^{m} I(d_j=k)\omega_{ij}(x_{ij}-\mu_j)^2/\sigma_j^2})^{\frac{1}{2}}, \tag{6}$$

where $i \in \{1, 2, ..., n\}$, $j \in \{1, 2, ..., m\}$ and $k \in \{1, 2, ..., \mathcal{D}\}$. In Equation 6, $I(d_j = k) = 1$ if $d_j = k$ is true, and $I(d_j = k) = 0$ otherwise. Though it is hard to get a close-form solution for each of the parameters, we can get the estimation of the parameters by iteratively computing the values based on Equations 5 and 6. To start the iterative process, we first set the initial values for the user expertise to be 1 ($u_k^i = 1, \forall i, k$), and then use the three equations to iteratively compute the values of μ_j and σ_j and u_k^i until the calculated values converge.

B. Dynamic Update of User Expertise

With the aforementioned method, user expertise is estimated based on the observations of the existing tasks. After new tasks are created, user expertise should be updated based on new observations. Specifically, the dynamic hierarchical clustering is first used to identify the expertise domains of new tasks. After adding these new tasks, there may three types of changes to the existing expertise domains (as shown in Figure 3), resulting to three types of updates to user expertise, which are discussed as follows.

1) Adding new tasks to existing expertise domain: When new tasks are added to an existing expertise domain (as shown in Figure 3 (a)), user expertise in this domain should be updated by incorporating users' observed data for these new tasks. User expertise is updated based on Equation 6, which is used to compute the user expertise u_k^i in the warm-up period. For the part inside the parentheses, which is $(u_k^i)^2$, the numerator and the denominator have different meanings. The numerator $\sum_{j=1}^{m} I(d_j = k)\omega_{ij}$, denoted as $N(u_k^i)$, represents the number of tasks from expertise domain k that have been finished by user i. The denominator $\sum_{j=1}^{m} I(d_j = k)\omega_{ij}(x_{ij} - \mu_j)^2/\sigma_j^2$, denoted as $D(u_k^i)$, represents the sum of squared estimation errors for those tasks.

To update user expertise u_k^i, we maintain the numerator $N(u_k^i)$ and the denominator $D(u_k^i)$ of the entity $(u_k^i)^2$. Suppose the current time step is T, and the t is the length of one time step. After new tasks from expertise domain k are finished by user i during the new time step, $N(u_k^i)$ and $D(u_k^i)$ are updated as follows:

$$N(u_k^i)^{T+t} = \alpha \cdot N(u_k^i)^T + \sum_{j=1}^{m'} I(d_j = k)\omega_{ij}, \tag{7}$$

$$D(u_k^i)^{T+t} = \alpha \cdot D(u_k^i)^T + \sum_{j=1}^{m'} I(d_j = k)\omega_{ij}(x_{ij} - \mu_j)^2/\sigma_j^2, \tag{8}$$

where $\alpha \in [0, 1]$ is the decaying factor placed on the original value to undermine the influence of the historical tasks, and m' is the number of tasks created in the current time step. Then the user expertise is updated based on $N(u_k^i)^{T+t}$ and $D(u_k^i)^{T+t}$:

$${u_k^i}^{T+t} = (\frac{N(u_k^i)^{T+t}}{D(u_k^i)^{T+t}})^{\frac{1}{2}} \tag{9}$$

When computing $D(u_k^i)^{T+t}$ with Equation 8, the true value μ_j and base number σ_j for new task j are unknown *a priori*. To address this problem, μ_j and σ_j are first estimated using Equations 5, in which the user expertise is initialized to the original values in time T. Then, we can update user expertise based on Equation 9. Since the values of μ_j and σ_j computed from Equations 5 may be changed after the user expertise is updated, we apply the same iterative process to update μ_j, σ_j and $({u_k^i}^{T+t})$ until they converge. At the end of the iterative process, the user expertise is updated, and the truth values $\mu_j, j \in \{1, ..., m'\}$ for the new tasks are identified.

2) Creating a new expertise domain: If there exist some new tasks that are close to each other but far from any existing expertise domains, a new expertise domain $\mathcal{D} + 1$ is created for them (as shown in Figure 3 (b)). The user expertise in the new domain, the truth and the base number of the new tasks in this domain are estimated using Equations 5–6, just as those in the warm-up period. The number of expertise domain $\mathcal{D}$ is added by 1.

