No Oops, You Won't Do It Again: Mechanisms for Self-correction in Crowdsourcing

Nihar B. Shah NIHAR@EECS.BERKELEY.EDU
Dept. of EECS, University of California, Berkeley

Dengyong Zhou DENGYONG.ZHOU@MICROSOFT.COM
Microsoft Research, Redmond

Abstract

Crowdsourcing is a very popular means of obtaining the large amounts of labeled data that modern machine learning methods require. Although cheap and fast to obtain, crowdsourced labels suffer from significant amounts of error, thereby degrading the performance of downstream machine learning tasks. With the goal of improving the quality of the labeled data, we seek to mitigate the many errors that occur due to silly mistakes or inadvertent errors by crowdsourcing workers. We propose a two-stage setting for crowdsourcing where the worker first answers the questions, and is then allowed to change her answers after looking at a (noisy) reference answer. We mathematically formulate this process and develop mechanisms to incentivize workers to act appropriately. Our mathematical guarantees show that our mechanism incentivizes the workers to answer honestly in both stages, and refrain from answering randomly in the first stage or simply copying in the second. Numerical experiments reveal a significant boost in performance that such "self-correction" can provide when using crowdsourcing to train machine learning algorithms.

1. Introduction

The emergence of deep learning and other complex machine learning tools have resulted in a need for huge amounts of labeled data (Raykar et al., 2010; Deng et al., 2009; Carlson et al., 2010). One of the most popular means of obtaining labeled data is crowdsourcing, where data is labeled by crowds of semi-skilled workers through the Internet typically in exchange from some monetary payments. Crowdsourcing is widely used in many real-world applications, and is particularly popular for collecting training labels for machine learning powered systems like web search engines (Burges et al., 2005; Alonso & Mizzaro, 2009; Kazai, 2011) or to supplement automated algorithms (Khatib et al., 2011; Lang & Rio-Ross, 2011; Von Ahn et al., 2008). The labels obtained from crowdsourcing, however, have significant amounts of error (Kazai et al., 2011; Vuurens et al., 2011; Wais et al., 2010), thereby degrading the performance of the machine learning algorithms that use this data downstream. Consequently, there is much emphasis on gathering higher quality labels, since a lower noise implies requirement of fewer labels for obtaining the same accuracy in practice.

In a study from a few years back, Kahneman & Frederick (2002) asked the following question to many participants: *"A bat and ball cost a dollar and ten cents. The bat costs a dollar more than the ball. How much does the ball cost?"* (See also The New Yorker (2012).) A large number of respondents gave an incorrect answer of "10 cents", including a majority of the students surveyed at Harvard University, Princeton University and MIT. Indeed, making silly mistakes is a part and parcel of being human. In several domains of science and technology that deal with humans, there are special provisions to mitigate the effects of such inadvertent errors (Dijkstra, 1979; Ayewah & Pugh, 2009; Aggarwal et al., 2013). In this work, we consider the problem of mitigating silly mistakes in crowdsourcing.

Unsurprisingly, the data obtained from crowdsourcing also suffers from several forms of inadvertent errors. Examples of such errors include those resulting from not following instructions properly (Gupta et al., 2012), misreading questions (Chros & Sundell, 2011), mistakes when entering solutions (Gupta et al., 2012), incorrect recollection (Lasecki et al., 2013), framing effects (Levin et al., 1998), satisficing (Krosnick, 1991), and many others (e.g., see Tversky & Kahneman 1974; Fleurbaay & Eveleigh 2012).

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

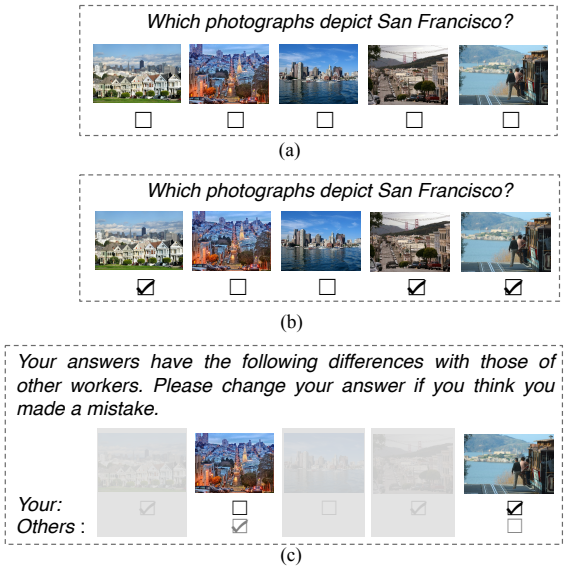

Figure 1: Illustration of self-correction in crowdsourcing. (a) Interface at the start of the task, comprising a set of 5 questions in this example. (b) In the first stage, the worker gives her answers to all the questions. (c) In the second stage, the worker is intimated of all the questions where her answers mismatched with a reference set of answers, and is allowed to change her answer to any of these questions

One approach towards mitigating these errors is to hire more workers to *independently* perform the same task and then aggregate their responses. This approach is the topic of several recent papers using of statistical aggregation algorithms to aggregate this data (e.g., see Raykar et al. (2010); Karger et al. (2011); Liu et al. (2012a); Zhou et al. (2015) and references therein). Most such existing work on crowdsourcing focuses on independent workers. Our approach is complementary to this line of work, and in fact, can nicely supplement these algorithms by providing them higher quality data at lower costs.

