An Iterative and Re-weighting Framework for Rejection and Uncertainty Resolution in Crowdsourcing

Sihong Xie[*] Wei Fan[†] Philip S. Yu[‡]

Abstract

In practical applications of crowdsourcing, labelers may be uncertain or refuse to label a particular instance (or reject) due to the inherent difficulty, and each labeler may be given a different set of instances for big dataset applications. These various issues lead to missing and uncertain labels. Existing crowdsourcing methods have limited capabilities when these two problems exist. In this paper, we propose an Iterative Re-weighted Consensus Maximization framework to address the missing and uncertain label problem. The intuitive idea is to use an iterated framework to estimate each labeler's hidden competence and formulate it as a spectral clustering problem in the functional space, in order to minimize the overall loss given missing and uncertain information. One main advantage of the proposed method from state-of-the-art Bayesian model averaging based approaches is that it uncovers the intrinsic consistency among different set of answers and mines the best possible ground truth. Formal analysis demonstrates that the proposed framework has lower generalization error than widely adopted majority voting techniques for crowdsourcing. Experimental studies show that the proposed framework outperforms state-of-the-art baselines on several benchmark datasets.

1 Introduction

Classification is a fundamental task in data mining and machine learning, such as webpage and image classification, etc. While it is expensive and sometimes impossible to obtain true labels, crowdsourcing provides an affordable alternative to collect supervision from non-experts. For example, given a set of webpages of news or product descriptions, it is relatively expensive to hire dedicated experts to read and categorize them. In contrast, one can outsource these tasks to non-experts on crowdsourcing platforms like Amazon Mechanical Turk.

Crowdsourcing has been a popular and inexpensive way to collect labels for such tasks. However, these low price tag labels usually come with a couple of drawbacks. First, the labels obtained could be noisy and uncertain due to the difficulty of the task, each labeler's perception, lack of interest and insufficient background knowledge. Second, it is not uncommon for labelers to have different opinions. For example, given an article talking about Google's acquisition of Motorola Mobility, it could be labeled as an information tech or a business article. The collected labels are usually inconsistent to some extent, such that one cannot simply treat these labels as ground truths. Third, missing and uncertain labels are ubiquitous in these tasks, especially for large scale data collected from real applications. In the example given above, a labeler might not be confident enough to label that article, so he/she simply omits the example. Collecting labels for all data is unrealistic for large datasets, or labelers are simply unable label all the data. For example, most of the Amazon reviewers review only a couple of products. Therefore, it is highly unlikely that all available labelers are able to provide labels for all queries. Lastly, labelers tend to have different but hidden competence for a task, depending on many factors such as background knowledge. Giving the same weights to all labelers is unreasonable, or at least suboptimal. As the labels provided by less competent labelers are more noisy and could therefore contaminate the labels from more competent labelers. Weighting labels from different labelers should perform better, yet how to learn the weights and exploit them effectively are nontrivial. Interestingly, it is hard to pre-assign a weight vector for labelers as their performance varies from task to task.

To summarize the challenges, we are given a set of uncertain, incomplete and inconsistent labels from which we wish to draw the ground truth while taking labelers' competencies into account. We use a toy example to demonstrate the challenges. In Table 1, the third label of $\mathbf{x}_1$ contradicts the other two labels, which are incorrect, causing majority voting to fail. For $\mathbf{x}_2$, the first two labels are missing, so there is not enough

[*]Department of Computer Science, University of Illinois at Chicago

[†]IBM T.J. Watson Research

[‡]Department of Computer Science, University of Illinois at Chicago and Computer Science Department King Abdulaziz University Jeddah, Saudi Arabia

Table 1: Uncertain and Incomplete Answers in Crowdsourcing

Data	Ground_truth	label 1	label 2	label 3
$\mathbf{x}_1$	-1	1	1	-1
$\mathbf{x}_2$	1	na	na	1
$\mathbf{x}_3$	1	1	-1	-1

information to infer the true label. Note that given an instance, not all labelers necessarily provide a label. This is different from the semi-supervised setting [15] (see Section 5).

We propose an IRCM framework (Iterative Reweighed Consensus Maximization) to address the above challenges. As shown in the flow chart in Figure 1, the proposed framework composes 3 main steps indicated by arrows in different lines. Unlike previous methods [16, 10, 15], which ignore missing labels, IRCM first fills up the missing labels by building classifiers ($f_1, \ldots, f_\ell$ in the chart) from the available labels (Y in the chart) and data (not shown), and then predicting missing labels. By doing so, one obtain a completed label matrix ("Y_c" in the chart). This step is shown in red dotted lines. Next, as shown in green solid lines in the chart, we infer ground truth ("$\mathbf{y}$" in the chart), by feeding weights of labelers (initially uniform distributed) and the completed label matrix to the proposed reweighted consensus maximization algorithm (*RCM* for short in the sequel). As the third step (blue dashed lines), the estimated ground truth and the completed label matrix are further used to update each labeler's competence ($\mathbf{w}$ in the chart). IRCM goes back to the second step, and the iteration continues until it converges. For a more detailed description, see Section 2. The summary of this work is as follows:

- We adopt and improve the CM (Consensus Maximization) framework for crowdsourcing that models labelers' latent consistencies and then use these consistencies to assign the best possible groundtruth. It is demonstrated to be theoretically sound and experimentally effective (Sections 3, 4).
- We adopt the EM framework to provide a probabilistic justification of IRCM, in which, we see the distinction between IRCM and existing state-of-the-art methods (Section 2.4), therefore explain the improved performance achieved via IRCM.
- We demonstrate in Section 3 that CM minimizes generalization error bound, which is not minimized by majority voting. While majority voting is widely adopted in existing crowdsourcing methods, we are the first to introduce CM to crowdsourcing.

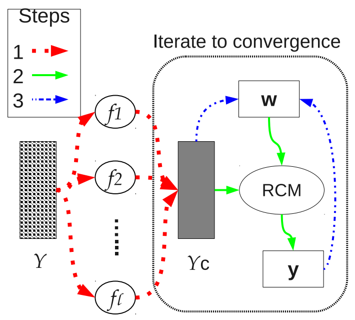


Figure 1: Flow Chart of the Proposed Algorithm

Table 2: Notations for Data

$\mathbf{x}_i \in \mathbb{R}^d$	An instance of data
$\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$	Collection of instances
$Z = [z_1, \ldots, z_n]^\top$	Ground truth labels of $\mathcal{D}$
$\mathbf{y}_i = [y_i^1, \ldots, y_i^\ell]$	Labels for $\mathbf{x}_i$ given by labelers
$Y = [\mathbf{y}_1, \ldots, \mathbf{y}_\ell]^\top$	Label matrix
$Y_c \in \{-1, 1\}^{n \times \ell}$	Completed label matrix
$\mathbf{w} = [w_1, \ldots, w_\ell]$	Competencies of labelers
$f : \mathbb{R}^d \to \{-1, 1\}$	Classification model

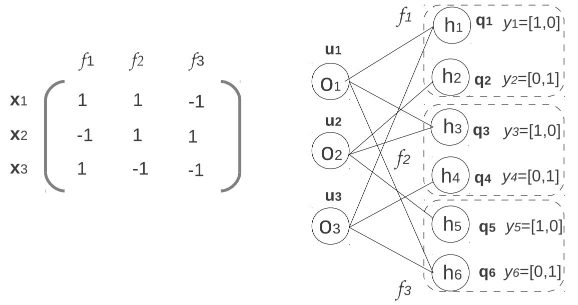
2 Framework

Assume that we are given a sample $\mathcal{D} = \{\mathbf{x}_i, i = 1, \ldots, n\}$, $\mathbf{x}_i \in \mathbb{R}^d$, with labels provided by ℓ labelers. These labels are denoted by $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^\top$ with $\mathbf{y}_i = [y_i^1, \ldots, y_i^\ell]$, which are the labels (probably missing) provided by ℓ labelers for $\mathbf{x}_i$. If an entry in $\mathbf{y}_i$ is missing, it has value 0, otherwise, it is either -1 or 1. The completed label matrix is denoted by Y_c with entries taking value -1 or 1. The objective is to model labelers' competencies $\mathbf{w} = [w_1, \ldots, w_\ell]$ and infer the ground truth labels $Z = [z_1, \ldots, z_n]^\top$ of $\mathcal{D}$. Each labeler can be seen as a mapping from $\mathbb{R}^d$ to $\{-1, 1\}$, denoted by $f_k(\mathbf{x}), k = 1, \ldots, \ell$. The k-th column in Y is the output when f_k evaluated on a subset of $\mathcal{D}$, while the k-th column in Y_c is f_k evaluated on all data in $\mathcal{D}$. These notations are summarized in Table 2.

