CrowdEval: A Cost-Efficient Strategy to Evaluate Crowdsourced Worker's Reliability

Chenxi Qiu, Anna Squicciarini, Dev Rishi Khare
College of Information Science and Technology, Pennsylvania State University
University Park, PA, USA
{czq3,acs20,dzk5384}@psu.edu

Barbara Carminati
Department of Theoretical and Applied Sciences, University of Insubria
Varese, Italy
barbara.carminati@uninsubria.it

James Caverlee
Department of Computer Science and Engineering, Texas A&M University
College Station, TX, USA
caverlee@cse.tamu.edu

ABSTRACT

Crowdsourcing platforms depend on the quality of work provided by a distributed workforce. Yet, it is challenging to dependably measure the reliability of these workers, particularly in the face of strategic or malicious behavior. In this paper, we present a dynamic and efficient solution to keep tracking workers' reliability. In particular, we use both gold standard evaluation and peer consistency evaluation to measure each worker performance, and adjust the proportion of the two types of evaluation according to the estimated distribution of workers' behavior (e.g., being reliable or malicious). Through experiments over real Amazon Mechanical Turk traces, we find that our approach has a significant gain in terms of accuracy and cost compared to state-of-the-art algorithms.

KEYWORDS

Crowdsourcing; gold standard evaluation; peer consistency evaluation; Amazon Mechanical Turk

ACM Reference Format:
Chenxi Qiu, Anna Squicciarini, Dev Rishi Khare, Barbara Carminati, and James Caverlee. 2018. CrowdEval: A Cost-Efficient Strategy to Evaluate Crowdsourced Worker's Reliability. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

1 INTRODUCTION

Crowdsourcing platforms successfully leverage the attention of millions of users to tackle traditionally repetitive problems that are difficult to automate. Through Crowdsourcing, individuals or companies can obtain services, ideas, or content by soliciting contributions from a large group of people, especially from the online community instead of traditional employees. Crowdsourcing can offer lower prices, compared to the price of hiring dedicated professionals for the same tasks. Furthermore, with the low price come a large number of workers who are available to work at any time, as Crowdsourcing platforms usually have a low barrier to entry.

Yet, these systems suffer from some important limitations. One such limitation is the reliability of the outcomes as they are generated by workers with diverse profiles [20]. As the responses to tasks are often too large to be verified, workers' quality is difficult to control and wrong answers are hard to be efficiently detected. Some workers may be unable or unwilling to put necessary efforts to complete tasks with accuracy. Others, referred to as *malicious* workers, may instead be driven by hidden agenda, and purposely provide biased answers (e.g., promote a certain object or sway classification tasks toward a certain value) [18, 19].

To this date, a number of approaches have been proposed to deal with inaccuracies of crowd worker responses (e.g., [15, 20]). A straightforward strategy is to pre-select workers based on their responses to a pre-screen questionnaire. Unfortunately, an initial filtering method is not sufficient to evaluate workers' reliability, as there is no continued incentive for workers to submit correct answers after finishing the initial screening. As a solution, some recent works propose to randomly inject *gold tasks* (i.e., tasks with a universal truth value, and which the truth answers are known by the task requester) into the work flow to estimate the workers' accuracy. This approach is called *gold standard evaluation* or *gold evaluation* [12, 17]. By checking responses to gold tasks one can always accurately measure whether the workers' responses are correct or not. Yet, this approach can be very costly: gold tasks need to be known before-hand and workers essentially will be paid for tasks for which truth value is known [12, 17]. To reduce costs, the requesters may employ a small sample of gold tasks. In turn, this possibly affects the effectiveness of the workers' evaluation since the sample ratio may be insufficient and repeated workers can game the system by only working hard on selected gold tasks [14].

Another widely used method to estimate workers' behavior is based on *peer consistency evaluation* or *peer evaluation* [8]. The idea of peer evaluation is to use the combined (or fused) answer provided by the workers as ground truth and each worker's performance is evaluated based on its consistency with the combined answers. Different from the idea of gold tasks, peer evaluation can generate abundant "gold-task"-like questions without repetition or additional costs. Unfortunately, with peer evaluation there is a non-zero chance of unfairly penalizing workers who give accurate responses when the combined answers used as ground truth and determined through estimation tools (e.g., majority voting) are incorrect. Moreover, peer evaluation is also easily exploitable by malicious workers, who may collude to flip the combined answer.

To clarify, we analyzed a trace records of users' answers for a set of crowdsourced tasks from Mechanical Turk or MTurk [13] in Figure 1. The trace includes 25 rounds, where each round is composed of 20 binary questions (see more trace details in Section 5). After

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

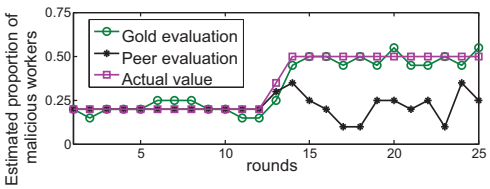


Figure 1: Comparison of two evaluation methods from an MTurk trace.

the 13th round, around half of workers become malicious and their combined answers' accuracy drops dramatically. The requester used majority voting [15] as the combination method to estimate the true answers from workers. From Figure 1, we can observe that peer evaluation cannot accurately keep track of the proportion of malicious workers after the 13th round using the low accuracy combined answers. Furthermore, once the workers' reliability is inaccurately estimated, the possibility of incorrect combined answer in the next round highly increases.

In order to estimate workers' reliability in an accurate and efficient fashion, in this paper, we build a framework, namely *CrowdEval*, with strong theoretical foundations combining gold evaluation with peer evaluation. In contrast to prior work, e.g., [12, 14], our approach dynamically adjusts the proportion of gold tasks (or gold ratio) according to the estimated proportion of different types of workers (aka *worker distribution*) in the pool and uses peer evaluation for assessment of accuracy. Importantly, our framework relies on the analysis of the distribution of reliability of workers in the pool as a whole rather than individual worker's reliability. Therefore, we are not subjected to the difficult task of working on individuals' reliability estimation and their availability over time.

With respect to performance, results based on experiments carried out over MTurk demonstrate that our approach outperforms state-of-the-art algorithms in terms of both estimation accuracy and cost (total compensation paid to all workers) on gold-tasks. In particular, with the same compensation paid to workers, our approach reduces the estimation error rate by at least 59.1% when the proportion of the malicious workers increases to around 50%. In addition, experiments on a synthetic dataset, i.e., with over 5,000 synthetic answers, demonstrate that our method outperforms state-of-the art methods with respect to the different number of workers and different proportion of malicious workers.