Specifically, we suggest a two stage "self-correction" setting. In the first stage, all workers are required to independently accomplish the crowdsourcing task which consists of a set of questions (see Figure 1a and Figure 1b). Standard crowdsourcing setups stop at this stage; we will term these settings as "single-stage" or "without self-correction" settings. Moving on, our self-correction setting is associated to a second stage in which the worker's answers are compared with a reference set of answers. For instance, the reference could be an aggregate of the responses of other workers who may have performed this task.[1] Alternatively, the reference could be the output of a (potentially noisy) machine learning algorithm. For every question that the worker answered differently from the reference (Figure 1), we offer her a second chance, and allow her to change her answer if she wishes to (see Figure 1c). In this paper we will often refer to the proposed self-correction setting as a "two-stage" setting.

The self-correction setting exploits the well-understood fact that reviewing a task for mistakes takes much less time than processing a new task (Gu, 2015; Haas et al., 2015). Moreover, incentivizing the workers to look at the feedback has an additional positive consequence of improving their understanding and performance in subsequent tasks (Jiang & Matsubara, 2012; Dow et al., 2011).

Contributions of the paper With a broad goal of designing ways to improve the quality of labeled data for machine learning algorithms, the specific contributions of this paper are three-fold. First, we introduce and mathematically formulate such a two-stage crowdsourcing setting for self-correction to mitigate inadvertent errors. Second are our primary contributions — theoretical results on mechanism design for the two-stage setting for self-correction. In particular, we consider tasks involving binary-choice questions. We design payment mechanisms to ensure that the workers are indeed incentivized to report truthfully in both stages of the task. (Any such mechanism is called "incentive compatible.") The problem of designing incentive-compatible mechanisms in this setting is challenging since on one hand we must ensure that the worker doesn't simply copy the reference answer, while on the other, we must also ensure that the worker cannot earn greater amounts by deliberately providing a false report in the first round and then changing her answer in the second round. The mechanism must accommodate all possible beliefs of the worker regarding the distribution of the true and the reference answers. We also theoretically prove attractive additional guarantees offered by our mechanism such as minimum slack (to be defined later) and uniqueness of the mechanism. Third, we conduct extensive numerical experiments that reveal how our self-correction setting can result in a significant improvement in the end-to-end accuracy of machine learning systems that use crowdsourced training data.

Related literature Our proposal to use other workers' answers as a reference is inspired from the benefits of communication studied in the literature on psychology. Several papers in the field show that interaction in a group can improve overall group performance in decision making (Kerr & Tindale, 2004; Kozlowski & Ilgen, 2006). However, the amount of shared information has to be limited and controlled, for example, by the so-called Delphi technique (Clayton, 1997; Rowe & Wright, 1999; Hasson et al., 2000). Otherwise, if the interaction in a group is rich, such as face-to-face discussions, it could lead to social bias

[1] The first worker who does this task will operate under the single-stage setting.

(Muchnik et al., 2013). The relationships among social pressure, attention to the stimulus, doubt about one's own judgment, and conformity have been thoroughly explored in psychology (Tesser et al., 1983), political science (Gerber et al., 2008), and consumer research (Bearden & Rose, 1990). These observations influenced the design of the proposed two-stage setting.

Our self-correction setting is related to but fundamentally different from the examination-verification methods in the crowdsourcing literature (Bernstein et al., 2010; Gao et al., 2011; Miller & Steyvers, 2011; Liu et al., 2012b; Su et al., 2012; Hara et al., 2013). Both allow limited information sharing among crowdsourcing workers rather than letting them work independently. However, in the examination-verification approaches, workers sequentially work on a task. Every worker examines the results from her predecessor and revises them when she disagrees. Consequently, workers do not have a chance for self-correction. Moreover, to the best of our our best knowledge, there is no incentive mechanism proposed for these examination-verification approaches. A worker may thus be incentivized to simply approve all the answers from her predecessor.

Several other works in the literature focus on design mechanisms for crowdsourcing (e.g., see Prelec 2004; Miller et al. 2005; Ranade & Varshney 2012; Shah & Zhou 2015; Shah et al. 2015 and references therein), a subset of which share our focus on the aspect of better labels for machine learning algorithms. However, these works all consider various forms of the single-stage setup. While the variants of the single-stage setup analyzed in these works are indeed non-trivial, as we will see in the sequel, the proposed two-stage self-correction setting on the other hand, comes with a set of very unique challenges.

Strictly proper scoring rules (Brier, 1950; Savage, 1971; Gneiting & Raftery, 2007) provide a general theory of mechanism design for eliciting private beliefs about the prediction of an event. Our setting of the design of payment mechanisms falls into the broad framework of strictly proper scoring rules.

2. Problem formulation

We begin with a formal description of the problem setting.

2.1. The task interface

There are N questions asked to a worker. We focus on binary-valued questions. The questions are objective, that is, for every question exactly one of the two options is correct. We will denote the two options for any questions as "A" and "B". The task proceeds in two stages:

- *Stage 1:* The worker is shown N questions. For every question, the worker selects either A or B as her answer.
- *Stage 2:* The worker's answers to all the questions are matched to a reference set of answers. For each question whose answer does not match, the worker is alerted about this mismatch and is given an option to either retain her own answer or copy the reference answer.

In order to evaluate the worker's performance, it is a common practice to include some "gold standard" questions in the task, that is, questions to which the answers known apriori to the mechanism designer. Specifically, we assume that the set of N questions contain G "gold standard" questions ($1 \leq G \leq N$), mixed uniformly at random in the entire set of questions. The worker does not know the identities of the gold standard questions. It is important to note that the gold standard questions are used only for evaluating the worker's performance at the end of the entire task, and are separate from the reference answer.

2.2. Beliefs of the worker

The worker has her own subjective probabilities with respect to the true answer and the reference answer for every question. During the first stage, from the *point of view of the worker*, for any question $i \in [N]$, let[2]

- $p_{A,i}$ be the probability that the correct answer is A
- $p_{B,i}$ $(= 1 - p_{A,i})$ be the probability that the correct answer is B
- $q_{A,i}$ be the probability that the reference answer is A
- $q_{B,i}$ $(= 1 - q_{A,i})$ be the probability that the reference answer is B.

