Learning-Based Incentive Mechanism Design for Crowdsourcing

Anonymous Authors[1]

Abstract

XXXX

1. Introduction

The ability to quickly collect large scale and high quality labeled datasets is crucial for Machine Learning, and more generally Artificial Intelligence. Among all proposed solutions, one of the most promising ones is crowdsourcing (Slivkins and Vaughan, 2014; Difallah *et al.*, 2015; Simpson *et al.*, 2015). The idea is neat - instead of using a centralized amount of efforts, the to-be-labeled tasks are disseminated to a decentralized crowd of workers, leveraging the power of human computation, to parallelize the collection procedure. Nonetheless, it has been noted that crowdsourced data often suffers from quality issues, due to its unique feature of no monitoring and no ground-truth verification of workers' contributed data (Waggoner and Chen, 2014).

The above quality control challenge has been attempted to resolve by two rather disconnected research communities separately. From the more Machine Learning side, quite a few inference techniques have been developed for inferring true labels from crowdsourced and potentially noisy labels (Dawid and Skene, 1979; Raykar *et al.*, 2010; Liu *et al.*, 2012; Chen *et al.*, 2015; Zheng *et al.*, 2017). They often work as a one-shot, post-processing procedure facing a static set of workers, whose labeling accuracy is fixed and *informative*. Despite its nice theoretical contribution and empirical success, the above methods ignored the effects of *incentives* when dealing with human inputs. It has been observed both in theory and practice that xxx(Liu and Chen, 2017b), without appropriate incentive, selfish and rational workers can easily chose to contribute low quality, uninformative, or even malicious data. Existing inference algorithms are very vulnerable in these cases - either much more redundant crowdsourced labels will need to be collected (low quality inputs), or the methods will simply fail to work (the case with uninformative and malicious inputs).

The above data quality control question has also been studied in the context of *incentive mechanism design*, mostly in a non-ML setting (except for (Liu and Chen, 2017a)). In particular, a family of mechanisms, jointly called *peer prediction*, has been proposed towards addressing above incentive challenges (Prelec, 2004; Gneiting and Raftery, 2007; Jurca *et al.*, 2009; Witkowski and Parkes, 2012; Radanovic and Faltings, 2013; Dasgupta and Ghosh, 2013). Existing peer prediction mechanisms focus on achieving incentive compatibility (IC) defined as reporting a high quality data will maximize the expected payment issued to workers. They achieve IC via comparing the reports between the targeted worker, and a randomly selected reference worker.

We note several undesirable properties of existing peer prediction mechanisms. Firstly, from the label inference studies (Zheng *et al.*, 2017), we can know that all the collected labels are correlated and this correlation contains a wealth of information about the true labels and the quality of workers. However, existing peer prediction mechanisms purely rely on the reports of the reference worker, which only represents a tiny share of the information. This way of design will inevitably lower the robustness but meanwhile increase the variance of incentives. Secondly, existing peer prediction mechanisms simplify workers' responses to the incentive mechanism by assuming that workers are all fully rational and only follow the utility-maximizing strategy. However, there have been many studies showing that human beings may be bounded rational and even keep improving their strategies in practice (Simon, 1982; McKelvey and Palfrey, 1995; Chastain *et al.*, 2014). Thus, these peer prediction mechanisms that are fancy in theory may yet perform extremely poor when facing the real human workers.

Currently, the connection between machine learning and incentive mechanism design is far from being satisfactory and unnecessary. In this paper, we propose a *learning-based incentive mechanism*, aiming to marry and extend the techniques in the two areas to address the caveats discussed above. The high level idea is as follows: we divide the large dataset into relatively small task packages. At each time step, we employ workers to handle one task package and use a certain inference algorithm to learn the true labels and worker models. Then, we use the learned worker models to determine the payments for workers. Meanwhile, driving in the background, we develop a reinforcement learning algorithm to adjust the payments based on workers' historical responses to incentives. By doing so, our incentive mechanism can be adapted to different types of workers.

However, as the first work to combine the label inference and reinforcement learning algorithms with incentive mechanism design, there is a line of challenges. Here, we summarize three core contributions of this paper as follows:

- Machine learning algorithms never consider incentive compatibility. Thus, starting from scratch, we prove the incentive compatibility of our incentive mechanism not only in each time step but also in the long term.
- Since workers' states and label accuracy both are unobservable, we need to estimate them via the label inference algorithm. However, these estimates, which are the means over all tasks, are approximately corrupted with Gaussian noise. Thus, we develop our reinforcement learning algorithm based on the data-driven Gaussian process regression.
- Existing inference algorithms are severely biased on estimating the label accuracy. This bias is allowed for existing studies because they only care the true labels. However, it will mislead the reinforcement learning algorithm in our mechanism. Thus, we develop a novel Bayesian inference algorithm by firstly deriving the explicit posterior distribution of true labels and then employing Gibbs sampling for inference.

Besides, we conduct empirical evaluation, and the results show that our mechanism can improve the robustness and lower the variance of payments. Meanwhile, in the long term, our mechanism significantly increase the utility of the data requester under different worker models, such as fully rational, bounded rational and learning agent models.

2. Related Work

There have been a few pioneering studies which improves incentive mechanisms by incorporating machine learning techniques. For example, to improve the long-term utility of the data requester in crowdsourcing, Liu and Chen (2017b) develop a multi-armed bandit algorithm to adjust the state-of-the-art peer prediction mechanism, DG13 (Dasgupta and Ghosh, 2013). However, both the bandit algorithm and DG13 still need to assume that workers are fully rational and will behave as we desired. Instead of randomly choosing a reference worker, Liu and Chen (2017a) propose to use supervised learning algorithms to generate the reference reports based on the contextual information of tasks and derive the incentive compatibility conditions for the supervised-learning-based peer prediction mechanisms. In this paper, without assuming the contextual information about tasks, we develop an unsupervised-learning algorithm to learn the worker models and true labels from. Our payments are determined based on the learned worker models, and we derive the incentive compatibility conditions for our unsupervised-learning-based mechanism. Besides, in e-commerce, to be adapted to different kind of agents, Cai *et al.* (2017; 2018) propose to build incentive mechanisms based on reinforcement learning. However, focusing on the empirical analysis, they never consider the theoretical incentive compatibility. In this paper, we also incorporate reinforcement learning to get rid of the assumption that workers are fully rational. When analyzing our incentive mechanism, we go one-step further by not only providing the empirical analysis but also present a novel proof for the incentive compatibility related with reinforcement learning.

3. Problem Formulation

Suppose one data requester assigns M tasks with binary answer space $\{1,2\}$ to $N \geq 3$ candidate workers at each time step t. We denote the tasks and workers by $\mathcal{T}^t = \{1,2,\ldots,M\}$ and $\mathcal{C} = \{1,2,\ldots,N\}$, respectively. Meanwhile, we use $L_i^t(j)$ to denote the label generated by worker $i \in \mathcal{C}$ for task $j \in \mathcal{T}^t$. If $L_i^t(j) = 0$, we mean that task j is not assigned to worker i at step t.

