Learning to Predict from Crowdsourced Data

Wei Bi [†] **Liwei Wang** [‡] **James T. Kwok** [†] **Zhuowen Tu** [*]
[†] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong
[‡] Department of Computer Science, University of Illinois at Urbana-Champaign, IL, United States
[*] Department of Cognitive Science, University of California San Diego, CA, United States

Abstract

Crowdsourcing services like Amazon's Mechanical Turk have facilitated and greatly expedited the manual labeling process from a large number of human workers. However, spammers are often unavoidable and the crowdsourced labels can be very noisy. In this paper, we explicitly account for four sources for a noisy crowdsourced label: worker's dedication to the task, his/her expertise, his/her default labeling judgement, and sample difficulty. A novel mixture model is employed for worker annotations, which learns a prediction model directly from samples to labels for efficient out-of-sample testing. Experiments on both simulated and real-world crowdsourced data sets show that the proposed method achieves significant improvements over the state-of-the-art.

1 INTRODUCTION

Supervised learning requires labels. However, the collection of labeled data from users is often expensive, tedious and time-consuming. Recently, the use of crowdsourcing allows this mundane process of obtaining manual labels from a great number of human workers to be greatly expedited. For example, in Amazon's Mechanical Turk (AMT), a "requester" can pose tasks known as HITs (Human Intelligence Tasks). Workers then choose to complete any of the existing HITs and get rewarded by a certain amount of monetary payment set by the requester. Researchers in different areas, such as computer vision (Sorokin and Forsyth, 2008) and natural language processing (Snow et al., 2008), have benefited from these crowdsourcing services and acquired labels for large data sets.

However, in practice, the crowdsourced labels are often noisy. On one hand, their quality depends on the labeling task. For example, if the labeling task is not well designed or not clearly described by the requester, the worker's motivation to participate may decrease, and the noisy level of the crowdsourced labels will increase (Zheng et al., 2011). Moreover, different labeling tasks can have different difficulties. If samples in one task are very challenging to annotate, the obtained crowdsourced labels may be less reliable (Whitehill et al., 2009; Yan et al., 2010; Zhou et al., 2012). On the other hand, workers' qualities can vary drastically and lead to different noise levels in their annotations. For example, their expertise differs due to their diverse knowledge backgrounds (Whitehill et al., 2009; Welinder et al., 2010). Moreover, their dedications to performing the task can also greatly affect their annotation accuracies. In the worst case, some workers may just randomly guess the labels without actually looking at the samples (Welinder et al., 2010). In particular, it is common to have "spammers", who provide wrong labels most of the time. The extraction of "true" labels from a large pool of crowdsourced labels is thus very important.

A popular and simple approach is to perform a majority vote on workers. However, it implicitly assumes that all workers are equally accurate, which is rarely the case in practice. It can also be misleading when there is a significant portion of spammers. To obtain a more accurate consensus, a number of algorithms have been proposed that model different aspects of the labeling noise (such as worker expertise and sample difficulty) (Whitehill et al., 2009; Welinder et al., 2010; Raykar and Yu, 2012; Liu et al., 2012; Zhou et al., 2012). Interested readers are referred to the recent survey in (Sheshadri and Lease, 2013). Yet, these models can only make estimations for samples with crowdsourced labels. For out-of-sample testing (i.e. prediction on an unseen test sample), the user has to first crowdsource its labels before these algorithms can be run.

To alleviate this problem, one can build a prediction model directly from the sample to the label. A popular approach is the two-coin model (Raykar et al., 2010). It assumes that each worker generates its label by flipping the ground-truth label with a certain probability. Depending on whether the true label being zero or one, the flipping probabilities are in general different. A prediction model is then built on the hidden "denoised" labels. This is further extended in

Table 1: Comparison between the existing methods and ours.

method	prediction model from samples to labels	worker expertise	sample difficulty	worker dedication
majority voting	×	×	×	×
Whitehill et al. (2009)	×	✓	✓	×
Welinder et al. (2010)	×	✓	×	×
Liu et al. (2012)	×	✓	×	×
Zhou et al. (2012)	×	✓	✓	×
Raykar and Yu (2012)	×	✓	×	×
Raykar et al. (2010)	✓	✓	×	×
Yan et al. (2010)	✓	✓	✓	×
Kajino et al. (2012)	✓	✓	×	×
Kajino et al. (2013)	✓	✓	×	×
proposed method	✓	✓	✓	✓

(Yan et al., 2010) by allowing the flipping probability to be different from sample to sample. Another approach is to formulate the crowdsourcing problem as a multitask learning problem (Evgeniou and Pontil, 2004). Each worker is considered a task, and the final prediction model is a linear combination of the worker models (Kajino et al., 2012, 2013). However, this may not be robust when many workers are spammers or incompetent.

In this paper, we propose a novel model for the generation of crowdsourced labels. Specifically, we assume that the label noise can come from four sources: (i) the worker is not an expert; (ii) the worker is not dedicated to the task; (iii) the worker's default label judgement is incorrect; and (iv) the sample is difficult. Note that some of these have been considered in the literature (Table 1). Moreover, they can be highly inter-correlated. For example, if a sample is easy, even an uncommitted non-expert can output the correct label. On the other side, if the sample is very difficult, even a dedicated expert can only rely on his default judgement. If his prior knowledge happens to be incorrect, the label will be wrong.

With these various factors, we employ a mixture model for the worker annotation of the crowdsourced data. If the worker is dedicated to the labeling task or if he considers the sample as easy, the corresponding label is generated according to his underlying decision function. Otherwise, the label is generated based on his default labeling judgement. To model sample difficulty, we use the usual intuition that a sample is difficult if it is close to the worker's underlying decision boundary, and vice versa. Obviously, we do not know in which way the worker generates the label of a sample. For inference, we use the expectation maximization (EM) algorithm (Dempster et al., 1977).

The rest of this paper is organized as follows. Section 2 presents our worker annotation model, and Section 3 presents the inference procedure. Experiments are presented in Section 4, and the last section gives some concluding remarks.

2 PROPOSED MODEL

In this paper, we assume that the crowdsourced task is a binary classification problem, with T workers and N query samples. The ith sample $\mathbf{x}^{(i)} \in \mathbb{R}^d$ is annotated by the set of workers $S_i \subseteq \{1, 2, \ldots, T\}$. The annotation provided by the tth worker (with $t \in S_i$) is denoted $y_t^{(i)} \in \{0, 1\}$.