Simply put, our contributions can be summarized as follows:
1) We formulate a new optimization problem, *the gold ratio control (GR-Control) problem*, to minimize the estimation error of workers' reliability and to reduce the cost of gold evaluation through controlling the gold ratio.
2) We derive a GR-Control function that determines the gold ratio given the current worker distribution. We use maximum likelihood estimator (MLE) to infer the worker distribution, and study convergence results for the GR-Control function. Considering that peer evaluation has a non-zero probability to underestimate the proportion of malicious workers, we add a correction factor to adjust the gold ratio and facilitate convergence.
3) We evaluate our approach's performance by conducting extensive simulations using both synthetic data and actual workers' data. The results demonstrate the superiority of our approach in terms

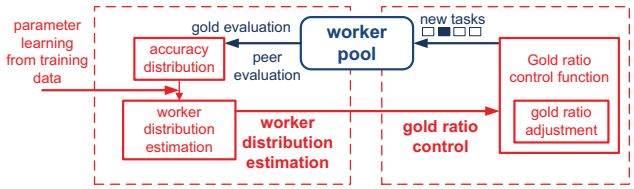


Figure 2: Outline of the system

of accuracy of workers' reliability estimation and the cost of gold evaluation.

The remainder of the paper is organized as follows: The next section gives an outline of our approach. In Section 3, we present our system model and formally define the gold ratio control problem. In Section 4.1, we present the worker distribution estimation method. In Section 4.2 and Section 4.3, we describe the gold ratio control function. In Section 5, we evaluate the performance of our method using both simulation and real world experiment, with comparisons to existing approaches. Finally, we present related work in Section 6 and conclude in Section 7.

2 OVERALL APPROACH

We organize tasks submitted by requesters in time intervals, each of which is referred to as *round*. Each round of tasks is then sequentially posted to a *worker pool*, i.e., a set of pre-screened and available workers. Tasks may or may not be binary, but we presume that a true universal answer exists or can be found. Within a round, workers may complete any number of tasks, and their behavior is assumed static. If new workers attempt to be added to the worker pool, they are added prior to the next round. Each new worker has to pass an initial pre-screening exam, which includes a set of gold tasks.

Pre-screening might help requesters to infer worker's reliability and thus correctness of answers from the collected responses. However, pre-screening obviously will not avoid workers to provide poor quality responses later on.

To overcome these issues, our approach aims at improving workers' quality control on the-fly, i.e., as crowdsourced tasks are completed by workers. The proposed system periodically injects gold tasks in a round to accurately estimate workers' reliability. In order to minimize the cost of these additional tasks, we propose a novel *dynamic gold ratio control mechanism*, i.e., a strategy to dynamically adjust the ratio of the gold tasks assigned to workers (aka gold ratio) according to the workers' behavioral distribution within a given round.

The *gold ratio control* mechanism (or GR-Control) includes two parts (highlighted as two dashed boxes in Figure 2):
1. **Worker distribution estimation** (Section 4.1), where the distribution of different classes of workers is estimated (e.g., reliable, sloppy), via the observation of workers' accuracy.
2. **Gold ratio control** (Section 4.2 and Section 4.3), in which the gold ratio is adjusted and controlled, according to the estimated worker distribution from the first part.

3 PROBLEM STATEMENT AND BASIC NOTATIONS

Crowdsourcing tasks are grouped into rounds, and each round refers to the tasks posted within a finite time interval t as *round* t.

Each round t contains $K(t)$ tasks, and a set of workers (or *worker pool*) can individually select to complete the tasks. Among $K(t)$ tasks, the system assigns $K_{\text{gold}}(t)$ gold tasks into the work flow in a random fashion, where the *gold ratio (GR)* is defined as

$$r(t) = K_{\text{gold}}(t)/K(t). \tag{1}$$

Assume that the set of workers can be categorized according to their response and accuracy rate, and results into m types (e.g., reliable workers, sloppy workers, etc.). Then, we can describe the worker distribution by $\mathbf{x}(t)$

$$\mathbf{x}(t) = [x_1(t), ..., x_l(t), ..., x_m(t)] \in \mathcal{X} \subset [0,1]^m, \tag{2}$$

where $\mathcal{X}$ represents the *worker distribution space* and $x_l(t)$ represents the proportion of workers of type l at round t ($\sum_{l=1}^{m} x_l(t) = 1$). According to the worker distribution in the last round $t-1$, the system adjusts the gold ratio (GR) $r(t)$ in the next round t. This adjustment process, *GR-Control*, aims to minimize the estimation error of the worker distribution with minimum gold ratio.

We use $y_i(t)$ and $\hat{y}_i(t)$ to represent the actual reliability and the estimated reliability of worker i at round t, respectively. Here, to estimate workers' accuracy in each round t, the system maintains a reference vector, composed of $K_{\text{gold}}(t)$ answers for gold tasks and $K(t) - K_{\text{gold}}(t)$ combined answers via majority voting [15] (i.e., answers of tasks provided by workers only in previous rounds, of which the truth is unknown) for normal tasks. Then, by checking the consistency of each worker i's answer with the reference vector, the system can estimate the worker i's accuracy $\hat{y}_i(t)$.

Moreover, given $\mathbf{y}(t) = [y_1(t), ..., y_{N(t)}(t)]$ and $\hat{\mathbf{y}}(t) = [\hat{y}_1(t), ..., \hat{y}_{N(t)}(t)]$, where $N(t)$ denotes the number of workers in round t, we define the workers' *reliability estimation error (REE)* in each round t as the square error of $\mathbf{y}(t)$ and $\hat{\mathbf{y}}(t)$[1] :

$$\varepsilon(t) = \sum_{i=1}^{N(t)} \left(y_i(t) - \hat{y}_i(t)\right)^2. \tag{3}$$

According to Equation (3), to calculate $\varepsilon(t)$ we first need to have the value of each $y_i(t)$, which is unknown. Nevertheless, researchers (e.g., [3]) have found that the accuracy of Crowdsourced workers are likely to follow a certain probability distributions, e.g., Gaussian distribution, and the distribution parameters can be estimated from workers' historical records. Hence, in what follows, we use a random variable $Y_i(t)$ to represent worker i's accuracy at round t, and we calculate the *expected REE* by

$$\varepsilon(t) = \mathbf{E}\left(\sum_{i=1}^{N(t)} \left(Y_i(t) - \hat{y}_i(t)\right)^2\right). \tag{4}$$

For simplicity, in the remaining of this paper, "REE"means the expected REE defined in Equation (4).

We note that workers' estimated accuracy $\hat{y}_i(t)$ can be affected by both gold ratio and combined answer accuracy, where the combined answer accuracy further depends on the actual worker distribution. Hence, the workers' REE is also correlated to both actual worker distribution and gold ratio. Accordingly, we can rewrite $\varepsilon(t)$ in form of $\mathbf{x}(t)$ and $r(t)$: $\varepsilon(t) = F(\mathbf{x}(t), r(t))$, where the details of F's derivation will be introduced in Section 4.2.

[1]In the following, $\varepsilon(t)$ is normalized with respect to $N(t)$.

