Leveraging Peer Communication to Enhance Crowdsourcing

Abstract

Crowdsourcing has become a popular tool for large-scale data collection where it is often assumed that crowd workers complete the work *independently*. In this paper, we relax such independent property and explore the usage of *peer communication*—a kind of direct interaction between workers—in crowdsourcing. In particular, in the crowdsourcing setting with peer communication, a *pair* of workers are asked to complete the same task together by first generating their initial answers to the task independently and then freely discussing the task with each other and updating their answers after the discussion. We first experimentally examine the effects of peer communication on crowdwork for three different types of tasks, and our results consistently suggest that the work quality is significantly improved in tasks with peer communication compared to tasks where workers complete the work independently. To better utilize peer communication in crowdsourcing, using binary classification tasks as an example, we then explore how to effectively aggregate data when some of them is generated from tasks with peer communication, and how to intelligently decide whether and when to adopt peer communication in the crowdsourcing process. We derive an aggregation method using weighted majority voting and show that it produces the maximum likelihood estimation. Moreover, we model the decision-making problem of whether and when to use peer communication in crowdsourcing as a constrained Markov decision process, and propose an algorithmic approach to solve it. Our proposed approach is empirically shown to bring higher overall work quality without exceeding the budget compared to baseline approaches.

Introduction

Crowdsourcing has gained increasing popularity recently as a scalable data collection tool for various purposes, such as obtaining labeled data for training machine learning algorithms and getting high-quality yet cheap transcriptions for audio files. Requesters who have a large amount of work at hand (e.g., label 1,000 images or transcribe 1,000 audio clips) can post such work on crowdsourcing platforms like Amazon's Mechanical Turk (MTurk) as a batch of "microtasks." Workers on the platforms can then work on each of these tasks for a short amount of time in exchange for some financial rewards, and requesters typically recruit multiple workers to work on the same task *independently* for quality assurance. A substantial amount of research in crowdsourcing, thus, focuses on how to effectively aggregate independent contributions from multiple workers (Raykar et al. 2010; Cholleti et al. 2008; Whitehill et al. 2009), and how to intelligently decide the number of independent workers needed for each task at the first place (Chen, Lin, and Zhou 2013; Li et al. ; Gurari and Grauman 2017).

More recently, researchers have started to explore the possibility of removing the independence property of crowdwork. For example, Drapeau et al. (2016) and Chang, Amershi, and Kamar (2017) have shown that in labeling tasks, workers can improve their labeling accuracy when some form of *indirect* and *structured* interactions are enabled (e.g., showing one worker the alternative answer and associated argument generated by a previous worker who has worked on the same task; workers thus "interact" with one another through their answers and arguments). While these results show the promise of enabling worker interactions in crowdwork, the particular formats of worker interactions used in these studies are quite task-specific. Therefore, it is unclear whether such interaction methods can be easily adapted to other tasks and whether similar findings can be obtained in different contexts.

On the other hand, a more general form of interaction that can be easily adopted in various contexts is to allow workers of the same task to *directly* interact with each other while they are working on the task. Inspired by the concept of peer instruction in education (Crouch and Mazur 2001), in this paper, we study one specific kind of direct interaction, which we refer to as *peer communication*. In particular, we operationalize peer communication as a procedure where a *pair* of workers working on the same task are asked to first provide an independent answer each, then freely discuss the task with each other, and finally provide an updated answer after the discussion.

We first ask whether introducing peer communication in crowdwork can bring up benefits like improvement in work quality. To this end, we design and conduct randomized experiments with three commonly seen tasks on crowdsourcing platforms: image labeling, optimal character recognition, and audio transcriptions. Our results suggest that for all types of tasks in our experiments, we have consistently observed that workers with peer communication perform significantly better than workers who work independently.

Next, we explore how to effectively utilize peer communication in crowdsourcing. Specifically, similar to questions that have been asked for independent crowdwork, we ask: (1) How should the requester *aggregate* multiple contributions on a task if some of them are produced by pairs of workers following the peer communication procedure? (2) Given that tasks with peer communication may bring up work of higher quality with higher cost compared to tasks where workers work independently, how should the requester intelligently decide *whether* and *when* peer communication is needed for each task in his batch at the first place?

In this paper, we focus on answering these questions for binary classification tasks. With respect to aggregation, not surprisingly, we find that data generated through tasks with peer communication is *correlated*. To carefully leverage such correlated data, we derive an aggregation method using weighted majority voting. We then show that, with the appropriately-chosen weights, weighted majority voting leads to the maximum likelihood estimation.

Moreover, we propose an algorithmic framework based on constrained Markov decision process to help requesters adaptively learn to decide whether and when to use peer communication for each task in his batch. We evaluate the effectiveness of the proposed algorithmic approach on real data collected through our experimental study. Results show that using our approach, requesters achieve higher overall quality across all tasks in his batch within a given budget, compared to when baseline approaches are adopted where the requester always stick with the traditional method of recruiting workers to work on each task independently.