2.1 GENERATION OF GROUND TRUTH

We assume that for each sample $\mathbf{x}^{(i)}$, its ground truth label $y^{*(i)} \in \{0, 1\}$ is generated by a logistic regression model with parameter $\mathbf{w}^*$. In other words, $y^{*(i)}$ follows the Bernoulli distribution

$$p(y^{*(i)} = 1 | \mathbf{w}^*, \mathbf{x}^{(i)}) = \sigma(\mathbf{w}^{*T} \mathbf{x}^{(i)}), \tag{1}$$

where $\sigma(z) = 1/(1 + \exp(-z))$ is the logistic function. To avoid over-fitting, we assume a normal prior on $\mathbf{w}^*$:

$$\mathbf{w}^* | \gamma \sim \mathcal{N}\left(\mathbf{0}, \frac{1}{\gamma}\mathbf{I}\right),$$

where $\gamma > 0$ is a constant (in the experiments, this is tuned by the validation set). Other priors can also be readily added. For example, if $\mathbf{w}^*$ is expected to be sparse, the Laplace prior can be used instead.

As will be seen later, training the model only requires access to the features but not the ground-truth labels. This is more realistic in many crowdsourced applications, as the features can often be readily extracted using standard unsupervised feature extraction.

2.2 WORKER ANNOTATION: EXPERTISE AND DEDICATION

For worker t, we assume that his failure in correctly annotating $\mathbf{x}^{(i)}$ is due to two reasons according to his dedication to the queried sample $\mathbf{x}^{(i)}$. First, he may have tried to annotate with the best effort, but still fails because his expertise is not strong enough. We model this by assuming

that worker t's annotation $y_t^{(i)}$ follows a similar Bernoulli distribution as (1):

$$p(y_t^{(i)} = 1|\mathbf{w}_t, \mathbf{x}^{(i)}) = \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)}), \quad (2)$$

where $\mathbf{w}_t$ is the worker's "estimation" of $\mathbf{w}^*$, and is sampled from the following normal distribution

$$\mathbf{w}_t|\mathbf{w}^*, \delta_t \sim \mathcal{N}(\mathbf{w}^*, \delta_t^2 \mathbf{I}). \quad (3)$$

A small δ_t means that $\mathbf{w}_t$ is likely to be close to $\mathbf{w}^*$, and thus worker t is an expert, and vice versa. When no additional information on the worker'expertise is available, a uniform hyperprior on $\{\delta_t\}_{t=1}^T$ can be used.

The second reason for worker t's failure in correctly annotating $\mathbf{x}^{(i)}$ is simply that he is not dedicated to the task and has not even looked at $\mathbf{x}^{(i)}$. In this case, he randomly annotates according to some default judgement. This can be modeled by another Bernoulli distribution:

$$p(y_t^{(i)} = 1|b_t) = b_t, \quad (4)$$

where $b_t \in [0, 1]$. Again, when no additional information on the worker's default labeling judgement are available, a uniform prior on $\{b_t\}_{t=1}^T$ will be used.

Combining these two causes, we have

$$p(y_t^{(i)}|\mathbf{x}^{(i)}, \mathbf{w}_t, b_t, z_t^{(i)}) = p(y_t^{(i)}|\mathbf{x}^{(i)}, \mathbf{w}_t)^{z_t^{(i)}} p(y_t^{(i)}|b_t)^{(1-z_t^{(i)})}, \quad (5)$$

where $z_t^{(i)} \in \{0, 1\}$ determines whether (3) or (4) should be used to generate $y_t^{(i)}$. Intuitively, an expert worker should have an accurate prediction model (δ_t is small), and be dedicated to the task ($z_t^{(i)} = 1$ on most $\mathbf{x}^{(i)}$'s); whereas a spammer either has a large δ_t or $z_t^{(i)} = 0$ most of the time.

2.3 INCORPORATING SAMPLE DIFFICULTY

The difficulty of a sample can greatly affect the annotation quality (Whitehill et al., 2009; Yan et al., 2010; Zhou et al., 2012). If a sample is vaguely described or too hard, even an expert may have to make a random guess and thus acts as if he has not looked at the sample. On the contrary, if a sample is very easy, even a spammer (especially the lazy ones) can quickly make a correct decision.

To model this effect on worker t, we incorporate the difficulty of $\mathbf{x}^{(i)}$ into the modeling of $z_t^{(i)}$. Intuitively, if $\mathbf{x}^{(i)}$ is difficult to annotate, $z_t^{(i)}$ should be close to 0. From (5), the annotation made is then independent of the decision model of worker t. To measure sample difficulty, we use the popular notion that worker t will perceive $\mathbf{x}^{(i)}$ as difficult if it is close to his decision boundary (Tong and Koller, 2002; Dong et al., 2013; Welinder et al., 2010). Thus, we arrive at the following Bernoulli distribution on $z_t^{(i)}$:

$$p(z_t^{(i)} = 1|\mathbf{x}^{(i)}, \mathbf{w}_t, \lambda_t) = 2\sigma\left(\lambda_t \frac{\|\mathbf{w}_t{}^T \mathbf{x}^{(i)}\|^2}{\|\mathbf{w}_t\|^2}\right) - 1. \quad (6)$$

Here, $\frac{\|\mathbf{w}_t{}^T \mathbf{x}^{(i)}\|}{\|\mathbf{w}_t\|}$ is the distance of $\mathbf{x}^{(i)}$ from worker t's decision boundary $\mathbf{w}_t{}^T \mathbf{x} = 0$, and $\lambda_t \geq 0$ models the sensitivity of worker t's annotation to sample difficulty. Depending on each worker's expertise (as reflected by his $\mathbf{w}_t$), one worker may consider sample $\mathbf{x}^{(i)}$ difficult while another worker may consider it easy. Moreover, a small λ_t makes an easy sample (with a large $\frac{\|\mathbf{w}_t{}^T \mathbf{x}^{(i)}\|}{\|\mathbf{w}_t\|}$) look difficult and worker t will rely more on his default judgement, and vice versa. As we are only interested in the value of $\lambda_t \frac{\|\mathbf{w}_t{}^T \mathbf{x}^{(i)}\|^2}{\|\mathbf{w}_t\|^2}$ in (6), to simplify inference, we reparameterize (6) as

$$p(z_t^{(i)} = 1|\mathbf{x}^{(i)}, \mathbf{w}_t, \lambda_t) = 2\sigma(\lambda_t \|\mathbf{w}_t{}^T \mathbf{x}^{(i)}\|^2) - 1. \quad (7)$$

After obtaining $\mathbf{w}_t$, the sensitivity of worker t's annotation to sample difficulty can be recovered as $\lambda_t \|\mathbf{w}_t\|^2$.

A graphical model representation for the complete model is shown in Figure 1.

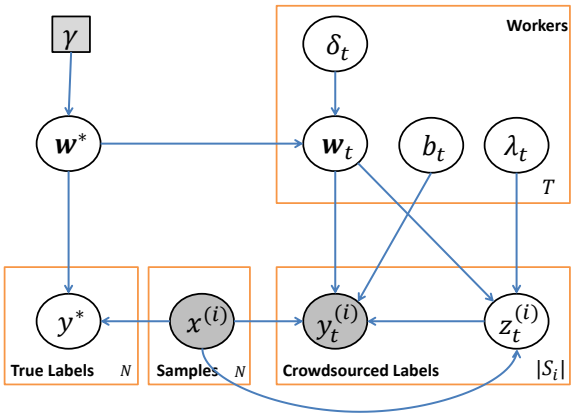


Figure 1: The proposed model incorporating sample difficulty and two sources for worker annotation.