2.1 Preliminary We briefly introduce the Consensus Maximization (CM) framework [6]. Quite different from traditional Bayesian model averaging approaches, CM combines the outputs of multiple models by considering their cross-example correlation and consistencies. Essentially, CM can resolve conflicts among labelers' opinions and infer the best posisslbe ground truth in crowdsourcing. In the proposed, we improve CM in order to infer each labeler's competence.

Table 3: Notations for Consensus Maximization

o_i	Object node for $\mathbf{x}_i$
h_j	Group node for the $\lceil j/2 \rceil$-th model
$\mathbf{u}_i$	Probability distribution over o_i
$\mathbf{q}_j$	Probability distribution over h_j
$\bar{\mathbf{y}}_j$	Initial probability distribution over h_j
$\mathbf{a}_j$	Indicates the o_i that h_j connects to



(a) Three instances with their labels given by three models (b) The bipartite graph representing the labeling

Figure 2: Bipartite graph used in CM

In CM, classification results of multiple labelers can be represented by a bipartite graph with two types of nodes: object and group nodes. $\mathbf{x}_i$ is represented by an object node o_i. For model k, $k = 1, \ldots, \ell$, there are 2 group nodes $h_{2\times k-1}$ and $h_{2\times k}$ associated with it, representing class -1 and 1, respectively. Object node o_i is connected to group node $h_{2\times k-1}$ (or $h_{2\times k}$) if instance $\mathbf{x}_i$ is classified to -1 (or 1) by the k-th model. Object node o_i is associated with a random vector $\mathbf{u}_i = [u_{i1}, u_{i2}]$ for $i = 1, \ldots, n$, $u_{i1} + u_{i2} = 1$, where u_{i1} and u_{i2} represent the probability that node o_i belongs to class -1 and 1, respectively. Similarly, group node h_j is associated with a random vector $\mathbf{q}_j = [q_{j1}, q_{j2}]$, $q_{j1} + q_{j2} = 1$ for $j = 1, \ldots, v$, $v = 2\ell$ where q_{j1} and q_{j2} are the probabilities that h_j belongs to class -1 and 1, respectively. These random vectors can be organized into two matrices $U_{n\times 2} = [\mathbf{u}_1, \cdots, \mathbf{u}_n]^\top$ and $Q_{v\times 2} = [\mathbf{q}_1, \cdots, \mathbf{q}_v]^\top$. The meanings of U and Q will be clear in Eq.(2.1) and (2.2). The connections between object and group nodes are given by matrix $A_{n\times v} = [\mathbf{a}_1, \ldots, \mathbf{a}_v]$, where $a_{ij} = 1$ if o_i is connected to h_j and 0 otherwise. Each group node has an initial class prediction $\bar{\mathbf{y}}_j = [1, 0]$ if node h_j represents class -1 and $\bar{\mathbf{y}}_j = [0, 1]$ otherwise. Note that $\bar{\mathbf{y}}_j$ are fixed and should be distinguished from the label vector $\mathbf{y}_i = [y_i^1, \ldots, y_i^\ell]$ of instance $\mathbf{x}_i$ given by ℓ labelers. Let $\bar{Y}_{v\times 2} = [\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_v]^\top$. Table 3 summarizes these notations.

Figure 2 demonstrates how to construct a bipartite graph in CM. Assume the missing labels in Table 1 are filled up and the resulting labels are shown in Figure 2(a). The corresponding bipartite graph is shown in Figure 2(b). Here we have 3 models (f_1, f_2 and f_3) and two classes, so we need 6 group nodes sitting on the right of the bipartite graph. On the left hand side there are 3 object nodes $\{o_1, o_2, o_3\}$ representing $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. $\mathbf{x}_1$ is classified to $1, 1$ and -1 by models f_1, f_2 and f_3 (row 1 in Figure 2(a)). Therefore, o_1 is connected to $h_{2\times 1-1}$, $h_{2\times 2-1}$ and $h_{2\times 3}$. Similarly, o_2 is connected to $h_{2\times 1}$, $h_{2\times 2-1}$ and $h_{2\times 3-1}$, since it is classified to $-1, 1, 1$ by three models, respectively (row 2). Lastly, row 3 gives connections between o_3 and h_1, h_4 and h_6.

The objective of CM can be expressed as follows:

$$\begin{aligned} \min_{Q,U} \quad & \textstyle\sum_{i=1}^{n} \sum_{j=1}^{v} a_{ij} \|\mathbf{u}_i - \mathbf{q}_j\|^2 + \alpha \sum_{j=1}^{v} \|\mathbf{q}_j - \bar{\mathbf{y}}_j\|^2 \\ \text{s.t.} \quad & u_{i1}, u_{i2} \geq 0, u_{i1} + u_{i2} = 1, i = 1, \ldots, n \\ & q_{j1}, q_{j2} \geq 0, q_{j1} + q_{j2} = 1, j = 1, \ldots, v \end{aligned}$$

The term $\sum_{j=1}^{v} a_{ij} \|\mathbf{u}_i - \mathbf{q}_j\|^2$ enforces the probability distribution of the object node $\mathbf{o}_i$ to be close to those of the group nodes it connects to and respect the original class predictions to some extent. Parameter α controls how much the consensus results $\mathbf{q}_j$ to be consistent with the initial classification models' outputs $\bar{\mathbf{y}}_j$: a larger α encourages each $\mathbf{q}_j$ to be closer to $\bar{\mathbf{y}}_j$. The optimization problem is solved via block-wise gradient descent, where Q and U are updated using:

$$Q^t = (D_v + \alpha I)^{-1}(A^\top U^{t-1} + \alpha \bar{Y}) \tag{2.1}$$

$$U^t = D_n^{-1} A Q^t \tag{2.2}$$

where $D_n = \text{diag}\{\sum_{j=1}^{v} a_{ij}, i = 1, \ldots, n\}$, and $D_v = \text{diag}\{\sum_{i=1}^{n} a_{ij}, j = 1, \ldots, v\}$, $\bar{Y}_{v\times 2} = [\bar{\mathbf{y}}_1, \cdots, \bar{\mathbf{y}}_v]^\top$ and the superscript t on U and Q denotes the number of iterations. It is easy to see that the probability distribution $\mathbf{u}_i$ is the average of the $\mathbf{q}_j$'s that are $\mathbf{u}_i$'s neighbors defined by A. Similarly, $\mathbf{q}_j$ is defined by the average of probability distributions of the object nodes $\mathbf{u}_i$ it connects to, plus $\alpha\bar{\mathbf{y}}_j$.

In the next section, we transform CM's formulation and interpret it as functional spectral clustering, that essentially use group nodes to associate and link similar and consistent behaviors among labelers, and in the same time, observe labelers' initial class assignments.

