Supervised Collective Classification for Crowdsourcing

Pin-Yu Chen*, Chia-Wei Lien†, Fu-Jen Chu‡, Pai-Shun Ting*, Shin-Ming Cheng§
*Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, USA
{pinyu, paishun}@umich.edu
†Amazon Corporate LLC, chiaweil@amazon.com
‡Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, USA
fujenchu@gatech.edu
§Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, smcheng@mail.ntust.edu.tw

***Abstract*—Crowdsourcing utilizes the wisdom of crowds for collective classification via information (e.g., labels of an item) provided by labelers. Current crowdsourcing algorithms are mainly unsupervised methods that are unaware of the quality of crowdsourced data. In this paper, we propose a supervised collective classification algorithm that aims to identify reliable labelers from the training data (e.g., items with known labels). The reliability (i.e., weighting factor) of each labeler is determined via a saddle point algorithm. The results on several crowdsourced data show that supervised methods can achieve better classification accuracy than unsupervised methods, and our proposed method outperforms other algorithms.**

I. Introduction

In recent years, collective decision making based on the wisdom of crowds has attracted great attention in different fields [1], particularly for social networking empowered technology [2]–[5] such as the trust-based social Internet of Things (IoT) paradigm [6]–[8]. Collective classification leverages the wisdom of crowds to perform machine learning tasks by acquiring multiple labels from crowds to infer groundtruth label. For instance, websites such as *Galaxy Zoo* asks visitors to help classify the shapes of galaxies, and *Stardust@home* asks visitors to help detect interstellar dust particles in astronomical images. In addition, new business model based on crowdsourcing [9] has emerged in the past few years. For instance, *Amazon Mechanical Turk (MTurk)* and *CrowdFlower* provide crowdsourcing services with cheap prices. For *MTurk*, a minimum of 0.01 US dollar is paid to a labeler/worker when she makes a click (i.e., generates a label) on an item. An illustrating figure can be found in Fig. 1.

Despite its cheap costs for acquiring labels, one eminent challenge for collective classification lies in dealing with these massive yet potentially incorrect labels provided by labelers. These incorrect labels may hinder the accuracy of collective classification when unsupervised collective classification methods (e.g., majority vote) are used for crowdsourcing. Unsupervised collective classification using the expectation maximization (EM) algorithm [10] is firstly proposed in [11]. A refined EM algorithm is then proposed in [12], which is shown to outperform majority vote. In [13], a minimax entropy regularization approach is proposed to minimize the Kullback- Leibler (KL) divergence between the probability generating function of the observed data and the true labels. Some data selection heuristics are proposed to identify high-quality labels/labelers for collective classification based on weighted majority votes [14], [15].

By allowing a fairly small amount of items with known labels for collective classification, it is shown in [13], [16]–[18] that supervised collective classification can improve the classification accuracy within affordable costs. Typical supervised classification algorithms include binary support vector machine [19], multi-class support vector machine [20], naive Bayes sampler [13], [17], [21], and multi-class Adaboost [22].

This paper provides an overview of the aforementioned methods and our goal is to propose a supervised collective classification algorithm that assigns weights to each labeler based on the accuracy of their labels. The weights are determined by solving a saddle point algorithm and they reflect the reliability of labelers. In addition to crowdsourcing, the proposed method can be applied to other communication paradigms such as mobile sensing and cooperative wireless network by assigning more weights to reliable users. For performance evaluation, we compare supervised and unsupervised collective classification algorithms on several crowdsourced datasets, which include a canonical benchmark dataset and the exam datasets that we collected from exam answers from junior high and high school students in Taiwan[1]. The results show that the proposed method outperforms others in terms of classification accuracy.

II. Problem Statement and Notations

Consider there are L labelers, N items for classification, and K label classes for items. Let $i \in \{1, 2, \ldots, L\}$ denote the i-th labeler, $j \in \{1, 2, \ldots, N\}$ denote the j-th item, and $x_{ij} \in \{0, 1, 2, \ldots, K\}$ denote the label of item j given by labeler i. $x_{ij} = 0$ if item j is not labeled by labeler i. For supervised data, each item is associated with a set of labels and a true label $\{X_j, y_j\}_{j=1}^{N_s}$, where $X_j = [x_{1j}, x_{2j}, \ldots, x_{Lj}]^T$ and N_s is the number of supervised/training items.

[1]The collected exam data is publicly available at the first author's personal website https://sites.google.com/site/pinyuchenpage

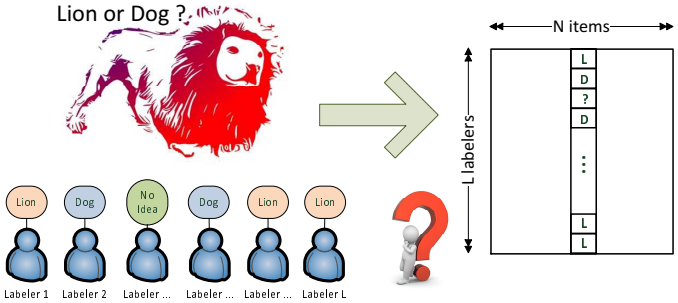

Fig. 1. Illustration of collective classification for crowdsourcing.