In the second stage, the questions for which the worker's answers do not match the reference are displayed to the worker. The worker updates her subjective probabilities accordingly as, for any question $i \in [N]$ displayed in the second stage,

- $p'_{A|B,i}$ $(\leq p_{A,i})$ be the probability that the correct answer is A given that the reference answer was B
- $p'_{B|A,i}$ $(\leq p_{B,i})$ be the probability that the correct answer is B given that the reference answer was A.

We also define $p'_{A|A,i} = 1 - p'_{B|A,i}$ and $p'_{B|B,i} = 1 - p'_{A|B,i}$.

We make the standard game theoretic assumptions that the workers aim to maximize their expected payment, and that her beliefs about the different questions are independent. With respect to further rationality, we consider two types of workers:

[2] We adopt the standard notation of letting $[N]$ denote the set $\{1, \ldots, N\}$ for any positive integer N.

- *Fully rational:* The worker ensures her beliefs are restricted to obey the law of total probability

$$p_{A,i} = q_{A,i}p'_{A|A,i} + q_{B,i}p'_{A|B,i}, \tag{1a}$$
$$p_{B,i} = q_{A,i}p'_{B|A,i} + q_{B,i}p'_{B|B,i}, \tag{1b}$$

  for all i.

- *Partially rational:* The worker may only have a "bounded" view of the probabilities involved, in which case the worker may assume values of $p_{A,i}$, $q_{A,i}$, $p'_{A|B,i}$ and $p'_{B|A,i}$ without the restriction imposed in (1).

In this paper we will support both fully and partially rational workers. The mechanisms designed subsequently will be incentive-compatible for both these types of workers.

Finally note that the values of the worker's beliefs are, of course, unknown to us. The goal is to design mechanisms that are incentive compatible for arbitrary values of these beliefs, as formalized below.

2.3. Requirements

The goal is to design a payment mechanism that incentivizes the worker to act as follows. Consider any choice of a fixed threshold $T \in \left[\frac{1}{2}, 1\right)$. The choice of the threshold T is made by the system designer based on the application at hand, and in this paper we will assume that the threshold is given to us. For any question $i \in [N]$, for arbitrary values of the worker's beliefs, the worker should be incentivized to select her answers in the following manner.

- First stage: For every question $i \in [N]$, the worker should be incentivized to select the option that she thinks is most likely to be correct, namely

$$\text{select} \begin{cases} \text{option "A"} & \text{if } p_{A,i} > \frac{1}{2} \\ \text{option "B"} & \text{if } p_{A,i} < \frac{1}{2}. \end{cases}$$

- Second stage: For every question $i \in [N]$ that had a mismatch in the first stage, the worker should copy the reference answer if and only if she is really sure about the reference answer. Formally, if the worker selected option "A" in the first stage, then she should

$$\text{select} \begin{cases} \text{"Copy"} & \text{if } p'_{B|B,i} > T \\ \text{"Retain"} & \text{if } p'_{B|B,i} < T \end{cases},$$

  and if the worker selected option "B" in the first stage, then she should

$$\text{select} \begin{cases} \text{"Copy"} & \text{if } p'_{A|A,i} > T \\ \text{"Retain"} & \text{if } p'_{A|A,i} < T. \end{cases}$$

Observe that in our model, we have restricted T to take a value of $\frac{1}{2}$ or more.[3] When $T = \frac{1}{2}$, the setting reduces to the conventional setting requiring the worker to select the option she thinks is most likely to be correct. When T is chosen to be strictly greater than a half, the worker should copy the reference answer only if she is really sure. This choice helps avoid the bias of simply believing in the reference and copying it.

The worker's final performance is evaluated based on her responses to the G gold standard questions. The worker's selection for any question in the gold standard may get evaluated to one of six possibilities, denoted by $\{+\mathfrak{M}, -\mathfrak{M}, +\mathfrak{R}, -\mathfrak{R}, +\mathfrak{C}, -\mathfrak{C}\}$, and defined as:

- $+\mathfrak{M}$: Match in the first round, and correct
- $-\mathfrak{M}$: Match in the first round, and incorrect
- $+\mathfrak{R}$: Mismatch in the first round, retained in the second round, and correct
- $-\mathfrak{R}$: Mismatch in the first round, retained in the second round, and incorrect
- $+\mathfrak{C}$: Mismatch in the first round, copied in the second round, and correct
- $-\mathfrak{C}$: Mismatch in the first round, copied in the second round, and incorrect.

Here "match" and "mismatch" respectively stand for whether the answer to a question given by a worker is same as the answer to that question in the reference or not. The terms "correct" and "incorrect" respectively refer to whether the option selected by the worker was correct (that is, matched the gold standard) or not.

Let μ denote the maximum pay a worker can receive in this task. The value of μ should be chosen based on application-specific conditions such as the recommended hourly wage for the worker; in this paper, we assume that the value of μ is given to us. In accordance with the requirements of crowdsourcing platforms, we will also assume that the payments made to the workers are non-negative.

Given the notation introduced thus far, we can mathematically represent any payment mechanism as a function $f : \{+\mathfrak{M}, -\mathfrak{M}, +\mathfrak{R}, -\mathfrak{R}, +\mathfrak{C}, -\mathfrak{C}\}^G \to [0, \mu]$. Then by definition of the parameter μ, we have $\max f(\cdot) = \mu$.

As mentioned earlier in Section 2.2, we assume that the worker aims to maximize her expected payment. The expectation of the payment f is taken over the random distribution of the G gold standard questions among the N questions, and over the worker's uncertainties $\{p_{A,i}, p_{B,i}, q_{A,i}, q_{B,i}, p'_{A|B,i}, p'_{B|A,i}\}_{i\in[N]}$ about the correctness of her own answers and of the reference answers.