2.2 CM as Functional Spectral Clustering Let $b_j = \sum_{i=1}^n a_{ij}$ and define $D_\lambda = \text{diag}\{\frac{b_j}{b_j+\alpha}, j = 1, \ldots, v\}$ and $D_{1-\lambda} = \text{diag}\{\frac{\alpha}{b_j+\alpha}, j = 1, \ldots, v\}$. According to [6], Eq.(2.1) and (2.2) can be re-written as

$$\begin{aligned}
Q^t &= D_\lambda(D_v^{-1}A^\top D_n^{-1}A)Q^{t-1} + D_{1-\lambda}\bar{Y} \\
&= D_\lambda P Q^{t-1} + D_{1-\lambda}\bar{Y} \\
&= (D_\lambda P)^t Q^0 + \left(\sum_{i=0}^{t-1}(D_\lambda P)^i\right) D_{1-\lambda}\bar{Y}
\end{aligned}$$

where $P = D_v^{-1}A^\top D_n^{-1}A$. Let the similarity matrix $P^* = A^\top A$. Then P is the transition probability matrix with entries $P_{kj} = P^*_{kj}/d_k$ where $d_k = \sum_{j=1}^v P^*_{kj}$. To see this, $\mathbf{a}_{2\times l-1} + \mathbf{a}_{2\times l} = \mathbf{1}$ so $d_k = \mathbf{a}_k^\top(\sum_{j=1}^v \mathbf{a}_j) = \ell \cdot |\mathbf{a}_k|$ ($|\cdot|$ is the ℓ_1 norm). Also $(D_n)_{ii} = \sum_{j=1}^v a_{ij} = \ell, i = 1, \ldots, n$, since each object node is connected to exactly ℓ group nodes. So $P_{kj} = (1/\ell)(D_v)^{-1}_{kk}P^*_{kj}$. But $(D_v)_{kk} = |\mathbf{a}_k|$, $P_{kj} = (1/(\ell|\mathbf{a}_k|))P^*_{kj} = (1/d_k)P^*_{kj}$. Similar to the results in [17], as $t \to \infty$

$$Q^* = (I - D_\lambda P)^{-1}D_{1-\lambda}\bar{Y}$$

We make the following observations. First, α is small compared to b_j, which is the number of object nodes connected to h_j, especially when the number of data n is large ($\alpha = 2$ in [6]). Therefore, $D_\lambda \approx 1$ and $D_{1-\lambda} \approx 0$ but not equal to 0. $Q^* \approx (I - P)^{-1}D_{1-\lambda}\bar{Y}$. We can see that CM is similar to spectral clustering of group nodes, where the objective is to find a configuration of $\mathbf{q}_j$ such that similar group nodes (similarity measured by how many object nodes they share) are close in distribution. More specifically, the solution Q^* is the harmonic solution which minimizes the energy function [18]

$$E(Q) = \frac{1}{2}\sum_{j,k} P_{jk}(\mathbf{q}_j - \mathbf{q}_k)^2 \tag{2.3}$$

In this point of view, the class labels provided by each base classification model are not used to estimate the final class distributions, but provide a way to measure similarity between any two group nodes. This accounts for the uncertainty of labelers' opinions in crowdsourcing. For example, one labeler classifies a set of examples to be positive, while another labeler classifies a subset of these examples to be negative, then there are conflicts. It is hard to say whose desicion is more "correct" than the other's. CM resolves these conflicts by adjusting the classification distributions of group nodes (each of them defines how a labeler classifies a subset of examples), such that the agreement of labeling between these two subsets respects their similarity. In Section 3, we justify the effectiveness of CM using this formulation.

2.3 Iterative Re-weighted Consensus Maximization It is natural that different labelers have different competencies in labeling data. Treating all labelers equally when aggregating their decisions is not optimal. Instead, one should pick up the outputs of the best ones to make final decisions, or weigh their outputs using competencies. Nonetheless, the original CM does not take competence or accuracy of base classifiers into account and therefore, is not optimal. To incorporate weights of base classifiers in CM, there are two questions to be addressed. First, without any ground truth or supervision, how can one estimate the competence of each labeler? Second, even if the weights are available, how to use this information effectively in CM?

For the first question, instead of requiring accurate computation of competencies, we ask only for a rough estimation, such that the relative order is preserved. For example, if labeler Bob is doing better than labeler Alice, we only need an estimation of competencies which weighs Bob higher than Alice, instead of an estimation close to the real competence of the labelers. Assume that CM outputs a reasonable estimation of ground truth, we use this information to estimate the competencies of labelers, denoted by $w_j, j = 1, \ldots, \ell$.

The second problem is more difficult to solve. A straightforward solution is to put the weights in the original CM algorithm by weighting $\|\mathbf{q}_j - \bar{\mathbf{y}}_j\|^2$ using $w_j, j = 1, \ldots, v$. The higher the w_j, the more penalty will be incurred when $\mathbf{q}_j$ deviates more from $\bar{\mathbf{y}}_j$, while a lower weight allows the final model $\mathbf{q}_j$ to go relatively far away from its original distribution $\bar{\mathbf{y}}_j$. However, we experimentally find out that this seemingly reasonable and simple method does not work as expected.

We propose to weight the base models in CM via functional space sampling. Specifically, we emphasize "good" functions by sampling them more frequently. We generate a sample of functions by adding new columns to the completed label matrix Y_c. The resulting matrix is called the extended label matrix, also denoted by Y_c, but with new columns added. The new columns are generated entry-wise. That is, for each entry of a new column in Y_c, we randomly pick the entries in the same row in the first ℓ columns of Y_c, where the probability that an entry being picked is proportional to the weight on that column. More formally, suppose we need to add C new columns,

$$Y_c(i, j') = Y_c(i, j) \text{ with probability } w_j/K \tag{2.4}$$

where $j' = \ell + 1, \ldots, \ell + C$, $j = 1, \ldots, \ell$ and $K =$

$\sum_{j=1}^{v} w_j$. In this way, the new columns can be seen as a weighted sample of the original ones. The IRCM algorithm is given in Algorithms 1 and 2. Note that Y_c in line 10 of Algorithm 1 refers to the completed label matrix without extension. The extension of Y_c is done in Algorithm 2 with a new estimation of $\mathbf{w}$.

The idea of column generation is similar to importance sampling [13], which emphasizes on certain instances by sampling them more frequently. Instead of sampling instances in the usual sample space, we are sampling functions in functional space. Suppose we have a space of functions $\mathcal{F} = \{f : \mathbb{R}^d \to \{-1, 1\}\}$. Each column in the label matrix can be seen as a function $f \in \mathcal{F}$ evaluated on data points in $\mathcal{D}$. Though we have no access to the entire functional space $\mathcal{F}$, existing columns can be seen as a finite sample of functions from $\mathcal{F}$. We denote this sample of functions by $\mathcal{F}_0$. Assume that the other good functions outside $\mathcal{F}_0$ are similar to the good ones in $\mathcal{F}_0$, we would like to uncover more good functions in $\mathcal{F}$. Since the "goodness" and "badness" of a function in $\mathcal{F}_0$ are only estimated on a finite number of data, there are some uncertainties associated with them. There is no reason to assert that a good function in $\mathcal{F}_0$ can be generalized well to all instances. We also cannot simply reject "bad" functions from $\mathcal{F}_0$, as they might be doing well on some unseen samples. A good function should be a mixture of functions in $\mathcal{F}_0$ with the constraint that they are closer to good functions than to bad functions in $\mathcal{F}_0$. Therefore, we do not take one of the existing columns as a whole to form a new column, but sample at the entry-wise level (see Eq.(2.4)). We show in the next section that this functional space sampling method might also be viewed as finding a good similarity measure between instances. Also, in Section 3, we justify this sampling method using generalization error bound analysis. We denote the newly generated functions together with the original functions $\mathcal{F}_0$ as $\sigma(\mathcal{F}_0)$, which can be seen as equivalent to Y_c.