2.4 EXTENSIONS

When the crowdsourced task is a multiclass classification problem, one can simply replace the Bernoulli distribution with a multinomial distribution. Similarly, for regression problems, the normal distribution can be used instead.

For ease of exposition, we use the linear logistic regression model in (1) and (2). This has also been used in most previous works (Raykar et al., 2010; Kajino et al., 2012). It can be easily replaced by any binary classifier. For example, to use a nonlinear kernelized version, one can replace $\mathbf{w}^{*T} \mathbf{x}^{(i)}$ in (1) by $\sum_{j=1}^N \alpha^{*(j)} k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$, where $k(\cdot, \cdot)$ is an appropriate kernel function. Similarly, $\mathbf{w}_t^T \mathbf{x}^{(i)}$ in (2) is replaced by $\sum_{j=1}^N \alpha_t^{(j)} k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$, where $\boldsymbol{\alpha}_t = [\alpha_t^{(1)}, \ldots, \alpha_t^{(N)}]$ serves as worker t's "estimation"

of the ground truth $\boldsymbol{\alpha}^* = [\alpha^{*(1)}, \ldots, \alpha^{*(N)}]^T$. Analogous to (3), we can assume that each $\boldsymbol{\alpha}_t$ is sampled from $\mathcal{N}(\boldsymbol{\alpha}^*, \delta_t^2\mathbf{I})$.

3 INFERENCE

In this section, we use the Expectation Maximization (EM) algorithm (Dempster et al., 1977) to obtain the model parameters [1] $\Theta = \{\mathbf{w}^*, \{\mathbf{w}_t\}_{t=1}^T, \{\delta_t\}_{t=1}^T, \{b_t\}_{t=1}^T, \{\lambda_t\}_{t=1}^T\}$. Let the samples $\mathbf{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ be independent. By treating $\mathbf{Y} = \{y_t^{(i)}\}$ as the observed data and $\mathbf{Z} = \{z_t^{(i)}\}$ as the missing data, the complete data likelihood can be written as

$$\begin{aligned}
&\mathcal{L}(\mathbf{Y}, \mathbf{Z}) \\
&= p(\mathbf{Y}, \mathbf{Z}|\mathbf{X}, \Theta) \\
&= p(\mathbf{Y}|\mathbf{Z}, \mathbf{X}, \{\mathbf{w}_t, b_t\}_{t=1}^T) p(\mathbf{Z}|\mathbf{X}, \{\mathbf{w}_t, \lambda_t\}_{t=1}^T) \\
&= \prod_{i=1}^{N} \prod_{t \in S_i} p(y_t^{(i)}|z_t^{(i)}, \mathbf{w}_t, \mathbf{x}^{(i)}, b_t) p(z_t^{(i)}|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t), \quad (8)
\end{aligned}$$

by assuming that the workers annotate independently. The posterior of Θ is then

$$\begin{aligned}
&p(\mathbf{w}^*, \{\mathbf{w}_t\}_{t=1}^T, \{\delta_t\}_{t=1}^T, \{\lambda_t\}_{t=1}^T, \{b_t\}_{t=1}^T | \mathbf{X}, \mathbf{Y}, \mathbf{Z}) \\
&\propto \mathcal{L}(\mathbf{Y}, \mathbf{Z}) p(\mathbf{w}^*) \prod_{t=1}^{T} p(\mathbf{w}_t|\mathbf{w}^*, \delta_t) p(\delta_t) p(\lambda_t) p(b_t) \quad (9)
\end{aligned}$$

3.1 E-STEP

Taking the log of (8), we have

$$\begin{aligned}
&\log \mathcal{L}(\mathbf{Y}, \mathbf{Z}) \\
&= \sum_{i=1}^{N} \sum_{t \in S_i} \Big(z_t^{(i)} \log p(y_t^{(i)}|\mathbf{w}_j, \mathbf{x}^{(i)}) p(z_t^{(i)} = 1|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) \\
&\quad + (1 - z_t^{(i)}) \log p(y_t^{(i)}|b_t) p(z_t^{(i)} = 0|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) \Big).
\end{aligned}$$

The expected value of $z_t^{(i)}$, denoted $\bar{z}_t^{(i)}$, is

$$\bar{z}_t^{(i)} = \frac{1}{Q_t^{(i)}} p(y_t^{(i)}|\mathbf{w}_t, \mathbf{x}^{(i)}) p(z_t^{(i)} = 1|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t),$$

where $Q_t^{(i)} = p(z_t^{(i)} = 1|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) p(y_t^{(i)}|\mathbf{w}_t, \mathbf{x}^{(i)}) + p(z_t^{(i)} = 0|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) p(y_t^{(i)}|b_t)$.

As can be seen, whether $\bar{z}_t^{(i)}$ is close to 1 is affected by both the sample difficulty (i.e., $p(z_t^{(i)} = 1|\mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t)$) and the confidence of $y_t^{(i)}$ generated from the current estimated function $\mathbf{w}_t$ (i.e., $p(y_t^{(i)}|\mathbf{w}_t, \mathbf{x}^{(i)})$).

[1] In the kernelized version, $\Theta = \{\boldsymbol{\alpha}^*, \{\boldsymbol{\alpha}_t\}_{t=1}^T, \{\delta_t\}_{t=1}^T, \{b_t\}_{t=1}^T, \{\lambda_t\}_{t=1}^T\}$ and the EM procedure is similar. In particular, the M-step updates $\boldsymbol{\alpha}^*$ and $\{\boldsymbol{\alpha}_t\}_{t=1}^T$ as $\mathbf{w}^*$ and $\{\mathbf{w}_t\}_{t=1}^T$.

3.2 M-STEP

Here, we use alternating minimization. At each step, one variable is minimized while the other variables are fixed.