The GR-Control problem. Once the system finds that the proportion of malicious workers in the worker pool is increasing (or decreasing), it will increase (or decrease) the gold ratio $r(t)$ correspondingly. Precisely, the requester aims to find the optimal ratio of gold tasks that simultaneously minimizes costs and maximizes expected tasks' accuracy (or minimizes errors). As far as we know, taking a linear combination of multiple objectives as the objective (also called *linear scalarization*) in problem formulation is a widely used approach [2]. Henceforth, we define the objective function of GR-Control as a linear combination of the *cost* (gold ratio) and the *estimation error* (REE). The problem can be then written as:

$$\min \quad \alpha r(t) + \beta F(\mathbf{x}(t), r(t)) \tag{5}$$

$$\text{s.t.} \quad 0 \le r(t) \le 1. \tag{6}$$

$$F(\mathbf{x}(t), r(t)) \le \epsilon. \tag{7}$$

where α and β are respectively the weights assigned to the cost and the estimation error in the objective function, and ϵ is the acceptable error rate for REE.

Next, we discuss the two main issues that need to be addressed to solve the GR-Control problem:

- Q1 *How to estimate $\mathbf{x}(t)$, given that workers' behavior cannot be observed directly (Section 4.1)?*
- Q2 *How to design the control function based on the estimated $\hat{\mathbf{x}}(t)$ (Section 4.2 and Section 4.3)?*

4 SYSTEM DESIGN

4.1 Worker Distribution Estimation

In this part, we aim to solve Q1, in which we need to infer which category each worker i belongs to in round t given their estimated accuracy.

Before categorizing workers into different types, the system first learns the accuracy distribution of each category from workers' historical record. Similar to [3], we assume that the accuracy of workers with each type l follows Gaussian distribution with mean μ_l and variance σ_l^2. We consider the following two types of workers (indicating $m = 2$ in Equation (2)):

- Type 1 *Reliable workers*, the workers who target on completing tasks with high μ_1.
- Type 2 *Malicious or careless workers*, the workers deliver relative lower accurate labels compared to reliable workers, i.e., $\mu_2 < \mu_1$. Malicious workers may be driven by a hidden agenda or try to complete tasks according to their own internal goal. For simplicity, in what follows, we call workers in type 2 malicious workers.

Here, μ_l and σ_l^2 can be learned from the workers' historical accuracy in type l. In particular, the system estimates the mean μ_l and variance σ_l^2 of each worker type by resorting to MLE [16]:

$$\hat{\mu}_l = \frac{1}{|\mathcal{A}_l|} \sum_{j \in \mathcal{A}_l} y_j \text{ and } \hat{\sigma}_l^2 = \frac{1}{|\mathcal{A}_l| - 1} \sum_{j \in \mathcal{A}_l} \left(\hat{y}_j - \hat{\mu}_l\right)^2. \tag{8}$$

where $\mathcal{A}_l$ represents the set of accuracy records of all workers in type l in the previous rounds. Then, given each observed accuracy $\hat{y}_i$ for worker i, we use MLE to estimate his behavior:

$$\hat{l}_i = \arg\max_{l=1,2} f(l; \hat{y}_i), \tag{9}$$

where $f(l;\hat{y}_i)$ is the probability density function defined by $f(l;\hat{y}_i) = \frac{1}{\sqrt{2\hat{\sigma}_l^2\pi}}\exp\left(-\frac{(\hat{y}-\hat{\mu}_l)^2}{2\hat{\sigma}_l^2}\right)$. After estimating each worker's behavior, we can obtain workers' distribution at round t, $x_1(t)$ and $x_2(t)$, by counting the number of workers with $\hat{l}_i = 1$ and $\hat{l}_i = 2$, respectively.

Note that our estimation method can be directly extended to the case of k types of workers by 1) estimating the distribution of each worker type and 2) using MLE to estimate which type each worker falls in.

4.2 Gold Ratio Control

Next, our goal to solve Q2, i.e., to design the gold ratio control function based on the estimated worker distribution $\hat{\mathbf{x}}(t)$. We start by introducing Theorem 4.1 to describe the relationship between the accuracy distribution estimation error $\varepsilon(t)$, the actual worker distribution $\mathbf{x}(t)$, and the gold ratio $r(t)$. Based on Theorem 4.1, we propose the GR-Control function, which determines the optimal gold ratio given the estimated worker distribution. We then analyze the convergence of the GR-control function by taking into account the worker distribution estimation error, i.e., the error between the estimated worker distribution and the actual value. Finally, we propose to increase the speed of GR-control function's convergence by adding a correction factor to the calculated optimal gold ratio.

Theorem 4.1. *The relationship between $\varepsilon(t)$, $\mathbf{x}(t)$, and $r(t)$ can be described by:*

$$\varepsilon(t) = F(\mathbf{x}(t), r(t)) = \frac{4(1-r(t))^2 \mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t))\mathcal{H}(\mathbf{x}(t))}{N(t)} \tag{10}$$

where

$$\mathrm{P}_{\mathrm{e}}(\mathbf{x}(t)) = \underbrace{1 - \sum_{k=1}^{\lfloor \frac{N(t)}{2} \rfloor} \sum_{j=1}^{k} (-1)^j \binom{k}{j} \mathbf{E}\left(Y^{N(t)-k+j}\right)}_{\text{the expected error probability of combined answer}}, \tag{11}$$

and

$$\mathcal{H}(\mathbf{x}(t)) = \underbrace{N(t)\left(\left(\frac{\sum_{l=1}^{2}\mu_l x_l}{\sum_{l=1}^{2} x_l} - \frac{1}{2}\right)^2 + \frac{\sum_{l=1}^{2} x_l^2\sigma_l^2}{\sum_{l=1}^{2} x_l^2}\right)}_{\text{reflect how much } \mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t)) \text{ can effect } F(\mathbf{x}(t), r(t))}. \tag{12}$$

Here $\mathbf{E}\left(Y^p\right) = \left(-\sqrt{-1}\sqrt{2}\sigma\right)^p \mathcal{U}\left(-\frac{p}{2}, \frac{1}{2}, -\frac{1}{2}\left(\frac{\mu}{\sigma}\right)^2\right)$, *$\mathcal{U}$ is Tricomi's function [16], $\mu = \sum_{l=1}^{2}\mu_l x_l$, and $\sigma = \sqrt{\sum_{l=1}^{2} x_l^2\sigma_l^2}$.*

Proof. As both Y_1 and Y_2 follow the Gaussian distribution, then given the worker distribution x_1 and x_2, the workers' accuracy in the worker pool (represented by $Y = x_1Y_1 + x_2Y_2$) also follows the Gaussian distribution with mean $x_1\mu_1 + x_2\mu_2$ and variance $\frac{x_1^2\sigma_1^2 + x_2^2\sigma_2^2}{x_1^2 + x_2^2}$. We first can calculate the expected error probability of combined answer as

$$\begin{aligned}\mathrm{P}_{\mathrm{e}}(\mathbf{x}(t)) &= 1 - \mathbf{E}\left(\sum_{k=1}^{\lfloor \frac{N(t)}{2} \rfloor}\left(Y^{N(t)-k}(1-Y)^k\right)\right) \\ &= 1 - \sum_{k=1}^{\lfloor \frac{N(t)}{2} \rfloor}\sum_{j=1}^{k}(-1)^j\binom{k}{j}\mathbf{E}\left(Y^{N(t)-k+j}\right).\end{aligned} \tag{13}$$