Algorithm 1 IRCM with Missing Labels

```
 1: Input:Data $\mathcal{D}$, incomplete label matrix $Y$
 2: Output:Inferred ground truths $Z$
 3: for all labeler $j \in \{1, \dots, \ell\}$ do
 4:     Build a classifier using $Y(:, j)$ and $\mathcal{D}$.
 5:     Complete $Y(:, j)$ using this classifier to get $Y_c(:, j)$.
 6: end for
 7: Infer $Z$ using CM from $Y_c$.
 8: Estimate competencies of labelers $\mathbf{w}$.
 9: while Not converge do
10:     Infer $Z$ from $Y_c$ and $\mathbf{w}$ using Algorithm 2.
11:     Re-estimate $\mathbf{w}$.
12: end while
```

Algorithm 2 RCM: Reweighted CM

```
Input: $Y_c$ (the completed label matrix), $\mathbf{w}$ (weights on each column), $C$ (number of columns to generate)
Output: Inferred ground truths $Z$.
for $j = \ell + 1 \to \ell + C$ and $i = 1 \to n$ do
    Assign value to $Y_c(i, j)$ according to Eq.(2.4).
end for
Estimate $Z$ by applying to $Y_c$.
```

2.4 A probabilistic View of IRCM We formulate IRCM using EM framework to interpret the functional space sampling method. Using this formulation we compare IRCM to the state-of-the-art crowdsourcing methods [16, 10] and bring out the distinction of IRCM. Given sample $\mathcal{D}$ and the extended and completed label matrix Y_c, we need to infer two groups of parameters: competencies $\mathbf{w}$ and ground truths Z. Let Z be the latent random variables, the goal is to maximize the log-likelihood

$$\Pr[Y|\mathcal{D}, \mathbf{w}] = \sum_{i=1}^{n} \ln \Pr[\mathbf{y}_i|\mathbf{w}, \mathbf{x}_i] \tag{2.5}$$

In the following, all probabilities depend on $\mathcal{D}$, but to keep the formula uncluttered, we only write down this dependency explicitly when necessary. Take the latent variables into account and use the total probability formula, $\Pr[\mathbf{y}_i|\mathbf{w}] = \mathbb{E}\{\Pr[\mathbf{y}_i|\mathbf{w}, z_i]\}$ where the expectation is taken over $\Pr[Z]$. It is difficult to maximize Eq.(2.5) directly, we instead maximize its lower bound $\sum_{i=1}^{n} \mathbb{E}\{\ln \Pr[\mathbf{y}_i|\mathbf{w}, z_i]\}$ by the concavity of the logarithmic function ln and Jensen inequality.

Assume either $\Pr[z = 1] = 1$ or $\Pr[z = -1] = 1$ and consider the competence as a labeler's accuracy, namely, $w_j = \Pr[y_i^j = z_i]$ for any $i = 1, \dots, n$. Then we model $\Pr[\mathbf{y}_i|\mathbf{w}, z_i]$ using Bernoulli model as in [16]

$$\Pr[\mathbf{y}_i|\mathbf{w}, z_i] = \prod_{j=1}^{\ell} w_j^{1-|y_i^j - z_i|/2}(1 - w_j)^{|y_i^j - z_i|/2}$$

and the lower bound of Eq.(2.5) can be written as

$$\sum_{i=1}^{n} \ln \left(\prod_{j=1}^{\ell} w_j^{1-|y_i^j - z_i|/2}(1 - w_j)^{|y_i^j - z_i|/2} \right) \tag{2.6}$$

M-step Maximize Take the derivative of Eq.(2.6) with respect to w_j and let it equal to 0 we get

$$w_j = \frac{\sum_{i=1}^{n}(1 - |y_i^j - z_i|/2)}{n} \tag{2.7}$$

E-step Compute Eq.(2.4) To compute Eq.(2.6), we need to know a particular assignment of labels to Z, which can be obtained via $\Pr[Z|\mathcal{D}]$. We propose a unique way to derive this probability. Similar to the work in [15] (refer to related work for more details), we impose a graph prior over the labels Z, namely, we construct a similarity graph of instances where nearby instances are assigned similar labels. Unlike their work, we incorporate labelers' competencies in the graph construction such that the inferred labels directly depend on the competencies.

In particular, given the competencies $\mathbf{w}$ and original functions $\mathcal{F}_0$, we can generate a sample of functions $\sigma(\mathcal{F}_0)$ (represented by the extended complete label matrix Y_c, see Eq.(2.4)). These generated functions can be seen as "virtual" labelers whose classification decisions are weighted combinations of real labelers. Next, based on $\sigma(\mathcal{F}_0)$, we derive a bipartite graph where functions (instances) are represented by group (object respectively) nodes and classification results are represented by the connections between group nodes and object nodes. This graph can be represented by a connection matrix A with $\ell + C$ columns. Rewrite Eq.(2.1) and Eq.(2.2) in the form of random walk iterations,

$$Z^t = \tilde{P}Z^{t-1} + \alpha D_n^{-1}A(D_v + \alpha I)^{-1}\bar{Y}$$

where we restrict the soft cluster membership U to hard partition indicators Z. It is easy to recognize this formula as the random walk with probability transition matrix $\tilde{P} = D_n^{-1}A(D_v + \alpha I)^{-1}A^\top$, which encodes the similarity between instances by the number of group nodes two instances share, normalized by some factors. The iteration converges to the solution maximizing the posterior:

$$\Pr[Z|\mathcal{D}] \propto \exp\{-Z^\top LZ\} \tag{2.8}$$

where $L = I - \tilde{P}$ is the graph Laplacian matrix. We can see that Z is inferred using the function sample $\sigma(\mathcal{F}_0)$.

The conjecture is that, with a better estimation of the competencies and sufficient sampling of the functional space $\mathcal{F}$, we would be able to recover the underlying cluster structure of the data. In this way, IRCM assigns a label to an instance by considering its neighbors' labels and *consensus* among labelers. Note that the methods in [10, 15, 16] assume that the labelers are independent given the sample $\mathcal{D}$. They also simply use labelers' competencies for weighted majority voting. In contrast, we are the first to introduce CM to model consensus among labelers in crowdsourcing. We are also the first to exploit labelers' competencies in the consensus maximization framework. As we shall in Sections 3 and 4, CM outperforms weighted majority voting theoretically and experimentally.

3 Formal Analysis

We adopt the functional spectral clustering formulation (see Section 2.2) to show that CM minimizes generalization error bound. We first define a probabilistic metric in the functional space, namely, the space of classifiers [1]. By a theorem provided in [7] and the objective of spectral clustering algorithm, we show that CM indeed selects a model with lower error bound than the simple majority voting. Finally, we justify the proposed importance sampling strategy in IRCM.

3.1 Metric in Functional Space In general, suppose $(\mathbb{R}^d, \mathcal{B}, \mu)$ is a measure space where $\mathcal{B}$ is a Borel algebra on $\mathbb{R}^d$ and μ is a measure on $\mathcal{B}$. The similarity between two measurable functions $f, g : \mathbb{R}^d \to \mathbb{R}$ can be defined by:

$$\langle f, g\rangle = \int \mathbf{1}(f = g)d\mu \tag{3.9}$$

where $\mathbf{1}(\cdot)$ is the indicator function. We may also calculate the probability that two functions agree, when the underlying measure μ is a probability measure $\Pr[\cdot]$:

$$\langle f, g\rangle = \Pr[f = g]$$

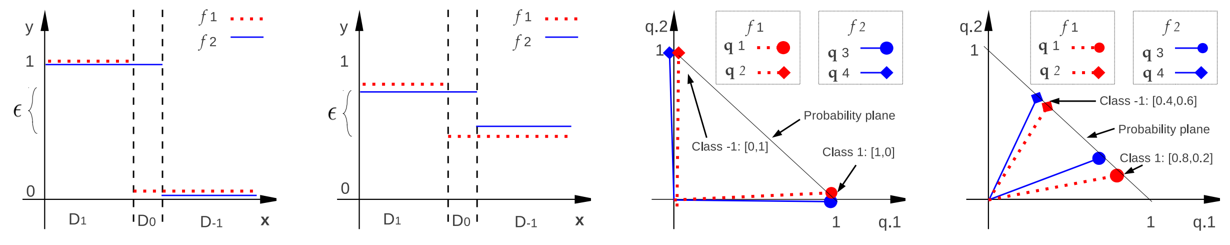
Accordingly, we define the distance between f and g as $1 - \langle f, g\rangle$. For example, suppose the model of ground truth is f_0, then the generalization error of model f is

$$\epsilon(f) = d(f, f_0) = 1 - \langle f, f_0\rangle = \Pr[f \neq f_0] \tag{3.10}$$