The collected exam data is publicly available at the first author's personal website https://sites.google.com/site/pinyuchenpage

For crowdsourced data, X_j might be a sparse vector, where sparsity is defined as the number of nonzero entries in a vector. The sparsity originates from the fact that a labeler might only label a small portion of items. For the collected multiple choice exam data where labels are answers provided by students, X_j is in general not a sparse vector. We aim to construct a classifier $f : \{0, 1, \ldots, K\}^L \mapsto \{1, 2, \ldots, K\}$ for collective classification based on supervised data. For binary classification, we will use the convention that $x_{ij} \in \{-1, 0, 1\}$ and $y_j \in \{-1, 1\}$. For multi-class classification, unless stated, we use the one-to-all classifier $f(X_j) = \arg\max_{k\in\{1,2,\ldots,K\}} f_k(X_j)$, where $f_k(X_j)$ is the binary classifier of item j such that the labels of class k are 1 and the labels of the other classes are -1. We also denote the predicted label of item j by $\hat{y}_j$ and the indicator function by $\mathbb{1}_{\{\cdot\}}$, where $\mathbb{1}_{\{e\}} = 1$ if event e is true and $\mathbb{1}_{\{e\}} = 0$ otherwise. We further define the weight of each labeler by w_i and define the weighting vector $w = [w_1, w_2, \ldots, w_L]^T$. For classifiers associated with weighting vector w, we have $f_k(X_j) = \text{sign}(w^T X_j)$, where $\text{sign}(z) = 1$ if $z \geq 0$ and $\text{sign}(z) < 0$ otherwise.

III. Overview of Crowdsourcing Algorithms

A. Majority Votes (MV)

Majority votes is the baseline (unsupervised) classifier for collective classification. The classifier is

$$f^{MV}(X_j) = \arg\max_{k\in\{1,2,\ldots,K\}} \sum_{i=1}^{L} \mathbb{1}_{\{x_{ij}=k\}}. \quad (1)$$

B. Weighted Averaging (WA)

Weighted averaging is a heuristic approach for weight assignment based on the classification accuracy of each labeler in the training data. Let q_i^{WA} be the number of correctly classified items out of the supervised items for labeler i, we define the weight to be

$$w_i^{WA} = \frac{q_i^{WA}}{\sum_{i=1}^{L} q_i^{WA}}. \quad (2)$$

C. Exponential Weighted Algorithm (EWA)

Exponential weighted algorithm sequentially adjusts weight of each labeler based on the loss of the predicted label and the true label [23]. For sake of convenience, let $x_{ij} \in \{0, 1\}$ be the binary labels and $\ell(\hat{y}_j, y_j) = (\hat{y}_j - y_j)^2$ be the loss function. Let $w_{i,j}^{EWA}$ be the weight of labeler i at stage j with initial value $w_{i,0}^{EWA} = 1/L$. The goal of EWA is to achieve low regret R_{N_s}, defined as

$$R_{N_s} = \sum_{j=1}^{N_s} \ell(\hat{y}_j, y_j) - \min_{i\in\{1,2,\ldots,L\}} \sum_{j=1}^{N_s} \ell(x_{ij}, y_j), \quad (3)$$

the difference of loss between collective classification and the best labeler. The predicted label for item j is

$$\hat{y}_j = \text{ceil}\left(\frac{\sum_{i=1}^{L} w_{i,j}^{EWA} x_{ij}}{\sum_{i=1}^{L} w_{i,j}^{EWA}} - \frac{1}{2} \right), \quad (4)$$

where $\text{ceil}(z)$ is the ceiling function that accounts for the smallest integer that is not less than z. The weight is updated according to

$$w_{i,j+1}^{EWA} = w_{i,j}^{EWA} \exp\left(-\eta \ell(x_{ij}, y_j)\right). \quad (5)$$

By setting $\eta = \sqrt{\frac{8 \ln L}{N_s}}$, it is proved in [24] that $R_{N_s} \leq \sqrt{\frac{N_s}{2} \ln L}$. That is, the regret to the best labeler scales with $O(\sqrt{N_s})$ and the average regret per training sample $\frac{R_{N_s}}{N_s}$ scales with $O(\frac{1}{\sqrt{N_s}})$.