We then define the following events: A_1 (worker i is correct); A_2 (task is a gold task); A_3 (combined answer is correct). Let A_i^{c} present the complement of A_i ($i = 1, 2, 3$). Given the accuracy of worker i, $Y_i(t)$, we can obtain the expected estimated accuracy $\hat{y}_i(t)$

$$\begin{aligned}\hat{y}_i(t) &= \mathrm{P}\left(A_1|A_2\right)\mathrm{P}\left(A_2\right) + \mathrm{P}\left(A_1|A_2^{\mathrm{c}}\right)\mathrm{P}\left(A_2^{\mathrm{c}}\right)\mathrm{P}\left(A_3|A_2^{\mathrm{c}}\right) \\ &+ \mathrm{P}\left(A_1^{\mathrm{c}}|A_2^{\mathrm{c}}\right)\mathrm{P}\left(A_2^{\mathrm{c}}\right)\mathrm{P}\left(A_3^{\mathrm{c}}|A_2^{\mathrm{c}}\right) \\ &= Y_i(t)r(t) + Y_i(t)(1-r(t))\left(1-\mathrm{P}_{\mathrm{e}}(\mathbf{x}(t))\right) \\ &+ (1-Y_i(t))(1-r(t))\mathrm{P}_{\mathrm{e}}(\mathbf{x}(t))\end{aligned} \tag{14}$$

from which we consequently derive that

$$\begin{aligned}\varepsilon(t) = F(\mathbf{x}(t), r(t)) &= \mathrm{E}\left(\sum_{i=1}^{N(t)}\left(Y_i(t) - \hat{y}_i(t)\right)^2\right) \\ &= 4(1-r(t))^2\mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t)) \\ &\times \underbrace{N(t)\left(\left(\frac{x_1\mu_1 + x_2\mu_2}{x_1 + x_2} - \frac{1}{2}\right)^2 + \frac{x_1^2\sigma_1^2 + x_2^2\sigma_2^2}{x_1^2 + x_2^2}\right)}_{\mathcal{H}(\mathbf{x}(t))}.\end{aligned}$$

□

According to Theorem 4.1, we rewrite the GR-Control problem:

$$\begin{aligned}\min \quad & \beta\mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t))\mathcal{H}(\mathbf{x}(t)) + \left(\alpha - 2\beta\mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t))\mathcal{H}(\mathbf{x}(t))\right)r(t) \\ + \quad & \beta\mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t))\mathcal{H}(\mathbf{x}(t))r^2(t) \qquad (15) \\ \text{s.t.} \quad & 0 \le r(t) \le 1,\ F(\mathbf{x}(t), r(t)) \le \epsilon. \qquad (16)\end{aligned}$$

from which we can derive the optimal gold ratio $r_{\mathrm{opt}}(t)$:

$$r_{\mathrm{opt}}(t) = h\left(\mathbf{x}(t)\right) = \min\left\{\begin{array}{c}\max\left\{0, 1 - \frac{\alpha}{2\beta\mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t))\mathcal{H}(\mathbf{x}(t))}\right\}, \\ \sqrt{\frac{\epsilon}{4\mathcal{H}(\mathbf{x}(t))\mathrm{P}_{\mathrm{e}}^2(\mathbf{x}(t))}} + 1\end{array}\right\} \tag{17}$$

where $h\left(\mathbf{x}(t)\right)$ is called the *GR-Control function*.

Algorithm 1 briefly describes the whole process of CrowdEval. Before inferring each worker's behavior type (i.e., reliable, sloppy, or malicious), the system first learns the mean and variance of historical accuracy in each category (line 2). Next, in each round t, the algorithm collects and combines the answers from workers to derive the reference vector (line 5), and then estimates the accuracy of each worker i by comparing their answers with the reference vector (line 7). According to the estimated accuracy of workers, workers are grouped into two types (line 8) and derives $\mathbf{x}(t)$ (line 9). Finally, the algorithm derives the optimal gold ratio in the next round, $t+1$, based on $\mathbf{x}(t)$ (line 10).

Note that, in Equation (17) (line 10 of Algorithm 1), the gold ratio $r(t)$ is calculated based on the estimation of worker distribution. However, the observations used in MLE are partially based on peer evaluation, leading to a non-null probability to overestimate the accuracy of workers (Proposition 4.2).

Proposition 4.2. *Peer evaluation via MV has a non-null probability to overestimate workers' average accuracy, but has zero probability to underestimate the average accuracy.*

Proof. Given a task with true answer unknown, suppose that there are n_1 correct answers and n_2 incorrect answers. Then, the average accuracy of the answers for this task is $\frac{n_1}{n_1+n_2}$. Consider the following two cases:

1) When $n_1 > n_2$, the combined answer is correct. Then, the correctness of all the workers' answers is accurately measured by

referring the combined answer as a standard answer. Therefore, the estimated average accuracy is equal to $\frac{n_1}{n_1+n_2}$.
2) When $n_1 < n_2$, the combined answer is incorrect. Then, the correctness of all the answers is inaccurately measured, indicating that the estimated average accuracy equals to $\frac{n_2}{n_1+n_2}$, which is higher than the actual average accuracy $\frac{n_1}{n_1+n_2}$. □

Algorithm 1: Pseudo-code of gold ratio control.

```
for each worker type l do
    Estimate the mean μ̂_l and variance σ̂_l^2 via MLE (Equation (8))
      based on workers' historical accuracy record;
// Main steps
for each round t do
    Collect and combine the answers from workers, and derive the
      reference vector;
    for each worker i do
        Compare his answer with the reference vector and calculate
          his estimated accuracy ŷ(t) ;
        Use MLE to estimate the type of worker i's behavior
          (Equation (9));
    Count the numumber of three types of workers to obtain x(t);
    Set the gold ratio in round t + 1 by gold ratio control function
      h(x(t)) (Equation (17))
```

Furthermore, when calculating $r(t)$, the system has no information of the worker distribution $\mathbf{x}(t)$ in the current round t. Hence, it can only use the worker distribution in the last round $\mathbf{x}(t-1)$ to derive $r(t)$, as described in Algorithm 1. Such lag of the response to worker distribution, along with the estimation error of workers' accuracy caused by peer evaluation, will lead to the gold ratio to converge slowly to the optimal value especially when the worker distribution changes over time.

4.3 Convergence of the GR-Control function.

In light of the observation above, let us consider the two cases when the proportion of malicious workers x_2 changes at round t: when $x_2(t) < x_2(t-1)$ and when $x_2(t) > x_2(t-1)$.

According to $\varepsilon(t) = F(\mathbf{x}(t), r(t))$ derived in Equation (10)-(12), we find that: the estimation error $\varepsilon(t)$

i) decreases with the increase of gold ratio $r(t)$,
ii) increases with the increase of the proportion of malicious workers $x_2(t)$.