[3] Our results also extend to the case of $T < \frac{1}{2}$. However, we choose to omit this case since we are interested in eliminating the bias towards simply copying the reference answer, and hence restrict attention to only $T \geq \frac{1}{2}$ in the narrative.

The goal is to design a mechanism f such that its expected value (from the point of view of the worker) is *strictly* maximized when in both stages, the worker answers as per the requirements stated above. Any such mechanism is termed an "incentive-compatible" mechanism.

3. One Stage: Trivial Mechanism

To set the ball rolling, let us first consider the standard setting of a single stage, which is typical of the crowdsourcing setups of today. Under such a setting, the worker must answer all the questions, and the payment is made to the worker based on these answers (to the gold standard questions). The condition of incentive compatibility requires that for all the questions, the worker must be incentivized to select the option which she thinks is most likely to be correct, i.e., to incentivize the worker to choose option A if $p_{A,i} > p_{B,i}$ and B if $p_{A,i} < p_{B,i}$ for any question $i \in [N]$. Of course, the mechanism designer does not know the values of $\{p_{A,i}, p_{B,i}\}_{i\in[N]}$.

Proposition 1 (trivial)**.** *Consider any values M_+ and M_- such that $M_+ > M_- \geq 0$ and $GM_+ = \mu$. Letting C denote the number of questions in the gold standard answered correctly, the following mechanism is incentive compatible:*

$$\text{Payment} = (M_+C + M_-(G - C)).$$

Proposition 1 presents just one of the many mechanisms that can be constructed for the single stage setting, and it is trivial to construct mechanisms that are incentive compatible if there was only one stage. The situation, however, changes dramatically upon introduction of the second stage, as is discussed in the rest of this paper.

4. Two stages: Where Things Get Interesting

We now consider the two-stage setting of Section 2.

4.1. Impossibility of incentive compatible mechanisms

Unlike the multitude of mechanisms available in the single-stage setting (Section 3), we are hit with an immediate roadblock in the two-stage case.

Theorem 1. *For any values of $N \geq G \geq 1$ and $T \in (\frac{1}{2}, 1)$, there is no mechanism that is incentive compatible.*

In order to circumvent this impossibility theorem[4], we will make a mild relaxation to our requirements.

[4]The theorem considers $T > \frac{1}{2}$. When $T = \frac{1}{2}$, the mechanism in the proof of Theorem 2 below (with the associated parameter $\xi = 0$) is incentive compatible.

4.2. Relax: Incentive compatibility with margins

Given the impossibility result of Theorem 1, in this section, we make a relaxation to the requirements outlined earlier in Section 2. Recall that the aforementioned setting requires that in the first stage, for every question $i \in [N]$, the worker must be incentivized to

$$\text{select} \begin{cases} \text{option "A"} & \text{if } p_{A,i} > \frac{1}{2} \\ \text{option "B"} & \text{if } p_{A,i} < \frac{1}{2}. \end{cases}$$

We relax this requirement as: in the first stage, for every question $i \in [N]$, the worker must be incentivized to

$$\text{select} \begin{cases} \text{option "A"} & \text{if } p_{A,i} > \frac{1}{2} + \xi \\ \text{option "B"} & \text{if } p_{A,i} < \frac{1}{2} - \xi, \end{cases}$$

for some parameter $\xi > 0$ whose value will be specified later. Thus, the incentivization for the first stage is changed from a hard threshold at $\frac{1}{2}$ to an interval between $\frac{1}{2} - \xi$ and $\frac{1}{2} + \xi$. The new formulation does not impose any requirements in the first stage when the confidence of the worker is in the range $[\frac{1}{2} - \xi, \frac{1}{2} + \xi]$. The incentivization requirement in the second stage remains the same as before.

It turns out that with this relaxation, perhaps surprisingly, for every value of $\xi > 0$ there exist infinitely many incentive compatible mechanisms.

Theorem 2. *For every value of $N \geq G$, $T \in [\frac{1}{2}, 1)$ and $\xi > 0$, there exists a mechanism that is incentive compatible. Moreover, there exist infinitely many mechanisms and the number of degrees of freedom in choosing any mechanism grows exponentially in G.*

The proof of Theorem 2 is constructive, that is, it provides explicit constructions of incentive-compatible mechanisms for every value of ξ.

The parameter ξ represents the amount by which a mechanism is allowed to slack as compared to the guarantees required in Section 2. Consequently, we would like to keep the value of ξ small. But Theorem 2 guarantees the existence of incentive compatible mechanisms for any positive value of ξ, and furthermore, points to the existence of infinitely many mechanisms. This result thus raises the following two questions:

- What value of ξ should be chosen?
- For the chosen ξ, what mechanism should be used?

Given that the proof of Theorem 2 constructs an explicit class of mechanisms for use, one may then be tempted to simply pick an arbitrary value of ξ and an arbitrary mechanism from that class. In this paper, however, we will take a principled approach towards this choice.

4.3. No-free-lunch axiom and a unique mechanism

In this section, we identify a simple and naturally desirable condition for any mechanism, that will help us answer the two questions raised above. Specifically, we impose the following requirement on the payment mechanism, which we term the 'no-free-lunch' axiom.

Definition 1 (No-free-lunch axiom)**.** *If all the answers (in the gold standard) given by a worker are either wrong or copied then the worker should get a zero payment, unless all answers given by the worker are correct. More formally, we require* $f(\mathbf{x}) = 0 \;\; \forall \; \mathbf{x} \in \{-\mathfrak{M}, -\mathfrak{R}, +\mathfrak{C}, -\mathfrak{C}\}^G \backslash \{+\mathfrak{C}\}^G$.