3) Merging existing expertise domains: If two existing expertise domains k_1 and k_2 are merged by including new tasks that are close to both k_1 and k_2 (as shown in Figure 3 (c)), the user expertise in the two domains should also be updated. The user expertise in domain k_1 is updated by incorporating tasks in k_2, and k_2 is deleted. Specifically, $N(u_{k_1}^i)$ and $D(u_{k_1}^i)$ in domain k_1 are updated by adding those in domain k_2:

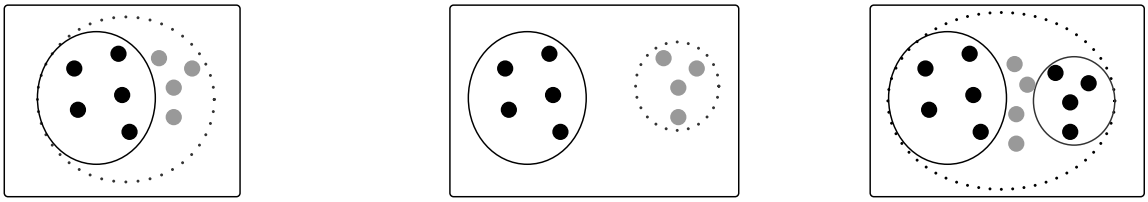
$$N(u_{k_1}^i) = N(u_{k_1}^i) + N(u_{k_2}^i), \tag{10}$$

$$D(u_{k_1}^i) = D(u_{k_1}^i) + D(u_{k_2}^i). \tag{11}$$

After merging these two expertise domains, $N(u_{k_1}^i)$ and $D(u_{k_1}^i)$ are updated by incorporating tasks in the new time step. The updating process is identical to adding new tasks to existing expertise domain as shown in Section IV-B1.

V. Expertise-Aware Task Allocation

As a task is created, it should be allocated to users with higher expertise for that task. However, since each user has limited processing capability, i.e., only limited time is available for completing tasks each day, it is possible that a user



(a) Adding new tasks to existing expertise domain (b) Creating a new expertise domain (c) Merging existing expertise domains

Fig. 3: The three types of changes for expertise domains (clusters) after new tasks (grey) are added. Solid circles represent existing expertise domains. Dotted circles represent new or updated expertise domains.

cannot finish all the assigned tasks. To address this problem, we formalize an optimization problem which maximizes the possibility that tasks are allocated to users with the right expertise, while ensuring the work load does not exceed the processing capability at each user. By proving the optimization problem to be NP-hard, we further propose a heuristic based algorithm as the approximation solution.

A. The Optimization Problem

1) Objective function: We first compute the probability that at least one user can provide accurate data for task j:

$$p_j = 1 - \prod_{i=1}^{n}(1 - p_{ij}) \tag{12}$$

where p_{ij} is the probability that user i can provide accurate data for task j. We consider the observed data to be accurate if its normalized error is smaller than ϵ, where the *normalized error* is computed as the error to the ground truth divided by the base number. ϵ is a small constant and set to 0.1 in the paper. Then, the probability that user i can provide accurate data for task j can be computed as follows:

$$p_{ij} = P(\frac{|x_{ij} - \mu_j|}{\sigma_j} < \epsilon) = \Phi(\epsilon u^i_{d_j}) - \Phi(-\epsilon u^i_{d_j}) \tag{13}$$

The objective function is computed as the sum of the probability that at least one user can provide accurate data for each task, which is as follows:

$$\begin{aligned}\sum_{j=1}^{m} p_j &= \sum_{j=1}^{m}[1 - \prod_{i=1}^{n}(1 - p_{ij})] \\ &= \sum_{j=1}^{m}[1 - \prod_{i=1}^{n}(1 - \Phi(\epsilon u^i_{d_j}) + \Phi(-\epsilon u^i_{d_j}))]\end{aligned} \tag{14}$$

By maximizing the objective function, the users with high expertise are selected with high priority, since users with high expertise are more likely to provide accurate information.

2) Constraints: The processing capability of each user i is denoted as c_i, which is the available time for user i to spend on processing tasks. The processing time for each task j is denoted as t_j. $s_{ij} \in \{0, 1\}$ is used to denote whether task j will be allocated to user i. Then, the constraints on the processing capability at each user can be represented as:

$$\sum_{j=1}^{m} t_j \cdot s_{ij} < c_i, \quad \forall i \in \{1, 2, ..., n\} \tag{15}$$

Thus, the optimization problem can be formalized as follows:

$$\begin{aligned} \max \quad & \sum_{j=1}^{m}[1 - \prod_{i=1}^{n}(1 - \Phi(\epsilon u^i_{d_j}) + \Phi(-\epsilon u^i_{d_j}))^{s_{ij}}] \\ s.t. \quad & \sum_{j=1}^{m} t_j \cdot s_{ij} < c_i, \quad \forall i \in \{1, ..., n\} \\ & s_{ij} \in \{0, 1\}, \quad \forall i \in \{1, ..., n\}, j \in \{1, ..., m\} \end{aligned} \tag{16}$$

The optimization problem is proved to be NP-hard. Considering the case where there is only one user, the problem can be easily reduced to a knapsack problem, where the knapsack is identical to the user with limited processing capability, and the items are identical to the tasks to be assigned. Since the knapsack problem has non-integer weight and item values, the problem is NP-hard [14]. Since the optimization problem with single user is NP-hard, the optimization problem with n users is also NP-hard.