In other words, to guarantee the estimation error of worker distribution to be lower than the acceptable error, the more malicious exist in the worker pool, the higher gold ratio is required.

Accordingly, *when* $x_2(t) < x_2(t-1)$, the gold ratio $r(t)$ calculated based on $x_2(t-1)$ is higher than the required gold ratio based on $x_2(t)$. Hence, while more costly than the optimal value, the estimation error of $x_2(t)$ is small (as there will be more gold tasks than actually needed) and the error of calculated gold ratio will fall in the acceptable error region immediately. In contrast, *when* $x_2(t) > x_2(t-1)$, the gold ratio $r(t)$ based on $x_2(t-1)$ is lower than the required gold ratio, leading to the estimation error of $x_2(t)$ to be high (i.e. malicious workers won't be checked with sufficient gold tasks) and the gold ratio's error higher than the acceptable error. Moreover, the estimation error of $x_2(t)$ in turn degrades the accuracy of gold ratio in the next round. Consequently, it will take numerous rounds to converge the gold ratio to the acceptable error region.

To analyze the expected number of rounds to converge when $x_2(t) > x_2(t-1)$, we need to know, in each round k, how the gold ratio $r^{(k)}$ is determined by the estimated $\hat{\mathbf{x}}^{(k)} = [\hat{x}_1^{(k)}, \hat{x}_2^{(k)}]$ and how $\hat{\mathbf{x}}^{(k+1)} = [\hat{x}_1^{(k+1)}, \hat{x}_2^{(k+1)}]$ in the next round will be effected by $r^{(k)}$. Here, we define $\hat{\mathbf{x}}^{(0)} = [\hat{x}_1^{(0)}, \hat{x}_2^{(0)}]$ as the worker distribution before change. These relationships can be represented mathematically:

$$r^{(k)} = h\left(\hat{\mathbf{x}}^{(k)}\right) \tag{18}$$

$$x_2 - \hat{x}_2^{(k+1)} = \Delta\hat{x}_2^{(k+1)} = g(x_2, r^{(k)}) \tag{19}$$

where $\Delta\hat{x}_2^{(k+1)}$ represents the proportion of malicious workers that is underestimated in round $k+1$ and g, called *feedback function*, reflects how the estimation error $\Delta\hat{x}_2^{(k+1)}$ in round $k+1$ will be effected by x_2 and $r^{(k)}$ in round k. Here, g is a monotonically increasing function of x_2 and a monotonically decreasing function of r. As it is non-trivial to derive the closed-form expression of g, we approximate g by the following linear equation:

$$g(x_2, r^{(k)}) \approx \hat{g}(x_2, r^{(k)}) \tag{20}$$

$$= \delta_2 r^{(k)} + \delta_1 x_2 + \delta_0, \tag{21}$$

where $\delta_2 < 0$, $\delta_1 > 0$, and δ_0 can be learned by linear regression from training data. Similarly, we approximate GR-Control function $\hat{h}()$ by a piecewise linear function (as shown in Figure 3, where $\beta_1 = 0.26, \beta_2 = 0.77, \theta = 1.96$ and $\theta_0 = -0.51$):

$$h\left(\hat{\mathbf{x}}^{(k)}\right) \approx \hat{h}\left(\hat{\mathbf{x}}^{(k)}\right) = \begin{cases} 0 & x_2 \in [0, \beta_1) \\ \theta\hat{x}_2^{(k)} + \theta_0 & x_2 \in [\beta_1, \beta_2) \\ 1 & x_2 \in [\beta_2, 1] \end{cases} \tag{22}$$

By solving the linear difference equations Equation (20) and Equation (22), we can derive the expected number of rounds for $\hat{x}_2$'s convergence (Proposition 4.3):

Proposition 4.3. *When* $x_2 \in [\beta_1, \beta_2)$*, the expected number of rounds* $\overline{k}$ *that* $\hat{x}_2^{(k)}$ *converges to the acceptable error region* $[x_2 - \epsilon_2, x_2 + \epsilon_2]$ *is* [2]

$$\overline{k} = \left\lceil \log_{-\delta_2\theta}\left(\frac{\epsilon_2(1+\theta\delta_2) + x_2(\delta_2\theta + \delta_1) + \delta_0 + \delta_0\theta_0}{x_2(1-\delta_1) - \hat{x}_2^{(0)}(1+\delta_2\theta)}\right)\right\rceil. \tag{23}$$

Proof. It is trivial to discuss the convergency of REE in the case $x_2 \in [\beta_2, 1)$, when the gold ratio equals 1 and can always accurately estimate the worker distribution, making REE converge right after the proportion changes. On the other hand, when $x_2 \in [0, \beta_1)$, although gold ratio equals 0, the proportion of malicious workers is extremely low and the accuracy of combined answer is high. Hence, the system can still quickly detect any change of x_2. Therefore, we only discuss the case of $x_2 \in [\beta_1, \beta_2)$ in what follows.

By approximating functions g and h with Equation (20) and Equation (22), we can derive $\hat{x}_2^{(k+1)}$ (i.e., the estimated proportion

[2]The cases $x_2 \in [0, \beta_1)$ and $x_2 \in [\beta_2, 1)$ are trivial to discuss and detailed explanation can be found in the proof in the supplementary material.

of malicious workers in round $k+1$) from Equation (18) and (19):

$$\hat{x}_2^{(k+1)} = (-\delta_2\theta)^k\left(\hat{x}_2^{(0)} - \frac{x_2(1-\delta_1)-\delta_0-\delta_0\theta_0}{1+\delta_2\theta}\right) + \frac{x_2(1-\delta_1)-\delta_0-\delta_0\theta_0}{1+\delta_2\theta}. \quad (24)$$

Let ϵ_2 denote the acceptable error for the estimation of x_2. Then, to satisfy the constraint Equation (7), we have

$$x_2 - \hat{x}_2^{(k+1)} = -(-\delta_2\theta)^k\left(\hat{x}_2^{(0)} - \frac{x_2(1-\delta_1)}{1+\delta_2\theta}\right) \quad (25)$$

$$+ \frac{x_2(\delta_2\theta+\delta_1)+\delta_0+\delta_0\theta_0}{1+\delta_2\theta} \le \epsilon_2, \quad (26)$$

and finally we can derive the minimum number of rounds is

$$\overline{k} = \left\lceil \log_{-\delta_2\theta}\left(\frac{\epsilon_2(1+\theta\delta_2)+x_2(\delta_2\theta+\delta_1)+\delta_0+\delta_0\theta_0}{x_2(1-\delta_1)-\hat{x}_2^{(0)}(1+\delta_2\theta))}\right)\right\rceil. \quad (27)$$

□

Correction factor. As discussed, peer evaluation has a non-null probability to underestimate x_2, and hence the derived gold ratio may be actually lower than the gold ratio required for accurate reliability detection. Hence, we need to adjust the gold ratio $r(t)$ by adding a *correction factor* $\xi^{(k)}$ to $r^{(k)}$:

$$\hat{h}'\left(\hat{\mathbf{x}}^{(k)}\right) = \hat{h}\left(\hat{\mathbf{x}}^{(k)}\right) + \xi^{(k)}. \quad (28)$$

Here, $\xi^{(k)}$ is essentially the value that the gold ratio is under-estimated due to the estimation of x_2.