The axiom is quite intuitive since if a worker gives only wrong answers or copies them from the reference, then these answers do not provide any new information to the mechanism designer.[5]

The no-free-lunch axiom stated above is a variant of the no-free-lunch axioms for other settings proposed in Shah & Zhou (2015); Shah et al. (2015). It is important to note that if the "zero payment" appears harsh, then one can replace the "zero" with any fixed positive value and all the results of this paper will continue to hold.

Given the natural requirement of the no-free-lunch axiom, in what follows, we will investigate the effects of this requirement under our self-correction setting.

Theorem 3. *For any values of* $N \geq G \geq 1$*, and any* $T \in \left[\frac{1}{2}, 1\right)$*, it is impossible to construct an incentive compatible mechanism satisfying the no-free-lunch axiom if* $\xi < \xi_{\min}$*, where* $\xi_{\min} \in \left(0, \frac{1}{2}\right)$ *is given by*

$$\xi_{\min} = \begin{cases} \frac{1}{2}\frac{1-T}{1+T} & \text{if } T \leq \frac{1}{\sqrt{2}} \\ \frac{1}{2}\left((2-T) - \sqrt{(5-T)(1-T)}\right) & \text{if } T \geq \frac{1}{\sqrt{2}} \end{cases}.$$

Theorem 3 thus prohibits the choice of any ξ below $\xi_{\min}$.

We now show that a slack of $\xi_{\min}$ is indeed feasible, i.e., it allows for incentive compatible mechanism(s) satisfying no-free-lunch. This helps *answer our first question on how to choose ξ: it is desirable to choose the smallest permissible value of the slack parameter, which turns out to be $\xi_{\min}$.* The rest of this section thus considers $\xi = \xi_{\min}$.

Consider the payment mechanism given in Algorithm 1.

As the following theorem shows, the proposed algorithm indeed works as desired.

Theorem 4. *For any choice of* $N \geq G \geq 1$*,* $T \in \left[\frac{1}{2}, 1\right)$ *and* $\xi = \xi_{\min}$*, the mechanism of Algorithm 1 satisfies the no-free-lunch axiom and is incentive compatible.*

[5]The exception of the case where all answers are correct is discussed subsequently in Section 4.4.

Algorithm 1 Incentive mechanism for self-correction

- Define function $\alpha : \{+\mathfrak{M}, -\mathfrak{M}, +\mathfrak{R}, -\mathfrak{R}, +\mathfrak{C}, -\mathfrak{C}\} \to \mathbb{R}_+$ as $\alpha(+\mathfrak{M}) = 1$, $\alpha(-\mathfrak{M}) = 0$, $\alpha(+\mathfrak{R}) = \frac{\frac{1}{2} - \xi_{\min}}{1-T}$, $\alpha(-\mathfrak{R}) = 0$, $\alpha(+\mathfrak{C}) = \frac{\frac{1}{2} - \xi_{\min}}{T}$ and $\alpha(-\mathfrak{C}) = 0$.
- If $(x_1, \ldots, x_G) \in \{+\mathfrak{M}, -\mathfrak{M}, +\mathfrak{R}, -\mathfrak{R}, +\mathfrak{C}, -\mathfrak{C}\}^G$ are the evaluations of the answers to the G questions in the gold standard, then the payment is

$$\text{Payment}(x_1, \ldots, x_G) = \kappa \prod_{i=1}^{G} \alpha(x_i)$$

where $\kappa = \mu \left(\max\left\{1, \frac{\frac{1}{2} - \xi_{\min}}{1-T}\right\}\right)^{-G}$.

It turns out that this mechanism is unique in the following sense.

Theorem 5. *For any* $N \geq G \geq 1$*,* $T \in \left[\frac{1}{2}, 1\right)$ *and* $\xi = \xi_{\min}$*, there is only one incentive-compatible mechanism satisfying the no-free-lunch axiom, and that is the mechanism of Algorithm 1.*

The uniqueness result of Theorem 5 thus *answers our second question about deciding which mechanism to choose.*

4.4. No stronger than no-free-lunch

The reader may have wondered about the "unless" clause in the definition of the no-free-lunch axiom (Definition 1). This section will investigate the implications of removing that clause. To this end, let us define a marginally stronger version of the no-free-lunch axiom.

Definition 2 (Strong no-free-lunch)**.** *If all the answers (in the gold standard) given by a worker are either wrong or copied, i.e, when the worker gives no correct answer on her own, then the worker should get a zero payment. More formally, we require* $f(\mathbf{x}) = 0 \; \forall \; \mathbf{x} \in \{-\mathfrak{M}, -\mathfrak{R}, +\mathfrak{C}, -\mathfrak{C}\}^N$.

Intuitively, if a worker's answers are all either wrong or simply copied then she is not contributing any new information. The strong no-free-lunch axiom is precisely the no-free-lunch axiom but without the 'unless' clause. The following theorem investigates this stronger requirement.

Theorem 6. *For any choice of* $N \geq G \geq 1$*,* $T \in \left[\frac{1}{2}, 1\right)$ *and* $\xi \in [0, \frac{1}{2})$*, there is no incentive-compatible mechanism satisfying the strong no-free-lunch condition.*

The result of this theorem thus justifies the inclusion of the 'unless' clause in the no-free-lunch axiom.