The generated label $L_i^t(j)$ depends both on the ground-truth label $L^t(j)$ and worker i's effort level e_i^t and reporting strategy r_i^t. Any worker i can potentially have two effort levels, High ($e_i^t = 1$) and Low ($e_i^t = 0$). Also, he/she can decide either to truthfully report his observation $r_i^t = 1$ or to revert the answer $r_i^t = 0$. Workers may act differently for different tasks. We thus define $e_i^t \in [0,1]$ and $r_i^t \in [0,1]$ as worker i's probability of exerting high efforts and being truthful, respectively. In this case, worker i's probability of being correct (PoBC) can be computed as

$$\begin{aligned} p_i^t = & r_i^t e_i^t p_{i,H} + r_i^t(1-e_i^t)p_{i,L} + \\ & (1-r_i^t)e_i^t(1-p_{i,H}) + (1-r_i^t)(1-e_i^t)(1-p_{i,L}) \end{aligned} \tag{1}$$

where $p_{i,H}$ and $p_{i,L}$ denote worker i's probability of observing the correct label when exerting high and low efforts, respectively. Following (Dasgupta and Ghosh, 2013; Liu and Chen, 2017b), we assume that the tasks are homogeneous and the workers share the same set of $p_{i,H}, p_{i,L}$, denoting by p_H, p_L, and $p_H > p_L = 0.5$. Here, $p_i^t = 0.5$ means that worker i randomly selects a label to report.

The data requester needs to pay each worker some money as the incentive for providing labels. We denote the payment for worker i at step t as P_i^t. At the beginning of each time step, the data requester promises the workers a certain rule of payment determination which acts the contract between two sides and cannot be changed until the next time step. The workers are self-interested and may change their reporting strategies (e_i^t and r_i^t) according to the payment rule. Workers' different reporting strategies will lead to the different values of workers' PoBCs and finally different levels of label quality. After collecting the labels from the workers, the data requester will infer the true labels by using a certain inference algorithm, and (Zheng *et al.*, 2017) provide

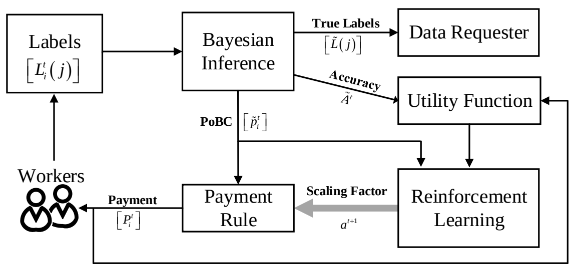


Figure 1. Layout of our incentive mechanism.

a good survey of existing inference algorithms. Denote the the inferred true label of task j by $\tilde{L}^t(j)$. Then, the label accuracy A^t and the utility u^t of the data requester satisfy

$$\begin{aligned} A^t &= \frac{1}{M}\sum_{j=1}^{M} 1\left[\tilde{L}^t(j) = L^t(j)\right] \\ u^t &= F(A^t) - \eta \sum_{i=1}^{N} P_i^t \end{aligned} \tag{2}$$

where $F(\cdot)$ is a non-decreasing monotonic function mapping accuracy to utility and η is a tunable parameter balancing label quality and costs. Intuitively, the $F(\cdot)$ function needs to be non-deceasing as higher accuracy is preferred.

The number of tasks in crowdsourcing is often very large, and the interaction between tasks and workers may last for hundreds of time steps. Thus, we introduce the cumulative utility $U(t)$ of the data requester from the current step t as

$$U(t) = \sum_{k=t}^{\infty} \gamma^{k-t} u^t \tag{3}$$

where $0 \leq \gamma < 1$ is the discount factor which determines the importance of future utilities. The objective of our study is to maximize $U(t)$ by optimally designing the payment rule and the ex-post adjustment algorithm of the payment rule, which we call as the incentive mechanism.

4. Incentive Mechanism for Crowdsourcing

We present the layout of our incentive mechanism in Figure **??**, where the estimate of a variable is denoted by adding an over-tilde. In our incentive mechanism, the Bayesian inference algorithm is responsible for estimating the true labels, workers' PoBCs and the label accuracy based on the collected labels at each time step. The payment rule is designed to ensure that reporting truthfully ($r_i^t = 0$) and exerting high efforts ($e_i^t = 1$) is the payment-maximizing strategy for all workers at any time step. By doing so, we wish to induce workers to generate high-quality labels and thus improve the label accuracy. The reinforcement learning algorithm adjusts the payment rule based on the historical data of payments, workers' PoBCs and the accuracy of aggregate labels. In this way, we can optimally balance the utility got from the labels and lost in the payments, which corresponds to $F(A^t)$ and $\sum_i P_i^t$ in Equation **??**, respectively. Besides, our incentive mechanism can ensure that always reporting truthfully ($r_i^t \equiv 0$) and exerting high efforts ($e_i^t \equiv 1$) is the payment-maximizing strategy for workers in the long term. This property prevents the clever manipulation which earns higher long-term benefits by sacrificing short-term ones.

Nevertheless, there are three challenges to achieve our design. Firstly, our empirical studies reveal that popular inference algorithms may be heavily biased towards overestimating the accuracy when the quality of labels is very low. For example, when there are 10 workers and $q_i^t = 0.55$, the estimated label accuracy of the EM estimator (Dawid and Skene, 1979; Raykar *et al.*, 2010) stays at around 0.9 while the real accuracy is only around 0.5. This heavy bias will cause the utility to be miscalculated and thus mislead our reinforcement adjustment. To reduce the inference bias, we develop our Bayesian inference algorithm by introducing the soft Dirichlet priors to both the true labels and workers' PoBCs. In this case, the posterior distribution cannot be expressed as any known distributions, which motivates us to derive the explicit posterior distribution at first and then employ Gibbs sampling to conduct inference. Secondly, the reinforcement adjustment expects the utility to be accurately calculated so that the direction of adjustment is clear. However, both the label accuracy and workers' PoBCs in our incentive mechanism are corrupted by noise. Considering that these estimates are calculated as an average over M tasks, the central limit theorem ensures that the inference noise approaches the Gaussian distribution. Therefore, to overcome the inference noise, we develop our reinforcement adjustment algorithm based on the Gaussian process. Lastly, the biggest challenge of our study is to prove that our incentive mechanism can ensure that reporting truthfully and exerting high efforts is the payment-maximizing strategy for workers in not only each time step and but also the long term. For clarity, we put the theoretical analysis in the next section. In this section, we focus on the first two challenges.

4.1. Payment Rule

Suppose, at time step t, worker i finishes M_i^t tasks. Then, the payment for worker i should be

$$P_i^t = M_i^t \cdot (a^t \phi_i^t + b) \quad , \quad \phi_i^t = \tilde{p}_i^t - 0.5 \tag{4}$$

where we call ϕ_i^t as worker i's score and $\tilde{p}_i^t$ will be calculated by our Bayesian inference algorithm. a^t is the scaling factor. It is determined by our reinforcement adjustment algorithm at the beginning of step t. We denote all the available values of a^t as set $\mathcal{A}$. Besides, $b \geq 0$ is the fixed base payment.