In reality, we have no access to the complete measure space, and approximation is required when only given $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$. Second, in CM, functions take continuous values in $[0, 1]$, rather than discrete values $\{-1, 1\}$, we should generalize the similarity measure to functions with real value domain. Therefore, we define the similarity between two real-valued functions on any $\mathcal{D} \subset \mathbb{R}^d$ to be

$$\langle f, g\rangle_{\mathcal{D}} = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}_i \in \mathcal{D}} \mathbf{1}(|f(\mathbf{x}_i) - g(\mathbf{x}_i)| \leq \epsilon) \tag{3.11}$$

where $\epsilon \in (0, 0.5)$ is a fixed small number used in the analysis only. An example is given in Figure 3(a) to demonstrate these concepts. Suppose we have two functions f_1,$f_2 : \mathbb{R} \to [0, 1]$. They can be seen as two classifiers for instances with only one real value feature. The vertical dash lines are the decision planes of f_1 and f_2. We also draw the length of ϵ on the y-axis. We can see that the value of two functions are less than ϵ on regions $\mathcal{D}_1$ and $\mathcal{D}_{-1}$ except $\mathcal{D}_0$ (their definitions are given in the next subsection). So their similarity is the sum of the lengths of $\mathcal{D}_1$ and $\mathcal{D}_{-1}$ according to Eq.(3.11)



(a) Two functions before CM (b) Two functions after CM (c) Two functions before CM in Q plane (d) Two functions after CM in Q plane

Figure 3: Functional spectral clustering

3.2 CM Reduces Functional Difference We claim and demonstrate that after CM, the distance between two functions become smaller. In CM, a function f_l is represented by two group nodes $h_{2\times l-1}$ and $h_{2\times l}$. Each group node is associated with a probability distribution $\mathbf{q}_j \in \mathbb{R}^2$, so functions are embedded in a 2-dimensional space. Figure 3(c) shows how f_1 and f_2 in Figure 3(a) are embedded in the plane $(q_{\cdot 1}, q_{\cdot 2})$, where $\mathbf{q}_j, j = 1, \ldots, 4$ are probability distributions for group nodes $h_j, j = 1, \ldots, 4$. For example, the group node for the positive class of f_2 (blue solid line on the top in Figure 3(a)) is h_3 with probability distribution $\mathbf{q}_3 = [1, 0]$. So in Figure 3(c), it is given by a blue solid round-ended line on the $q_{\cdot 1}$-axis (shifted for visibility, it overlaps the axis mathematically). We embed the other group nodes in the plane in the same way. The points on the line from $(0, 1)$ to $(1, 0)$ on the plane represents all classification probability distributions, as the sum of the coordinates of any point on the line is 1. During its iterations, CM gradually adjusts the coordinates of these group nodes until convergence. This actually changes the output of the functions in $\sigma(\mathcal{F}_0)$, such that the similarities between any two functions are preserved to the maximum extent. In the above example, the similarity between group nodes h_2 and h_3 are non-zeros (see Figure 3(a), the blue solid line in the top (h_3) overlaps the red dotted line in the bottom (h_2)), so their probability representation, $\mathbf{q}_3$ and $\mathbf{q}_2$, should be pulled closer after CM. Also, $\mathbf{q}_1$ should be even closer to $\mathbf{q}_3$ as h_1 and h_3 represent the same class and overlap more than that h_2 overlaps h_3, This also applies for $\mathbf{q}_2$ and $\mathbf{q}_4$.

In particular, before applying CM, a function or classification model is given by $f : \mathbb{R}^d \to \{-1, 1\}$ that partitions $\mathcal{D}$ into disjoint sets:

$$\mathcal{D}(f) = \bigcup_{c \in \{-1,1\}} \mathcal{D}_c(f) \tag{3.12}$$

where $\mathcal{D}_c(f) = \{\mathbf{x}_i \in \mathcal{D} : f(\mathbf{x}_i) = c\}$. Given a set of functions $\sigma(\mathcal{F}_0) = \{f_1, \ldots, f_{\ell+C}\}$, (see Eq.(2.4) and the paragraph after it), we want to show that CM minimizes the distance between any two pair of functions, in order to apply the generalization error bound in Eq.(3.15). Consider any two functions f_1 and f_2. As shown in Figure 3(a), we can partition $\mathcal{D}$ into three disjoint subsets $\mathcal{D} = \mathcal{D}_{-1} \cup \mathcal{D}_1 \cup \mathcal{D}_0$ where $\mathcal{D}_c = \mathcal{D}_c(f_1) \cap \mathcal{D}_c(f_2)$ for $c \in \{-1, 1\}$ and $\mathcal{D}_0 = \{\mathbf{x} \in \mathcal{D} : f_1(\mathbf{x}) \neq f_2(\mathbf{x})\}$. Then the similarity between f_1 and f_2 can be decomposed into three terms:

$$\langle f_1, f_2 \rangle_{\mathcal{D}} = \langle f_1, f_2 \rangle_{\mathcal{D}_{-1}} + \langle f_1, f_2 \rangle_{\mathcal{D}_1} + \langle f_1, f_2 \rangle_{\mathcal{D}_0} \tag{3.13}$$

We claim that the after applying CM, Eq.(3.13) cannot decrease. First, note that before applying CM, the third term is zero by Eq.(3.11), as $|f_1 - f_2| > \epsilon$ on $\mathcal{D}_0$. Second, if the similarity between two group nodes from different classes is non-zero (like h_2 and h_3 in the above example), then the distributions of these two nodes will be adjusted according to how similar they are, such that the third term in Eq.(3.13) might become non-zero. For example, in Figure 3(d), $\mathbf{q}_2$ and $\mathbf{q}_3$ are brought closer so that they have non-zero similarity, In contrast, they are perpendicular to each other in Figure 3(c) with 0 similarity. Accordingly, in Figure 3(b), the blue solid line in the top and the red dotted line in the bottom are brought closer such that their difference in y-axis is within the ϵ window and $\langle f_1, f_2 \rangle_{\mathcal{D}_0} > 0$. Therefore, the third term in Eq.(3.13) cannot decrease. On the other hand, group nodes representing the same class might not take the highest similarity value 1, so spectral clustering might pull them away from each other. However, they are still closer to each other than to those representing a different class. As spectral clustering preserves the similarity between group nodes, if the difference between the output of two functions for different classes (h_2 and h_3 in the example) is less than ϵ, so does the difference of the output of these functions for the same class (e.g. h_1 and h_2). Therefore, the first two terms in Eq.(3.13) should stay the same. For example, after CM, in Figure 3(d), $\mathbf{q}_3$ and $\mathbf{q}_1$ do not overlap each other perfectly as they do in Figure 3(c), because f_1 and f_2 do not agree on $\mathcal{D}_0$. However, in Figure 3(b), f_1 and f_2 are still within a ϵ window.

3.3 IRCM Minimizes Generalization Error Bound First, the functional sampling method is justified by minimizing Eq.(3.10). Assume the output of CM $\hat{f}$ is a reasonable estimation of f_0, namely, $d(\hat{f}, f_0)$ is relatively small. We would like to find a set of functions such that their center f' (CM averages all functions' output, see Eq.(2.2)) is close to f_0.

$$d(f', f_0) \leq d(f', \hat{f}) + d(\hat{f}, f_0) \tag{3.14}$$

$d(\hat{f}, f_0)$ is fixed, so we aim at minimizing $d(f', \hat{f})$, which can be reduced to finding a set of functions having their center close to $\hat{f}$. This is achieved by sampling the functions in $\mathcal{F}_0$ that are closer to $\hat{f}$ more frequently. Note that the similarity between $f \in \mathcal{F}_0$ and $\hat{f}$ is the accuracy of f when $\hat{f}$ is assumed to be the ground truths. Therefore, the proposed weighted sampling strategy in functional space searches a set of functions to minimize Eq.(3.14).