5. Numerical Experiments

In Section 4 we analytically proved the working and the optimality of our proposed mechanism, Algorithm 1, for the

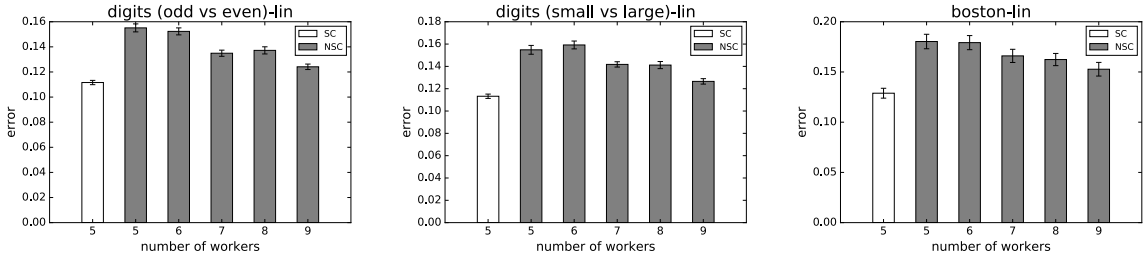

Figure 2: Error incurred by SVM with a linear kernel under the self-correction (SC) setting with 5 workers, compared to the error incurred under the standard setting with no self correction (NSC) with 5 to 9 workers.

two-stage setting. In this section we return to our primary hypothesis of the benefits of the two-stage self-correction setting, and via extensive numerical experiments, investigate the possible benefits of using a two-stage setting as compared to the standard one-stage setting without any self-correction. Such an examination is worthwhile since while the second stage would help to eliminate inadvertent errors and improve the quality of the data, it would also require each worker to spend more time on the task. In other words, for a fixed budget (under a fixed expected hourly wage), our two-stage setting trades off cleaner data with allowing for a slightly smaller number of workers. It turns out, as we will see below, that in machine learning systems that use crowdsourcing for labeled data, the self-correction setting results in a significant reduction in the end-to-end error rates as compared to the standard single-stage settings employed today.

Data We consider the labeling of the following two popular data sets: (a) UCI digits dataset (Lichman, 2013): Contains images of handwritten numeric digits from 0 to 9. We investigate two binary classification versions of this dataset: odd vs. even digits, and small values $0-4$ vs. large values $5-9$; (b) Boston housing dataset (Harrison & Rubinfeld; 1978): Contains information regarding housing in the area of Boston, USA. The binary classification problem is to predict whether the price of a house is greater than a certain value.

We then simulate the crowdsourced labeling procedure in the following manner. The number of workers hired in the two-stage self-correction setting and the standard single-stage setting may be different. In our simulations, the collection of workers are associated to a first-stage reliability parameter p and a second-stage improvement parameter q as follows. The workers have a reliability of p in the first stage, meaning that each worker, for each question, makes an error independently with probability $(1-p)$ in the first stage. In the second stage, the quality is assumed to improve by q due to self correction by the workers, that is, the reliability is $(p+q)$ at the end of the second stage. In the simulations, we investigate the effects of different values of p, q and the number of workers.

Machine learning algorithms We study the performance of two popular binary classification algorithms:

- support vector machine (SVM) with a linear kernel, and
- SVM with a radial basis function (RBF) kernel.

We perform the following operations separately for each of the two classification algorithms, for each of the three classification problems mentioned above, and for the two settings of with and without self-correction. The data is split into two equal halves, which are used for training and testing respectively. The labels for the training data are noisy, where the noise comes from the crowdsourced labelling described above. The test set is used to measure and compare the final performance of the classification algorithms, and is hence free of errors. The hyperparameters of the algorithms, including the regularization parameter and the kernel bandwidth, are chosen via 5-fold cross-validation.

Results In each of the plots to follow, each data point is averaged over 50 runs. We plot the results for SVM with linear kernel here in the main text, and noting that the results for the RBF kernel are almost identical to that for the linear kernel, we relegate the plots of SVM with RBF kernel to Appendix B.

We first investigate how the error under the self-correction setting compares with the error in the setting with no self-correction, for various amounts of redundancies in the task. More specifically, we fix the number of workers per question as 5 in the self-correction setting and vary the number of workers per question from 5 to 9 in the setting with no self-correction. For this set of experiments, we set $p = 0.6$ and $q = 0.15$. In each case, we use the aggregate of the worker's answers as training data for the two classification algorithms described earlier. Figure 2 plots the amount of error incurred by the SVM algorithm with the linear kernel. In each case, the performance of the algorithms when supplied the data from the self-correction setting outperforms the performance when data comes from the standard crowdsourcing setup with no self-correction. It is notewor-

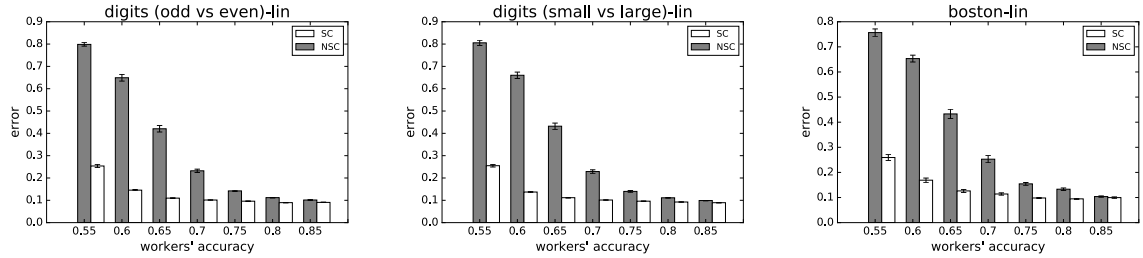

Figure 3: Error incurred by SVM with a linear kernel for different reliabilities (p) of the worker in the first stage. The no-self-correction (NSC) setting has 7 workers whereas the self-correction (SC) setting has only 5 workers.