Related Work

A major line of research in crowdsourcing is to better leverage the wisdom of crowds through effective quality control methods, and most work in this line has made the assumption that workers independently complete the tasks. One theme in the quality control literature is to develop better methods for aggregating workers' answers. Assuming a batch of noisy inputs, the EM algorithm (Dempster, Laird, and Rubin 1977) can be adopted to learn the skill levels of workers and obtain estimates of the best answer (Raykar et al. 2010; Cholleti et al. 2008; Jin and Ghahramani 2003; Whitehill et al. 2009; Dawid and Skene 1979). Building upon such aggregation methods, researchers further study the optimal decision-making problem for requesters, for example, by considering how to appropriately assign tasks to workers (Karger, Oh, and Shah 2011b; Karger, Oh, and Shah 2011a; Ho, Jabbari, and Vaughan 2013) and how to efficiently allocate budget across tasks (Chen, Lin, and Zhou 2013; Li et al. ; Gurari and Grauman 2017). In addition, practitioners and researchers have also conducted extensive studies on designing effective extrinsic incentives (Mason and Watts 2009; Horton and Chilton 2010; Yin, Chen, and Sun 2013; Ho et al. 2015) as well as intrinsic motivation (Law et al. 2016; Rogstadius et al. 2011; Shaw, Horton, and Chen 2011) to encourange better performance from workers.

Recently, researchers have started to explore the potential of incorporating *indirect* worker interactions in crowdsourcing to further improve crowdwork. One such approach is to decompose a task into several subtasks, and workers working on *different* subtasks are coordinated through *workflows* which define the input-output handoffs and thus the dependency between them (Little et al. 2010; Dai et al. 2013; Noronha et al. 2011; Kittur et al. 2011). These workflow-based approaches enable crowdsourcing to solve not only microtasks but also more complex tasks. In addition to designing workflows, some recent studies examine whether allowing indirect interactions among workers working on the *same* task through, for example, presenting a worker with other workers' answers or arguments, can improve the quality of work (Drapeau et al. 2016; Chang, Amershi, and Kamar 2017). While their results suggest a positive answer, these indirect interactions are designed for the target applications and are not trivial to generalize to other tasks. Therefore, we choose to study a more general form of *direct* worker interactions called *peer communication* in this work.

In addition to empirically showing the benefits on work quality brought up by peer communication, we further use binary classification tasks as an example and engage ourselves with addressing the problem of how to better leverage peer communication in crowdsourcing. For example, we adopt a constrained Markov decision process (MDP) framework to help requesters adaptively determine whether and when to deploy peer communication. Our MDP framework is built on top of the work by Chen, Lin, and Zhou (2013), in which they help requesters to sequentially decide which task in the batch needs an additional worker to work on given a budget constraint. Compared to Chen, Lin, and Zhou (2013), our algorithmic framework has two key differences: First, in addition to choose which task needs further work, we further decide on how to design that piece of work—recruiting one worker to work on the task independently or recruiting two workers to work on the task with peer communication? Second, since peer communication and independent work incurs *different cost*, the requester's decision-making problem does *not* degenerate into a finite-horizon Markov decision process. Thus, we explicitly model the problem as a *constrained* Markov decision process.

Understanding the Effects of Peer Communication

In this section, we first present our experimental study, in which we carefully examine the effects of introducing peer communication between pairs of workers on the quality of crowdwork through a set of randomized experiments conducted on Amazon's Mechanical Turk (MTurk).

Experimental Design

Independent vs. Discussion Tasks. To evaluate the effects of peer communication in crowdsourcing, in our experiments, we considered two ways to structure the tasks:

- *Independent tasks* (tasks without peer communication): In an independent task, workers are instructed to complete the task on their own.
- *Discussion tasks* (tasks with peer communication): Inspired by the concept of "peer instruction" in educational settings (Crouch and Mazur 2001), we designed

a procedure which guides workers in a discussion task to communicate with each other and complete the task together. Specifically, each worker is paired with another "co-worker" on a discussion task. Both workers in the pair are first asked to work on the task and submit their answers independently. Then, the pair enters a chat room, where they can see each other's independent answer, and they have two minutes to discuss the task freely. Workers are instructed to explain to each other why they believe their answers are correct. After the discussion, both workers get the opportunity to update and submit their final answers.

Experimental Treatments. The most straight-forward experimental design for examining the effects of peer communication would include two treatments, where workers in one treatment are asked to work on a sequence of independent tasks while workers in the other treatment complete a sequence of discussion tasks. However, if we adopt such a design, the different nature of independent and discussion tasks (i.e., discussion tasks require more time and effort from workers but can be more interesting to workers) implies the possibility of observing severe self-selection biases in the experiments (i.e., workers may self-select into the treatment that they can complete tasks faster or find more enjoyable).

To overcome the drawback of this simple design, we designed our experimental treatments in a way that each treatment consists of the same number of independent tasks *and* discussion tasks, such that neither treatment appears to be obviously more time-consuming or enjoyable. In particular, we bundled 6 tasks in each HIT (i.e., Human Intelligence Task on MTurk). When a worker accepted our HIT, she was told that there are 4 independent tasks and 2 discussion tasks in the HIT. There are two treatments in our experiments:

- *Treatment 1*: Workers are asked to complete 4 independent tasks followed by 2 discussion tasks.
- *Treatment 2*: Workers are asked to complete 2 discussion tasks followed by 4 independent tasks.

Importantly, we did *not* tell workers the ordering of the 6 tasks, which helps us to minimize the self-selection biases as the two treatments look the same to workers. Such design, thus, allows us to examine the effects of peer communication on work quality by comparing the work quality produced in *the first two tasks of the HIT* between the two treatments.[1]

Experimental Tasks. We conducted our experiments on three common types of tasks:

- *Image labeling*: In each task, the worker is asked to identify whether the dog shown in an image is a Siberian Husky or a Malamute. Dog images we use are collected from the Stanford dogs dataset (Khosla et al. 2011).
- *Optical character recognition (OCR)*: In each task, the worker is asked to transcribe a vehicle's license plate numbers from photos. The photos are taken from the dataset provided by Shah and Zhou (2015).
- *Audio transcription*: In each task, the worker is asked to transcribe an audio clip which contains about 5 seconds of speech. The audio clips are collected from VoxForge[2].