D. Multi-class Adaboost (MC-Ada)

For multi-class Adaboost [22], each labeler acts as a weak classifier $f_i(\cdot)$, and in the training stage it finds the best labeler according to a specified error function. In each round the weights of the supervised data are updated so that the algorithm can find a labeler with better classification capability for the misclassified training samples. The weight of each classier is determined according to the error function and the final classier is the weighted combination of every labeler. The multi-class Adaboost algorithm proposed in [22] is summarized as follows:

1) Initialize all weights of the training data to be $\alpha_j = \frac{1}{N_S}$. For $i = 1, 2, ..., L$ repeat Step 2 to Step 5
2) Find the best labeler that minimizes the error $err_i = \sum_{j=1}^{N_S} \frac{\alpha_j \mathbb{1}_{\{y_j \neq f_i(X_j)\}}}{\sum_{j=1}^{N_S} \alpha_j}$
3) Set $w_i = \log \frac{1-err_i}{err_i} + \ln(K-1)$
4) Set $\alpha_j \leftarrow \alpha_j \cdot \exp(w_i \mathbb{1}_{\{y_j \neq f_i(X_j)\}})$ for $j = 1, 2, \ldots, N_s$
5) Normalize α to have unit norm
6) The final classifier is $f^{MC-Ada}(X_j) = \arg\max_{k=\{1,2,\ldots,K\}} \sum_{i=1}^{L} w_i \mathbb{1}_{\{k=f_i(X_j)\}}$

E. Conventional Support Vector Machine (C-SVM)

Given supervised data $\{X_j, y_j\}_{j=1}^{N_s}$, conventional SVM aims to solve the optimization problem [23]

$$\min_{w,b,\xi} \ \frac{1}{2}\|w\|_2^2 + \frac{C}{N_s} \sum_{j=1}^{N_s} \xi_j \quad (6)$$

subject to $y_j(w^T X_j - b) \geq 1 - \xi_j,\ \xi_j \geq 0\ \forall j \in \{1, 2, \ldots, N_s\}$,

where $\xi_j \geq 0$ is the soft margin and C is a tuning parameter. By representing (6) in dual form, it is equivalent to solving

$$\max_{\sigma} -\frac{1}{2}\sum_{j=1}^{N_s}\sum_{z=1}^{N_s}\sigma_j\sigma_z y_j y_z X_j^T X_z + \sum_{j=1}^{N_s}\sigma_j \quad (7)$$
$$\text{subject to } 0 \leq \sigma_j \leq \frac{C}{N_s}\ \forall j \in \{1,2,\ldots,N_s\}.$$

Let $\sigma^* = [\sigma_1, \sigma_2, \ldots, \sigma_{N_s}]^T$ be the solution of (7), then the optimal weight and intercept in (6) are $w^* = \sum_{j=1}^{N_s}\sigma_j^* y_j X_j$ and b^* is the average value of $y_j - {w^*}^T X_j$ for all j such that $0 < \sigma_j^* < \frac{C}{N_s}$. Therefore the binary classifier is

$$f^{C-SVM}(X_j) = \text{sign}\left({w^*}^T X_j + b^*\right) = \text{sign}\left(\sum_{z=1}^{N_s}\sigma_z^* y_z X_z^T X_j + b^*\right). \quad (8)$$

F. Multi-class Support Vector Machine (MC-SVM)

In [20], a multi-class support vector machine approach is proposed by imposing a generalized hinge loss function as a convex surrogate loss function of the training error. Let $M = [M_1, M_2, \ldots, M_K]^T$ be the matrix containing all weighting vectors $M_k \in \mathbb{R}^L$ for class k, the generalized hinge loss function is defined as $\max_{r\in\{1,2,\ldots,K\}}\{M_r^T X_j - 1 - \delta_{y_j,r}\} - M_k^T X_j$, where $\delta_{y_j,r}$ is the Kronecker delta function such that $\delta_{y_j,r} = 1$ if $y_j = r$ and $\delta_{y_j,r} = 0$ otherwise.

By introducing the concepts of soft margins and canonical form of separating hyperplanes, MC-SVM aims to solve the optimization problem

$$\min_{M,\xi}\ \frac{\lambda}{2}\|M\|_2^2 + \sum_{j=1}^{N_s}\xi_j \quad (9)$$
$$\text{subject to } M_{y_j}^T X_j + \delta_{y_j,k} - M_k^T X_j \geq 1 - \xi_j\ \forall j,k,$$

where $\xi = [\xi_1, \xi_2, \ldots, \xi_{N_s}]^T$ is the vector of slack variables accounting for margins with $\xi_j \geq 0$, $\|M\|_2^2 = \sum_{k,j} M_{kj}^2$ is defined as the ℓ_2-norm of the vector represented by the concatenation of M's rows, and λ is the regularization coefficient.