4.2. Bayesian Inference

Now, we present the details of our inference algorithm. For the simplicity of notations, we omit the superscript t in this subsection. The joint distribution of the collected labels $\mathcal{L} = [L_i(j)]$ and the true labels $\boldsymbol{L} = [L(j)]$ satisfies

$$
\begin{aligned}
&P(\mathcal{L}, \boldsymbol{L}|\boldsymbol{p}, \boldsymbol{\tau}) = \\
&\quad \prod_{j=1}^{M}\prod_{k=1}^{K}\left\{\tau_k \prod_{i=1}^{N} p_i^{\delta_{ijk}}(1-p_i)^{\delta_{ij(3-k)}}\right\}^{\xi_{jk}}
\end{aligned}
\tag{5}
$$

where $\boldsymbol{p} = [p_i]_N$ and $\boldsymbol{\tau} = [\tau_1, \tau_2]$. τ_1 and τ_2 denote the distribution of answer 1 and 2 among all tasks, respectively. Besides, $\delta_{ijk} = \mathbb{1}(L_i(j) = k)$ and $\xi_{jk} = \mathbb{1}(L(j) = k)$. Here, we assume Dirichlet priors $\text{Dir}(\cdot)$ for p_i and $\boldsymbol{\tau}$ as

$$
[p_i, 1-p_i] \sim \text{Dir}(\alpha_1, \alpha_2)\ ,\ \boldsymbol{\tau} \sim \text{Dir}(\beta_1, \beta_2). \tag{6}
$$

Then, the joint distribution of $\mathcal{L}$, $\boldsymbol{L}$, $\boldsymbol{p}$ and $\boldsymbol{\tau}$ satisfies

$$
\begin{aligned}
&P(\mathcal{L}, \boldsymbol{L}, \boldsymbol{p}, \boldsymbol{\tau}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = P(\mathcal{L}, \boldsymbol{L}|\boldsymbol{p}, \boldsymbol{\tau}) \cdot P(\boldsymbol{p}, \boldsymbol{\tau}|\boldsymbol{\alpha}, \boldsymbol{\beta}) \\
&= \frac{1}{B(\boldsymbol{\beta})}\prod_{k=1}^{K}\tau_k^{\hat{\beta}_k - 1} \cdot \prod_{i=1}^{N}\frac{1}{B(\boldsymbol{\alpha})}p_i^{\hat{\alpha}_{i1}-1}(1-p_i)^{\hat{\alpha}_{i2}-1}
\end{aligned}
\tag{7}
$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2]$, $\boldsymbol{\beta} = [\beta_1, \beta_2]$ and

$$
\begin{aligned}
\hat{\alpha}_{i1} &= \sum_{j=1}^{M}\sum_{k=1}^{K}\delta_{ijk}\xi_{jk} + \alpha_1 \\
\hat{\alpha}_{i2} &= \sum_{j=1}^{M}\sum_{k=1}^{K}\delta_{ij(3-k)}\xi_{jk} + \alpha_2 \\
\hat{\beta}_k &= \sum_{j=1}^{M}\xi_{jk} + \beta_k.
\end{aligned}
\tag{8}
$$

Besides, $B(x, y) = (x-1)!(y-1)!/(x+y-1)!$ denotes the beta function. The convergence of our inference algorithm requires $\alpha_1 > \alpha_2$. To simplify the theoretical analysis, we set $\alpha_1 = 1.5$ and $\alpha_2 = 1$ in this paper. Meanwhile, we employ the uniform distribution for $\boldsymbol{\tau}$ by setting $\beta_1 = \beta_2 = 1$. In this case, we can conduct marginalization via integrating Equation 7 over $\boldsymbol{p}$ and $\boldsymbol{\tau}$ as

$$
P(\mathcal{L}, \boldsymbol{L}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{B(\hat{\boldsymbol{\beta}})}{B(\boldsymbol{\beta})} \cdot \prod_{i=1}^{N}\frac{B(\hat{\boldsymbol{\alpha}}_i^*)}{[B(\boldsymbol{\alpha})]^2} \tag{9}
$$

where $\hat{\boldsymbol{\alpha}}_i^* = [\hat{\alpha}_{i1} + 0.5, \hat{\alpha}_{i2}]$ and $\hat{\boldsymbol{\beta}} = [\hat{\beta}_1, \hat{\beta}_2]$. Following Bayes' theorem, we can know that

$$
P(\boldsymbol{L}|\mathcal{L}) = \frac{P(\mathcal{L}, \boldsymbol{L}|\boldsymbol{\alpha}, \boldsymbol{\beta})}{P(\mathcal{L}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \propto B(\hat{\boldsymbol{\beta}})\prod_{i=1}^{N}B(\hat{\boldsymbol{\alpha}}_i^*). \tag{10}
$$

Based on the joint posterior distribution $P(\boldsymbol{L}|\mathcal{L})$, we cannot derive an explicit formulation for the true label distribution of task j. Hence, we resort to Gibbs sampling for the inference based on $P(\boldsymbol{L}|\mathcal{L})$. More specifically, according to Bayes' theorem, we can know the conditional distribution of the true label of task j satisfies $P[L(j)|\mathcal{L}, \boldsymbol{L}(-j)] \propto P(\boldsymbol{L}|\mathcal{L})$. In this case, we can generate the samples of the true label vector $\boldsymbol{L}$ by using Algorithm 1. At each step of sampling (line 6-8), Algorithm 1 calculates the conditional distribution and generate a new sample of $L(j)$ to replace the old one in $\boldsymbol{L}$. Through traversing all tasks, Algorithm 1 generates a new sample of the true label vector $\boldsymbol{L}$. Repeating this process for W times, we can get the required posterior samples of $\boldsymbol{L}$, which is sequentially recorded in $\mathcal{S}$. Here, we write the s-th sample as $\boldsymbol{L}^{(s)}$. Since Gibbs sampling requires a burn-in process, we need to discard the first b samples in $\mathcal{S}$. Thus, we can estimate worker i's PoBC p_i as

Algorithm 1 Gibbs sampling for crowdsourcing

1: **Input:** the collected labels $\mathcal{L}$, the number of samples W
2: **Output:** the sample sequence $\mathcal{S}$
3: $\mathcal{S} \leftarrow \emptyset$, Initialize $\boldsymbol{L} = [L(j)]_M$ with the uniform distribution
4: **for** $s = 1$ **to** W **do**
5:   **for** $j = 1$ **to** M **do**
6:     Set $L(j) = 1$ and compute $x_1 = B(\hat{\boldsymbol{\beta}})\prod_{i=1}^{N}B(\hat{\boldsymbol{\alpha}}_i)$
7:     Set $L(j) = 2$ and compute $x_2 = B(\hat{\boldsymbol{\beta}})\prod_{i=1}^{N}B(\hat{\boldsymbol{\alpha}}_i)$
8:     $L(j) \leftarrow$ Sample $\{1, 2\}$ with $P(1) = x_1/(x_1 + x_2)$
9:   **end for**
10:   Append $\boldsymbol{L}$ to the sample sequence $\mathcal{S}$
11: **end for**

$$
\tilde{p}_i = \frac{\sum_{s=b+1}^{W}\left[\alpha_1 + \sum_{j=1}^{M}\mathbb{1}(L^{(s)}(j) = L_i(j))\right]}{(W-b)\cdot(\alpha_1 + \alpha_2 + M)} \tag{11}
$$

and the distribution of true labels $\boldsymbol{\tau}$ as

$$
\tilde{\tau}_k = \frac{\sum_{s=b+1}^{W}\left[\beta_1 + \sum_{j=1}^{M}\mathbb{1}(L^{(s)}(j) = k)\right]}{(W-b)\cdot(\beta_1 + \beta_2 + M)}. \tag{12}
$$