B. Heuristic Based Algorithm

Since the optimization problem is NP-hard, we propose a heuristic based algorithm. The basic idea is to greedily select user-task pairs which can add more value to the objective function while ensuring the tasks consume less processing time. For each selected user-task pair, the task will be assigned to that user.

In the heuristic based algorithm, we first define a concept *efficiency* for each user-task pair (denoted as *efficiency*(i, j) for user i and task j) to quantify its importance in increasing the value of the objective function. Let c'_i denote the remaining processing capability of user i. The *efficiency* concept is formally defined as follows:

Definition* 1.** *The* ***efficiency *of user-task pair (i, j) is calculated as the value increase of the objective function divided by the processing time t_j of task j if task j is assigned to user i. However, if the remaining processing capability c'_i of user i is not enough to finish task j, i.e., $t_j > c'_i$, the efficiency is set to zero.*

Specifically, the value increase of the objective function is computed as

$$p^{new}_j - p_j = [1 - (1 - p_j)(1 - p_{ij})] - p_j = p_{ij}(1 - p_j), \tag{17}$$

where p^{new}_j is the updated p_j after user i is added. Note that adding a user for task j does not change other tasks, so we

Algorithm 1 Expertise-aware Task Allocation

Input: $u_k^i, c_i, \forall i,k$, and $d_j, t_j, \forall j$
Output: $s_{ij}, \forall i,j$
1: **Initialize:** $s_{ij} \leftarrow 0$
2: Compute *efficiency*(i,j) according to Equation 18
3: **for** Each task $j \in \{1,2,...,m\}$ **do**
4: Find the max efficiency that can be achieved for task j: *maxeff*$_j$
5: Simultaneously find the user that achieves *maxeff*$_j$: *user*$_j$
6: **end for**
7: **while** $true$ **do**
8: Find max efficiency from *maxeff*$_j$, $\forall$ task $j \in \{1,..,m\}$
9: Simultaneously find the task that achieves the max efficiency: j^*
10: **if** max efficiency $= 0$ **then**
11: **break**
12: **end if**
13: Select the user-task pair: $s(user_{j^*}, j^*) \leftarrow 1$
14: Update efficiency for the user-task pairs associated with *user*$_{j^*}$
15: Update efficiency for the user-task pairs associated with task j^*
16: Simultaneously update *maxeff*$_{j^*}$ and *user*$_{j^*}$ for task j^*
17: **end while**

do not need to consider other tasks in the objective function. As a result, the efficiency of (i,j) is computed as:

$$\textit{efficiency}(i,j) = \begin{cases} \frac{p_j^{new} - p_j}{t_j} = \frac{p_{ij}(1-p_j)}{t_j}, & \text{if } c_i' \geq t_j \\ 0, & \text{if } c_i' < t_j \end{cases} \quad (18)$$

The heuristic based algorithm works by greedily adding the user-task pair (i,j) which achieves the maximum efficiency. The greedy process terminates when the maximum efficiency that can be achieved becomes *zero*, which means the users have used up all their processing capabilities. The general flow of the heuristic algorithm is outlined in Algorithm 1. First, the algorithm computes *efficiency*(i,j) for all user-task pair (Line 2). At the same time, the algorithm maintains the max efficiency *maxeff*$_j$ that can be achieved for each task j, and the user that achieves *maxeff*$_j$, denoted as *user*$_j$ (Lines $3 \sim 6$), so that when the greedy algorithm selects the maximum efficiency in each round, it can simply select from the maintained *maxeff*$_j$ for the m tasks (Line 8). Then, the algorithm greedily selects the user-task pair that achieves the maximum efficiency and updates the efficiency for other unselected pairs (Lines $8 \sim 16$). The algorithm terminates when the max efficiency is equal to *zero* (Lines $10 \sim 12$).

To select each user-task pair, $O(m)$ time is used to find the pair with the highest efficiency. After selecting the user-task pair, $O(m+n)$ time is used to update the efficiency of the remaining unselected user-task pairs, because only the pairs associated with the selected user or the selected task need to be updated. As a result, the total time used for selecting each user-task pair is $O(m)+O(m+n) = O(m+n)$. Assuming $\mathcal{K}$ user-task pairs are selected, the time complexity of the whole process is $O(\mathcal{K}(m+n))$.