- $\mathbf{w}_t$'s: From (9), the various $\mathbf{w}_t$'s can be learned independently. The optimization subproblem for $\mathbf{w}_t$ is

$$\begin{aligned}
\min_{\mathbf{w}_t} \quad & \frac{1}{\delta_t^2}\|\mathbf{w}_t - \mathbf{w}^*\|^2 - \sum_{i:t \in S_i} \Big(\bar{z}_t^{(i)} y_t^{(i)} \log \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)}) \\
& + \bar{z}_t^{(i)} (1 - y_t^{(i)}) \log(1 - \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)}) \\
& + \bar{z}_t^{(i)} \log(2\sigma(\lambda_t \|{\mathbf{w}_t}^T \mathbf{x}^{(i)}\|^2) - 1) \\
& + (1 - \bar{z}_t^{(i)}) \log(2 - 2\sigma(\lambda_t \|{\mathbf{w}_t}^T \mathbf{x}^{(i)}\|^2)) \Big).
\end{aligned}$$

  This can be maximized by gradient descent, and the gradient w.r.t. $\mathbf{w}_t$ is

$$\begin{aligned}
&\frac{2}{\delta_t^2}(\mathbf{w}_t - \mathbf{w}^*) - \sum_{i:t \in S_i} \Big(\bar{z}_t^{(i)} (y_t^{(i)} - \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)})) \mathbf{x}^{(i)} \\
&+ \frac{(\bar{z}_t^{(i)} - 2\sigma(v_t^{(i)}) + 1)\sigma(v_t^{(i)}) \lambda_t {\mathbf{w}_t}^T \mathbf{x}^{(i)} \mathbf{x}^{(i)}}{2\sigma(v_t^{(i)}) - 1} \Big),
\end{aligned}$$

  where $v_t^{(i)} = \lambda_t \|{\mathbf{w}_t}^T \mathbf{x}^{(i)}\|^2$.

- $\mathbf{w}^*$: The optimization subproblem for $\mathbf{w}^*$ is

$$\min_{\mathbf{w}^*} \sum_{t=1}^{T} \frac{1}{\delta_t^2}\|\mathbf{w}_t - \mathbf{w}^*\|^2 + \gamma\|\mathbf{w}^*\|^2,$$

  with the closed-form solution

$$\mathbf{w}^* = \frac{\sum_{t=1}^T \frac{1}{\delta_t^2} \mathbf{w}_t}{\gamma + \sum_{t=1}^T \frac{1}{\delta_t^2}}. \quad (10)$$

  Note that $\mathbf{w}^*$ is a weighted average of all the $\mathbf{w}_t$'s, with contributions from the experts (those with small δ_t's) weighted heavier.

- δ_t: The optimization subproblem for δ_t is

$$\begin{aligned}
&\min_{\delta_t} \ \frac{1}{\delta_t^2}\|\mathbf{w}_t - \mathbf{w}^*\|^2 + \log\det(\delta_t^2 \mathbf{I}) \\
&= \min_{\delta_t} \ \frac{1}{\delta_t^2}\|\mathbf{w}_t - \mathbf{w}^*\|^2 + 2d \log \delta_t,
\end{aligned}$$

  where $\mathbf{I}$ is the $d \times d$ identity matrix, and d is the number of input features. By setting its derivative w.r.t. δ_t to 0, we obtain

$$\delta_t = \frac{1}{\sqrt{d}}\|\mathbf{w}_t - \mathbf{w}^*\|.$$

- b_t: The optimization subproblem is

$$\max_{b_t} \sum_{i:t \in S_i} (1 - \bar{z}_t^{(i)})(y_t^{(i)} \log b_t + (1 - y_t^{(i)}) \log(1 - b_t)).$$

By setting its derivative w.r.t. b_t to 0, we have

$$\sum_{i:t\in S_i}(1-\bar{z}_t^{(i)})\left(\frac{y_t^{(i)}}{b_t}-\frac{1-y_t^{(i)}}{1-b_t}\right)=0.$$

Rearranging gives the closed-form solution

$$b_t=\frac{\sum_{i:t\in S_i}(1-\bar{z}_t^{(i)})y_t^{(i)}}{\sum_{i:t\in S_i}(1-\bar{z}_t^{(i)})}.$$

Recall that $\bar{z}_t^{(i)}\in[0,1]$ is the expectation of $y_t^{(i)}$ generated by worker t's default judgement. Hence, b_t is simply the average of worker t's labels that are generated by his default judgement.

- $\{\lambda_t\}_{t=1}^T$: The optimization subproblem is

$$\max_{\lambda_t}\sum_{i:t\in S_i}\bar{z}_t^{(i)}\log(2\sigma(\lambda_t\|{\mathbf{w}_t}^T\mathbf{x}^{(i)})\|^2)-1)$$
$$+(1-\bar{z}_t^{(i)})\log(2\text{–}2\sigma(\lambda_t\|{\mathbf{w}_t}^T\mathbf{x}^{(i)}\|^2)).$$

Again, this can be solved by projected gradient (as $\lambda_t\geq 0$), with the gradient w.r.t. λ_t given by

$$\sum_{i:t\in S_i}\frac{(\bar{z}_t^{(i)}-2\sigma(v_t^{(i)})+1)\sigma(v_t^{(i)})\|{\mathbf{w}_t}^T\mathbf{x}^{(i)}\|^2}{2\sigma(v_t^{(i)})-1}.$$

4 EXPERIMENTS

In this section, we perform two sets of experiments to evaluate the performance of the proposed method. Section 4.2 simulates a crowdsourced environment with synthetic workers using a standard benchmark data set; while Section 4.3 uses data sets with real labels crowdsourced from the AMT.

4.1 SETUP

The proposed model will be compared with the following groups of algorithms:

1. Algorithms that learn prediction models directly from samples to labels (Table 1). In particular, we will compare with
   - MTL: The multitask formulation in (Kajino et al., 2012). Each worker is considered as a task, and the prediction model is a rescaled average of all the learned worker models.
   - RY: The two-coin model in (Raykar et al., 2010). It considers the annotation generated by flipping the ground truth label with a certain biased probability.
   - YAN: This model is proposed in (Yan et al., 2010), and an extension of (Raykar et al., 2010). Its flipping probability is sample-specific and varies with sample difficulty. However, unlike ours, it does not have a clear connection with the worker's decision function.
2. Algorithms that do not learn a prediction model from samples to labels (Table 1). In particular, we will compare with
   - GLAD (Whitehill et al., 2009) [2]: It models each sample's difficulty level and each worker's expertise.
   - CUBAM (Welinder et al., 2010) [3]:: It considers sample competence, worker expertise and bias.
   - MV : Majority voting, a popular baseline which essentially treats all the workers as equally accurate.

   For prediction on an unseen test sample, these algorithms have to first crowdsource its labels. To avoid this problem, we will proceed as follows: (i) Estimate the "true" labels of the training samples using each of these algorithms; (ii) Use the estimated labels to train a logistic regression model; (iii) Use the trained regression model to make predictions on the test samples.
3. We also include an ideal baseline (Ideal), which is a logistic regression model trained using the training samples with ground truth labels.

For performance evaluation, we follow (Raykar et al., 2010) and report the area under ROC curve (AUC). The ROC curve is obtained by varying the prediction threshold. Parameters in all the models are tuned by a validation set (which is constructed by using 20% of the training data). With the chosen parameters, a prediction model is then learned using all the training data.

4.2 UCI DATA SET

Following (Kajino et al., 2012), we use the red wine data in the UCI Wine-Quality data set[4]. There are a total of 1,599 samples, each with 11 features. The original multiclass labels are binarized such that samples with quality levels below 7 are labeled as 0, and 1 otherwise. 70% of the samples are randomly chosen for training, and the remaining 30% for testing. To reduce statistical variability, results are averaged over 5 repetitions.