Let $\mathbf{1}_z$ be a vector of zero entries except that its zth entry being 1, $\mathbf{1}$ be the vector of all ones, and $\tau = [\tau_1, \tau_2, \ldots, \tau_{N_s}]$ be a K-by-N_s matrix. The optimization problem in (9) can be solved in dual form by

$$\max_{\tau}\ -\frac{1}{2}\sum_{j=1}^{N_s}\sum_{z=1}^{N_s}(X_j^T X_z)\cdot(\tau_j^T\tau_z) + \lambda\sum_{j=1}^{N_s}\tau_j^T e_{y_j} \quad (10)$$
$$\text{subject to } \tau_j \leq \mathbf{1}_{y_j},\ \mathbf{1}^T\tau_j = 0\ \forall j.$$

Consequently the classifier for MC-SVM is

$$f^{MC-SVM}(X_j) = \arg\max_{k\in\{1,2,\ldots,K\}}\left\{\sum_{z=1}^{N_s}\tau_{kz}X_z^T X_j\right\}. \quad (11)$$

For algorithmic implementation using multi-class SVM, X_j is extended to a $K \times L$-by-1 vector, where the label for item j provided by labeler i is represented as $\mathbf{1}_{x_{ij}}$. For instance, the label 3 of a 4-class SVM is represented by $[0\ 0\ 1\ 0]^T$.

G. EM Algorithm

For sake of convenience, let $x_{ij} \in \{0,1\}$ be the binary labels. Following the definitions in [12], let $\alpha_i = P(x_{ij} = 1|y_j = 1)$ be the probability of correct classification of labeler i when $y_j = 1$ and $\beta_i = P(x_{ij} = 0|y_j = 0)$ be the probability of correct classification of labeler i when $y_j = 0$. Given the crowdsourced data $X = \{X_1, X_2, \ldots, X_N\}$, the task is to estimate the parameters $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_L]^T$ and $\beta = [\beta_1, \beta_2, \ldots, \beta_L]^T$, where we denote the parameters by $\theta = \{\alpha, \beta\}$.

Assuming the items are independently sampled, the likelihood function of observing X given θ is

$$P(X|\theta) = \prod_j^N P(x_{1j}, x_{2j}, \ldots, x_{Lj}|\theta) = \prod_j^N P(X_j|\theta). \quad (12)$$

The maximization problem can be simplified as we apply EM algorithm [10], which is an efficient iterative procedure to compute the maximum-likelihood solution in presence of missing/hidden data. Here, we regard the unknown hidden true label y_j as the missing data. Define

$$a_j = \prod_{j=1}^N {\alpha_i}^{x_{ij}}(1-\alpha_i)^{1-x_{ij}};\ b_j = \prod_{j=1}^N {\beta_i}^{x_{ij}}(1-\beta_i)^{1-x_{ij}}; \quad (13)$$

$$u_j = P(y_j = 1|X_j,\theta) \propto P(X_j|y_j = 1)\times PD(y_j = 1|\theta) = \frac{a_j v}{a_j v + b_j(1-v)} \quad (14)$$

by the Bayes rule with $v = \frac{1}{N}\sum_{j=1}^N u_j$. The complete loglikelihood can be written as $\ln P(X, y|\theta) = \sum_{j=1}^N y_j \ln v a_j + (1-y_j)\ln(1-v)b_j$. The EM algorithm is summarized as follows:
E-step: Since $\mathbb{E}[\ln P(X, y|\theta)] = \sum_{j=1}^N u_j \ln v a_j + (1-u_j)\ln(1-v)b_j$, where the expectation is with respect to $P(y|X,\theta)$, we compute $v = \frac{1}{N}\sum_{j=1}^N u_j$ and update $u_j = \frac{a_j v}{a_j v b_j(1-v)}$.
M-step: Given the updated posterior probability u_j and the observed data X, the parameters θ can be estimated by maximizing the conditional expectation of correct specification probability, i.e., α_i and β_i, by

$$\alpha_i = \frac{\sum_{j=1}^N u_j x_{ij}}{\sum_{j=1}^N u_j};\ \beta_i = \frac{\sum_{j=1}^N (1-u_j)(1-x_{ij})}{\sum_{j=1}^N (1-u_j)}; \quad (15)$$

The binary classifier is built upon the converged posterior probability u_j, i.e., $f^{EM}(X_j) = \text{ceil}\left(u_j - \frac{1}{2}\right)$. For initial condition, the conventional (unsupervised) EM algorithm adopts $u_j = \frac{1}{L}\sum_{j=1}^N x_{ij}$. Since supervised data are available, we also propose to modify the initial condition to be $u_j = \frac{1}{L}\sum_{j=1}^N w_i^{WA} x_{ij}$, where the weight w_i^{WA} is defined in Sec. III-B. The experimental results show that the collective classification accuracy can be improved by setting the initial condition as the weighted average of the supervised data.