For all 3 types of tasks, we evaluated the work quality in it using the notion of *error*. In image labeling tasks, we defined error as the binary classification error—the error is 0 for correct labels and 1 for wrong labels. For OCR and audio transcription tasks, we defined error as the edit distance between the worker's answer and the correct answer, divided by the number of characters in the correct answer. Naturally, for all tasks, a lower rate of error implies higher work quality.

Notice that designing *indirect* interactions between workers following the previous methods (e.g., (Drapeau et al. 2016; Chang, Amershi, and Kamar 2017)) might be feasible for tasks like image labeling, but not for the other ones (i.e., OCR and audio transcription) when the space of possible answers become large. Our experiments thus focus on understanding that for a broader set of tasks where indirect worker interactions may or may not be feasible, whether and how *peer communication*—a direct, synchronous, and free-style communication between pairs of workers—can affect work quality produced in these tasks.

Experimental Procedure. To enable synchronous communication between crowd workers, we synchronized the work pace of workers by dynamically matching pairs of workers and sending them to simultaneously start working on the same sequence of tasks. In particular, when each worker arrived at our HIT, we first checked whether there was another worker in our HIT who didn't have a co-worker yet—if yes, she would be matched to that worker and assigned to the same treatment and task sequence as that worker, and the pair then started working on their sequence of tasks together. Otherwise, the worker would be *randomly* assigned to one of the two treatments as well as a *random* sequence of tasks, and she would be asked to wait for another co-worker to join the HIT for a maximum of 3 minutes. In the case where no other workers arrived at our HIT within 3 minutes, we asked the worker to decide whether she was willing to complete all tasks in the HIT on her own (and we dropped the data for the analysis but still payed her accordingly) or get a 5-cent bonus to keep waiting for another 3 minutes.

We provided a base payment of 60 cents for all our HITs. In addition to the base payments, workers were provided with the opportunity to earn performance-based bonuses, that is, workers can earn a bonus of 10 cents in a task if the final answer they submit for that task is correct. Our experiment HITs were open to U.S. workers only, and each worker was only allowed to take one HIT for each type of tasks.

[1] The middle two (independent) tasks of the HIT are not essential for the purpose of examining whether introducing peer communication in a task can affect work quality produced in it; we included them for answering a separate question (i.e., whether peer communication can be used to train workers towards better independent performance) which is not the focus of the current paper. The last two tasks of the HIT were added for balancing the number of tasks of different types in the HIT. Comparisons on work quality produced in the last two tasks should *not* be used to estimate the effects of peer communication on work quality, because workers in the last two tasks of the two treatments differed on *two* dimensions: the provision of peer communication in the current task, and the previous exposure to tasks with peer communication.

[2] `http://www.voxforge.org`

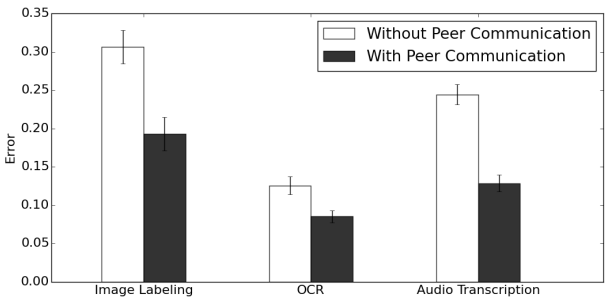

Figure 1: Comparisons of work quality produced in tasks with or without peer communication. Error bars indicate the mean $\pm$ one standard error.

Experimental Results

In total, we had 388, 382, and 250 workers who successfully formed pairs and completed the image labeling, OCR, and audio transcription tasks in our experiments, respectively.

In Figure 1, we plot the average error rate for workers' final answers in the first two tasks of Treatment 1 HITs (or Treatment 2 HITs) using white (or black) bars. Visually, it is clear that for all three types of tasks, the work quality is higher in discussion tasks (i.e., the first two tasks of Treatment 2 HITs) when workers are able to communicate with others about the work, compared to that in independent tasks (i.e., the first two tasks of Treatment 1 HITs) where workers need to complete the work on their own. We further conduct two-sample t-tests to check the statistical significance of the differences, and p-values for image labeling, OCR and audio transcription tasks are 2.42×10^{-4}, 5.02×10^{-3}, and 1.95×10^{-11} respectively, suggesting that introducing peer communication in crowdsourcing indeed improve the work quality produced significantly.[3]

Utilizing Peer Communication in Crowdsourcing

Our experimental study confirms the conjecture that introducing peer communication in crowdwork leads to positive impacts on the work quality. Naturally, the next question to ask is in practice, how can a requester better utilize the peer communication strategy. Specifically, similar to questions that have been asked in the traditional crowdsourcing setting where workers are assumed to complete their work independently, for the crowdsourcing setting where peer communication is enabled, we are interested in exploring: (1) How should the requester aggregate multiple contributions on a task, where some of them may be generated by pairs of workers following the peer communication procedure? (2) Given that tasks with peer communication may bring up work of higher quality with higher cost compared to tasks where workers work independently, how should the requester intelligently decide whether and when peer communication is needed for each task in his batch at the first place? We focus on answering these questions for binary classification tasks, in which each worker is asked to provide a label in $\{0, 1\}$ for the given labeling task.