VI. Performance Evaluations

In this section, we evaluate the performance of ETA^2 by conducting experiments based on two real-world datasets.

A. Datasets

1) Survey-based Dataset: The first dataset is collected through a survey including 60 participants on our campus, after IRB approval. In the survey, each of the participants is required to answer 89 questions about their daily life and basic knowledge on various topics. Some sample questions are listed as follows:

- *How many parking lots on campus are open to students in this semester?*
- *What is the estimated driving hours to another city in the local state?*
- *What is the average salary for an entry-level software engineers in United States?*

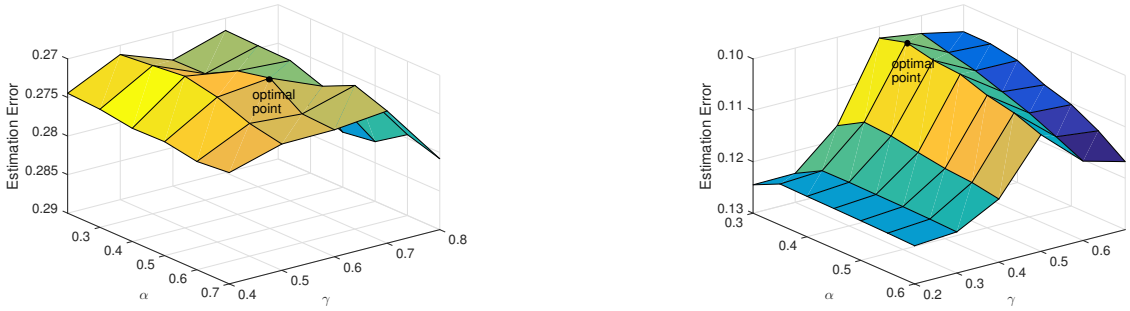
The answers to some questions may depend on specific time and location. For example, the available parking lots to students may be different during weekdays and weekends, and the driving hours may be different during early morning and late afternoon in each day. Therefore, some questions are replicated to consider the conditions at different time and locations. As a result, there are finally 150 questions in the dataset.

The provided answers are noisy, and then can be used for evaluating our approaches. Assume these questions are the sensing tasks provided by the server, and the content of the question is the task description. If a user is queried with a sensing task, the collected data value is the answer to the corresponding question. For the purpose of performance evaluation, the ground truth of every question is carefully inspected and added by the researchers.

2) SFV Dataset: The second dataset [15] is extracted from the Slot-Filling Validation (SFV) task of the 2013 Text Analysis Conference (TAC) Knowledge Base Population (KBP) track. In the task, 18 slot-filling systems are required to answer a set of questions about 100 entities, including famous persons or organizations. The questions are about various properties of the entities, like the age, birthday and name. There are about 2,000 questions for these 100 entities in the original dataset. Similar to the survey-based dataset, each question is treated as a sensing task. The 18 slot-filling systems are treated as users, and theirs answers to the questions are treated as the collected sensing data. In the dataset, documents that describe the entities and their properties are also given, from which we can easily compose the descriptions of tasks. The ground truth of the tasks is also included in the original dataset, which can be used for performance evaluations.

B. Experimental Setting

In these two datasets, some important information like the processing time required for each task and the processing capability of each user is not provided. Without loss of generality, these quantities are generated using uniform distribution. For the survey-based dataset, the processing time required for each task is randomly generated between $[2,4]$ hours. For the SFV dataset, it is randomly generated between $[1,2]$ hours. The processing capability of each user is randomly generated between $[\tau-4, \tau+4]$ hours, where τ is a variable representing the average processing capability. τ is set to 12, but can be changed to evaluate how the processing capability affects the performance. The length of the time step is *a day* in the experiment. The sensing tasks are assumed to be generated and evenly distributed during *five* days. In the beginning of



(a) Survey-based dataset

(b) SFV dataset

Fig. 4: Estimation error with different parameter settings

each day, a group of sensing tasks are generated and allocated to the selected users. At the end of the day, the truth for each task is estimated based on all the collected data values.

In order to have statistically converging results, we set different seeds to randomly select tasks in each day and take the average as the results. Specifically, every experiment is run 100 times by setting different seeds to achieve statistical convergence.