To determine $\xi^{(k)}$, we need to first estimate how $\hat{x}_2^{(k)}$ is related to the proportion of malicious workers that is underestimated, $\Delta x_2^{(k)}$. From Equation (19) and Equation (20), we can derive the relationship between $\hat{x}_2^{(k)}$ and $\Delta x_2^{(k)}$

$$\Delta x_2^{(k)} = \frac{\delta_1}{1-\delta_1}\hat{x}_2^{(k)} + \frac{\delta_2}{1-\delta_1}r^{(k-1)} + \frac{\delta_0}{1-\delta_1}. \quad (29)$$

According to the approximated GR-function defined in Equation (22), we can estimate how much the gold ratio is under-estimated, which is the product of $\Delta x_2^{(k)}$ and θ,

$$\xi^{(k)} = \theta\Delta x_2^{(k)} = \frac{\theta\delta_1}{1-\delta_1}\hat{x}_2^{(k)} + \frac{\theta\delta_2}{1-\delta_1}r^{(k-1)} + \frac{\theta\delta_0}{1-\delta_1}. \quad (30)$$

We then analyze the convergence of the gold ratio with the correction factor added: First, according to Equation (18)-(22), and Equation (28):

$$\Delta\hat{x}_2^{(k+1)} = \delta_2(\theta\hat{x}_2^{(k)} + \theta_0 + \xi^{(k)}) + \delta_1 x_2 + \delta_0 \quad (31)$$

$$= \delta_2 r + \delta_1 x_2 + \delta_0 \le \epsilon_2 \quad (32)$$

which implies that the gold ratio adjusted by the correction factor immediately satisfies the requirement in the 1st round.

Note that the adjusted gold ratio might be higher than it is required, then it will be in Case $x_2(t) < x_2(t-1)$, and REE will immediately converge to the acceptable error region in the next round. Figure 4 depicts how the correlation factor can increase the converge ratio of gold ratio to the acceptable error region using our dataset (see the description of the trace in Section 5).

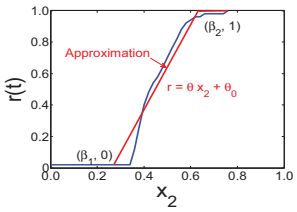


Figure 3: Approximation of the feedback function h

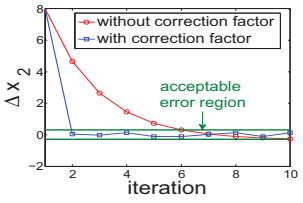


Figure 4: Convergence of the GR-Control function

5 PERFORMANCE EVALUATION

In this section, we turn our attention to practical applications of our reliability evaluation strategy and examine the performance of the system in realistic settings. Specifically, we implement simulation using both real trace from MTurk (Section 5.1) and synthetic data (Section 5.2), and carry out an online experiment on MTurk (Section 5.3).

Baseline methods and comparison metrics. We compare our approach CrowdEval with three baseline methods:

1) *Gold evaluation (GE) [14].* GE randomly injects gold tasks into the work flow with a constant ratio. The answers generated by the workers are evaluated by the gold tasks. In particular, we set the gold ratio by 0.3.

2) *Peer evaluation (PE) [8].* PE solely uses workers' combined answers as ground truth to evaluate each workers' performance.

3) *Gold evaluation and peer evaluation (GP).* GP uses both gold evaluation and peer evaluation, where gold tasks are distributed randomly in the work flow with the constant gold ratio 0.3. For the non-gold tasks, we use peer evaluation to estimate each worker's accuracy.

4) *GP to individuals (GI).* GI uses GP to evaluate all workers but specifies the gold ratio of each individual according to his own accuracy. Particularly, the gold ratio for each worker i in round t is set by $1-\hat{y}_i(t')$, where t' is the round of worker i's last recorded task responses.

The metrics we compare include:

1) *The total reliability estimation error (REE)*: $\sum_t \varepsilon(t)$ (defined by Equation (3)).

2) *The total worker distribution estimation error (WDEE)*, defined as the sum of the mean squared error of $\mathbf{x}(t)$ and $\hat{\mathbf{x}}(t)$ over t, i.e., $\sum_t \sum_{l=1}^{2}(x_l(t)-\hat{x}_l(t))^2$.

3) *The total gold cost*, i.e., the total number of gold tasks used, i.e., $K(t)\sum_t r(t)$.

4) *The relative cost*, a linear combination of the total REE and the total gold cost: $\alpha\sum_t r(t) + \beta\sum_t \varepsilon(t)$. This metric reflects the overall benefit of the requester considering both REE and gold cost. In the following we set α by 20 and β by 1 by default[3].

5.1 Trace-driven simulation

We collected 10,947 traces for "news" tasks from 267 MTurk workers. In these traces (for which ground truth values are known to us), Turk workers were asked to decide whether a given URL was a

[3]In the simulation, we assume the requester has similar preference to accuracy and cost, and we set higher value for α than for β because $r(t)$ has a smaller scale than $\varepsilon(t)$.

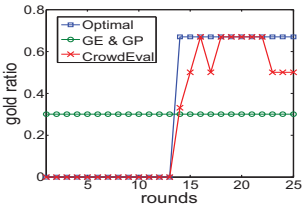


Figure 5: Gold ratio comparison in different methods

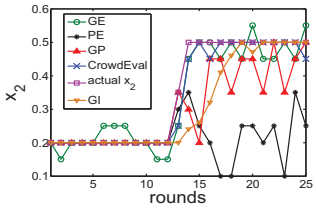


Figure 6: Comparison of x_2's estimation error

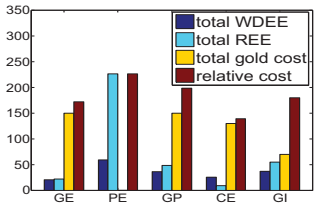


Figure 7: Overall comparison *CE denotes CrowdEval

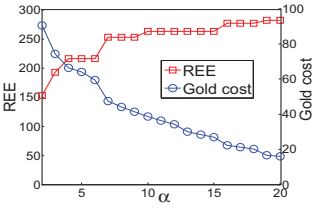


Figure 8: Performance of CrowdEval with different α

reliable news site or not. Each task has up to 10 workers to complete, and all the tasks are completed in 25 rounds. We labeled part of tasks as "gold" based on the gold task assignments and we assume the system doesn't know the true answer of a task if it is not labeled as "gold". Note that all the gold tasks are randomly injected in the work flow and workers cannot realize which tasks are gold, hence the change of gold ratio won't affect workers' answers in the trace.