Aggregating Data When Peer Communication is Used

When peer communication is used in a task, a pair of workers directly interact with each other. Naturally, their contributions (e.g., labels in image labeling tasks) might be correlated. Formally, let X, Y be the random variables representing the answers generated by a pair of workers for the same task. The correlation of workers' answers can be formulated using covariance $cov(X, Y)$, defined as

$$cov(X, Y) = E[XY] - E[X]E[Y]$$

By definition, when a pair of answers X, Y are independent, the covariance should be 0. To see whether the answers from a pair of workers are correlated when they work together on a task with peer communication, for each of the 20 image labeling tasks in our experiment, we calculate the covariance between pairs of labels generated in independent tasks and discussion tasks[4], in which we use the empirical average to replace the expectation in the definition. The results are shown in Figure 2. Perhaps not surprisingly, data collected in independent tasks is mostly independent (with covariance close to 0), while data collected in discussion tasks is correlated to various degrees.

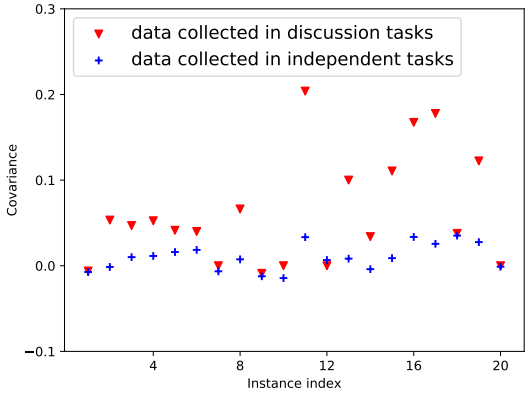

Figure 2: Covariance for data collected in independent tasks and discussion tasks in the image labeling HITs.

This observation thus poses a non-trivial question to requesters: How should a requester aggregate multiple contributions on a task if some of them can be correlated due to the communication happened between workers?

Maximum-Likelihood Aggregation Consider a binary classification task for which the true label $z \in \{0, 1\}$. When peer communication is deployed in a task, we obtain a pair of

[3] As a secondary result, comparing work quality produced in the middle two independent tasks between the two treatments suggest that previous interactions with other workers on the same type of tasks does *not* help workers to improve their independent performance, at least for short interactions as the ones we operationalized in our peer communication procedure (i.e., two minutes).

[4] Recall that in our experiment, we always send a pair of workers to work on a same sequence of tasks. Thus, an independent task is also completed by a pair of workers except that they don't communicate with each other. This allows us to directly calculate the covariance for labels generated in independent tasks.

labels, which can be $\{1,1\}$, $\{0,1\}$, or $\{0,0\}$, and we use the "*meta-label*" s_{11}, s_{01}, and s_{00} to denote them, respectively. Moreover, denote s_1 and s_0 as the label 1 and 0 obtained from a single worker who works independently.

Assume workers are homogeneous. We first propose a model to characterize the correlation in data produced in tasks with peer communication as follows: Denote p as the probability of independent workers providing correct labels, i.e., $p = P(s_1|z=1) = P(s_0|z=0)$. Additionally, we denote p_+, p_0, p_- as the probability for workers in tasks with peer communication to contribute two correct labels, one correct and one incorrect label, and two incorrect labels:

$$p_+ = P(s_{11}|z=1) = P(s_{00}|z=0)$$
$$p_- = P(s_{00}|z=1) = P(s_{11}|z=0)$$
$$p_0 = P(s_{01}|z=1) = P(s_{01}|z=0)$$

Note that when the pair of labels are independent, and the probability for each worker in the pair to submit a correct label is still p, we should have $p_+ = p^2, p_- = (1-p)^2$, and $p_0 = 2p(1-p)$. When the correlation between a pair of labels is 1 (i.e., the two labels are always the same), we have $p_0 = 0$. Therefore, this model provides a principled way to capture different levels of correlation.

As workers in different pairs do not communicate, we assume each *meta-label* is drawn independently. Now we aim to answer the following question: for a task with unknown $z \in \{0,1\}$, given a set of N labels (or meta-labels) $\boldsymbol{L} = \{l_1, ..., l_N\}$, where $l_i \in \{s_{11}, s_1, s_{01}, s_0, s_{00}\}$, how should we aggregate these labels and estimate the value of z? We first define the maximum likelihood estimator for this problem:

Definition 1. ***(Maximum likelihood estimator)*** *Let the ground truth of the task be z. Given a set of labels $\boldsymbol{L} = \{l_1, \ldots, l_N\}$. $\hat{z}$ is a maximum likelihood estimator if*

$$\hat{z} = \begin{cases} 1 & \text{if } P(\boldsymbol{L}|z=1) \geq P(\boldsymbol{L}|z=0), \\ 0 & \text{otherwise.} \end{cases}$$

To simplify the analysis, we assume p, p_+, p_0, and p_- are all known. We will relax this assumption later in our MDP formulation where we adopt a Bayesian setting to learn how to aggregate the data over time without prior knowledge on values of these parameters. However, when such prior knowledge is available, we show that a weighted majority voting rule leads to maximum likelihood estimation:

Lemma 1. *Given a set of labels $\boldsymbol{L}$. Let $n_{11}, n_1, n_{01}, n_0, n_{00}$ denote the number of labels $s_{11}, s_1, s_{01}, s_0, s_{00}$ in $\boldsymbol{L}$. Consider the following weighted majority voting rule that generates a aggregation $\hat{z}$*

$$\hat{z} = \begin{cases} 1 & \text{if } w_{11}n_{11} + w_1 n_1 \geq w_{00}n_{00} + w_0 n_0 \\ 0 & \text{if } w_{11}n_{11} + w_1 n_1 < w_{00}n_{00} + w_0 n_0 \end{cases}$$

This weighted majority voting rule leads to maximum likelihood estimation when the weights are set as: $w_{11} = w_{00} = \ln \frac{p_+}{p_-}$, and $w_1 = w_0 = \ln \frac{p}{1-p}$.