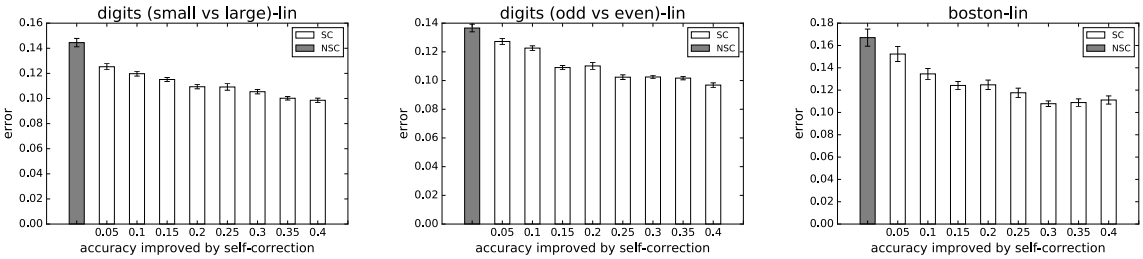

Figure 4: Error incurred by SVM with a linear kernel for different values of the improvement in accuracy (q) via self-correction. The no-self-correction (NSC) setting has 7 workers whereas the self-correction (SC) setting has only 5 workers.

thy that the self-correction setting shows an improved performance even when the number of workers in the standard setup is almost twice that in the self-correction setup.

Next, we compare the performance of the two settings for various values of the first-stage reliability p of the worker. To this end, we consider the self-correction setting with 5 workers per question and the setting with no self-correction having 7 workers per question. We vary the reliability p of each worker in the range 0.55 to 0.85, fixing $q = 0.15$. As before, the data obtained is employed to train an SVM algorithm with a linear kernel for the three datasets. The accuracy of the algorithm is shown in Figure 3. Observe that the self-correction setting consistently outperforms the standard setting with no self-correction. The improvement is particularly striking in high-noise conditions (i.e., when p is small).

Finally, we now compare the performance in these two settings when the second-stage accuracy parameter q is varied, keeping p fixed. In particular, we again consider the self-correction setting with 5 workers per question and the setting with no self-correction having 7 workers per question; we set $p = 0.6$ and vary q from 0.05 to 0.4. We observe (Figure 4) that even if the second stage offers marginal improvements (such as $q \leq 0.1$), we can still get significant gains from the two-stage setting as compared to a one-stage setting, despite the one stage setting having more workers.

All in all, the numerical experiments indicate significant improvements in the quality of the labels due to self-correction, and a corresponding increase in the accuracy of machine learning algorithms. Such improvements arise even in cases when the amount of self-correction may be quite small and when the setting without self correction has more workers than the setting with self correction.

6. Discussions

In this paper we proposed a two-stage setting for self-correction to overcome the various inadvertent errors that are observed widely in crowdsourcing. We showed the potential of such a self-correction setting via numerical experiments where we observed significant gains in the end-to-end performance of machine learning algorithms based on crowdsourced data. On the theoretical front, we investigated incentive mechanisms to ensure that workers report truthfully in both stages. (The modeling choices underlying the theory are discussed further in Appendix A.)

Our work leads to a number of interesting directions for future work. We addressed crowdsourcing tasks with binary-choice problems – such tasks are very popular in practice and quite challenging to analyze theoretically. We hope to use our results as building blocks for addressing more complex tasks. Second, our numerical experiments reveal that our proposed two-stage setup can offer significant gains as compared to standard single-stage setups. It remains to evaluate these mechanisms in real crowdsourcing platforms, which however, will necessitate sufficient training and exposure of the workers to this new setting.

References

Aggarwal, V., , Srikant, S., and Shashidhar, V. Principles for using machine learning in the assessment of open response items: Programming assessment as a case study. In *NIPS Workshop on Data Driven Education*, 2013.

Alonso, O. and Mizzaro, S. Can we get rid of TREC assessors? Using Mechanical Turk for relevance assessment. In *SIGIR Workshop on the Future of IR Evaluation*, pp. 15–16, 2009.

Ayewah, N. and Pugh, W. Using checklists to review static analysis warnings. In *Workshop on Defects in Large Software Systems (at ACM ISSTA)*, 2009.

Bearden, W. and Rose, R. Attention to social comparison information: An individual difference factor affecting consumer conformity. *Journal of Consumer Research*, 1990.

Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., Crowell, D., and Panovich, K. Soylent: a word processor with a crowd inside. In *UIST*, 2010.

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. Learning to rank using gradient descent. In *ICML*, 2005.

Carlson, A., Betteridge, J., Wang, R.and Hruschka Jr, E., and Mitchell, T. Coupled semi-supervised learning for information extraction. In *WSDM*, 2010.

Chros, O. and Sundell, S. Digitalkoot: Making old archives accessible using crowdsourcing. In *Human Computation*, 2011.

Clayton, M. J. Delphi: a technique to harness expert opinion for critical decision-making tasks in education. *Educational Psychology*, 17(4):373–386, 1997.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE CVPR*, 2009.

Dijkstra, E. W. On the foolishness of "natural language programming". In *Program Construction*. 1979.

Dow, S., Kulkarni, A., Bunge, B., Nguyen, T., Klemmer, S., and Hartmann, B. Shepherding the crowd: managing and providing feedback to crowd workers. In *CHI*, 2011.

Fleurbaay, E. and Eveleigh, A. Crowdsourcing: Prone to error? In *International Council on Archives*, 2012.

Gao, H., Barbier, G., and Goolsby, R. Harnessing the crowdsourcing power of social media for disaster relief. *IEEE Intelligent Systems*, 2011.

Gerber, A., Green, D., and Larimer, C. Social pressure and voter turnout: Evidence from a large-scale field experiment. *American Political Science Review*, 2008.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 2007.

Gu, L. Crowdsourcing for complex tasks: How to ensure quality output, September 2015. http://engineering.godaddy.com/crowdsourcing-for-complex-tasks-how-to- ensure-quality-output/.

