Semi-supervised Learning from Crowds Using Deep Generative Models

Kyohei Atarashi
Hokkaido University
atarashi_k@complex.ist.hokudai.ac.jp

Satoshi Oyama
Hokkaido University/RIKEN AIP
oyama@ist.hokudai.ac.jp

Masahito Kurihara
Hokkaido University
kurihara@ist.hokudai.ac.jp

Abstract

Although supervised learning requires a labeled dataset, obtaining labels from experts is generally expensive. For this reason, crowdsourcing services are attracting attention in the field of machine learning as a way to collect labels at relatively low cost. However, the labels obtained by crowdsourcing, i.e., from non-expert workers, are often noisy. A number of methods have thus been devised for inferring true labels, and several methods have been proposed for learning classifiers directly from crowdsourced labels, referred to as learning from crowds. A more practical problem is learning from crowdsourced labeled data and unlabeled data, i.e., semi-supervised learning from crowds. This paper presents a novel generative model of the labeling process in crowdsourcing. It leverages unlabeled data effectively by introducing *latent features* and a data distribution. Because the data distribution can be complicated, we use a deep neural network for the data distribution. Therefore, our model can be regarded as a kind of deep generative model. The problems caused by the intractability of latent variable posteriors is solved by introducing an inference model. The experiments show that it outperforms four existing models, including a baseline model, on the MNIST dataset with simulated workers and the Rotten Tomatoes movie review dataset with Amazon Mechanical Turk workers.

1 Introduction

Machine learning, especially supervised learning, techniques have achieved excellent performance in many tasks. However, supervised learning requires a labeled dataset, and obtaining ground truth labels from experts is costly in terms of money and time for the machine learning user and tedious for the expert. Hence, crowdsourcing services like Amazon's Mechanical Turk (AMT) are being used more and more to obtain labels.

Although one can obtain labels at relatively low cost by crowdsourcing, the labels collected from crowdsourcing workers are usually noisy because they are not experts. Furthermore, some workers are simply spammer who provide random labels to easily earn money. Using these noisy labels in supervised learning may result in an inaccurate classifier.

A straightforward way to solve this problem is repeated labeling, i.e., obtaining multiple labels for each data point from multiple workers and inferring the true ones by using a majority voting strategy. Although majority voting is widely used, it is implicitly based on an unreasonable assumption that all workers have the same ability for all data points. Several methods have been proposed for accurately inferring the true labels (Dawid and Skene 1979; Welinder et al. 2010; Whitehill et al. 2009; Oyama et al. 2013). They take into account the expertise of each worker, the difficulty of each instance, and so on. Many of the previous studies for inferring the true labels have proposed the use of a generative model of the labeling process in which the true labels are included as latent labels. In such models, the expertise of each worker, the difficulty of each instance etc. are included as latent variables or parameters. They also proposed estimating the true labels and the model parameters by using the expectation-maximization (EM) algorithm.

One can obtain a classifier by using labels inferred as described above, i.e., by first obtaining labels through repeated labeling and majority voting (or a more sophisticated method) and then applying a general supervised learning method along with the inferred labels. However, after the inferring the true labels, several types of additional information about each instance, e.g., the difficulty of it, are lost. In a supervised learning scenario, the goal is not to infer the true labels for training data but to obtain an accurate classifier for future unknown data, and additional information from crowds may be useful for that goal as well as for training. A number of studies have proposed methods for obtaining accurate classifiers from crowdsourced labels, called *learning from crowds* (Raykar et al. 2010; Kajino, Tsuboi, and Kashima 2012; Yan et al. 2010b; 2010a; Bi et al. 2014; Rodrigues, Pereira, and Ribeiro 2013). Many of these studies also have proposed using a generative model, which includes the true labels as latent variables generated from the classifier. Like the above-mentioned methods for inferring the true labels, the parameters of these models are usually estimated using an EM algorithm. One can obtain a classifier after estimating the parameters, including the parameters of the distribution of true labels.

Although one can obtain labels at relatively low cost by crowdsourcing, the amount of data that can be accessed is becoming larger and larger, so it is not realistic to require labels for all data. Practically, one may be faced with having repeatedly labeled data from crowdsourcing and much more unlabeled data. This problem setting can be regarded as a kind of semi-supervised learning, so we call it *semi-supervised learning from crowds*. However, there have been few studies on semi-supervised learning from crowds (Yan et al. 2010a; Yi et al. 2012).

In this paper, we propose a novel generative model for semi-supervised learning from crowds. Our model effectively uses unlabeled data by introducing *latent features* and the distribution of data. Unlike the model proposed by Dawid and Skene (1979), each worker in our model is assumed to label on the basis of not only true labels but also latent features. This assumption is reasonable and makes the labeling process model more flexible. Because the distribution of data can be complicated, we use a deep neural network for representing the distribution of data. Hence our model can be regarded as a kind of deep generative model. To the best of our knowledge, although there have been a few studies of deep-learning-based methods for inferring true labels (Yin et al. 2017; Gaunt, Borsa, and Bachrach 2016), our study is the first that focused on a deep-learning-based method for learning from crowds.