Proof. We can write the probabilities on both sides as follows:

$$P(\boldsymbol{L}|z=1) = p_+^{n_{11}} p^{n_1} p_0^{n_{01}} (1-p)^{n_0} p_-^{n_{00}}$$
$$P(\boldsymbol{L}|z=0) = p_-^{n_{11}} (1-p)^{n_1} p_0^{n_{01}} p^{n_0} p_+^{n_{00}}$$

Therefore, we have

$$\frac{P(\boldsymbol{L}|z=1)}{P(\boldsymbol{L}|z=0)} = \left(\frac{p_+}{p_-}\right)^{n_{11}} \left(\frac{p}{1-p}\right)^{n_1} \left(\frac{1-p}{p}\right)^{n_0} \left(\frac{p_-}{p_+}\right)^{n_{00}}$$

Note that, in maximum likelihood estimator, $\hat{z} = 1$ if $P(\boldsymbol{L}|z=1)/P(\boldsymbol{L}|z=0) \geq 1$. Therefore, $\hat{z} = 1$ if

$$\left(\frac{p_+}{p_-}\right)^{n_{11}} \left(\frac{p}{1-p}\right)^{n_1} \geq \left(\frac{p}{1-p}\right)^{n_0} \left(\frac{p_+}{p_-}\right)^{n_{00}}$$

The proof is completed by taking logarithm on both sides. □

As a sanity check, we can easily see that when a pair of labels are independent (i.e., $p_+ = p^2$ and $p_- = (1-p)^2$), we have $w_{11} = w_{00} = 2w_1 = 2w_0$, implying that the weight of $\{1,1\}$ label is twice as the weight of $\{1\}$ label, and this is essentially a simple majority voting.

A caveat to note here is that the number of meta-label s_{01} actually does *not* play a role in the aggregation process. In other words, we may consider a pair of workers in peer communication as a *meta-worker* who generates a *meta-label*, and with probability p_+ (or p_-) she generates a correct (or incorrect) label, while with probability p_0 she generates *no* label at all. The above weighted majority voting rule then simply indicates that different weights need to be used for labels generated by independent workers or meta-workers.

Finally, we would like to note that this aggregation rule can be extended to the setting with heterogeneous workers, if we assume we know their skill levels (as represented by the set of (p, p_+, p_0, p_-)). In the case that the skill levels are unknown, we can apply standard techniques in label aggregation literature and apply EM-like approach to learn worker skills and true labels simultaneously.

Dynamically Deciding the Use of Peer Communication

We now turn to our second question on better utilizing peer communication in crowdsourcing. In particular, while our experimental study clearly shows that introducing peer communication in crowdwork leads to significant improvement in work quality, such improvement also comes with extra cost, such as the financial payment incurred to recruit more workers (e.g., at least two workers are needed to form a pair and work together for peer communication to happen) and the additional administrative cost for synchronizing the work pace of worker pairs. As a result, in practice, a requester faces the quality-cost tradeoff, and he needs to strategically decide *whether* and *when* to use peer communication for each task in his batch. Inspired by the method discussed by Chen, Lin, and Zhou (2013) to optimally allocate budget among task instances in crowdsourcing data collection, we propose an algorithmic approach to help requesters dynamically deploy tasks with peer communication in an optimal way by modeling the decision-making problem as a Bayesian Markov decision process (MDP). However, compared with the work by Chen, Lin, and Zhou (2013), there are additional challenges in our setting since deploying peer communication or not incurs different cost and produces different data. Therefore, we have formulated our problem as a *constrained* MDP to address these challenges.

Problem Setup. Suppose a requester gets a budget of $\mathcal{B}$ and a batch of K binary classification tasks, and he needs to estimate the label for each of these tasks. The goal of the requester is to maximize the average accuracy of the estimated labels across all tasks through spending the budget to solicit labels from crowd workers and then aggregating the collected labels. Assume the K tasks are independent from each other, and $Z_k \in \{0, 1\}$ represents the true label for task k ($1 \leq k \leq K$). We characterize the difficulty of task k using a parameter $\theta_k \in [0, 1]$, which is defined as the probability for a "reliable" worker to submit a label of 1 in task k. We further assume that θ_k is consistent with Z_k, that is, $Z_k = 1$ if and only if $\theta_k \geq 0.5$.[5] Intuitively, task k is relatively difficult when θ_k is close to 0.5.

The requester recruits workers to label his tasks in a sequential manner. Specifically, at each time step t, the requester decides on a task k_t to work on, and he can solicit label(s) from crowd workers on this task using one of the two strategies (the strategy is denoted as x_t): first, the requester can recruit a *single* worker to work on the task ($x_t = 0$), and thus obtain *one* label for that task; second, the requester may recruit a *pair* of workers to work on the task together following the peer communication procedure ($x_t = 1$), and thus obtain *two* labels for the task. Following the discussion in the previous section, below we denote a *meta-worker* as a pair of workers recruited with peer communication and a *meta-label* as the two labels obtained from a meta-worker,