C. Approaches in Comparison

We compare our proposed expertise-aware solution with three most well-known approaches that estimate truth by calculating source reliability, and a baseline approach serving as the lower bound. These approaches in comparison are as follows:

- *Hubs and Authorities* [16]: The reliability of a source is the sum of the credibility of the data items it provides, and the credibility of a data item is the sum of the reliability of sources that provide the data.
- *Average·Log* [5]: The reliability of each source is calculated by multiplying the average credibility of its provided data item and the logarithm of the number of its provided data item.
- *TruthFinder* [4]: The credibility of an observed data item is the probability that it is accurate and the reliability of the source is the probability that it provides accurate data. Specifically, the credibility of a data item is computed as the probability that at least one source can provide accurate data. A source's reliability is calculated by averaging the credibility of its provided data.
- *Baseline*: The truth is estimated as the mean value of the observed data.

These approaches in comparison only specify the method for truth analysis. To allocate new tasks to users, the first three reliability-based approaches greedily allocate tasks to users with high reliability. Considering users only have limited processing capability, we prioritize the tasks with lower sensing time to be allocated to users with high-reliability, so that these high-reliability users can finish as many tasks as possible. For the baseline approach, the tasks are allocated to users randomly.

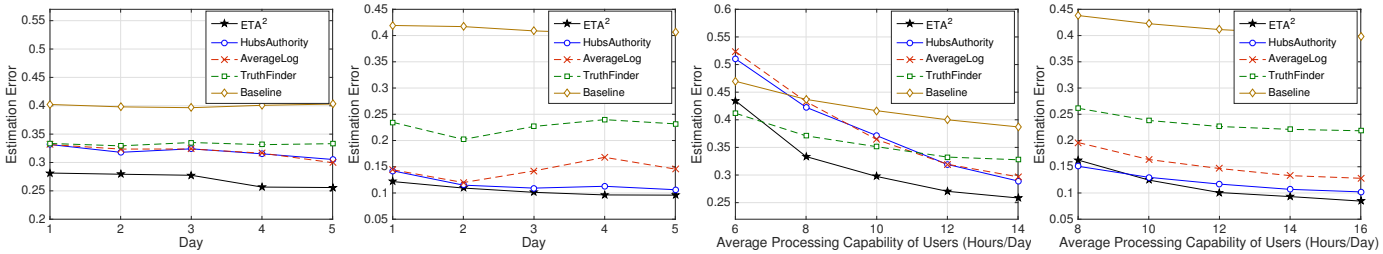
D. Evaluation Results

In the evaluations, we first show how the parameters γ and α affect the performance of ETA^2 approach. Then, we evaluate the overall performance of ETA^2 by comparing it with existing approaches. Finally, we evaluate the effectiveness of the three modules of ETA^2. In the evaluation, the performance metric is the *estimation error*, which is computed as the average of the normalized estimation error for all sensing tasks.

1) The Effects of Parameters: ETA^2 has two parameters α and γ. When updating user expertise to include the new tasks, a decaying factor $\alpha \in [0, 1]$ is placed on the historical tasks to reduce the influence of the historical tasks. Parameter $\gamma \in [0, 1]$ is used to determine the *minimum allowed distance* between clusters, which determines when the merging process terminates. It is crucial to set appropriate α and γ so that the performance of our approach can be optimized.

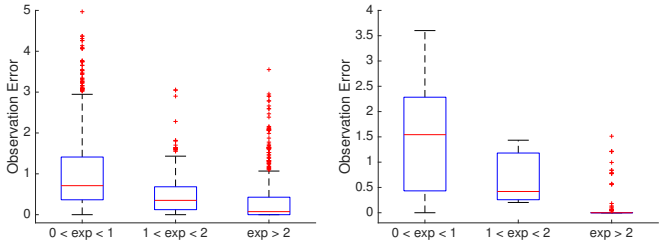
We first conduct experiments to evaluate how the two parameters affect the performance of ETA^2. The objective is to evaluate the estimation error under different parameter settings and find the set of parameters which result in best performance. The results are shown in Figure 4 (a) and Figure 4 (b) for the two datasets, respectively. In these two figures, the z axis is upside down for better visualization, so that the point with the smallest estimation error is shown in the upmost place. As we can see, the best performance can be achieved when $\alpha = 0.5$ and $\gamma = 0.6$ for the survey-based dataset, and $\alpha = 0.1$ and $\gamma = 0.5$ for the SFV dataset. Since the parameters for different datasets may be different, when ETA^2 is implemented, the parameters are first evaluated based on a pre-set warm-up period and then the chosen values are applied to the experiments. In the rest of the evaluation, α and γ will be set to these values that achieve the best performance according our evaluations.