Second, we justify CM by a theorem in [7]

Theorem 1. *Let $\hat{f}$ be the center of functions in F_0. Then*

$$\varepsilon(\hat{f}) \leq \inf_{f \in \mathcal{G}_0} \sup_{g \in \mathcal{G}_0} \hat{d}(f, g) + \sup_{g', g}(d - \hat{d})(g', g) \tag{3.15}$$

In the theorem, $\mathcal{G}_0$ is a set of consistent classifiers, $\hat{d}$ is the empirical distance of functions. The final output of CM is the average of all models, which corresponds to $\hat{f}$ in the theorem. The second term in the bound can be replaced by two terms independent of the functions in $\mathcal{G}_0$ (see Corollary 1 in [7]). Since we do not have access to $\mathcal{G}_0$, we have to use some approximation by estimating the first term by replacing $\mathcal{G}_0$ with $\sigma(\mathcal{F}_0)$. Therefore, the problem reduces to minimizing the term $\inf_{f \in \sigma(\mathcal{F}_0)} \sup_{g \in \sigma(\mathcal{F}_0)} \hat{d}(f, g)$. We have proved in previous subsections that the distance of each pair of functions in $\sigma(\mathcal{F}_0)$ is made smaller via spectral clustering in functional space. Therefore, CM does lower the upper-bound of generalization error.

Table 4: Summary of Datasets

Tasks	# Features	# Instances
pltcs_vs_bsnss	1389	596
pltcs_vs_tech	1409	597
bsnss_vs_tech	1326	597

4 Experiments

To create datasets with instances labeled by multiple labelers, we ask 5 human annotators to label articles crawled from the Yahoo! news website[1]. We choose 3 categories of news for experiments (politics (pltcs), business (bsnss) and technologies (tech)). The gold standard labels are provided according to the classification of Yahoo! news. The reason of using these 3 classes is that the selected articles can usually be classified into more than one categories, therefore, it is more confusing for human to label, introducing more noise in the labels. For each category, we fetched roughly 300 articles from the website (900 articles in total). Then we mix them together and select 5 subsets of 90 articles randomly. Each labeler labels a subset of articles. Different labelers could label the same article and their opinions might contradict those of others. The resulting label matrix has 90% missing labels. We then create 3 binary classification problems by combining articles from any 2 out of all 3 categories. After text preprocessing such as stop words elimination, TF-IDF transformation, we vectorize the articles in each problem, the properties of the data of 3 problems are summarized in Table 4.

4.1 Baseline Methods In the experiments, we compare the proposed algorithm with two state-of-the-art methods dealing with crowdsourcing data in [16] (EM-PMLA) and [10] (EM-SLME). We refer the readers to the related work for more details. In EM-PMLA, we use LBFGS[2] in the M-step. In EM-SLME, we use the L1-regularized logistic regression provided by Liblinear package [4]. To compare these methods with the proposed method, we feed label matrices with different levels of completeness (see Section 4.2 and 4.3). Besides these sophisticated methods, we consider simplified versions of IRCM. The simplest version is just to use the first step of IRCM to fill up the missing labels and then apply majority voting (MV). Next, after the first round of IRCM, an initial set of competence estimates is obtained. Even without the iteration, we can apply these rough estimates to perform weighted majority voting (WMV). We also consider a degenerated case of IRCM

[1]`http://news.yahoo.com/`

[2]`http://users.eecs.northwestern.edu/~nocedal/lbfgs.html`

which performs CM alone after filling up the missing labels without taking competence into consideration, we called this fCM. As we shall see, while filling up missing labels can help, introducing rough competence estimate has only marginal effect. Although fCM is more effective than majority voting, the iteration step to refine the competence estimates is indeed critical to boost up the performance in IRCM. We adopt libsvm[3] to fill up the missing labels.

4.2 Overall Performance Study We show the overall accuracy of the proposed method and the baseline methods in Table 5 and Figure 4. The number of columns generated in IRCM is fixed to 60 and the number of iterations is set to 15 (see next section for sensitivity study). For a given binary classification problem and a number of labelers (corresponding to one column in the table), we run IRCM 100 times and EM-PMLA 20 times using different combinations of labelers on all three classification problems. The average accuracy and standard variance are recorded for each of such combination. Then these statistics are averaged over all combinations of labelers. For example, when there are 3 labelers, then we have $\binom{5}{3} = 10$ combinations of labelers. All algorithms run on these 10 combinations and each gets 10 copies of accuracies and standard deviations. The accuracies under the header "3 Labelers" in Table 5 are the averages of these 10 copies. For EM-SLME, there is no randomness associated with it, so we do not need to repeatedly run it as other algorithms. If the standard variance is greater than 0.02 (0.01, respectively), then the corresponding accuracy is underlined (in italics font, respectively). For a given number of labelers, we also average performance of each algorithm over 3 classification problems, as shown in the columns with header "avg".

From the table and the bar charts, we have the following observations. First, IRCM has the best performance 7 out of 9 tasks, with the exceptions that fCM slightly better than IRCM. Second, EM-PMLA and EM-SLME can sometimes perform even worse than weighted majority voting. Third, EM-PMLA is unstable, as it has the highest standard deviation in all methods across tasks. In contrast, the performance of IRCM are stable as none of the standard variance is higher than 0.01. Lastly, though sometimes EM-PMLA and EM-SLME have performance close to the proposed method, their accuracy can go down to a very low level. For example, in task *b_vs_t* with 4 labelers, EM-SLME is worse than random guess with only 37.86% accuracy.

The following conclusions can be drawn out of these observations, First, even using the first step of IRCM to fill up missing labels can be helpful for simple strategy like MV. In contrast, the weights learned by these EM-based methods are not effective when the labels are sparse. As we have seen, even the simple MV and WMV (using our weights) work better than weighted voting in EM-PMLA and EM-SLME. Second, The weights learned from IRCM are good indicators to pick up labelers. When there is a tie in majority voting with an even number of labelers, the weights can be helpful to decide which labelers to trust more. Third, As we show in formal analysis, CM is able to find a consensus results among labelers such that the generalization error bound is minimized. This is confirmed by the improved performance of fCM compared to MV and WMV. Lastly, IRCM iteratively refines and exploits competencies of labelers. The iterative re-weighted method achieves even better results than fCM, which does not consider labelers' competencies. This demonstrates that, in situations with multiple labelers such as crowdsourcing, competence is a critical factor to improving classification performance. This also shows that the functional space sampling method is an effective way to incorporate weights in the original CM.

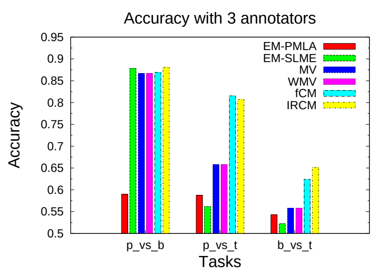
4.3 The effectiveness of filling up missing labels The proposed framework first predicts the missing labels before estimating competencies of labelers. Here, we demonstrate the effectiveness of this missing label filling up step. Different percentage of *missing labels* are filled up, and accuracies are obtained in the same way as we do in the last section. The number of EM iterations in EM-PMLA and EM-SLME is set to 15. From Figure 5, we can see that in most of the cases, the performance of EM-PLMA (7 out of 9 cases) and EM-SLME (6 out of 9 cases) go up as missing label are filled up, therefore, the proposed framework does help existing state-of-the-art methods gain performance.