These two recruiting strategies have a few key differences. First, recruiting a single worker incurs a cost of c_s, while recruiting a meta-worker incurs a cost of c_p ($c_p > c_s$). In terms of the accuracy of the labels, we use $\alpha_s \in [0, 1]$ to represent the skill level of a worker when she works independently, and it is defined as the probability that a worker's independent answer agrees with the answer from a reliable worker. Thus, when $x_t = 0$, following the notations for collected labels in the previous section, we have:

$$p_{k_t,s,1} = \mathbb{P}(y_t = s_1|\theta_{k_t}, \alpha_s) = \alpha_s\theta_{k_t} + (1 - \alpha_s)(1 - \theta_{k_t})$$
$$p_{k_t,s,0} = \mathbb{P}(y_t = s_0|\theta_{k_t}, \alpha_s) = 1 - p_{k_t,s,1}$$

where y_t is the label the requester gets from an independent worker at time step t. Intuitively, a higher value of α_s implies a higher accuracy. In addition, we use $p_{k_t,0} \in [0, 1]$ and $\alpha_p \in [0, 1]$ to characterize the skill level of a meta-worker. Specifically, $p_{k_t,0}$ is the probability of a meta-worker generating a meta-label of s_{01} in task k_t which, recall from the last section, will effectively be "discarded" when the requester aggregates the data. Conditioned on a meta-worker contributing a meta-label other than s_{01}, α_p is similarly defined as α_s. Formally, when $x_t = 1$, we have:

$$\begin{aligned} q_{k_t} &= \mathbb{P}(y_t = s_{01}|\theta_{k_t}, \alpha_s) = p_{k_t,0} \\ p_{k_t,p,1} &= \mathbb{P}(y_t = s_{11}|\theta_{k_t}, \alpha_s) \\ &= (1 - p_{k_t,0})(\alpha_p\theta_{k_t} + (1 - \alpha_p)(1 - \theta_{k_t})) \\ p_{k_t,p,0} &= \mathbb{P}(y_t = s_{00}|\theta_{k_t}, \alpha_s) = 1 - p_{k_t,p,1} - q_{k_t} \end{aligned}$$

[5] In other words, θ_k is the *soft label* for task k, and a worker is reliable if she always randomly samples her submitted hard label in a task based on the soft label.

Naturally, the requester's activity in each time step can be summarized through the tuple (k_t, x_t, y_t). By the time $t_{\mathcal{B}}$ that the requester exhausts his budget, his activity history is $\mathcal{H}_{\mathcal{B}} = \{(k_0, x_0, y_0), ..., (k_{t_{\mathcal{B}}}, x_{t_{\mathcal{B}}}, y_{t_{\mathcal{B}}})\}$. The requester then aggregates the data he has collected and infers the true labels for each of the K tasks such that the expected accuracy across all K tasks conditioned on the activity history $\mathcal{H}_{\mathcal{B}}$ is maximized. In other words, the requester determines a set of tasks $S_{\mathcal{B}}$ with the inferred label being 1 by solving the optimization problem:

$$S_{\mathcal{B}} = \underset{S \subset \{1,...,K\}}{\operatorname{argmax}} \mathbb{E}(\sum_{i \in S} \mathbf{1}(Z_i = 1) + \sum_{i \notin S} \mathbf{1}(Z_i = 0)|\mathcal{H}_{\mathcal{B}})$$

where $\mathbf{1}(.)$ is the indicator function.

A Constrained Markov Decision Process Formulation. We now formally model the requester's decision-making problem as a constrained Markov decision process:

- **States**: the state s_t is a $K \times 4$ matrix, where $s_t(k, \cdot)$ is a 1×4 vector with each entry representing before time t, the number of label (or meta-labels) s_0, s_1, s_{00}, s_{11} obtained for task k. Note that since the meta-label s_{01} does not affect the aggregation, we do not include the count of it in the state representation.
- **Actions**: $a_t = (k_t, x_t)$, where k_t is the task to work on at time t, and $x_t \in \{0, 1\}$ represents the worker recruiting strategy, with 0 being recruiting a single worker working independently and 1 being recruiting a pair of workers to follow the peer communication procedure.
- **Transition probabilities**: When $a_t = (k_t, x_t = 0)$, $Pr(s_{t+1}|s_t, a_t) =$

$$\begin{cases} p_{k_t,s,1} & s_{t+1} = s_t + (\mathbf{0}, \mathbf{e_{k_t}}, \mathbf{0}, \mathbf{0}) \\ p_{k_t,s,0} & s_{t+1} = s_t + (\mathbf{e_{k_t}}, \mathbf{0}, \mathbf{0}, \mathbf{0}) \\ 0 & \text{otherwise} \end{cases}$$

  where $\mathbf{e_{k_t}}$ is a $K \times 1$ vector with value 1 at the k_t-th entry and 0 at all other entries. On the other hand, when $a_t = (k_t, x_t = 1)$, $Pr(s_{t+1}|s_t, a_t) =$

$$\begin{cases} p_{k_t,p,1} & s_{t+1} = s_t + (\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{e_{k_t}}) \\ 1 - p_{k_t,p,1} - q_{k_t} & s_{t+1} = s_t + (\mathbf{0}, \mathbf{0}, \mathbf{e_{k_t}}, \mathbf{0}) \\ q_{k_t} & s_{t+1} = s_t \\ 0 & \text{otherwise} \end{cases}$$

- **Rewards**: We adopt the same reward function as that used by Chen, Lin, and Zhou (2013). Specifically, we assume the parameters $\theta_k, \alpha_s, \alpha_p$ are sampled from three separate Beta prior distributions, and we update the posteriors of these distributions through variational approximation where hyper-parameters are decided by moment matching. Doing so, we can then define the reward as $R(s_t, a_t) = \mathbb{E}(h(P_{k_t}^{t+1}) - h(P_{k_t}^t))$, where P_k^t is the probability of the parameter θ_k taking on a value of at least 0.5 given the posterior of θ_k at time t, $h(x) = \max(x, 1 - x)$, and the expectation is taken over all possible label y_t observed after action a_t.
- **Constraint**: Different from the setting in the work by Chen, Lin, and Zhou (2013), as different actions imply different costs, we need to explicitly characterize

the budget constraint for our problem. Formally, the requester needs to ensure the budget constraint is satisfied. $\sum_{t=0}^{t_{\mathcal{B}}} c_s \mathbf{1}(x_t = 0) + c_p \mathbf{1}(x_t = 1) \leq \mathcal{B}$.