Furthermore, we define the log-ratio of task j as

$$
\tilde{\sigma}_j = \log\frac{P[L(j) = 1]}{P[L(j) = 2]} = \log\left(\frac{\tilde{\tau}_1}{\tilde{\tau}_2}\prod_{i=1}^{N}\tilde{\lambda}_i^{\delta_{ij1} - \delta_{ij2}}\right) \tag{13}
$$

where $\tilde{\lambda}_i = \tilde{p}_i/(1-\tilde{p}_i)$. Then, we decide the true label estimate $\tilde{L}(j)$ as 1 if $\tilde{\sigma}_j > 0$ and as 2 if $\tilde{\sigma}_j < 0$. Correspondingly, the label accuracy A can be estimated as

$$
\tilde{A} = \mathbb{E}A = \frac{1}{M}\sum_{j=1}^{M}e^{|\tilde{\sigma}_j|}\left(1 + e^{|\tilde{\sigma}_j|}\right)^{-1}. \tag{14}
$$

Note that, both W and b should be large values, and in this paper, we set $W = 1000$ and $b = 100$.

4.3. Reinforcement Incentive Adjustment

In this subsection, we formally introduce our reinforcement learning (RL) algorithm, which adjusts the incentive scaling level at each time step t. Stepping back and viewing it under the large picture, the reinforcement learning serves as the glue to connect each other component in our framework.

In an RL problem, an agent interacts with an unknown environment and attempts to maximize its utility. The environment is commonly formalized as a Markov Decision

Process (MDP) defined as $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$ (). At time t the agent is in state $s_t \in \mathcal{S}$ where it takes an action $a_t \in \mathcal{A}$ leading to the next state $s_{t+1} \in \mathcal{S}$ according to the transition probability kernel $\mathcal{P}$, which encodes $Pr(s_{t+1} \mid s_t, a_t)$. In most RL problems, P is unknown to the agent. The agent's goal is to learn the optimal policy, a conditional distribution $\pi(a \mid s)$ that maximizes the sate value function

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{k=1}^{\infty} \gamma^k r_{t+k} \mid s = s_t\right].$$

The crowdsourcing problem we aim to tackle in this paper can perfectly fit into this RL formalization. To be more specific, the data requester is the agent and it interacts with workers (i.e. the environment); incentive scaling levels are actions; the implicit utility after paying workers (see Formula 2) is reward; how workers respond to different incentives and potentially change their effort levels and reporting strategies thereafter forms the transition kernel, which is unknown to the data requester; which incentive scaling level to be picked at each time t given workers' labelling constructs the policy, which needs to be learned by the data requester.

Figure 1 visualizes how our RL algorithm interacts with the environment and the rest of the framework; it takes as input workers' PoBC, reward signal, and internally its action history, and outputs the current incentive scaling level. The latest determined incentive scaling level gets plugged back into the payment rule, and by following formula (4), the exact payment to each worker is decided.

As a critical step towards improving a given, it is a standard practice for RL algorithms to learn a state-action value function (i.e. Q-function), denoted:

$$Q^{\pi}(s, a) = \mathbb{E}\left[\mathcal{R}(s_t, a_t, s_{t+1}) + \gamma V^{\pi}(s_{t+1}) \mid s = s_t, a = a_t\right]$$

In real-world problems, in order to achieve a better generalization, instead of learning a value for each state-action pair, it is more common to learn an approximate value function: $Q^{\pi}(s, a; \theta) \approx Q^{\pi}(s, a)$. A standard approach is to learn a feature-based state representation $\phi(s)$ of the state s. Due to the popularity of Deep Reinforcement learning, it has been a trend to deploy neural networks to automatically extract high-level features(). However, most deep RL methods' success is only demonstrated in the domains where the environment is very high-dimensional(). Unfortunately, this prerequisite does not hold in most crowdsourcing problems, where the number of workers are limited to be fewer than thousands. Due to this fact, we turn our attention to designing a high-quality human-engineered feature representation, which embodies our knowledge of this domain. Several studies also reveal that a carefully designed static feature representation can achieve performance as good as the most sophisticated state-of-the-art deep RL models, even in those most challenging problems. ()

Recall that the data requester's implicit utility at each time t only depends on the aggregate probability of being correct averaged across the whole worker body. Such observation already points out to a representation design which guarantees generalization. To be more specific, we design our state representations as

$$\phi(s) = \frac{1}{N} \cdot \sum_{k=1}^{2} Pr(L = k) \cdot \sum_{i=1}^{N} c_{ikk}.$$

Further recall, when deciding the scaling level a_t the data requester does not observe the latest labelling and thus cannot estimate the current $\phi(s_t)$ via Bayesian inference. Due to this one-step delay, we have to build our state representation using the previous observation. Since most workers would only change their effort levels and reporting strategies after receiving a new incentive, there exists some imperfect mapping function $\phi(s_t) \approx f(\phi(s_{t-1}), a_{t-1})$. Putting into another perspective, the combination of $\phi(s_{t-1}, a_{t-1}$ also reflects our best knowledge of the current state. Utilizing this implicit function, we formally introduce an augmented state representation for our RL algorithm

$$\hat{s_t} = \langle \phi(s_{t-1}), a_{t-1} \rangle.$$

Since the data requester never possesses the ground truth for each task, the utility u_t is not accurately observed. Also $\hat{s}_t$ is not accurate, as the mapping function can not be perfect. Combining both together, it would not be a surprise that some noise that cannot be directly learned exists in our state-action value function. As section 3 mentions, according to the central limit theorem, these noise can be modeled using Gaussian process. To be more specific, we calculate our temporal difference (TD) as

$$r_t \approx Q^{\pi}(\hat{s_t}, a_t) - \gamma V^{\pi}(\hat{s_{t+1}}) + \epsilon_t$$

where the noise ϵ follows a Gaussian process $\mathcal{N}(\hat{s_t}, \hat{s_{t+1}})$. Note we gain two benefits doing so. First, it greatly simplifies our derivation of the update for the Q-function. Secondly, our empirical results later show that this Gaussian approximation has achieved robust performance under different worker models.

Under the Gaussian process approximation, we can put all the observed rewards and the corresponding Q-function up to the current step t together and obtain

$$\boldsymbol{r} = \boldsymbol{H}\boldsymbol{Q} + \boldsymbol{N} \tag{15}$$

where $\boldsymbol{r}$, $\boldsymbol{Q}$ and $\boldsymbol{N}$ denote the collection of rewards, Q values, and residual values up to step t, respectively. Due to the Gaussian process assumption of the residual, $\boldsymbol{N} \sim$

$\mathcal{N}(\mathbf{0},\boldsymbol{\sigma}^2)$, where $\boldsymbol{\sigma}^2 = \mathrm{diag}(\sigma^2,\ldots,\sigma^2)$. The hat matrix $\boldsymbol{H}$ satisfies that $\boldsymbol{H}(k,k) = 1$ and $\boldsymbol{H}(k,k+1) = -\gamma$ for $k = 1,\ldots,t$. Then, the Q-function can be learned effectively using the online Gaussian process regression algorithm (Engel *et al.*, 2005). Furthermore, we use the classic ϵ-greedy method to construct policy from the learned Q-function.