To test how our method can accurately keep track of the workers' reliability when the worker distribution changes significantly over time, about 20% workers and 50% workers are guided to be malicious in the first 12 rounds and the last 13 rounds, respectively.

Comparison of gold ratio over rounds. Figure 5 compares the gold ratio of different methods with the optimal gold ratio, which is derived by the GR-Control function with the worker distribution known. From the figure, we find that optimal gold ratio is 0 in round $1-12$ and is around 0.7 in round $13-25$. Hence, GE and GP have insufficient gold tasks in round $13-25$. Compared with GE, GP, the gold ratio of CrowdEval is much closer to the optimal gold ratio over the whole 25 rounds.

Comparison of x_2 tracking. Figure 6 compares how close GE, GP, PE, GI, and CrowdEval can track the proportion of malicious workers x_2. Not surprisingly, the estimated $\hat{x}_2$ of GE fluctuates around the actual x_2. It is because that GE solely relies on gold evaluation with an insufficient source. Compared with GE, PE and GP have more accurate estimation in round 1–12. It can be attributed to the use of peer evaluation, which has a very high probability of being accurate when the proportion of malicious workers is low. However, both PE and GP underestimate x_2 in round 13–25, because peer evaluation has a non-null probability to overestimate workers' accuracy (Property 4.2), especially when x_2 is high. Although the gold tasks assigned by GP can decrease the deviation caused by peer evaluation, the estimation errors of peer evaluation still cannot be avoided since GP's gold ratio is still not high enough. Different from GP, CrowdEval can accurately track x_2 in the whole 25 rounds since 1) CrowdEval increases the gold ratio when detecting more malicious workers in round 13–25, and 2) CrowdEval adds correlation factor to eliminate the error generated by peer evaluation. Finally, it takes GI around 6-7 rounds to detect the increase of x_2. GI specifies the gold ratio for each individual according to his own accuracy in the last attendance. Since workers' participation is probably inconsistent (e.g. workers may complete a few rounds, log off and later work on additional tasks), the workers' performance that GI uses to infer gold ratio may be outdated, leading to GI's slower response to the immediate change of workers' distribution.

Overall comparison in different metrics. We compare the total WDEE, total REE, total cost, and relative cost of the above methods in Figure 7. For WDEE, we find that CrowdEval and GE have significant advantage over PE, GP, and GI, since both CrowdEval and GE can estimate x_2 more accurately in round 13–25. As for total REE, we have: CrowdEval $<$ GE $\ll$ GP $<$ GI $<$ PE, which is consistent with the observations in Figure 6. Specifically, CrowdEval reduces 59.1% REE compared to GE and reduces at least 81.3% REE compared to the other three methods. We also find that, compared with GP, CrowdEval can achieve higher performance in terms of REE and WDEE with lower gold cost. Referring to Figure 5, GP overuses the gold tasks in round 1-12, while CrowdEval always keeps the gold ratio in a suitable level. Finally, CrowdEval reduces the relative cost by at least 23.7% compared to the other methods.

CrowdEval's performance with different α. Recall that when defining the objective function of the GR-Control problem (Equation (5)), we assign weights α and β to the cost and the estimation error REE, respectively. Requesters can adjust these two weights according to their own preference to cost and accuracy. To observe how these two weights affect the optimal gold ratio, we change α from 1 to 20, and depict the total gold cost and the total REE of CrowdEval in Figure 8. We can observe that, with the increase of α the total REE increases and the total gold cost decreases, indicating that the more the requester focus on the cost, the gold ratio control function will be less accurate.

5.2 Synthetic analysis

We evaluate the performance of different methods through simulations on up to 50,000 synthetically generated data. We still consider binary questions and assume that the objectives of reliable workers and malicious workers are to provide correct and incorrect answers, respectively.

Comparison of cost efficiency. Intuitively, the budget spent for gold tasks could be used to hire more workers, as an alternative way to increase the overall task accuracy [9]. Accordingly, it is interesting to ascertain whether the use of gold tasks in CrowdEval is worthwhile, as opposed to hiring extra workers. Given the same total budget (100 USD), we compare the total REE of GE, PE, GP, GI, and CrowdEval in 25 rounds in Figure 9(a), where the total budget includes the compensation paid to both normal tasks and gold

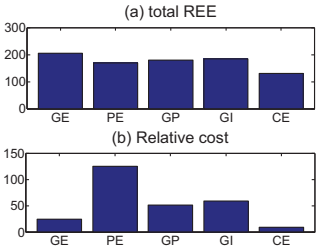


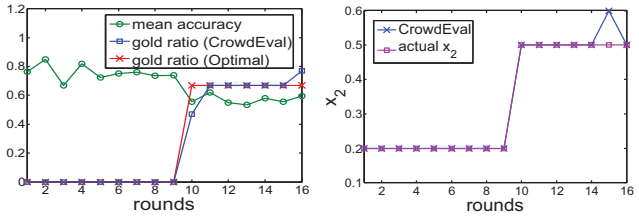
Figure 9: Comparison of cost efficiency.

tasks. Here, each round has 20 tasks and we paid 10 cents for each task. Note that GE and GP have to hire fewer workers than CrowdEval because GE and GP need to allocate more gold tasks as their average gold ratio ($r = 0.3$) is higher than CrowdEval's ($r = 0.26$). In contrast, PE and GI can afford to hire at least 30% and 12% more workers than the other three approaches, as they include no gold tasks and less gold tasks, respectively. We use the 10,947 answers from the "news" trace as the workers' answers. Considering that we need more workers in PE and GI, we additionally generated 3,000 synthetic answers, of which the accuracy follows the same distribution with the answer accuracy recorded in the "news" trace. Figure 9(a) demonstrates that CrowdEval has the lowest REE and hence it uses gold tasks most efficiently compared with the other methods. We also compare the total cost of the four methods with the same REE in Figure 9(b), which shows that GE, PE, GP costs more than CrowdEval to achieve the same degree of accuracy.

5.3 Online Experiment on MTurk

We carried out an experiment to test CrowdEval' performance on MTurk, using actual workers. The experiment includes 16 rounds, with each round composed of 20 "news" tasks. We hired 10 workers to solve each task. We asked part of workers to introduce large amount of erroneous responses to create a scenario where collusive malicious workers decrease their accuracy simultaneously. As malicious workers act in parallel, they "flip" the combined answers in peer evaluation. More specifically, we asked 20% workers and 50% workers to be malicious in the first 9 rounds and the last 7 rounds, respectively. Under these settings, we collected a total of 32,000 answers from workers. The website for data collection is built using Django [5], a python web framework, and hosted on Amazon Web Services [1].