4.4 Sensitivity Study We study how the performance of IRCM varies with the number of iterations with different number of labelers. IRCM in the first iteration is equivalent to fCM with uniform weights on labelers. In Figure 6, we plot the average accuracy with standard variance over 100 trials as error bars. From these figures, we can see that the accuracy of IRCM becomes better and better in 7 out of 9 cases, with two exceptions in Figure 6(b) and Figure 6(d), where the accuracies of IRCM go down slightly, but still better than those of the baseline methods. Therefore, we demonstrated the effectiveness of incorporating weights in CM in an iterative manner. The number of iterations required to converge is generally quite small (< 5).

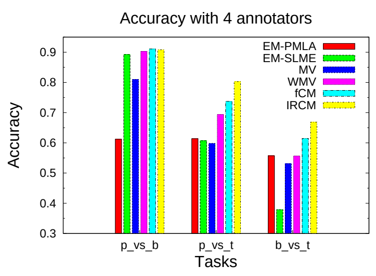
[3] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Table 5: Overall Performance

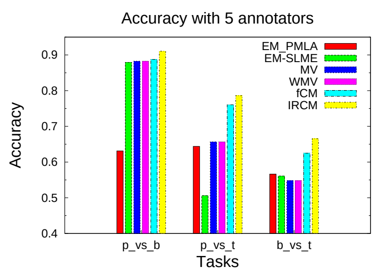
	3 Labelers				4 Labelers				5 Labelers			
Tasks	p vs b	p vs t	b vs t	avg	p vs b	p vs t	b vs t	avg	p vs b	p vs t	b vs t	avg
EM_PMLA	0.5899	0.5876	0.5429	0.5735	0.6120	0.6138	*0.5572*	0.5943	*0.6312*	0.6439	*0.5662*	0.6138
EM_SLME	0.8780	0.5618	0.5224	0.6541	0.8923	0.6070	0.3786	0.6260	0.8792	0.5059	0.5611	0.6487
MV	0.8664	0.6576	0.5578	0.6939	0.8101	0.5983	0.5307	0.6464	0.8826	0.6566	0.5477	0.6956
WMV	0.8664	0.6576	0.5578	0.6939	0.9027	0.6941	0.5568	0.7179	0.8826	0.6566	0.5477	0.6956
fCM	0.8693	**0.8154**	0.6240	0.7696	**0.9107**	0.7374	0.6144	0.7542	0.8876	0.7605	0.6248	0.7576
IRCM	**0.8809**	0.8068	**0.6513**	**0.7797**	0.9082	**0.8028**	**0.6683**	**0.7931**	**0.9107**	**0.7859**	**0.6654**	**0.7873**



(a) pltcs vs bsnss with 3 labelers



(b) pltcs vs bsnss with 4 labelers



(c) pltcs vs bsnss with 5 labelers

Figure 4: Overall accuracy comparison

5 Related Works

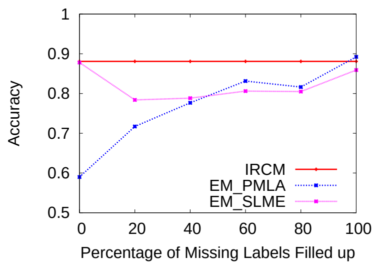
Crowdsourcing is a protocol to collect information using human intelligence. Both EM-PMLA[16] and EM-SLME [10] model the accuracy/expertise of labelers, which is utilized to learn a classification model and the ground truth simultaneously. [16] differs from [10] in that the former assumes that the expertise of a labeler depends not only on the given labels, but also on which data that labeler is labeling. In the high level of view, IRCM is similar to these works, as they all learns competencies and ground truths at the same time. IRCM also bears some similarity to the work in [15] by assuming the graph prior over the ground truths (see Eq.(2.8)). However, IRCM differs from these methods. First, IRCM fills up missing labels rather than assumes that they do not exist. When labels are noisy and sparse, ignoring missing labels can introduce more bias and variance to the estimation. Second, we introduce CM to infer the ground truths, while previous work use majority voting. Third, IRCM exploits labelers' competencies in a different way, namely, importance sampling in functional space, previous works simply use competencies as weights in majority voting.

In [14, 11, 3], they consider the problem of active learning in the multiple labelers settings. Since labels are not free, therefore, acquiring the most desirable labels should be preferred. Note that these strategies can produce label matrices with missing values. For example, different labelers are given different sets of instances according to the active learning strategy. Therefore, rather than being comparable to the proposed algorithm, the algorithms in these previous works can be incorporated in the proposed framework in this paper.

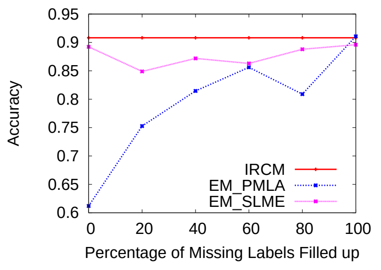
The problem of inferring ground truth from multiple annotators is similar to the problem of ensemble classification[2]. The difference is that the labels in ensemble methods usually come from classifiers, while the labels in crowdsourcing come from human efforts. However, they're similar in light of the noise and inconsistency in labels. One of the most common way to combine multiple classifiers is via majority voting, which can be generalized to weighted majority voting. As pointed out in the formal analysis (Section 3), majority voting ignores the correlations between instances, therefore is ineffective when the noise level in labels is high. Recently, consensus learning algorithms are receiving increasing interests. These learning algorithms aim at bringing consensus to multiple clustering and/or classification models and achieving better performance. [5, 12, 9, 8] are examples of consensus clustering. More recently, [6] proposes a general framework to combine the output of multiple classification and clustering models, demonstrating superior performance to simple majority voting. IRCM improves CM by introducing iterative importance sampling in functional space.

6 Conclusion

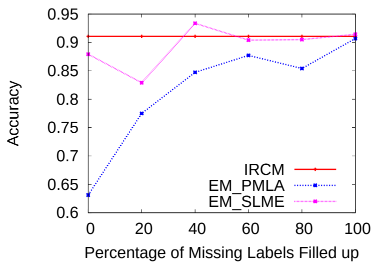
In this paper, we proposed an iterative re-weighted framework (IRCM) to solve the missing and uncertain label problems in crowdsourcing tasks. The main challenges of the problem are: no gold standard is available, labels for an instance are usually inconsistent and



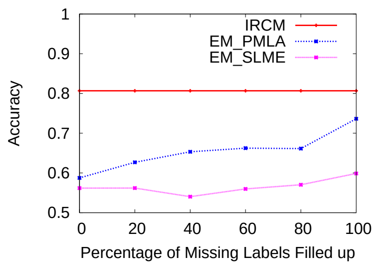
(a) pltcs vs bsnss with 3 labelers



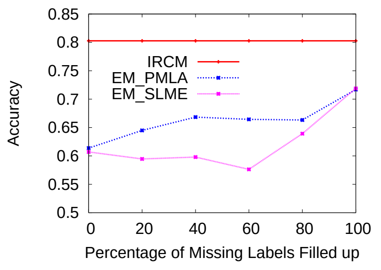
(b) pltcs vs bsnss with 4 labelers



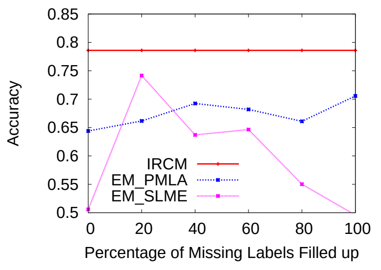
(c) pltcs vs bsnss with 5 labelers



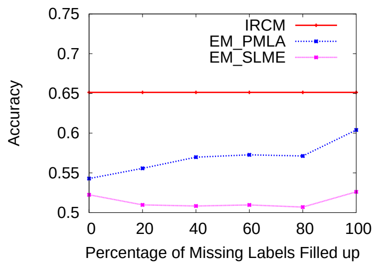
(d) pltcs vs tech with 3 labelers



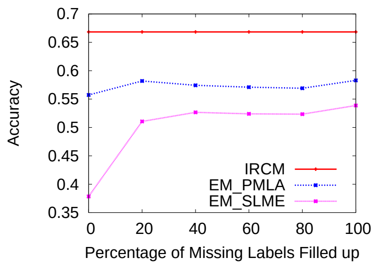
(e) pltcs vs tech with 4 labelers



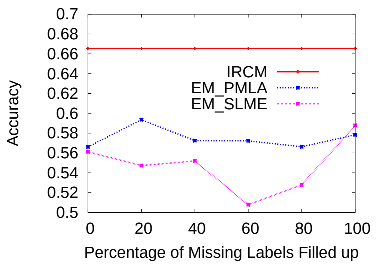
(f) pltcs vs tech with 5 labelers



(g) bsnss vs tech with 3 labelers



(h) bsnss vs tech with 4 labelers



(i) bsnss vs tech with 5 labelers

Figure 5: Accuracies of EM-PMLA and EM-SLME with different percentage of missing labels completed