5. Game-Theoretic Analysis

In this section, we present the game-theoretic analysis on our incentive mechanism. Our main results are as follows:

Proposition 1. *When $M \gg 1$ and $(2p_H)^{2(N-1)} \geq M$, in any time step t, reporting truthfully ($r_i^t = 0$) and exerting high efforts ($e_i^t = 1$) is the payment-maximizing strategy for any worker i if the other workers all follow this strategy. In other words, reporting truthfully and exerting high efforts is a Nash equilibrium for all workers in any time step.*

Proposition 2. *Suppose the conditions in Proposition 1 are satisfied. In our reinforcement learning algorithm, when $\tilde{Q}(s,a)$ approaches the real $Q(s,a)$ and*

$$\eta M(N-1)p_H \min_{a,b\in\mathcal{A}} |a-b| > \frac{F(1)-F(1-\psi)}{1-\rho}$$
$$\psi = 2(\tau_1\tau_2^{-1}+\tau_1^{-1}\tau_2)[4p_H(1-p_H)]^{\frac{N-1}{2}} \tag{16}$$

always reporting truthfully ($r_i^t \equiv 0$) and exerting high efforts ($e_i^t \equiv 1$) is the payment-maximizing strategy for any worker i in the long term if the other workers all follow this strategy. In other words, always reporting truthfully and exerting high efforts is a Nash equilibrium for all workers.

The proof of Proposition 1 relies on the convergence of our Bayesian inference algorithm, namely $\tilde{p}_i^t \to p_i^t$. To prove Proposition 2, we need to bound the effects of a single worker on our reinforcement learning algorithm. This analysis provides a novel tool to prevent self-interested agents from manipulating reinforcement learning algorithm.

5.1. Proof for Proposition 1

After the workers report their labels, the payment in our incentive mechanism is only decided by $\tilde{p}_i^t$ which only depends on the labels in the current step. Thus, in this subsection, we focus on analyzing our Bayesian inference algorithm and omit the superscript t in all equations for the simplicity of notations. From Equation 10, we can know the posterior distribution of the true labels satisfies

$$P(\boldsymbol{L}|\mathcal{L},\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{B(\hat{\boldsymbol{\beta}})\prod_{i=1}^N B(\hat{\boldsymbol{\alpha}}_i^*)}{C_p\cdot P(\mathcal{L}|\boldsymbol{\alpha},\boldsymbol{\beta})} \tag{17}$$

where C_p is the nomalization constant. Denote the labels generated by N workers for one task as vector $\boldsymbol{x}$. Then, we can compute the distribution of $\boldsymbol{x}$ as

$$P_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum\nolimits_{k=1}^2 \tau_k \prod\nolimits_{i=1}^N p_i^{1(x_i=k)}(1-p_i)^{1(x_i=3-k)} \tag{18}$$

where $\boldsymbol{\theta} = [\tau_1, p_1, \ldots, p_N]$ denotes all the parameters. For the denominator in Equation 17, we can have

Proposition 3. *When $M \to \infty$,*

$$P(\mathcal{L}|\boldsymbol{\alpha},\boldsymbol{\beta}) \to C_L(M)\cdot\prod\nolimits_{\boldsymbol{x}}[P_{\boldsymbol{\theta}}(\boldsymbol{x})]^{M\cdot P_{\boldsymbol{\theta}}(\boldsymbol{x})} \tag{19}$$

where $C_L(M)$ denotes a constant that depends on M.

Proof. Denote the prior distribution of $\boldsymbol{\theta}$ by π. Then,

$$P(\mathcal{L}|\boldsymbol{\alpha},\boldsymbol{\beta}) = \prod\nolimits_{j=1}^M P_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\int e^{[-M\cdot d_{KL}]}\mathrm{d}\pi(\hat{\boldsymbol{\theta}}) \tag{20}$$

$$d_{KL} = \frac{1}{M}\sum_{j=1}^M \log\frac{P_{\boldsymbol{\theta}}(\boldsymbol{x}_j)}{P_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}_j)} \to \mathrm{KL}[P_{\boldsymbol{\theta}}(\boldsymbol{x}), P_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x})] \tag{21}$$

where $\boldsymbol{x}_j$ denotes the labels generated for task j. The KL divergence $\mathrm{KL}[\cdot,\cdot]$, which denotes the expectation of the log-ratio between two probability distributions, is a constant for the given $\boldsymbol{\theta}$ and $\hat{\boldsymbol{\theta}}$. Thus, $\int e^{[-M\cdot d_{KL}]}\mathrm{d}\pi(\hat{\boldsymbol{\theta}}) = C_L(M)$. In addition, when $M \to \infty$, we can also have $\sum 1(\boldsymbol{x}_j = \boldsymbol{x}) \to M\cdot P_{\boldsymbol{\theta}}(\boldsymbol{x})$, which concludes Proposition 3. $\square$

Then, we move our focus to the posterior true label vector $\boldsymbol{L}$ generated by $P(\boldsymbol{L}|\mathcal{L},\boldsymbol{\alpha},\boldsymbol{\beta})$. We introduce n and m to denote the number of tasks of which the posterior true label is correct and wrong, respectively. Besides, for the simplicity of notations, we employ the convention that $\bar{p} = 1-p$, $\hat{p} = \max\{p,\bar{p}\}$ and $p_0 = \tau_1$. Hence, we can have

Proposition 4. *When $M \gg 1$,*

$$\mathbb{E}[m/M] \lesssim (1+e^{\delta})^{-1}(\varepsilon+e^{\delta})(1+\varepsilon)^{M-1} \tag{22}$$
$$\mathbb{E}[m/M]^2 \lesssim (1+e^{\delta})^{-1}(\varepsilon^2+e^{\delta})(1+\varepsilon)^{M-2} \tag{23}$$

where $\varepsilon^{-1} = \prod_{i=0}^N(2\hat{p}_i)^2$, $\delta = O[\Delta\cdot\log(M)]$ and

$$\Delta = \sum\nolimits_{i=1}^N[1(p_i<0.5)-1(p_i>0.5)].$$

Proof. Firstly, we introduce a set of variables to describe the real true labels and the collected labels. Among the n tasks of which the posterior true label is correct,

- x_0 and y_0 denote the number of tasks of which the real true label is 1 and 2, respectively.
- x_i and y_i denote the number of tasks of which worker i's label is correct and wrong, respectively.

Also, among the remaining $m = M - n$ tasks,

- w_0 and z_0 denote the number of tasks of which the real true label is 1 and 2, respectively.
- w_i and z_i denote the number of tasks of which worker i's label is correct and wrong, respectively.

Thus, we can have $x_i + y_i = n$ and $w_i + z_i = m$. Besides, we use ξ_i to denote the combination (x_i, y_i, w_i, z_i).