2) Overall Performance of ETA^2: We further evaluate the performance of ETA^2 by comparing it with existing approaches. First, we keep track of the estimation error in different days. The results for the two datasets are shown in Figure 5 (a) and (b), respectively. As shown in the figures, the estimation error of ETA^2 drops overtime in both two datasets. This is because in the beginning, there is no initial knowledge on user expertise, and the tasks are randomly allocated to



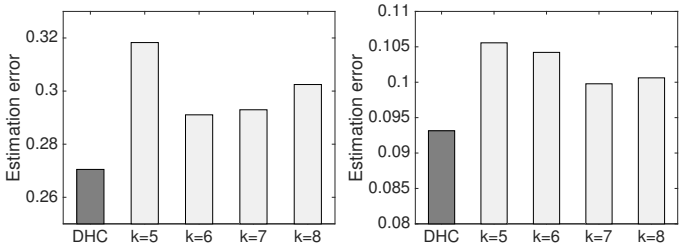
(a) Survey-based dataset (b) SFV dataset (c) Survey-based dataset (d) SFV dataset

Fig. 5: Comparison of estimation error (a)(b) in different days and (c)(d) with different processing capability.



(a) Survey-based dataset (b) SFV dataset

Fig. 6: The effects of user expertise on observation error



(a) Survey-based dataset (b) SFV dataset

Fig. 7: The comparison between dynamic hierarchical clustering (DHC) and k-means

users. After more tasks have been finished, user expertise is better estimated, which can be used in the following days to make better decisions on task allocation. Also, ETA^2 outperforms other approaches. Specifically, for the survey-based dataset, its estimation error is 15% to 20% smaller compared with the existing approaches. For the SFV-based dataset, the estimation error is at least 5% to 15% smaller compared with other approaches.

Second, we change the average processing capability τ. A large τ means that users have more time to complete the tasks and therefore more users can be selected for each task. As shown in Figures 5 (c) and (d), the estimation error decreases as the average processing capability increases. When the processing capability is small, ETA^2 underperforms *TruthFinder* in the survey dataset and *Hubs and Authority* in the SFV dataset. This is because there are very limited users assigned for each task, and then the user expertise cannot be accurately estimated. As the processing capability increases, ETA^2 significantly outperforms other approaches.

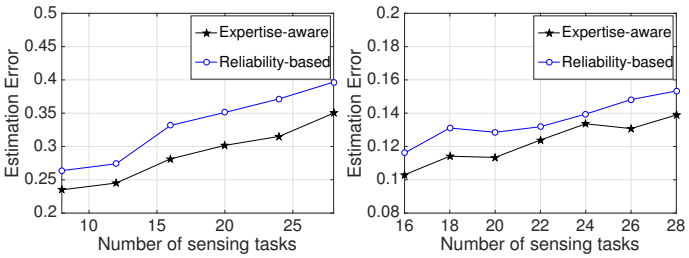
Verifying the importance of expertise: To further verify the importance of the expertise information, we conduct a small experiment to figure out how user expertise affects the data they observe. Specifically, the experiments measure the *observation error*, i.e., the error of the observed data, under different user expertise. The results are shown in Figure 6. As we can see, with the increase of user expertise, there is a clear decrease of the observation error. When the user expertise is larger than 2, most observation errors are close to zero (the red line inside the box indicates the median of observation errors). This result demonstrates that user expertise can be exploited to improve data quality in ETA^2.

3) Evaluation of Different Modules in ETA^2: Finally, we respectively evaluate the effectiveness of the three modules of ETA^2.

Task Expertise Identification: In ETA^2, the expertise domain of the task is identified by applying a dynamic hierarchical clustering (DHC) method over the task descriptions. We evaluate the effectiveness of DHC by comparing it with a well-known clustering method, *k-means*, where k stands for the number of expertise domains. We set k from 5 to 8, since it is less likely that the number of expertise domains falls out of this range for these two datasets. As shown in Figure 7 (a) and (b), the estimation error using DHC is much smaller than those using *k-means*. This is because our DHC method is more flexible and can dynamically update the expertise domains as new tasks arrive.

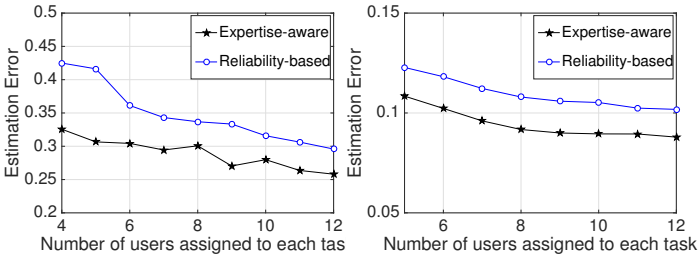
Expertise-aware Truth Analysis: In this experiment, tasks are randomly allocated to users. Then, the truth associated with each task is identified by applying the expertise-aware approach or the reliability-based approach. For the reliability-based approach in comparison, we test all three reliability-based truth analysis approaches listed in Section VI-C, but only show the result of the approach that achieves the smallest estimation error.