H. Naive Bayes (NB) Sampler/Classifier

Naive Bayes sampler is a generative classifier that assumes conditional independence among components in X_j. The classifier can be represented as $f^{NB}(X_j) =$

$\arg\max_{k=\{1,2,\ldots,K\}} \hat{\pi}_k \hat{g}_k(X_j)$, where $\hat{\pi}_k$ is the estimated prior of the training data, and $\hat{g}_k(X_j)$ is the estimated probability mass function $g_k(X_j) = \prod_{i=1}^{L} g_k^{(i)}(x_{ij})$, and $g_k^{(i)}(x_{ij})$ is the marginal probability mass function of the random variable $X_{ij}|Y = k$.

The prior is estimated using the Dirichlet prior, which accounts for uniformly distributed prior. This alleviates the problem that data samples of some classes do not appear in the training data. We thus have the estimators $\hat{\pi}_k = \frac{\phi_k}{N_s+K}$ and $\hat{g}_k^{(j)}(z_l) = \frac{\phi_{kl}^j+1}{\phi_k+K}$, where $\phi_k = |\{j : y_j = k\}|$ and $\phi_{kl}^{(i)} = |\{j : y_j = k \wedge x_{ij} = z_l\}|$.

IV. The Proposed Supervised Collective Classification Method

We propose a saddle point algorithm for supervised collective classification, where the weight of each labeler is the solution of a convex optimization problem of the form

$$\min_w \; T\left(w, \{X_j\}_{j=1}^{N_s}, \{y_j\}_{j=1}^{N_s}\right) + \lambda R(w), \tag{16}$$

and λ is the regularization parameter. The function T is a convex surrogate loss function associated with the training error $\frac{1}{N_s}\sum_{j=1}^{N_s} \mathbb{1}_{\{f(X_j)\neq y_j\}}$. In particular, we consider hinge loss function $h(z) = \max(0, 1-z)$ and therefore $T = \frac{1}{N_s}\sum_{j=1}^{N_s} h(y_j w^T X_j)$. The function $R(w)$ is a convex regularization function on weighting vector w. In this paper, we consider the ℓ_1-norm regularization functions, i.e., $R(w) = \|w\|_1 = \sum_{i=1}^{L} |w_i|$. The ℓ_1-norm regularization function favors the sparsity structure of the weighting vector w and therefore it aims to assign more weights on the experts (labelers with high classification accuracy) hidden in the crowds.

With the hinge loss function, the formulation in (16) can be rewritten as

$$\begin{aligned} &\min_w \; \frac{1}{N_s}\sum_{j=1}^{N_s} \xi_j + \lambda R(w) \\ &\text{subject to } y_j w^T X_j \geq 1-\xi_j,\; \xi_j \geq 0, j = 1, 2, \ldots, N_s, \end{aligned} \tag{17}$$

where ξ_j accounts for the soft margin of the classifier [23].

The Lagrangian of (17) is

$$\begin{aligned} \mathfrak{L}(w, \xi, \alpha, \beta) = &\frac{1}{N_s}\sum_{j=1}^{N_s} \xi_j + \lambda R(w) \\ &- \sum_{j=1}^{N_s} \alpha_j (y_j w^T X_j - 1 + \xi_j) - \sum_{j=1}^{N_s} \beta_j \xi_j, \end{aligned} \tag{18}$$

where $\alpha_j, \beta_j \geq 0$ are the Lagrange multipliers. The dual optimization problem of (17) becomes

$$\max_{\alpha,\beta,\alpha_j,\beta_j \geq 0} \; \min_{w,\xi} \mathfrak{L}(w, \xi, \alpha, \beta). \tag{19}$$

Fixing α, β, and w, the value ξ that minimizes $\mathfrak{L}$ will satisfy the following equation:

$$\frac{\partial \mathfrak{L}}{\partial \xi_j} = \frac{1}{N_s} - \alpha_j - \beta_j = 0. \tag{20}$$

Note that (20) implies $0 \leq \alpha_j, \beta_j \leq \frac{1}{N_s}$. Substituting (20) to (18), the Lagrangian can be simplified to

$$\mathfrak{L}(w, \alpha) = \lambda R(w) - \sum_{j=1}^{N_s} \alpha_j (y_j w^T X_j - 1). \tag{21}$$

Therefore the dual optimization problem becomes

$$\max_{\alpha,\; 0\leq\alpha_j\leq\frac{1}{N_s}} \; \min_w \mathfrak{L}(w, \alpha). \tag{22}$$

The solution to (22) is a saddle point of $\mathfrak{L}$ that can be obtained by iteratively solving the inner and outer optimization problem and updating the corresponding parameters in (22) [25].