Proposed Algorithm. We adopt the method of Lagrangian multipliers to solve the above constrained optimization problem, which converts the problem of maximizing the total reward (i.e., $\sum_{t=0}^{t_{\mathcal{B}}} R(s_t, a_t)$) under the budget constraint into a simpler problem of maximizing the auxiliary function $\sum_{t=0}^{t_{\mathcal{B}}} R(s_t, a_t) - \lambda \sum_{t=0}^{t_{\mathcal{B}}} (c_s \mathbf{1}(x_t = 0) + c_p \mathbf{1}(x_t = 1))$. Notice this optimization problem is equivalent to solve a (unconstrained) Markov decision process where reward in each step is redefined as $R^{'}(s_t, a_t) = R(s_t, a_t) - \lambda(c_s \mathbf{1}(x_t = 0) + c_p \mathbf{1}(x_t = 1))$. We use the optimistic knowledge gradient technique introduced in (Chen, Lin, and Zhou 2013) to solve the optimal policy of this MDP, which produces a single-step look-ahead policy that maximizes the highest reward at each step. Note that in theory, we can compute the optimal value of λ for solving the constrained MDP. In practice, we have experimented with multiple different λ values and find that the choice of λ has limited influence on the performance of our algorithmic approach.

Evaluation of the Algorithmic Approach. We evaluated the effectiveness of our algorithmic approach using the real data that we collected in the $K = 20$ image labeling tasks of our experimental study. We set the cost of recruiting a single worker as $c_s = 1.0$, while we vary the cost of recruiting a pair of workers to work on a task with peer communication (i.e., a meta-worker) c_p from 1.6 to 2.4. The prior distribution for θ_k is set as Beta$(1, 1)$, where the prior distributions for α_s and α_p are all set to be Beta$(4, 1)$. For this evaluation, we only considered the final labels that workers in our experimental study submit in the *first two tasks* of the image labeling HIT[6]. Thus, when $a_t = (k_t, 0)$, we randomly sampled a label from Treatment 1 workers who had completed task k_t in their first two (independent) tasks, and when $a_t = (k_t, 1)$, we randomly sampled a label from Treatment 2 workers who had completed task k_t in their first two (discussion) tasks. We experimented with different values of λ and found that it had limited impact on the performance of the proposed approach. Thus, we set a fixed $\lambda = 0.01$ throughout the experiments.

The performance of our algorithmic approach is compared against the following baseline approaches:

- *Round robin*: when the requester decides which task to work on in a round robin fashion, and he always recruit a single worker to work on the task independently.
- *Single workers only*: when the requester always recruit single workers to work on tasks independently, although he optimally decides which task to work using our algorithm.

We conducted this evaluation on a wide range of budget level from 60 to 380 with an interval of 20. At each budget level, we implemented each of the decision-making strategies (our proposed approaches with different levels of c_p, and 2 baseline approaches) for 100 times, and we report the average level of overall accuracy the requester obtains across the 20 tasks when he exhausts the budget in Figure 3. It is clear from the figure that our proposed algorithmic approach outperforms the two baseline strategies when the requester always recruit workers to work on his tasks independently. Indeed, given the same budget level, the requester can always obtain a higher average accuracy across all his tasks following the policy generated by our algorithm.

[6]Recall that we set out to examine the effects of peer communication using the first two tasks in each HIT.

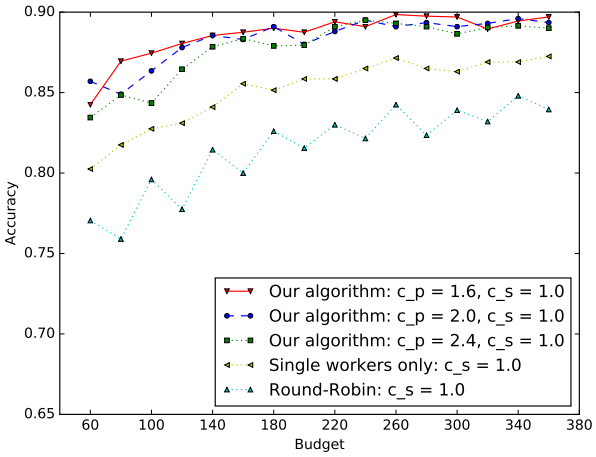

Figure 3: Evaluating the performance of the proposed approach on real datasets.