To compute the expectation of m/M, we need to analyze the probability distribution of m. According to Equation 10, we can know that $P(m)$ satisfies

$$P(m) \approx \frac{C_M^m}{Z} \sum_{\xi_0,\ldots,\xi_N} \prod_{i=0}^{N} P(\xi_i|m) B(\hat{\boldsymbol{\beta}}) \prod_{i=1}^{N} B(\hat{\boldsymbol{\alpha}}_i^*) \quad (24)$$

where $Z = C_p C_L \prod_{\boldsymbol{x}} [P_{\boldsymbol{\theta}}(\boldsymbol{x})]^{M \cdot P_{\boldsymbol{\theta}}(\boldsymbol{x})}$ is independent of ξ_i and m. Meanwhile, $\hat{\beta}_1 = x_0 + z_0 + 1$, $\hat{\beta}_2 = y_0 + w_0 + 1$, $\hat{\alpha}_{i1}^* = x_i + z_i + 2$ and $\hat{\alpha}_{i2}^* = x_i + z_i + 1$. When the m tasks of which the posterior true label is wrong are given, we can know that $x_i \sim \mathrm{Bin}(n, p_i)$ and $w_i \sim \mathrm{Bin}(m, p_i)$, where $\mathrm{Bin}(\cdot)$ denotes the binomial distribution. In addition, x_i and y_i are independent of w_i, z_i and $\xi_{k \neq i}$. Also, w_i and z_i are independent of x_i and y_i and $\xi_{k \neq i}$. Thus, we can further obtain $P(m) \approx \hat{Z}^{-1} \cdot C_M^m Y(m)$, where

$$\begin{aligned} &Y(m) = e^{\log H(m, p_0; M, 0) + \sum_{i=1}^{N} \log H(m, p_i; M, 1)} \\ &H(m, p; M, t) = \sum\nolimits_{x=0}^{n} \sum\nolimits_{w=0}^{m} 2^{M+1} C_n^x C_m^w \times \\ &\quad p^{x+w}(1-p)^{y+z} B(x + z + 1 + t, y + w + 1) \end{aligned} \quad (25)$$

and $\hat{Z} = 2^{-(N+1)(M+1)} Z$. Considering $\sum_{m=1}^{M} P(m) = 1$, we can know that $\hat{Z} \approx \sum_{m=1}^{M} C_M^m Y(m)$.

The biggest challenge of computing $P(m)$ exists in the analysis of function $H(m, p; M, t)$ which we put in the supplementary file because of the space limitation. Here, we directly use the obtained lower and upper bounds of the H function (Lemmas 17 and 18) and can have

$$\begin{cases} e^{C - K_l m} \lesssim Y(m) \lesssim e^{C - K_u m} & 2m \leq M \\ e^{C + \delta - K_l n} \lesssim Y(m) \lesssim e^{C + \delta - K_u n} & 2m > M \end{cases} \quad (26)$$

where $C = H(0, p_0; M, 0) + \sum_{i=1}^{N} H(0, p_i; M, 1)$ and

$$\begin{aligned} &K_l = \sum\nolimits_{i=0}^{N} \log \hat{\lambda}_i \ , \ K_u = 2 \sum\nolimits_{i=0}^{N} \log (2\hat{p}_i) \\ &\delta = \Delta \cdot \log(M) + \sum\nolimits_{i=1}^{N} (-1)^{1(p_i > 0.5)} \phi(\hat{p}_i) \\ &\hat{\lambda}_i = \max \left\{ \frac{p_i}{\bar{p}_i + \frac{1}{M}}, \frac{\bar{p}_i}{p_i + \frac{1}{M}} \right\} \ , \ \phi(p) = \log \frac{2p - 1}{p}. \end{aligned}$$

Besides, we set a convention that $\phi(p) = 0$ when $p = 0.5$. Thereby, the expectations of m and m^2 satisfy

$$\mathbb{E}[m] \lesssim \frac{\sum_{m=0}^{M} m e^{-K_u m} + \sum_{m=0}^{M} m e^{\delta - K_u n}}{\sum_{m=0}^{k} e^{-K_l m} + \sum_{m=k+1}^{M} e^{\delta - K_l n}} \quad (27)$$

$$\mathbb{E}[m^2] \lesssim \frac{\sum_{m=0}^{M} m^2 e^{-K_u m} + \sum_{m=0}^{M} m^2 e^{\delta - K_u n}}{\sum_{m=0}^{k} e^{-K_l m} + \sum_{m=k+1}^{M} e^{\delta - K_l n}} \quad (28)$$

where $k = \lfloor M/2 \rfloor$. By using Lemmas 4, 5, 6 and 7, we can know the upper bounds of the numerator in Equations 27 and 28 are $M(\varepsilon + e^{\delta})(1+\varepsilon)^{M-1}$ and $[M^2\varepsilon^2 + M\varepsilon + e^{\delta}(M^2 + M\varepsilon)](1+\varepsilon)^{M-2}$, respectively, where $\varepsilon = e^{-K_u}$. On the other hand, by using Lemma 8, we can obtain the lower bound of the denominator as $(1 + e^{\delta})[1 - e^{-c(\omega)M}](1 + \omega)^M$, where $\omega = e^{-K_l}$ and $c(\omega) = 0.5(1-\omega)^2(1+\omega)^{-2}$. Considering $M \gg 1$, we can make the approximation that $e^{-c(\omega)M} \approx 0$ and $(1+e^{\delta})\varepsilon/M \approx 0$. Besides, $(1+\omega)^M \geq 1$ holds because $\omega \geq 0$. In this case, Proposition 4 can be concluded by combining the upper bound of the numerator and the lower bound of the denominator. □

Lastly, focusing on worker i, we calculate the difference between the estimated PoBC $\tilde{p}_i$ and the real PoBC p_i when the other workers all exert high efforts and report truthfully. When $M \gg 1$, according to Equation 11, we can know that $\tilde{p}_i \approx \mathbb{E}_{\boldsymbol{L}}(x_i + z_i)/M$, where $\mathbb{E}_{\boldsymbol{L}}$ denotes the expectation based on the posterior distribution $P(\boldsymbol{L}|\mathcal{L})$. Meanwhile, in the proof of Proposition 4, according to the law of large numbers, $p_i \approx (x_i + w_i)/M$. Thus, we can have

$$|\tilde{p}_i - p_i| \approx \mathbb{E}_{\boldsymbol{L}} |w_i - z_i|/M \leq \mathbb{E}_{\boldsymbol{L}} [m/M] . \quad (29)$$

If workers except for worker i all report truthfully and exert high efforts, then $\Delta \leq -1$ in Proposition 4 because we require $N \geq 3$ in Section 3. Considering $M \gg 1$, we can make the approximation that $e^{\delta} \approx 0$. In addition, considering $2\hat{p}_i \geq 1$,we can have $\varepsilon^{-1} \geq (2p_H)^{2(N-1)}$. When $(2p_H)^{2(N-1)} \geq M$, $\varepsilon \leq M^{-1}$. Thus, the upper bound in Proposition 4 can be further calculated as

$$\mathbb{E}\left[\frac{m}{M}\right] \lesssim \frac{C_1}{M \cdot C_2} \ , \ \mathbb{E}\left[\frac{m}{M}\right]^2 \lesssim \frac{C_1}{M^2 \cdot C_2^2} \quad (30)$$

where $C_1 = (1 + M^{-1})^M \approx e$ and $C_2 = 1 + M^{-1} \approx 1$. Then, $m/M \approx 0$ because $\mathbb{E}[m/M] \approx 0$ and $\mathrm{Var}[m/M] = \mathbb{E}[m/M]^2 - (\mathbb{E}[m/M])^2 \approx 0$. In this case, $\tilde{p}_i \approx p_i$. Thereby, worker i can only get the maximal payment when reporting truthfully and exerting high efforts, namely, when $p_i = p_H$, which concludes Proposition 1.