The results for the two datasets are shown in Figure 8 (a) and Figure 8 (b), respectively. As we can see, when there are different sensing tasks, the expertise-aware approach consistently outperforms the reliability-based solutions. This is because the expertise-aware truth analysis assigns higher weights to users with higher expertise, as shown in Equation 5. Since high-expertise users usually observe data more accurately, the estimated truth is more accurate. For reliability-



(a) Survey-based dataset (b) SFV dataset

Fig. 8: The comparison between expertise-aware truth analysis and reliability-based truth analysis



(a) Survey-based dataset (b) SFV dataset

Fig. 9: The comparison between expertise-aware task allocation and reliability-based task allocation

based truth analysis, even though it can assign higher weight to users that are inferred to have "high reliability", these users may not actually provide accurate data.

Expertise-aware Task Allocation: In addition to truth analysis, we also do experiments to evaluate if the expertise-aware task allocation can effectively allocate tasks to users, compared with the reliability-based solution. In the experiments, we fix the number of users to be assigned for each task, and apply the expertise-aware solution or the reliability-based solution to allocate tasks to users. As shown in Figure 9 (a) and Figure 9 (b), the expertise-aware solution outperforms the reliability-based solution. This is especially true when there is less number of users to be assigned to each task in the survey-based dataset. The possible reason is that with the expertise-aware task allocation, the tasks are mainly assigned to the users with high expertise, so that data with better quality are always collected.

VII. Related Work

In mobile crowdsourcing, task allocation determines how to select appropriate users to complete tasks. Researchers have designed solutions to address this problem by focusing on different perspectives. Specifically, the authors in [17][18] consider energy-efficient task allocation which recruits users to reduce the energy consumption while achieving the objective on spacial coverage. Wang *et al.* [2] and Wu *et al.* [19] consider image sensing with smartphones and propose user selection approach to maximize the photo utility and minimize the resource consumption. Cheung *et al.* [20] propose a technique that enable mobile users to select tasks in a distributed manner, with the objective to collect time-sensitive and location-dependent information. Researchers in [8][7][9] study how the quality of data source affects the provided information and provide solutions to identify and recruit high-quality data sources.

In addition to task allocation, other research problems in mobile crowdsourcing include privacy issues and the design of incentive mechanisms. Specifically, researchers in [21][22][23] consider privacy issues related to the crowdsourcing participants and design privacy-preserving schemes for data collection. Researchers in [1][24][25] design incentive mechanisms and propose techniques to recruit users while minimizing cost. In addition to minimize the cost, researchers in [26][27] propose incentive mechanisms to consider the quality of information and encourage users to provide higher-quality data by paying more to users. These mechanisms on privacy protection and incentive are orthogonal to the contributions of this paper, but can be easily built on top of our strategy.

Truth analysis in crowdsourcing has received considerable attention recently. Most truth analysis techniques estimate truth by inferring the reliability of users. One of the earliest techniques is Hubs and Authorities [16], which iteratively evaluates the correctness of information by evaluating the reliability of data sources. Pasternack *et al.* [5] extend these frameworks to a more general scenario by incorporating prior knowledge of the information. TruthFinder proposed in [4] is also based on an iterative method to infer the information correctness and user reliability. Researchers also propose truth analysis techniques by designing and investigating statistical models, such as bayesian inference [28] and expectation-maximization (EM) [6]. In addition to these techniques, Zhang *et al.* [29] propose a resource-aware truth analysis approach where data can be adaptively collected from networks to further improve data quality. Also, researchers in [30][31][32] further study other issues related to truth analysis, such as truth existence, truth evolution and data sparsity.

VIII. Conclusion

In this paper, we proposed an ETA^2 approach for expertise-aware truth analysis and task allocation in mobile crowdsourcing. Specifically, we first identified the expertise domains of the tasks by designing a novel semantic analysis method to extract the semantic information of tasks and proposing a dynamic hierarchical clustering approach to cluster tasks based on their semantic information. Then, we proposed an expertise-aware truth analysis solution, in which we built an expertise-aware statistical model and applied *maximum likelihood estimation* to estimate truth and learn user expertise. Finally, we designed an expertise-aware task allocation technique by formalizing an optimization problem to maximize the probability that tasks are allocated to users with the right expertise while ensuring the work load does not exceed the processing capability at each user. Experimental results based on two real-world datasets demonstrate that ETA^2 has much lower estimation error compared with the existing approaches.

REFERENCES

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

[2] Y. Wang, W. Hu, Y. Wu, and G. Cao, "Smartphoto: a resource-aware crowdsourcing approach for image sensing with smartphones," in *ACM MobiHoc*, 2014.