[2]Code is from `http://mplab.ucsd.edu/~jake/`

[3]Code is from `http://www.vision.caltech.edu/welinder/cubam.html`

[4]`http://archive.ics.uci.edu/ml/datasets/Wine+Quality`

Table 2: Testing AUCs on the wine data set. The best results and those that are not statistically worse (according to the paired t-test with p-value less than 0.05) are in bold.

	#workers	proposed	MTL	RY	YAN	GLAD	CUBAM	MV	Ideal
set 1	20	**0.81** $\pm$ 0.01	0.79 $\pm$ 0.04	0.38 $\pm$ 0.04	0.49 $\pm$ 0.03	0.52 $\pm$ 0.02	0.66 $\pm$ 0.05	0.48 $\pm$ 0.06	0.87 $\pm$ 0.02
	40	**0.79** $\pm$ 0.01	0.75 $\pm$ 0.07	0.51 $\pm$ 0.01	0.53 $\pm$ 0.03	0.51 $\pm$ 0.03	0.58 $\pm$ 0.04	0.34 $\pm$ 0.02	0.87 $\pm$ 0.03
set 2	20	**0.81** $\pm$ 0.01	**0.80** $\pm$ 0.01	0.50 $\pm$ 0.03	0.49 $\pm$ 0.03	0.49 $\pm$ 0.05	0.52 $\pm$ 0.04	0.49 $\pm$ 0.01	0.87 $\pm$ 0.01
	40	0.73 $\pm$ 0.04	**0.76** $\pm$ 0.02	0.49 $\pm$ 0.01	0.50 $\pm$ 0.03	0.50 $\pm$ 0.03	0.54 $\pm$ 0.03	0.50 $\pm$ 0.03	0.79 $\pm$ 0.02
set 3	20	**0.80** $\pm$ 0.01	**0.82** $\pm$ 0.01	**0.80** $\pm$ 0.03	**0.78** $\pm$ 0.03	**0.84** $\pm$ 0.05	**0.84** $\pm$ 0.04	0.48 $\pm$ 0.03	0.84 $\pm$ 0.01
	40	**0.80** $\pm$ 0.04	**0.82** $\pm$ 0.02	**0.80** $\pm$ 0.01	0.76 $\pm$ 0.04	**0.84** $\pm$ 0.05	**0.84** $\pm$ 0.03	0.59 $\pm$ 0.03	0.85 $\pm$ 0.02

4.2.1 Generation of Labels

We generate three sets of simulated crowdsourced labels based on different model assumptions:

- Set 1: The crowdsourced labels are generated using the proposed annotation process. The "optimal" $\mathbf{w}^*$ is obtained by training a logistic regression model on all the training and test samples. We generate different numbers (20 and 40) of noisy workers. For each worker, we generate $\mathbf{w}_t$ as in (3) with different settings of δ_t's:
  1. $\frac{1}{4}$ of the workers have $\delta_t = 10$ (high expertise);
  2. $\frac{1}{2}$ of the workers have $\delta_t = 100$ (moderate expertise); and
  3. $\frac{1}{4}$ of the workers have $\delta_t = 1000$ (low expertise).

  Sample difficulty is generated as in Section 2.3:

  1. For the expert workers, we set $\lambda_t = 10,000$, and so most of the samples appear easy;
  2. For workers with moderate expertise, set $\lambda_t = 100$; and
  3. For workers with low expertise, set $\lambda_t = 1$ (and so most of the samples appear difficult).

  For each sample i, we set $z_t^{(i)} = 1$ with probability given in (7). If $z_t^{(i)} = 1$, $y_t^{(i)}$ is labeled 1 with probability defined in (2); otherwise, $y_t^{(i)}$ is always labeled 1 (i.e., b_t in (4) is set to 1).

  In summary, $\frac{1}{4}$ of the workers are experts, $\frac{1}{2}$ of them are non-experts but not very noisy; while the remaining $\frac{1}{4}$ are very noisy workers.

- Set 2: The crowdsourced labels are generated using the MTL assumption in (Kajino et al., 2012). Specifically, from the $\mathbf{w}_t$ generated in Set 1, we generate $y_t^{(i)} = 1$ with probability $\sigma(\mathbf{w}_t^T \mathbf{x}^{(i)})$.

- Set 3: The crowdsourced labels are generated using the two-coin assumption in (Raykar et al., 2010). For worker t, let α_t (resp. β_t) be the probability that a ground truth label with value 1 (resp. 0) is flipped.
  1. For $\frac{1}{4}$ of the workers, we set $\alpha_t = \beta_t = 0.05$ (experts);
  2. For $\frac{1}{2}$ of the workers, set $\alpha_t = \beta_t = 0.25$ (non-experts but not very noisy); and
  3. For $\frac{1}{4}$ of the workers, set $\alpha_t = \beta_t = 0.55$ (very noisy workers).

4.2.2 Results on ROC Curves

Figure 2 shows the obtained testing ROC curves (with each point averaged over the five repetitions). The corresponding averaged AUC values are shown in Table 2. As can be seen, the proposed model performs well under various noise generation scenarios.

On Set 1, since the labels are generated using the proposed annotation process, the proposed method performs significantly better than the others as expected. MTL is the second best, as it also builds a prediction model for each worker. RY, YAN, GLAD, CUBAM and MV perform poorly, as their model assumptions are very different from the underlying data generation process.

On Set 2, MTL is the best. The proposed model also yields comparable performance; while the others do not perform well.

On Set 3, MTL, RY, GLAD, CUBAM and the proposed method have comparable performance. Their performance gaps with Ideal are also quite small, which is consistent with the results in (Kajino et al., 2012). As various methods can perform well here, it suggests that the noise generated by the two-coin model is easier to remove than those in the previous two settings.