5.2. Proof for Proposition 2

To prove Proposition 2, we need to analyze worker i's effects on our reinforcement learning algorithm. If worker i wishes to get higher payments in the long term, he/she must push our reinforcement learning algorithm to at least increase the scaling factor from a to $b > a$ at a certian state s. In the ϵ-greedy strategy used by our reinforcement learning algorithm, the random selection part is independent of worker i. Thus, worker i must mislead the greedy part by letting $\tilde{Q}(s, a) \leq \tilde{Q}(s, b)$. In this proof, we will show that, under the condition defined in Equation 16, there does not exist $b \in \mathcal{A}$ that can achieve this objective. In other words, our reinforcement learning algorithm will never increase the scaling factor to please a single worker. On the other hand, in any time step t, worker i will loss some money if $p_i^t < p_H$. Thereby, the payment-maximizing strategy for worker i is to report truthfully and exert high efforts in all time steps, which concludes Proposition 2.

Since Proposition 2 requires $\tilde{Q}(s,a) \approx Q(s,a)$ as one of the conditions, we now focus on proving that $Q(s,a) - Q(s,b) > 0$ always holds. Suppose all workers except for worker i report truthfully and exert high efforts in all time steps. According to Equations **??** and **??**, we can have $Q(s,a) - Q(s,b) \geq X(a) - X(b) + Y$, where

$$X(a) = \sum_{k=0}^{\infty} \rho^k \cdot \mathbb{E}F(\tilde{A}^{k+t} | s_t = s^*, a^t = a) \tag{31}$$

denotes the expected long-term utility that we get from the labels. $Y = \eta M(N-1)p_H(b-a) > 0$ denotes the payment increment for workers except worker i. To attract our reinforcement learning algorithm to increase the scaling factor, worker i must increase p_i^t when a^t is increased from a to b. Otherwise, we will get less accurate labels with higher payments, which is impossible for the greedy strategy used in our reinforcement learning algorithm. In this case, the payment for worker i will also increase. However, we do not know p_i. Thus, we regard the payment increment as 0 when deriving the lower bound of $Q(s,a) - Q(s,b)$.

Here, to bound $X(a) - X(b)$, we analyze the effects of worker i on the estimated accuracy $\tilde{A}$. Since our analysis is satisfied in all time steps, we omit the time step t for the simplicity of notations. From Equation 14, we can know that, when $M \gg 1$, the estimated accuracy $\tilde{A}$ satisfies

$$\tilde{A} \approx 1 - \mathbb{E}g(\tilde{\sigma}_j) \ \ , \ \ g(\tilde{\sigma}_j) = 1/(1 + e^{|\tilde{\sigma}_j|}). \tag{32}$$

From the proof of Proposition 1, we can know that $\tilde{p}_i^t \approx p_i^t$. In this case, according to Equation 13, we can have

$$\tilde{\sigma}_j(p_i) \approx \log\left(\frac{\tau_1}{\tau_2}\lambda_i^{\delta_{ij1}-\delta_{ij2}} \prod_{k\neq i} \lambda_H^{\delta_{kj1}-\delta_{kj2}}\right). \tag{33}$$

where $\lambda_i = p_i/(1-p_i)$ and $\lambda_H = p_H/(1-p_H)$.

Considering the case that worker i exert low efforts and reports randomly, namely $p_i = 0.5$, we can eliminate λ_i from Equation 33 because $\lambda_i = 1$. Furthermore, according to Lemma 11 in the supplementary file, we can know that $g(\tilde{\sigma}_j) < e^{\tilde{\sigma}_j}$ and $g(\tilde{\sigma}_j) < e^{-\tilde{\sigma}_j}$ both hold. Thus, we build a more tight upper bound of $g(\tilde{\sigma}_j)$ by dividing all the combinations of δ_{kj1} and δ_{kj2} in Equation 33 into two sets and using the smaller one of $e^{\tilde{\sigma}_j}$ and $e^{-\tilde{\sigma}_j}$ in each set. By using this method, if the true label is 1, we can have $\mathbb{E}_{[L(j)=1]} g(\tilde{\sigma}_j) < q_1 + q_2$, where

$$\begin{aligned} q_1 &= \frac{\tau_2}{\tau_1} \sum_{n=K+1}^{N-1} C_{N-1}^n (\frac{1}{\lambda_H})^{n-m} p_H^n (1-p_H)^m \\ q_2 &= \frac{\tau_1}{\tau_2} \sum_{n=0}^{K} C_{N-1}^m {\lambda_H}^{n-m} p_H^n (1-p_H)^m \\ n &= \sum_{k\neq i} \delta_{kj1} \ , \ m = \sum_{k\neq i} \delta_{kj2} \ , \ K = \lfloor (N-1)/2 \rfloor. \end{aligned}$$

Here, we use $e^{-\tilde{\sigma}_j}$ and $e^{\tilde{\sigma}_j}$ as the upper bound of $g(\tilde{\sigma}_j)$ when $n \in (K, N-1]$ and $n \in [0, K]$, respectively. By using Lemma 12 in the supplementary file, we can thus get

$$\mathbb{E}_{[L(j)=1]} g(\tilde{\sigma}_j) < c_\tau [4p_H(1-p_H)]^{\frac{N-1}{2}}. \tag{34}$$

where $c_\tau = \tau_1 \tau_2^{-1} + \tau_1^{-1}\tau_2$. Similarly,

$$\mathbb{E}_{[L(j)=2]} g(\tilde{\sigma}_j) < c_\tau [4p_H(1-p_H)]^{\frac{N-1}{2}}. \tag{35}$$

Thereby, $\tilde{A} > 1 - 2c_\tau[4p_H(1-p_H)]^{\frac{N-1}{2}} = 1 - \psi$.

We then consider another case where worker i exerts high efforts but reports falsely, namely $p_i = 1 - p_H$. In this case, we can rewrite Equation 33 as

$$\tilde{\sigma}_j(1-p_H) \approx \log\left(\frac{\tau_1}{\tau_2}\lambda_H^{x-y} \prod_{k\neq i} \lambda_H^{\delta_{kj1}-\delta_{kj2}}\right). \tag{36}$$