Before the experiment, we used the 10,947 traces of "news" tasks (of which the worker type has been known) to learn the mean and variance of reliable workers and malicious workers. Figure 10(a) shows the average accuracy of workers' answers, CrowdEval's gold ratio, and the optimal gold ratio over rounds. In the 10th round, the average accuracy goes down immediately since half workers become malicious, and CrowdEval adjusts the gold ratio from 0 to around 0.7 in round 9-10, correspondingly. We observe that the gold ratio calculated by the GR-Control is close to the optimal gold ratio, even when workers' behavior changes. Figure 10(b) compares the estimated proportion and the actual proportion of malicious workers. This result is consistent with results obtained in our simulations (Figure 6), and consequently demonstrates that CrowdEval can accurately keep track of the proportion of malicious workers in the real test when a large portion of workers turn to be malicious together.



(a) Accuracy and gold ratio (b) Total WDEE

Figure 10: Online experiment on MTurk

6 RELATED WORK

A widely used approach for quality control of workers' answers in Crowdsourcing is to allocate tasks or select workers according to workers' reliability [4, 6, 7, 10, 11, 22]. For example, Cao et al. modeled the worker selection process as the well-known Jury Selection Problem [4], of which the objective is to select a jury from a set of candidate jurors to maximize the accuracy of majority voting, where each candidate juror is associated with a payment and error rate. Ho et al. [7] generalized this model to allow heterogeneous tasks, so that the probability that a worker completes a task correctly may depend on the particular task. However, all the above methods rely on accurate real-time evaluation of workers' reliability, which is non-trivial to achieve. The closer related works to ours are from [3, 6, 8, 12, 14], which focus on accurate evaluation of workers' performance. For example, Le et al. [12] studied how to inject the gold tasks into work flow to improve the reliability of collected responses. Oleson et al. [14] noticed that gold task is difficult and costly to get, and solved this problem by using previous matched answers with high confidence as "programmatic gold" to expand the size of gold standard data. Instead of using gold task, Huang et al. [8] proposed to use combined answers from workers as a standard to evaluate each worker' performance, namely peer consistent evaluation, or shortly peer evaluation. Similarly, Estrada et al. [6] evaluated workers by majority voting without gold tasks in Crowdsensing systems. The drawback of peer evaluation is its risk of unfairly penalizing workers who give accurate response when the answers determined through peer evaluation are incorrect. As both gold evaluation and peer evaluation have their merits and drawbacks, we believe that striking a balance between these two methods will further improve the estimation of workers' reliability. In addition, Yu et al. proposed a similar idea in [21], which dynamically adjusts the proportion of various information sources in multi-agent trust system to improve the overall estimation accuracy. However, Yu's approach evaluates different resources and always picks up the one with highest trust, while our approach strikes a tradeoff between accuracy and cost, by balancing the proportion of two evaluation methods.

7 CONCLUSIONS

In this paper, we presented a dynamic and efficient solution that can keep tracking of workers' reliability in real time. Different from the existing works, we use both gold evaluation and peer evaluation to measure each workers' performance and adjust the proportion of the two types of tests according to the estimated worker distribution. We demonstrate that our approach outperforms state-of-the-art methods using the real-world data.

We plan to investigate the impact of the behavior of sophisticated malicious workers (e.g., those who may eavesdrop or sniff to get the information of gold tasks), and also consider collusive behavior among malicious workers. On the other hand, we will consider how to optimize the distribution of gold tasks in the work flow, rather than randomly injecting the gold tasks.

8 ACKNOWLEDGEMENTS

This work was supported in part by AFOSR grant FA9550-15-1-0149 and PSU CSRE grant.

REFERENCES

[1] AWS. [n. d.]. https://aws.amazon.com/. ([n. d.]). [Online; accessed September 2017].

[2] S. Boyd and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

[3] M. J. Bragg and D. S. Weld. 2016. Optimal Testing for Crowd Workers. In *Proc. of AAMAS*.

[4] Y. Tong C. Cao, J. She and L. Chen. 2012. Whom to ask? jury selection for decision making tasks on micro-blog services. In *Proc. of VLDB*.

[5] Django. [n. d.]. https://www.djangoproject.com/. ([n. d.]). [Online; accessed September 2017].

[6] R. Estrada, R. Mizouni, H. Otrok, A. Ouali, and J. Bentahar. 2017. A Crowd-Sensing Framework for Allocation of Time-Constrained and Location-based Tasks. *IEEE Transactions on Services Computing* (2017).

[7] C.-J. Ho, S. Jabbari, and J. W. Vaughan. 2013. Adaptive task assignment for crowdsourced classification. In *Proc. of ICML*.

[8] S. Huang and W. Fu. 2013. Enhancing Reliability Using Peer Consistency Evaluation in Human Computation. In *Proc. of CSCW*.

[9] P. G. Ipeirotis, F. Provost, V. S. Sheng, and J. Wang. 2014. Repeated labeling using multiple noisy labelers. In *Proc. of KDD*.

[10] D. Karger, S. Oh, and D. Shah. 2011. Budget-optimal task allocation for reliable crowdsourcing systems. In *CoRR*.

[11] D. Karger, S. Oh, and D. Shah. 2011. Iterative learning for reliable crowdsourcing systems. In *Proc. of NIPS*.

[12] J. Le, A. Edmonds, V. Hester, and L. Biewald. 2010. Ensuring quality in crowdsourced search relevance evaluation: The effects of training qustion distribution. In *Proc. of the SIGIR Workshop on Crowdsourcing for Search Evaluation*.

[13] MTurk. [n. d.]. https://www.mturk.com/mturk/welcome. ([n. d.]). [Online; accessed September 2017].

[14] D. Oleson, A. Sorokin, G. Laughlin, V. Hester, J. Le, and L. Biewald. 2011. Programmatic Gold: Targeted and Scalable Quality Assurance in Crowdsourcing. In *Proc. of AAAI Workshop*.

[15] C. Qiu, A. C. Squicciarini, B. Carminati, J. Caverlee, and D. R. Khare. 2016. Crowdselect: Increasing accuracy ofcrowdsourcing tasks through behavior prediction anduser selection. In *Proc. of CIKM*.

[16] S. M. Ross. 2003. *Introduction to Probability Models*. Amsterdam: Academic Press.

[17] A. Sorokin and D. Forsyth. 2008. Utility data annotation with amazon mechanical turk. In *Proc. of Computer Vision and Pattern Recognition Workshops*.

[18] G. Wang, T. Wang, H. Zheng, and B. Y. Zhao. 2014. Man vs. machine: Practical adversarial detection of malicious crowdsourcing workers. In *Proc. of Usenix Security*.

[19] T. Wang, G. Wang, X. Li, H. Zheng, and B. Y. Zhao. 2013. Characterizing and detecting malicious crowdsourcing. In *Proc. of Sigcomm*.

[20] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Proc. of NIPS*.

[21] H. Yu, Z. Shen, C. Miao, B. An, and C. Leung. 2014. Filtering trust opinions through reinforcement learning. *Decision Support Systems* (2014).

[22] Z. Zhao, F. Wei, M. Zhou, W. Chen, and W. Ng. 2015. Crowd-selection query processing in crowdsourcing databases: A task-driven approach. In *Proc. of EDBT*.