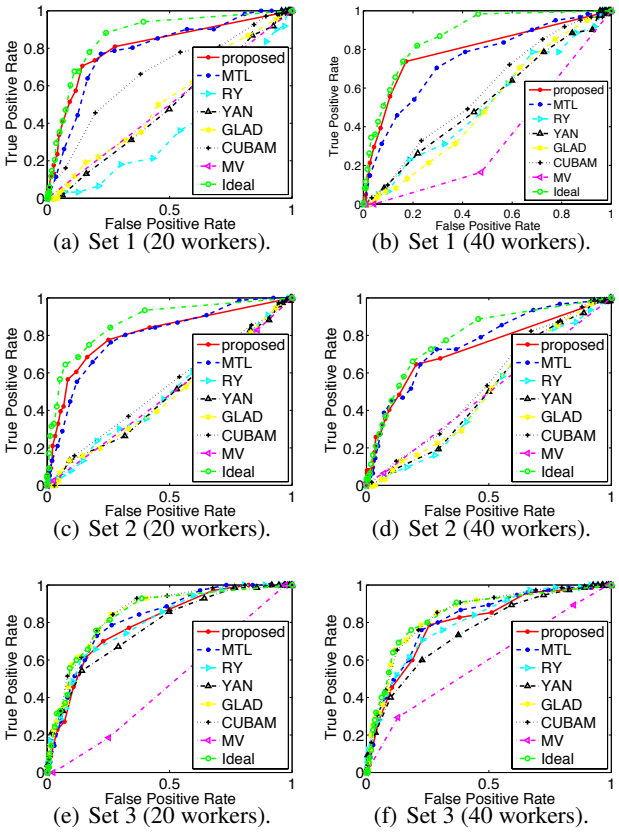
4.2.3 Separating Experts from Noisy Workers

In this section, we examine the proposed model's ability to separate experts from noisy workers using the two criteria: worker expertise and worker dedication. Because of the lack of space, we will only show results (averaged over the 5 repetitions) on Set 1.

First, we check if the proposed model can detect workers with high expertise. Figures 3(a) and 3(b) show the contri-

bution of $\mathbf{w}_t$ in $\mathbf{w}^*$ in (10) (i.e., $\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$). As can be seen, all the nonzero contributions are from the experts, while the other workers are barely used.

Next, we check if the proposed model can find the dedicated workers. Recall that for experts, most of his $z_t^{(i)}$'s should be close to 1; while most of the $z_t^{(i)}$'s for non-dedicated workers are close to 0. Figures 3(c) and 3(d) show the value of $\hat{z}_t = \sum_{i:t\in S_i} \bar{z}_t^{(i)}$ for each worker. As expected, the $\hat{z}$'s of experts are large; while those of the others are usually much smaller (especially for the noisy workers).

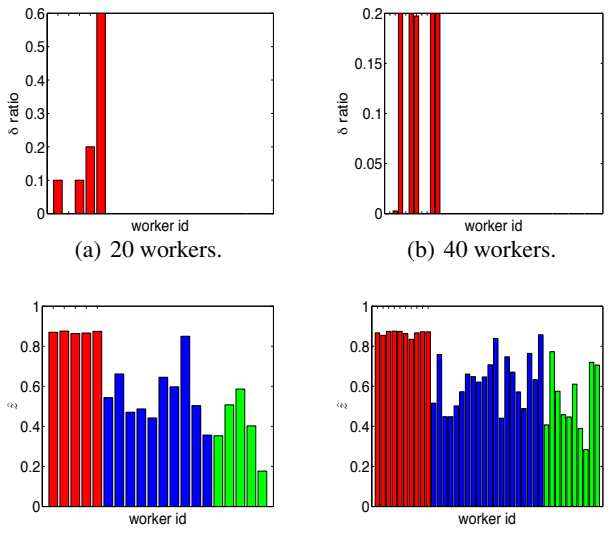


(a) Set 1 (20 workers). (b) Set 1 (40 workers).

(c) Set 2 (20 workers). (d) Set 2 (40 workers).

(e) Set 3 (20 workers). (f) Set 3 (40 workers).

Figure 2: Testing ROC curves on the wine data set.



(a) 20 workers. (b) 40 workers.

(c) 20 workers. (d) 40 workers.

Figure 3: Worker expertise and worker dedication on the Set 1 data. 3(a) and 3(b): Contribution of each worker t towards $\mathbf{w}^*$ ($\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$). 3(c) and 3(d): Average $\hat{z}$'s of the workers. Columns in red/blue/green correspond to experts/non-experts/noisy workers. In 3(a) and 3(c): workers 1-5 are experts; 6-15 are non-experts; and 16-20 are noisy workers. In 3(b) and 3(d): workers 1-10 are experts; 11-30 are non-experts ; and 31-40 are noisy workers.

4.3 DATA SETS CROWDSOURCED FROM AMT

4.3.1 Data Collection and Feature Extraction

For better performance evaluation, it is desirable for the data set to satisfy the following three conditions: (i) It is labeled by a sufficient number of workers so that workers with different expertise and dedications are all involved; (ii) Each worker labels a sufficient amount of data so that one can reliably model the annotating behavior of each worker ; (iii) The ground truth labels are provided. To our best knowledge, very few crowdsourced data sets meet all these requirements. Thus, in the following, we build a crowdsourced data set based on the Stanford Dog data set[5] (Khosla et al., 2011). It contains images of 120 breeds (categories) of dogs collected from the ImageNet[6] (Deng et al., 2009).

For an image, its raw pixel representation is very high-dimensional and also sensitive to image changes such as scales, object locations, illuminations. Consequently, various image features have been studied by the computer vision community to better represent the image from low level (e.g. SIFT (Lowe, 1999)) to mid-level descriptors (Wang et al., 2012). In this experiment, we extract 4,096-dimensional features from images using the DeCAF (deep convolutional activation feature) algorithm (Donahue et al., 2014). These features are outputs from the intermediate layers of a pre-trained deep convolutional neural network (Krizhevsky et al., 2012). It has been shown that they can be used as generic representations for various vision tasks, and have achieved good performance even when combined with simple linear classifiers (Donahue et al., 2014).

[5] http://vision.stanford.edu/aditya86/ImageNetDogs/

[6] http://www.image-net.org/

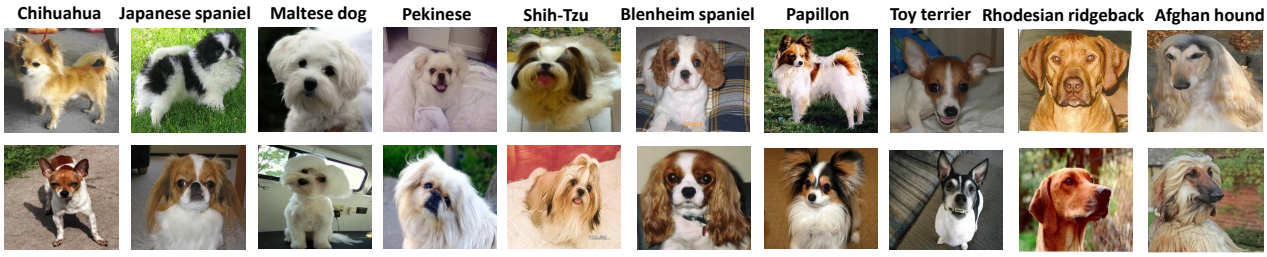


Figure 4: Sample images of the 10 dog categories.

4.3.2 Setup

We select the 10 categories that are most difficult to classify (Khosla et al., 2011) (Figure 4). For each category, images belonging to this category are taken as positive samples; while images from the other categories are treated as negative samples. Some statistics of the data sets are shown in Table 3. The constructed data sets are then randomly split into HITs on the AMT. Each HIT contains 50 images and is labeled by 6 workers. There are a total of 65 HITS and 21 workers over the 10 categories.