Gupta, A., Thies, W., Cutrell, E., and Balakrishnan, R. mclerk: enabling mobile crowdsourcing in developing regions. In *SIGCHI*, 2012.

Haas, D., Ansel, J., Gu, L., and Marcus, A. Argonaut: macrotask crowdsourcing for complex data processing. *VLDB*, 2015.

Hara, K., Le, V., and Froehlich, J. Combining crowdsourcing and google street view to identify street-level accessibility problems. In *SIGCHI*, 2013.

Harrison, D. and Rubinfeld, D. Boston housing dataset. `http://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html`. Retrieved: October 13, 2015.

Harrison, D. and Rubinfeld, D. L. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, 5(1):81–102, 1978.

Hasson, F., Keeney, S., and McKenna, H. Research guidelines for the delphi survey technique. *Journal of advanced Nursing*, 32(4):1008–1015, 2000.

Jiang, H. and Matsubara, S. Improving crowdsourcing efficiency based on division strategy. In *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology*, 2012.

Kahneman, D. and Frederick, S. Representativeness revisited: Attribute substitution in intuitive judgment. *Heuristics and biases: The psychology of intuitive judgment*, 49, 2002.

Karger, D. R., Oh, S., and Shah, D. Iterative learning for reliable crowdsourcing systems. In *NIPS*, 2011.

Kazai, G. In search of quality in crowdsourcing for search engine evaluation. In *Advances in information retrieval*, 2011.

Kazai, G., Kamps, J., Koolen, M., and Milic-Frayling, N. Crowdsourcing for book search evaluation: impact of HIT design on comparative system ranking. In *ACM SIGIR*, 2011.

Kerr, N. and Tindale, R. Group performance and decision making. *Annual Review of Psychology*, 2004.

Khatib et al., F. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature structural & molecular biology*, 2011.

Kozlowski, S. W. and Ilgen, D. R. Enhancing the effectiveness of work groups and teams. *Psychological science in the public interest*, 7(3):77–124, 2006.

Krosnick, J. A. Response strategies for coping with the cognitive demands of attitude measures in surveys. *Applied cognitive psychology*, 5(3):213–236, 1991.

Lang, A. and Rio-Ross, J. Using Amazon Mechanical Turk to transcribe historical handwritten documents. *The Code4Lib Journal*, 2011.

Lasecki, W., Miller, C., and Bigham, J. Warping time for more effective real-time crowdsourcing. In *SIGCHI*, 2013.

Levin, I., Schneider, S., and Gaeth, G. All frames are not created equal: A typology and critical analysis of framing effects. *Organizational behavior and human decision processes*, 1998.

Lichman, M. UCI digits dataset (UCI machine learning repository). `http://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits`, 2013. Retrieved: October 13, 2015.

Liu, Q., Peng, J., and Ihler, A. Variational inference for crowdsourcing. In *NIPS*, 2012a.

Liu, X., Lu, M., Ooi, B. C., Shen, Y., Wu, S., and Zhang, M. Cdas: a crowdsourcing data analytics system. In *VLDB*, 2012b.

Miller, B. and Steyvers, M. The wisdom of crowds with communication. In *Conference of the Cognitive Science Society*, 2011.

Miller, N., Resnick, P., and Zeckhauser, R. Eliciting informative feedback: The peer-prediction method. *Management Science*, 51(9):1359–1373, 2005.

Muchnik, L., Aral, S., and Taylor, S. J. Social influence bias: A randomized experiment. *Science*, 2013.

Prelec, D. A Bayesian truth serum for subjective data. *Science*, 306(5695):462–466, 2004.

Ranade, G. and Varshney, L. To crowdsource or not to crowdsource. In *HCOMP workshop at AAAI*, 2012.

Raykar, V., Yu, S., Zhao, L., Valadez, G., Florin, C., Bogoni, L., and Moy, L. Learning from crowds. *JMLR*, 2010.

Rowe, G. and Wright, G. The delphi technique as a forecasting tool: issues and analysis. *International Journal of Forecasting*, 15(4):353–375, 1999.

Savage, L. J. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66(336):783–801, 1971.

Shah, N. B. and Zhou, D. Double or nothing: Multiplicative incentive mechanisms for crowdsourcing. In *NIPS*, 2015.

Shah, N. B., Zhou, D., and Peres, Y. Approval voting and incentives in crowdsourcing. In *ICML*, 2015.

Su, H., Deng, J., and Fei-Fei, L. Crowdsourcing annotations for visual object detection. In *Workshops at AAAI*, 2012.

Tesser, A., Campbell, J., and Mickler, S. The role of social pressure, attention to the stimulus, and selfdoubt in conformity. *European Journal of Social Psychology*, 1983.

The New Yorker. Why smart people are stupid. `http://www.newyorker.com/tech/frontal-cortex/why-smart-people-are-stupid`, June 2012.

Tversky, A. and Kahneman, D. Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131, 1974.

Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., and Blum, M. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 2008.

Vuurens, J., de Vries, A. P., and Eickhoff, C. How much spam can you take? An analysis of crowdsourcing results to increase accuracy. In *SIGIR Workshop on Crowdsourcing for Information Retrieval*, 2011.

Wais, P., Lingamneni, S., Cook, D., Fennell, J., Goldenberg, B., Lubarov, D., Marin, D., and Simons, H. Towards building a high-quality workforce with Mechanical Turk. *NIPS workshop on computational social science and the wisdom of crowds*, 2010.

Zhou, D., Liu, Q., Platt, J. C., Meek, C., and Shah, N. B. Regularized minimax conditional entropy for crowdsourcing. *arXiv preprint arXiv:1503.07240*, 2015.