Since our regularization function $R(w) = \|w\|_1$ is not differentiable when $w_i = 0$ for some i. We use the subgradient method [25], [26] to solve the inner optimization problem. The subgradient g of $\|w\|_1$ at a point w_0 has to satisfy $\|w\|_1 \geq \|w_0\|_1 + g^T(w - w_0)$ for all w. Consider a one-dimensional regularizer function $|w|$. Since $|w|$ is everywhere differentiable except when $w = 0$, substituting $w_0 = 0$ we have the constraint on the subgradient at 0 that $g \leq \frac{|w|}{w} \in [-1, 1]$. For $w \neq 0$, g is the gradient of $|w|$ that $g = 1$ if $w > 0$ and $g = -1$ if $w < 0$. Extending these results to $R(w) = \|w\|_1$, we define the (entrywise) projection operator of a L-dimensional function g as $Proj_g(\theta) = [Proj_g(g_1), \ldots, Proj_g(g_L)]^T$, where

$$Proj_g(g_i) = \begin{cases} g_i, & \text{if } |g_i| \leq 1, \\ \frac{g_i}{\|g\|_\infty}, & \text{if } |g_i| > 1, \end{cases} \tag{23}$$

and $\|g\|_\infty = \max_i g_i$ is the infinity norm of g. Therefore the projection operator $Proj_g$ guarantees that the function g to be a feasible subgradient of $\|w\|_1$.

Fixing α, differentiating $\mathfrak{L}$ with respect to w by using the subgradient g as the gradient at the non-differentiable points gives

$$g = \frac{1}{\lambda}\sum_{j=1}^{N_s} \alpha_j y_j X_j. \tag{24}$$

By the subgradient method the iterate of w at stage $t+1$ given $\alpha^{(t)}$ is updated by

$$w^{(t+1)} = w^{(t)} \pm s_w Proj_g\left(\frac{1}{\lambda}\sum_{j=1}^{N_s} \alpha_j^{(t)} y_j X_j\right), \tag{25}$$

where s_w is the constant step length and we set $w^{(0)} = \mathbf{0}$, the vector of all zeros. The sign of the subgradient is determined so that $\mathfrak{L}(w^{(t+1)}, \alpha^{(t)}) \leq \mathfrak{L}(w^{(t)}, \alpha^{(t)})$.

Similarly, for the outer optimization problem, given $w^{(t+1)}$, the gradient of $\mathfrak{L}$ in (21) with respect to α_j is $1 - y_j {w^{(t+1)}}^T X_j$. Since $0 \leq \alpha_j \leq \frac{1}{N_s}$, define the (entrywise) projection operator of a N_s-dimensional function α as

$$Proj_\alpha(\alpha_j) = \begin{cases} \alpha_j, & \text{if } 0 \leq |\alpha_j| \leq \frac{1}{N_s}, \\ \frac{\alpha_j}{\|\alpha\|_\infty N_s}, & \text{if } \alpha_j > \frac{1}{N_s}, \\ 0, & \text{if } \alpha_j < 0. \end{cases} \tag{26}$$

The projection operator $Proj_\alpha$ projects α onto its feasible set.

The iterate of α at stage $t+1$ given $w^{(t)}$ is updated by

$$\alpha^{(t+1)} = Proj_{\alpha}\left(\alpha^{(t)} + s_{\alpha}\text{vec}\left(1 - y_j {w^{(t+1)}}^T X_j\right)\right), \quad (27)$$

where s_α is the constant step length, and $\text{vec}\left(1 - y_j {w^{(t+1)}}^T X_j\right) = [1 - y_1 {w^{(t+1)}}^T X_1, \ldots, 1 - y_{N_s} {w^{(t+1)}}^T X_{N_s}]^T$. Since α relates to the vector of importance of the training samples, we set $\alpha^{(0)} = \frac{1}{N_s}\mathbf{1}$ as the initial point, which means that all training samples are assumed to be equally important in the first place. The algorithm keeps updating the parameters α and w until they both converge. In this paper we set the convergence criterion to be the ℓ_2 norm (Euclidean distance) between the old and newly updated parameters (e.g., the ℓ_2 norm is less than 0.01). The proposed algorithm is summarized as follows:

Algorithm 1 The proposed supervised collective classification algorithm

Input: training samples $\{X_j\}_{j=1}^{N_s}$, training labels $\{y_j\}_{j=1}^{N_s}$, regularization parameter λ
Output: optimal weighting vector w^*
Initialization: $\alpha^{(0)} = \frac{1}{N_s}\mathbf{1}$, $w^{(0)} = \mathbf{0}$, $t = 0$
while $\alpha^{(t)}$ and $w^{(t)}$ do not converge **do**
  Compute $g = \frac{1}{\lambda}\sum_{j=1}^{N_s} \alpha_j^{(t)} y_j X_j$.
  if $\mathfrak{L}\left(w^{(t)} - s_w Proj_g(g), \alpha^{(t)}\right) \leq \mathfrak{L}\left(w^{(t)}, \alpha^{(t)}\right)$ **then**.
    $w^{(t+1)} = w^{(t)} - s_w Proj_g(g)$
  else
    $w^{(t+1)} = w^{(t)} + s_w Proj_g(g)$
  end if
  $\alpha^{(t+1)} = Proj_{\alpha}\left(\alpha^{(t)} + s_{\alpha}\text{vec}\left(1 - y_j {w^{(t+1)}}^T X_j\right)\right)$.
  $t = t + 1$
end while
For robust algorithm, set $w_i^* = w_i^* \mathbb{1}_{\{w_i^* > 0\}}$

Since (16) imposes no positivity constraint on the elements of the weighting vector w, some entries of w can be negative, which implies that one should not trust the labels generated by labelers with negative weights for collective classification. However, in practice altering labeler's labels might be too aggressive and resulting in non-robust classification for the test data. One way to alleviate this situation is to truncate the weights by setting $\widetilde{w}_i = w_i$ if $w_i > 0$ and $\widetilde{w}_i = 0$ if $w_i \leq 0$. That is, the labels from reliable labelers are preserved, whereas the labels from unreliable labelers are discarded. We refer to this approach as the proposed robust method. Note that the proposed saddle algorithm can be adjusted to different convex surrogate loss functions T and convex regularization function R following the same methodology.

V. Performance Evaluation

We compare the crowdsourcing algorithms introduced in Sec. III with the proposed method in Sec. IV on a canonical crowdsourced dataset and the collected multiple-choice exam datasets. The canonical dataset is the text relevance judgment dataset provided in Text REtrieval Conference (TREC) Crowdsourcing Track in 2011 [27], where labelers are asked to judge the relevance of paragraphs excerpted from a subset of articles with given topics. Each labeler then generates a binary label that is either "relevant" or "irrelevant". This dataset is a sparse dataset in the sense that in average each labeler only labels roughly 26 articles out of 394 articles in total. The exam datasets contains science exam with 40 questions and math exam with 30 questions. There are 4 choices for each question and therefore this is a typical multi-class machine learning task. These datasets are quite dense in the sense that almost every student generates an answer for each question.

The oracle classifier to be compared is the performance of the best labeler in the crowds. All tuning parameters are determined by leave-one-out-cross-validation (LOOCV) approach swiping from 0 to 200 for the training data. The classification accuracy are listed in Table V, where the parentheses in the row of best labeler means the number of correctly specified items of the best labeler, and the classifier of the highest classification accuracy is marked by bolded face. For the TREC2011 dataset, when 10 percent of items (40 items) are used to train the classifier, majority votes leads to around 0.8 classification rate. The classification accuracy has notable improvement by using weighted averaging, conventional SVM, and supervised EM algorithm. Naive Bayes sampler has worse performance due to limited training samples. Note that the proposed robust algorithm outperforms others by assigning more weights to reliable labelers and discarding labels from unreliable labeler.

The science dataset is perhaps the most challenging one since there are no perfect experts (i.e., labelers with classification accuracy 1) in the crowds and most of students do not provide accurate answers. Despite its difficulties, our proposed method still outperforms others. Note that in this case the proposed non-robust and robust methods have the same classification accuracy since this dataset is non-sparse and the accuracy of answers in the training data and test data are highly consistent.

The math dataset is a relatively easy task since the majority of students have correct answers. Consequently unsupervised methods tend to have the same performance as the oracle classifier. In case of limited training data size (5 training samples), some supervised methods such as naive Bayes sampler and multi-class SVM suffer performance degradation due to insufficient training samples, whereas the proposed method attains perfect classification.

VI. Conclusion

This paper provides an overview of unsupervised and supervised algorithms for crowdsourcing and proposes a supervised collective classification method where the weights of each labeler is determined via a saddle point algorithm. The proposed method is capable of distinguishing reliable labelers from unreliable ones to enhance the classification accuracy with limited training samples. The results on a benchmark crowdsourced dataset and the exam datasets collected by the authors show that the proposed method outperforms other algorithms. This suggests that supervised collective classification methods with limited training samples can be crucial for crowdsourcing and relevant applications.

TABLE I
DESCRIPTIONS OF CROWDSOURCED DATASETS AND CLASSIFICATION ACCURACY.