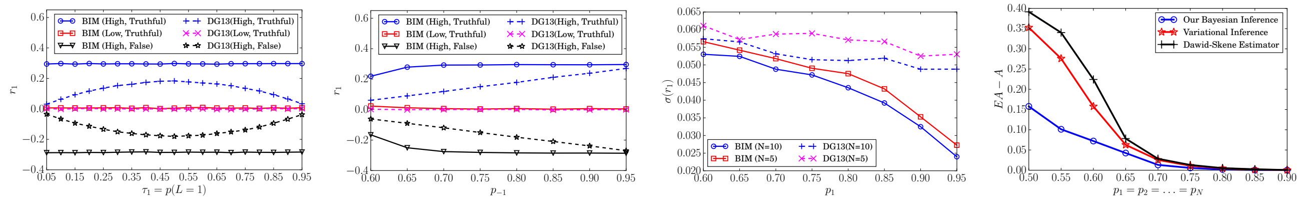
where $x = \delta_{ij2}$ and $y = \delta_{ij1}$. Since $p_i = 1 - p_H$, x and y actually has the same distribution as δ_{kj1} and δ_{kj2}. Thus, the distribution of $\tilde{\sigma}_j(1-p_H)$ is actually the same as $\tilde{\sigma}_j(p_H)$. In other words, since Proposition 1 ensures p_i to be accurately estimated, our Bayesian inference algorithm uses the information provided by worker i via flipping the label when $p_i < 0.5$. Thus, $p_i = 0.5$ actually lowers $\tilde{A}$ to the utmost because worker i provides no information about the true label in this case. Thus, $\tilde{A} \geq 1 - \psi$ always holds. On the other hand, $\tilde{A} \leq 1.0$ also always holds. Considering $F(\cdot)$ is a non-decreasing monotonic function, we can get $X(a) \geq (1-\rho)^{-1}F(1-\psi)$ while $X(b) \leq (1-\rho)^{-1}F(1)$. Thereby, when Equation 16 is satisfied, $X(a) - X(b) + Y > 0$ always holds, which concludes Proposition 2.

6. Empirical Experiments

In this section, we firstly test the one-step performance of our incentive mechanism by comparing it with the state-of-the-art incentive mechanism. Then, we show the advantages of including the reinforcement algorithm via conducting experiments on three representative worker models, including fully rational, bounded rational and self-learning agents.

6.1. One-Step Peroformance Analysis

In Figures 2a-c, we compare the average payments per task for worker 1 in our incentive mechanism with DG13, the



(a) (b) (c) (d)

Figure 2. One-step performance of our reinforcement incentive mechanism (RIM) (a) payment variation with the distribution of true labels (c) payment variation as the PoBC of other workers (d) the standard variance of the payment (d) the inference bias on label accuracy

state-of-the-art peer prediction mechanism for binary labels (Dasgupta and Ghosh, 2013; Liu and Chen, 2017b). In all these experiments, we fix the scaling factor $a^t = 1$ and set $M = 100$, $N = 10$, $p_H = 0.8$ and $b = 0$. The labels are generated by simulating workers' observation process. We firstly generate the true label for task j based on the true label distriubtion (τ_1, τ_2). Then, we generate worker i's label for task j based on worker i's PoBC p_i and the true label $L(j)$. For each point in these figures, we run the experiments for 1000 rounds and present the means.

In Figure 2a and b, we show the variation of the payment for worker 1 with the distribution of true labels and the strategies of other workers, respectively. More specifically, in Figure 2a, we let all the other workers report truthfully and exert high efforts ($p_{i\neq 1} = p_H$), and meanwhile increase τ_1 from 0.05 to 0.95. In Figure 2b, we let $\tau_1 = 0.5$, and increase $p_{i\neq 1}$ from 0.6 to 0.95. From these two figures, we can find that the payment for worker 1 in our mechanism almost only depend on worker 1's own strategy. By contrast, the payments in DG13 is severely affected by the distribution of true labels and the strategies of other workers. Furthermore, in Figure 2c, we present the standard variance of the payment for worker 1. We let $\tau_1 = 0.5$, $p_{i\neq 1} = p_H$ and meanwhile increase p_1 from 0.6 to 0.95. Form the figure, we can find that the payment vairance of our mechanism is much smaller than that of DG13. All in all, our mechanism is much fairer and more stable than DG13. This is because we can fully exploit the information provided by all workers while traditional peer prediciton mechanisms only compare the labels of two workers.

In Figure 2d, we compare our Bayesian inference algorithm with two popular inference algorithms in the studies of crowdsourcing, that is, the Dawid-Skene estimator (Dawid and Skene, 1979; Raykar *et al.*, 2010) and the variational inference estimator (Liu *et al.*, 2012; Chen *et al.*, 2015). Here, we set workers' PoBC p_i to be equal and increase the value of p_i from 0.5 to 0.9, which means the quality of labels is gradually improved. The other settings are the same as Figure 2b. From the figure, we can find that, when the quality of labels is very low, the inference bias of the Dawid-Skene and variational inference estimators on the label accuracy can be larger than 0.3 while the range of the label accuracy is only $[0.5, 1.0]$. This observation shows that these two estimators become over-optimistic for low-quality labels, which will be disastrous for our reinforcement algorithm. Thus, we develop a novel Bayesian inference algorithm which reduces the inference bias for low-quality labels by considering the connection between tasks.

References

Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. Reinforcement mechanism design for e-commerce. *CoRR*, abs/1708.07607, 2017.

Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. Reinforcement mechanism design for fraudulent behaviour in e-commerce. In *Proc. of AAAI*, 2018.

Erick Chastain, Adi Livnat, Christos Papadimitriou, and Umesh Vazirani. Algorithms, games, and evolution. *PNAS*, 111(29):10620–10623, 2014.

Xi Chen, Qihang Lin, and Dengyong Zhou. Statistical decision making for optimal budget allocation in crowd labeling. *Journal of Machine Learning Research*, 16:1–46, 2015.

Anirban Dasgupta and Arpita Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proc. of WWW*, 2013.

Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979.

Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G Ipeirotis, and Philippe Cudré-Mauroux. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In *Proc. of WWW*, 2015.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proc. of ICML*, 2005.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Radu Jurca, Boi Faltings, et al. Mechanisms for making crowds truthful. *Journal of Artificial Intelligence Research*, 34(1):209, 2009.

Yang Liu and Yiling Chen. Machine-learning aided peer prediction. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 63–80. ACM, 2017.

Yang Liu and Yiling Chen. Sequential peer prediction: Learning to elicit effort using posted prices. In *AAAI*, pages 607–613, 2017.

Qiang Liu, Jian Peng, and Alexander T Ihler. Variational inference for crowdsourcing. In *Proc. of NIPS*, 2012.

Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995.

Dražen Prelec. A bayesian truth serum for subjective data. *science*, 306(5695):462–466, 2004.

Goran Radanovic and Boi Faltings. A robust bayesian truth serum for non-binary signals. In *Proc. of AAAI*, 2013.

Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11(Apr):1297–1322, 2010.

Herbert Alexander Simon. *Models of bounded rationality: Empirically grounded economic reason*, volume 3. MIT press, 1982.

Edwin D Simpson, Matteo Venanzi, Steven Reece, Pushmeet Kohli, John Guiver, Stephen J Roberts, and Nicholas R Jennings. Language understanding in the wild: Combining crowdsourcing and machine learning. In *Proc. of WWW*, 2015.

Aleksandrs Slivkins and Jennifer Wortman Vaughan. Online decision making in crowdsourcing markets: Theoretical challenges. *ACM SIGecom Exchanges*, 12(2):4–23, 2014.

Bo Waggoner and Yiling Chen. Output agreement mechanisms and common knowledge. In *Proc. of HCOMP*, 2014.

Jens Witkowski and David C Parkes. Peer prediction without a common prior. In *Proc. of ACM EC*, 2012.

Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? *Proc. of the VLDB Endowment*, 10(5):541–552, 2017.