[3] Y. Wu, Y. Wang, W. Hu, X. Zhang, and G. Cao, "Resource-aware photo crowdsourcing through disruption tolerant networks," in *IEEE ICDCS*, 2016.

[4] X. Yin, J. Han, and P. S. Yu, "Truth discovery with multiple conflicting information providers on the web," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 6, pp. 796–808, 2008.

[5] J. Pasternack and D. Roth, "Knowing what to believe (when you already know something)," in *COLING*. Association for Computational Linguistics, 2010.

[6] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher, "On truth discovery in social sensing: A maximum likelihood estimation approach," in *ACM IPSN*, 2012.

[7] Z. He, J. Cao, and X. Liu, "High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility," in *IEEE INFOCOM*, 2015.

[8] M. Karaliopoulos, O. Telelis, and I. Koutsopoulos, "User recruitment for mobile crowdsensing over opportunistic networks," in *IEEE INFOCOM*, 2015.

[9] S. Reddy, D. Estrin, and M. Srivastava, "Recruitment framework for participatory sensing data collections," in *Pervasive Computing*. Springer, 2010, pp. 138–155.

[10] T. Hofmann, "Probabilistic latent semantic indexing," in *ACM SIGIR conference on Research and development in information retrieval*, 1999.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[13] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data*. Springer, 2006, pp. 25–71.

[14] S. Martello and P. Toth, *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.

[15] H. Ji, R. Grishman, H. T. Dang, K. Griffitt, and J. Ellis, "Overview of the tac 2010 knowledge base population track," in *Third Text Analysis Conference (TAC)*, vol. 3, no. 2, 2010, pp. 3–3.

[16] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM (JACM)*, vol. 46, no. 5, pp. 604–632, 1999.

[17] X. Sheng, J. Tang, and W. Zhang, "Energy-efficient collaborative sensing with mobile phones," in *IEEE INFOCOM*, 2012.

[18] Q. Zhao, Y. Zhu, H. Zhu, J. Cao, G. Xue, and B. Li, "Fair energy-efficient sensing task allocation in participatory sensing with smartphones," in *IEEE INFOCOM*, 2014.

[19] Y. Wu, Y. Wang, and G. Cao, "Photo crowdsourcing for area coverage in resource constrained environments," in *IEEE INFOCOM*, 2017.

[20] M. H. Cheung, R. Southwell, F. Hou, and J. Huang, "Distributed time-sensitive task selection in mobile crowdsensing," in *ACM MobiHoc*, 2015.

[21] J. Shi, Y. Zhang, and Y. Liu, "Prisense: privacy-preserving data aggregation in people-centric urban sensing systems," in *IEEE INFOCOM*, 2010.

[22] Q. Li and G. Cao, "Providing privacy-aware incentives for mobile sensing," in *IEEE PerCom*, 2013.

[23] L. Kong, L. He, X.-Y. Liu, Y. Gu, M.-Y. Wu, and X. Liu, "Privacy-preserving compressive sensing for crowdsensing based trajectory recovery," in *IEEE ICDCS*, 2015.

[24] I. Koutsopoulos, "Optimal incentive-driven design of participatory sensing systems," in *IEEE INFOCOM*, 2013.

[25] F. Tian, B. Liu, X. Sun, X. Zhang, G. Cao, and G. Lin, "Movement-based incentive for crowdsourcing," *IEEE Transactions on Vehicular Technology*, 2017.

[26] H. Jin, L. Su, D. Chen, K. Nahrstedt, and J. Xu, "Quality of information aware incentive mechanisms for mobile crowd sensing systems," in *ACM MobiHoc*, 2015.

[27] D. Peng, F. Wu, and G. Chen, "Pay as how well you do: A quality based incentive mechanism for crowdsensing," in *ACM MobiHoc*, 2015.

[28] B. Zhao, B. I. Rubinstein, J. Gemmell, and J. Han, "A bayesian approach to discovering truth from conflicting sources for data integration," *Proceedings of the VLDB Endowment*, vol. 5, no. 6, pp. 550–561, 2012.

[29] X. Zhang, Y. Wu, and G. Cao, "Resource-aware approaches for truth analysis in crowdsourcing," in *IEEE MASS*, 2016.

[30] S. Zhi, B. Zhao, W. Tong, J. Gao, D. Yu, H. Ji, and J. Han, "Modeling truth existence in truth discovery," in *ACM SIGKDD*, 2015.

[31] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han, "On the discovery of evolving truth," in *ACM SIGKDD*, 2015.

[32] D. Y. Zhang, R. Han, D. Wang, and C. Huang, "On robust truth discovery in sparse social media sensing," in *IEEE BigData*, 2016.