Conclusion

In this paper, we relax the common assumption of data independence in crowdsouring data collection. In particular we explore peer communication, in which a pair of crowd workers directly communicate when producing the data. We first examine the effects of peer communication on the quality of crowd work produced. Randomized experiments conducted on Amazon Mechanical Turk demonstrate that the work quality significantly improves in tasks with peer communication. We then study how to utilize peer communication in crowdsourcing data collection. In particular, we develop a general aggregation method based on weighted majority voting that can aggregate labels collected in tasks with and without peer communication. We derive how to optimally determine the weights of the voting rule to obtain a maximum likelihood estimator for the aggregation. With the data aggregation rule in place, we develop a constrained Markov decision process framework, built on top of the work by Chen, Lin, and Zhou (2013), which can optimally determine whether and when to deploy peer communication in crowdsourcing data collection. Experiments conducted on real data also demonstrate the advantage of our proposed algorithms over baseline approaches without considering peer communication.

Our results suggest the potential benefits of incorporating peer communication in crowdsourcing and provide a framework for better utilizing these benefits. We hope this work could open more discussions on designing and leveraging more complex, useful worker interactions to further enhance crowdsourcing.

References

[2017] Chang, J. C.; Amershi, S.; and Kamar, E. 2017. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI)*.

[2013] Chen, X.; Lin, Q.; and Zhou, D. 2013. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In *International Conference on Machine Learning (ICML)*.

[2008] Cholleti, S. R.; Goldman, S. A.; Blum, A.; Politte, D. G.; and Don, S. 2008. Veritas: Combining expert opinions without labeled data. In *Proceedings 20th IEEE international Conference on Tools with Artificial intelligence (ICTAI)*.

[2001] Crouch, C., and Mazur, E. 2001. Peer instruction: Ten years of experience and results. *Am. J. Phys.* 69(9):970–977.

[2013] Dai, P.; Lin, C. H.; Mausam; and Weld, D. S. 2013. Pomdp-based control of workflows for crowdsourcing. *Artif. Intell.* 202(1):52–85.

[1979] Dawid, A. P., and Skene, A. M. 1979. Maximum likeihood estimation of observer error-rates using the EM algorithm. *Applied Statistics* 28:20–28.

[1977] Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B* 39:1–38.

[2016] Drapeau, R.; Chilton, L. B.; Bragg, J.; and Weld, D. S. 2016. Microtalk: Using argumentation to improve crowdsourcing accuracy. In *Fourth AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*.

[2017] Gurari, D., and Grauman, K. 2017. Crowdverge: Predicting if people will agree on the answer to a visual question. In *Proceedings of the 2017 Conference on Human Factors in Computing Systems (CHI)*.

[2015] Ho, C.-J.; Slivkins, A.; Suri, S.; and Vaughan, J. W. 2015. Incentivizing high quality crowdwork. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*.

[2013] Ho, C.; Jabbari, S.; and Vaughan, J. W. 2013. Adaptive task assignment for crowdsourced classification. In *The 30th International Conference on Machine Learning (ICML)*.

[2010] Horton, J. J., and Chilton, L. B. 2010. The labor economics of paid crowdsourcing. In *Proceedings of the 11th ACM conference on Electronic commerce (EC)*.

[2003] Jin, R., and Ghahramani, Z. 2003. Learning with multiple labels. In *Advances in Neural Information Processing Systems (NIPS)*.

[2011a] Karger, D. R.; Oh, S.; and Shah, D. 2011a. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Proc. 49th Annual Conference on Communication, Control, and Computing (Allerton)*.

[2011b] Karger, D. R.; Oh, S.; and Shah, D. 2011b. Iterative learning for reliable crowdsourcing systems. In *The 25th Annual Conference on Neural Information Processing Systems (NIPS)*.

[2011] Khosla, A.; Jayadevaprakash, N.; Yao, B.; and Fei-Fei, L. 2011. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*.

[2011] Kittur, A.; Smus, B.; Khamkar, S.; and Kraut, R. E. 2011. Crowdforge: Crowdsourcing complex work. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[2016] Law, E.; Yin, M.; Goh, J.; Chen, K.; Terry, M. A.; and Gajos, K. Z. 2016. Curiosity killed the cat, but makes crowdwork better. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI)*.

[] Li, Q.; Ma, F.; Gao, J.; Su, L.; and Quinn, C. J. Crowdsourcing high quality labels with a tight budget. In *Proceedings of the ninth acm international conference on Web Search and Data Mining (WSDM)*.

[2010] Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2010. Turkit: Human computation algorithms on mechanical turk. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[2009] Mason, W., and Watts, D. 2009. Financial incentives and the "performance of crowds". In *Proceedings of the 1st Human Computation Workshop (HCOMP)*.

[2011] Noronha, J.; Hysen, E.; Zhang, H.; and Gajos, K. Z. 2011. Platemate: Crowdsourcing nutritional analysis from food photographs. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[2010] Raykar, V.; Yu, S.; Zhao, L.; Valadez, G.; Florin, C.; Bogoni, L.; and Moy, L. 2010. Learning from crowds. *Journal of Machine Learning Research* 11:1297–1322.

[2011] Rogstadius, J.; Kostakos, V.; Kittur, A.; Smus, B.; Laredo, J.; and Vukovic, M. 2011. An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. *ICWSM* 11:17–21.

[2015] Shah, N. B., and Zhou, D. 2015. Double or nothing: Multiplicative incentive mechanisms for crowdsourcing. In *The 29th Annual Conference on Neural Information Processing Systems (NIPS)*.

[2011] Shaw, A. D.; Horton, J. J.; and Chen, D. L. 2011. Designing incentives for inexpert human raters. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work (CSCW)*.

[2009] Whitehill, J.; Ruvolo, P.; Wu, T.; Bergsma, J.; and Movellan, J. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems (NIPS)*.

[2013] Yin, M.; Chen, Y.; and Sun, Y.-A. 2013. The effects of performance-contingent financial incentives in online labor markets. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI)*.