Table 3: Statistics on the dog data sets.

data set	#positive sample	#negative sample	avg #samples per worker
Chihuahua	142	157	85
Japanese spaniel	142	157	85
Maltese dog	142	163	85
Pekinese	142	163	85
Shih-Tzu	142	157	83
Blenheim spaniel	142	207	89
Papillon	142	175	92
Toy terrier	142	175	86
Rhodesian ridgeback	142	207	88
Afghan hound	142	207	89

For each category, we randomly use 50% of the samples for training, and the rest for testing. To reduce statistical variability, results are averaged over 5 repetitions.

4.3.3 Results on ROC Curves

The ROC curves are shown in Figure 5, and the corresponding AUC values in Table 4. As can be seen, the proposed method yields the highest AUC on all 10 categories. It is then followed by CUBAM, GLAD, RY and YAN, which are very competitive on some categories. MTL can sometimes achieve good performance (e.g., Blenheim spaniel), but are often much inferior. Overall, the simple MV is the worst.

4.3.4 Experts vs Noisy Workers

As in Section 4.2.3, we examine the obtained δ_t's and $\bar{z}_t$'s on the Chihuahua, Japanese spaniel and Maltese dog categories. As the real experts and noisy workers are not known, we assume that workers with high overall accura-

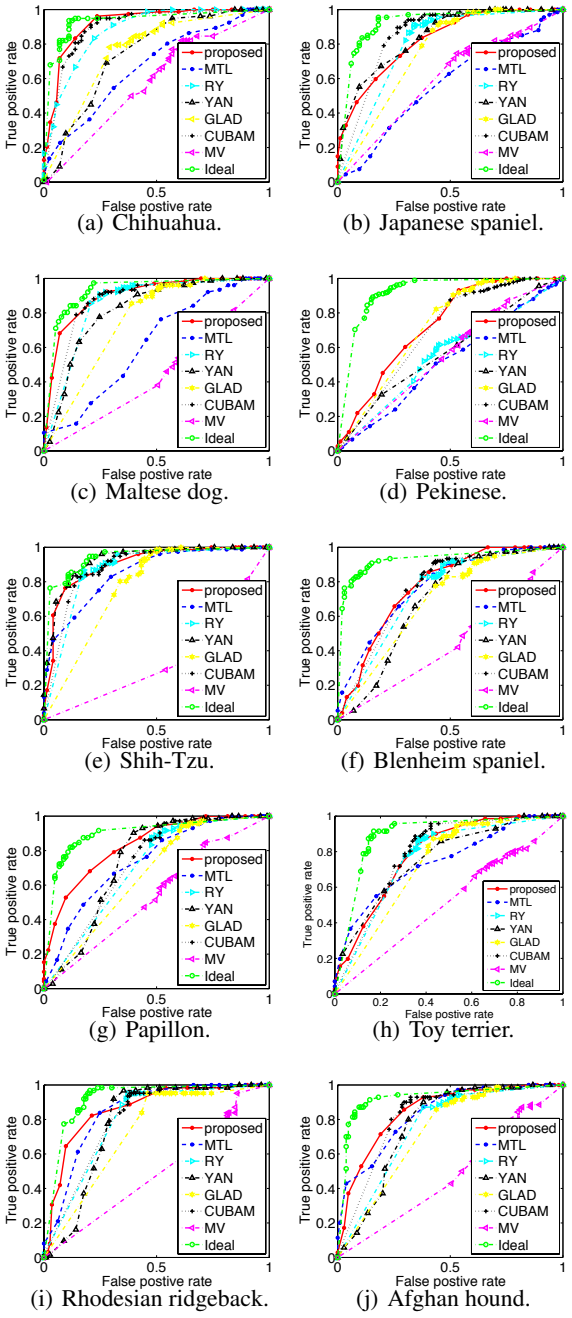


Figure 5: Testing ROC curves of the dog data sets.

Table 4: Testing AUCs on the dog data sets. The best results and those that are not statistically worse (according to the paired t-test with p-value less than 0.05) are in bold.

data set	proposed	MTL	RY	YAN	GLAD	CUBAM	MV	Ideal
Chihuahua	**0.92** $\pm$ 0.02	0.67 $\pm$ 0.01	0.88 $\pm$ 0.04	0.74 $\pm$ 0.08	0.76 $\pm$ 0.06	0.90 $\pm$ 0.02	0.58 $\pm$ 0.11	0.94 $\pm$ 0.02
Japanese spaniel	**0.83** $\pm$ 0.01	0.57 $\pm$ 0.01	0.80 $\pm$ 0.03	**0.84** $\pm$ 0.04	0.75 $\pm$ 0.04	**0.85** $\pm$ 0.03	0.60 $\pm$ 0.05	0.92 $\pm$ 0.01
Maltese dog	**0.90** $\pm$ 0.01	0.62 $\pm$ 0.05	0.85 $\pm$ 0.02	0.82 $\pm$ 0.03	0.76 $\pm$ 0.05	0.87 $\pm$ 0.03	0.43 $\pm$ 0.02	0.93 $\pm$ 0.02
Pekinese	**0.73** $\pm$ 0.05	0.53 $\pm$ 0.02	0.60 $\pm$ 0.03	0.58 $\pm$ 0.04	**0.72** $\pm$ 0.05	**0.72** $\pm$ 0.03	0.56 $\pm$ 0.09	0.92 $\pm$ 0.01
Shih-Tzu	**0.90** $\pm$ 0.02	0.85 $\pm$ 0.03	0.87 $\pm$ 0.04	**0.93** $\pm$ 0.03	0.77 $\pm$ 0.03	0.88 $\pm$ 0.03	0.35 $\pm$ 0.08	0.94 $\pm$ 0.03
Blenheim spaniel	**0.78** $\pm$ 0.03	**0.78** $\pm$ 0.03	0.74 $\pm$ 0.02	0.69 $\pm$ 0.05	0.69 $\pm$ 0.07	**0.77** $\pm$ 0.03	0.45 $\pm$ 0.03	0.93 $\pm$ 0.03
Papillon	**0.83** $\pm$ 0.03	0.74 $\pm$ 0.07	0.70 $\pm$ 0.05	0.74 $\pm$ 0.04	0.66 $\pm$ 0.04	0.72 $\pm$ 0.04	0.53 $\pm$ 0.06	0.90 $\pm$ 0.03
Toy terrier	**0.79** $\pm$ 0.02	0.75 $\pm$ 0.01	0.76 $\pm$ 0.03	0.76 $\pm$ 0.03	0.73 $\pm$ 0.02	**0.79** $\pm$ 0.04	0.51 $\pm$ 0.05	0.89 $\pm$ 0.03
Rhodesian ridgeback	**0.86** $\pm$ 0.04	0.85 $\pm$ 0.03	0.79 $\pm$ 0.02	0.78 $\pm$ 0.02	0.73 $\pm$ 0.05	0.79 $\pm$ 0.04	0.50 $\pm$ 0.05	0.92 $\pm$ 0.01
Afghan hound	**0.85** $\pm$ 0.02	0.83 $\pm$ 0.02	0.76 $\pm$ 0.01	0.77 $\pm$ 0.01	0.73 $\pm$ 0.04	0.81 $\pm$ 0.04	0.47 $\pm$ 0.05	0.93 $\pm$ 0.01