noisy. The proposed algorithm first fills up missing labels, then hidden competence of labelers are modeled and exploited in an iterative manner. Based on CM, we proposed a novel way to use labelers' competencies, namely, as weights for functional space sampling. Besides, the framework can be formulated as an EM algorithm, where we can see the difference between IRCM and other existing state-of-the-art methods. The framework is theoretically sound and outperforms other baseline methods most of the time.

7 Acknowledgements

This work is supported in part by NSF through grants IIS 0905215, DBI-0960443, CNS-1115234, IIS-0914934, OISE-1129076, and OIA-0963278, and Google Mobile 2014 Program.

References

[1] Yoshua Bengio and Nicolas Chapados. Extensions to metric based model selection. *JMLR*, 3:1209–1227, March 2003.

[2] Thomas G. Dietterich. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*, pages 1–15. Springer-Verlag, 2000.

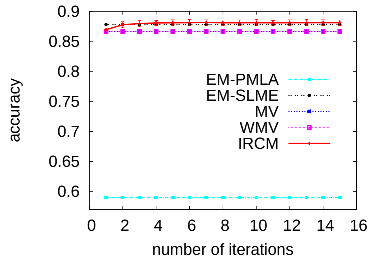
[3] Pinar Donmez and Jaime G. Carbonell. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. CIKM '08, pages 619–628, New York, NY, USA, 2008. ACM.

[4] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874, 2008.

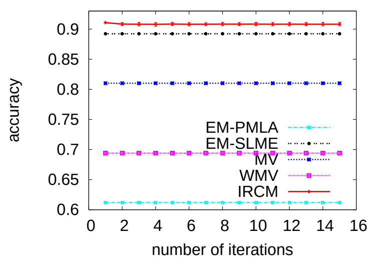
[5] Xiaoli Zhang Fern and Carla E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. ICML '04, New York, NY, USA, 2004. ACM.

[6] Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. Graph-based consensus maximization among multiple supervised and unsupervised models. NIPS '09, pages 585–593, 2009.

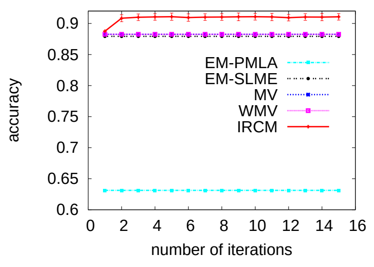
[7] Matti Kaariainen. Generalization error bounds using



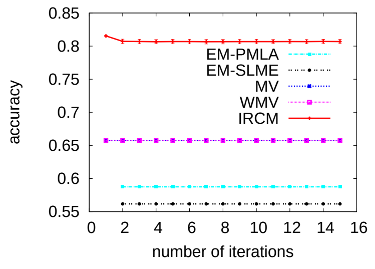
(a) pltcs vs bsnss with 3 labelers



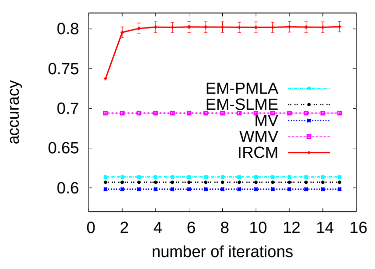
(b) pltcs vs bsnss with 4 labelers



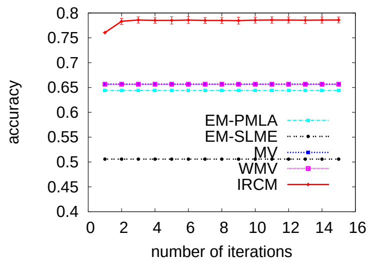
(c) pltcs vs bsnss with 5 labelers



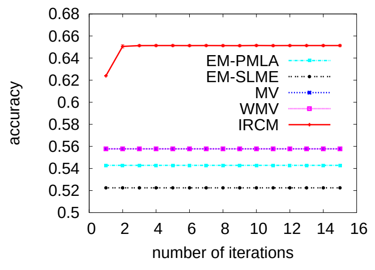
(d) pltcs vs tech with 3 labelers



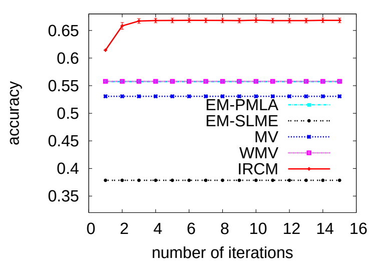
(e) pltcs vs tech with 4 labelers



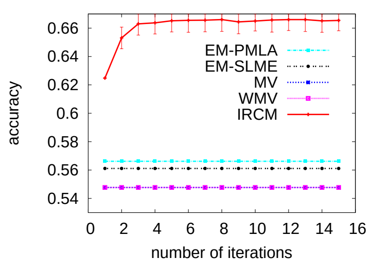
(f) pltcs vs tech with 5 labelers



(g) bsnss vs tech with 3 labelers



(h) bsnss vs tech with 4 labelers



(i) bsnss vs tech with 5 labelers

Figure 6: Accuracies of IRCM with varying number of iterations

unlabeled data. pages 127–142. COLT, 2005.

[8] Dijun Luo, Chris Ding, Heng Huang, and Feiping Nie. Consensus spectral clustering in near-linear time. *Data Engineering, International Conference on*, 0:1079–1090, 2011.

[9] Nam Nguyen and R. Caruana. Consensus clusterings. ICDM '07, pages 607 –612, oct. 2007.

[10] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: whom to trust when everyone lies a bit. ICML, New York, NY, USA, 2009. ACM.

[11] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. KDD '08, pages 614–622, New York, NY, USA, 2008. ACM.

[12] Alexander Strehl and Joydeep Ghosh. Cluster ensembles:a knowledge reuse framework for combining multiple partitions. *JMLR*, 3:583–617, March 2003.

[13] Surya T. Tokdar and Robert E. Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews Computational Statistics*, 2(1):54–60, 2010.

[14] Byron C. Wallace, Kevin Small, Carla E. Brodley, and Thomas A. Trikalinos. Who should label what? instance allocation in multiple expert active learning. In *SDM*, 2011.

[15] Yan Yan, Romer Rosales, Glenn Fung, and Jennifer Dy. Modeling multiple annotator expertise in the semi-supervised learning scenario. UAI '10, pages 674–682, Corvallis, Oregon, 2010. AUAI Press.

[16] Yan Yan, Rómer Rosales, Glenn Fung, Mark W. Schmidt, Gerardo Hermosillo Valadez, Luca Bogoni, Linda Moy, and Jennifer G. Dy. Modeling annotator expertise: Learning when everybody knows a bit of something. 2010.

[17] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schlkopf. Learning with local and global consistency. NIPS '04, pages 321–328. MIT Press, 2004.

[18] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. ICML '03, pages 912–919, 2003.