Description / Dataset	TREC2011	TREC2011	Science	Math	Math
training data size (N_s)	40	60	10	5	10
test data size ($N - N_s$)	354	334	30	25	20
number of labelers (L)	689	689	183	559	559
Method / Classification accuracy	TREC2011	TREC2011	Science	Math	Math
best labeler (oracle)	1 (82)	1 (84)	0.7 (30)	1 (25)	1 (20)
majority vote	0.7938	0.7964	0.4667	**1**	**1**
weighted averaging	0.8305	0.8323	0.4667	**1**	**1**
exponential weighted algorithm	0.8051	0.8084	0.2667	0.36	0.4
conventional SVM	0.8333	**0.8413**	0.5	0.96	**1**
multi-class SVM	X	X	0.4333	0.52	0.7
unsupervised EM	0.7881	0.7784	0.5	**1**	**1**
supervised EM	0.8277	0.8174	0.5	**1**	**1**
naive Bayes sampler	0.6921	0.6707	**0.5333**	0.64	**1**
multi-class Adaboost	0.8051	0.7994	0.5167	0.8489	0.885
The proposed method	0.8277	0.8323	**0.5333**	**1**	**1**
The proposed robust method	**0.8446**	**0.8413**	**0.5333**	**1**	**1**

ACKNOWLEDGEMENT

The first author would like to thank Tianpei Xie and Dejiao Zhang at the University of Michigan for useful discussions.

REFERENCES

[1] J. Surowiecki, *The Wisdom of Crowds*. Anchor, 2005.

[2] R. J. Prill, J. Saez-Rodriguez, L. G. Alexopoulos, P. K. Sorger, and G. Stolovitzky, "Crowdsourcing network inference: The dream predictive signaling network challenge," *Science Signaling*, vol. 4, no. 189, Sept. 2011.

[3] D. R. Choffnes, F. E. Bustamante, and Z. Ge, "Crowdsourcing service-level network event monitoring," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 387–398, Aug. 2010.

[4] C. Robson, "Using mobile technology and social networking to crowdsource citizen science," Ph.D. dissertation, Berkeley, CA, USA, 2012.

[5] J. Albors, J. Ramos, and J. Hervas, "New learning network paradigms: Communities of objectives, crowdsourcing, wikis and open source," *International Journal of Information Management*, vol. 28, no. 3, pp. 194–202, 2008.

[6] M. Nitti, R. Girau, and L. Atzori, "Trustworthiness management in the social internet of things," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1253–1266, May 2014.

[7] I.-R. Chen, F. Bao, and J. Guo, "Trust-based service management for social internet of things systems," *IEEE Trans. Dependable Secure Comput.*, vol. PP, no. 99, pp. 1–1, 2015.

[8] M. Nitti, L. Atzori, and I. Cvijikj, "Friendship selection in the social internet of things: Challenges and possible strategies," *IEEE Internet Things J.*, vol. 2, no. 3, pp. 240–247, June 2015.

[9] J. Howe, "The rise of crowdsourcing," *Wired Magazine*, vol. 14, no. 6, 2006.

[10] 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*, vol. 39, pp. 1–38, 1977.

[11] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the EM algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 20–28, 1979.

[12] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, "Learning from crowds," *J. Mach. Learn. Res.*, vol. 11, pp. 1297–1322, Aug. 2010.

[13] D. Zhou, J. Platt, S. Basu, and Y. Mao, "Learning from the wisdom of crowds by minimax entropy," in *Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 2204–2212.

[14] O. Dekel and O. Shamir, "Vox populi: Collecting high-quality labels from a crowd," in *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

[15] S. Ertekin, H. Hirsh, and C. Rudin, "Approximating the wisdom of the crowd," in *Advances in Neural Information Processing Systems (NIPS). Workshop on Computational Social Science and the Wisdom of Crowds*, 2011.

[16] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on amazon mechanical turk," in *Proceedings of the ACM SIGKDD Workshop on Human Computation*, 2010, pp. 64–67.

[17] W. Tang and M. Lease, "Semi-Supervised Consensus Labeling for Crowdsourcing," in *Proceedings of the ACM SIGIR Workshop on Crowdsourcing for Information Retrieval (CIR)*, July 2011, pp. 36–41.

[18] P. Ipeirotis, F. Provost, V. Sheng, and J. Wang, "Repeated labeling using multiple noisy labelers," *Data Mining and Knowledge Discovery*, vol. 28, no. 2, pp. 402–441, 2014.

[19] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sept. 1995.

[20] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *J. Mach. Learn. Res.*, vol. 2, pp. 265–292, Mar. 2002.

[21] 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 (EMNLP)*, 2008, pp. 254–263.

[22] J. Zhu, S. Rosset, H. Zou, and T. Hastie, "Multi-class AdaBoost," Tech. Rep., 2006.

[23] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics, 2001.

[24] N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," *Inf. Comput.*, vol. 108, no. 2, pp. 212–261, Feb. 1994.

[25] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

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

[27] *Text REtrieval Conference (TREC) Crowdsourcing Track*. National Institute of Standards and Technology (NIST), 2011. [Online]. Available: https://sites.google.com/site/treccrowd/home