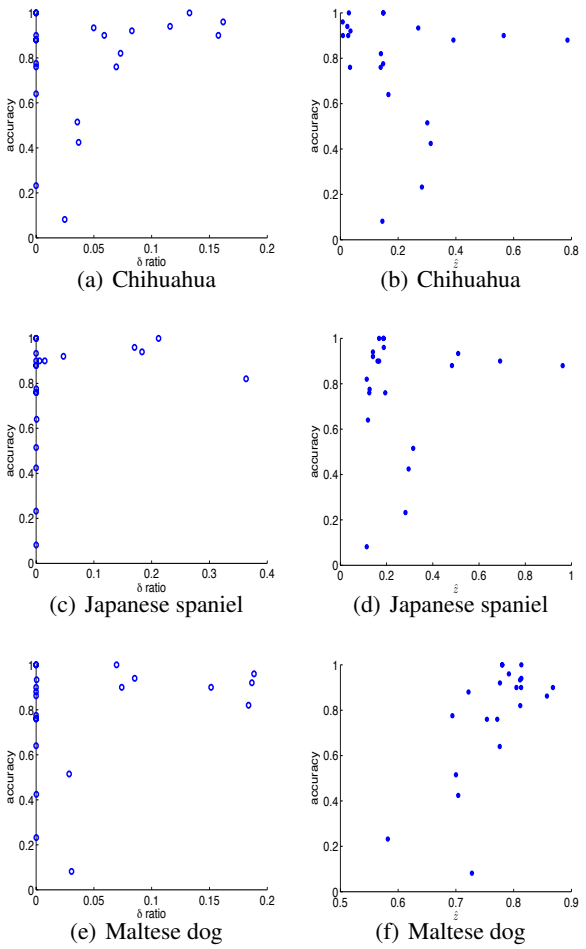
cies (that are computed based on both the training and test samples) are experts. In Figures 6(a),(c) and (e), we first plot the overall accuracies versus average weighting of the workers ($\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$) over five repetitions. As can be seen, workers with high weights, which are detected as experts in our model, generally have high overall accuracies. Next, we plot the overall accuracies versus average $\hat{z}_t$'s of the workers over five repetitions (Figures 6(b),(d) and (e)). Workers with high average $\hat{z}_t$'s are detected as dedicated workers and those with low average $\hat{z}_t$'s as lazy workers. As shown, the detected dedicated workers generally have high overall accuracies.

5 CONCLUSION

In this paper, we proposed a new model for crowdsourced labels that can perform out-of-sample prediction effectively. We observe that the worker's expertise and dedication to the task greatly affect the labeling process. We employed a mixture of distributions to model the annotation process: one models the worker's expertise and the other one depicts worker's labeling judgement with his random guess. We showed that this model can be easily extended to account for sample difficulty. The proposed model can be solved by the simple EM algorithm. Experiments on both UCI and real-world crowdsourced data sets demonstrate that the proposed method has significant improvements over other state-of-the-art approaches.

Acknowledgements

This research was supported in part by the Research Grants of the Hong Kong Special Administrative Region (Grant 614513), IIS-1360566 (NSF IIS-1216528) and NSF IIS-1360568 (IIS-0844566). Also, we thank Jiajun Wu for helping collecting part of the AMT data sets.



(a) Chihuahua (b) Chihuahua (c) Japanese spaniel (d) Japanese spaniel (e) Maltese dog (f) Maltese dog

Figure 6: Results on the Chihuahua, Japanese spaniel and Maltese dog data sets. Figures 6(a), 6(c) and 6(e): Overall accuracies vs average weighting of the workers ($\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$); Figures 6(b),6(d) and 6(f): Overall accuracies vs average $\hat{z}_t^{(i)}$'s of all workers.

References

A. P. Dempster, N. M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B*, pages 1–38, 1977.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL, USA, 2009.

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, Er. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the International Conference on Machine Learning*, pages 647–655, Beijing, China, 2014.

J. Dong, W. Xia, Q. Chen, J. Feng, Z. Huang, and S. Yan. Subcategory-aware object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 827–834, Portland, OR, USA, June 2013.

T. Evgeniou and M. Pontil. Regularized multi–task learning. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 109–117, Seattle, WA, USA, 2004.

H. Kajino, Y. Tsuboi, and H. Kashima. A convex formulation for learning from crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Toronto, Canada, 2012.

H. Kajino, Y. Tsuboi, and H. Kashima. Clustering crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Bellevue, WA, USA, 2013.

A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *the Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, Lake Tahoe, NV, USA, 2012.

Q. Liu, J. Peng, and A. Ihler. Variational inference for crowdsourcing. In *Advances in Neural Information Processing Systems 25*, pages 701–709, Lake Tahoe, NV, USA, 2012.

D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1150–1157, Ft. Collins, CO, USA, 1999.

V. C. Raykar and S. Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *Journal of Machine Learning Research*, 13:491–518, 2012.

V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *Journal of Machine Learning Research*, 99:1297–1322, 2010.

A. Sheshadri and M. Lease. SQUARE: A benchmark for research on computing crowd consensus. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, pages 156–164, Palm Springs, CA, USA, 2013.

R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast – but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii, 2008.

A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, Anchorage, AK, USA, 2008.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Joural of Machine Learning Research*, 2:45–66, 2002.

L. Wang, Y. Li, J. Jia, J. Sun, D. Wipf, and J. M. Rehg. Learning sparse covariance patterns for natural scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2767–2774, Providence, RI, USA, 2012.

P. Welinder, S. Branson, P. Perona, and S. J. Belongie. The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems 22*, pages 2424–2432, Vancouver, Canada, 2010.

J. Whitehill, T.-F. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22*, pages 2035–2043, Vancouver, Canada, 2009.

Y. Yan, R. Rosales, G. Fung, M. W. Schmidt, G. H. Valadez, L. Bogoni, L Moy, and J. G. Dy. Modeling annotator expertise: Learning when everybody knows a bit of something. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 932–939, Chia Laguna, Italy, 2010.

H. Zheng, D. Li, and W. Hou. Task design, motivation, and participation in crowdsourcing contests. *International Journal of Electronic Commerce*, 15(4):57–88, 2011.

D. Zhou, J. C. Platt, S. Basu, and Y. Mao. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems 25*, pages 2204–2212, Lake Tahoe, NV, USA, 2012.