The structure of the rest of this paper is as follows. In Section 2, we describe our problem settings and notation. We present our proposed model and the method for obtaining classifiers in Section 3. In section 4, we describe related work. Experimental results for an actual dataset and a simulated dataset are shown in Section 5. We summarize our work and discuss future work in Section 6.

2 Semi-supervised Learning from Crowds

In this section we define the problem of semi-supervised learning from crowds. Although we focus on a multiclass classification problem in this paper, it is easy to extend it to a regression problem.

We assume that there are N i.i.d data points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_i \in \mathbb{R}^d$, and each data point has an unknown true label. We denote the true label of $\boldsymbol{x}_i$ as $\boldsymbol{t}_i$, which is a 1-of-K encoded K dimensional vector. We also assume that the first N_c data points $\mathcal{X}_c = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_c}\} = \{\boldsymbol{x}_i \mid i \in \Lambda_c\}$ have crowdsourced labels and the other $N_u = N - N_c$ data points $\mathcal{X}_u = \{\boldsymbol{x}_{N_c+1}, \ldots, \boldsymbol{x}_N\} = \{\boldsymbol{x}_i \mid i \in \Lambda_u\}$ do not. Let $\mathcal{T}_c$ and $\mathcal{T}_u$ be the sets of true labels of $\mathcal{X}_c$ and $\mathcal{X}_u$. Also we assume that there are J workers and denote a noisy label of $\boldsymbol{x}_i$ given by worker $j \in \{1, \ldots, J\}$ as $\boldsymbol{y}_{ij}$, which is also 1-of-K encoded. We denote the set of workers who give labels to $\boldsymbol{x}_i$ as $\mathcal{J}_i$ and the set of crowdsourced labels as $\mathcal{Y}_c$.

Our goal is to obtain an accurate classifier for predicting $\boldsymbol{t}$ from new unknown data point $\boldsymbol{x}$ by using $\mathcal{X}_c, \mathcal{X}_u$, and $\mathcal{Y}_c$.

3 Model and Method

Our generative model of the labeling process in crowdsourcing represents the true labels as latent variables as in previous studies. Unlike previous studies, each data point has *latent feature* $\boldsymbol{z}_i$, and the distribution of data is modeled. Although Welinder et al. proposed a generative model introducing latent features (Welinder et al. 2010), it is for inferring true labels and it does not use the feature vector of data, i.e., $\boldsymbol{x}$, or unlabeled data. By introducing latent features and the distribution of data, we can take advantage of large amounts of unlabeled data.

Our model does not explicitly have a classifier unlike the models proposed in previous studies of learning from crowds. Moreover, it is difficult to estimate the parameters by using an EM algorithm because computing the posteriors of the latent variables is intractable. We thus introduced an inference model (Dayan et al. 1995) for $\boldsymbol{t}$ and $\boldsymbol{z}$ that is like a variational autoencoder (VAE) (Kingma and Welling 2014; Kingma et al. 2014) and estimate the parameters of both the generative model and the inference model at the same time by minimizing the negative unbiased estimator of the lower bound of the marginal log-likelihood, the stochastic gradient variational Bayes (SGVB) estimator. After the parameters have been estimated, the inference model can be used for classifying $\boldsymbol{t}$ of unknown data $\boldsymbol{x}$.

Proposed Generative Model

In the model proposed by Dawid and Skene (1979), which is the most well-known model for inferring true labels and which we call the DS (Dawid-Skene) model, each worker is assumed to label only on the basis of the (unknown) true label of the target data point and to make random mistakes. This means that the labeling error rate does not depend on the data in the DS model. However, in the real world, the ease or difficulty of labeling can vary among data points. For example, in an image labeling task, blurry images are likely to be labelled incorrectly more often than sharp ones. Furthermore, there may be data points that are more easily labeled as belonging to a specific incorrect class than other data points belong to the same class. Therefore, we assume that the labeling does not depend on only the true label. This assumption is the same as one in several previous studies (Yan et al. 2010b; Welinder et al. 2010; Rodrigues, Pereira, and Ribeiro 2013). Further when a worker labels a given data, in many case, he or she does not see data from corner to corner but rather unconsciously extract prominent visual features and assign a label on the basis of those features. Hence, it is plausible that there are *latent features* for each data point and that workers label data on the basis of these latent features. These latent features may have information related to the true label but they can also have other information that does not depend on the true label. Furthermore, the assumption that a worker's label strongly depends on the true label is also reasonable. Hence, we explicitly divide these latent features into the true label $\boldsymbol{t}_i$ and other information $\boldsymbol{z}_i \in \mathbb{R}^{d_z}$ (hereinafter, $\boldsymbol{z}$ is called latent features), and assume that a worker's label $\boldsymbol{y}_{ij}$ is generated from $\boldsymbol{z}_i$ and $\boldsymbol{t}_i$. This modeling makes parameter learning easier. In the semi-supervised learning from crowds setting, there is a large amount of unlabeled data $\mathcal{X}_u$. To utilize $\mathcal{X}_u$ in learning, we introduce the distribution of data (i.e., the distribution of raw feature vectors). The true label $\boldsymbol{t}_i$ and the latent features $\boldsymbol{z}_i$ can be regarded as a summary of $\boldsymbol{x}_i$. Hence, we assume that the raw feature vector $\boldsymbol{x}_i$ is also generated from $\boldsymbol{t}_i$ and $\boldsymbol{z}_i$.

From these assumptions, we can express our model as a factorization of the joint distribution:

$$\begin{aligned} &p(\mathcal{X}_c,\mathcal{Y}_c,\mathcal{T}_c,\mathcal{Z}_c,\mathcal{X}_u,\mathcal{T}_u,\mathcal{Z}_u)\\ &= p(\mathcal{X}_c,\mathcal{Y}_c,\mathcal{T}_c,\mathcal{Z}_c)p(\mathcal{X}_u,\mathcal{T}_u,\mathcal{Z}_u)\\ &= p(\mathcal{X}_c \mid \mathcal{T}_c,\mathcal{Z}_c)p(\mathcal{Y}_c\mid\mathcal{T}_c,\mathcal{Z}_c)p(\mathcal{T}_c)p(\mathcal{Z}_c)\\ &\times p(\mathcal{X}_u\mid\mathcal{T}_u,\mathcal{Z}_u)p(\mathcal{T}_u)p(\mathcal{Z}_u)\\ &= \prod_{i\in\Lambda_c}\prod_{j\in\mathcal{J}_i} p(\boldsymbol{x}_i\mid \boldsymbol{t}_i,\boldsymbol{z}_i)p(\boldsymbol{y}_{ij}\mid\boldsymbol{t}_i,\boldsymbol{z}_i)p(\boldsymbol{t}_i)p(\boldsymbol{z}_i)\\ &\times\prod_{i\in\Lambda_u}p(\boldsymbol{x}_i\mid\boldsymbol{t}_i,\boldsymbol{z}_i)p(\boldsymbol{t}_i)p(\boldsymbol{z}_i), \end{aligned}\tag{1}$$

where $\mathcal{Z}_c$ and $\mathcal{Z}_u$ are the sets of the latent features of $\mathcal{X}_c$ and $\mathcal{X}_u$, and $p(\boldsymbol{t}_i)$ and $p(\boldsymbol{z}_i)$ are the prior distributions of true label $\boldsymbol{t}_i$ and latent features $\boldsymbol{z}_i$. We assume that $\boldsymbol{t}_i$ is generated from a categorical distribution and $\boldsymbol{z}_i$ is generated from a Normal distribution:

$$p(\boldsymbol{t}_i) = \mathrm{Cat}(\boldsymbol{t}_i \mid \boldsymbol{\pi}), \tag{2}$$

$$p(\boldsymbol{z}_i) = \mathcal{N}(\boldsymbol{z}_i \mid \mathbf{0}, \mathbf{I}), \tag{3}$$

where $\boldsymbol{\pi} = (1/K, \ldots, 1/K)^{\mathrm{T}}$.

Because the distribution of data points, $\boldsymbol{x}$, can be complex, we assume that $p(\boldsymbol{x}_i \mid \boldsymbol{t}_i, \boldsymbol{z}_i)$ is a Normal distribution and parameterize the model by using a deep neural network like VAE (Kingma and Welling 2014):

$$p(\boldsymbol{x}_i \mid \boldsymbol{t}_i, \boldsymbol{z}_i) = \mathcal{N}\big(\boldsymbol{x}_i \mid \mu_{\boldsymbol{\theta}}(\boldsymbol{t}_i, \boldsymbol{z}_i), \mathrm{diag}(\sigma^2_{\boldsymbol{\theta}}(\boldsymbol{t}_i, \boldsymbol{z}_i))\big), \tag{4}$$

where $\mu_{\boldsymbol{\theta}}(\boldsymbol{t}_i, \boldsymbol{z}_i)$ and $\sigma^2_{\boldsymbol{\theta}}(\boldsymbol{t}_i, \boldsymbol{z}_i)$ are two separate outputs from the top layer in a deep neural network parameterized by $\boldsymbol{\theta}$, so this deep neural network can be regarded as a decoder. In the case of $\boldsymbol{x}_i \in \{0, 1\}^d$, we assume that $p(\boldsymbol{x}_i \mid \boldsymbol{t}_i, \boldsymbol{z}_i)$ is a Bernoulli distribution and that its parameters are output from the top layer in a deep neural network. Although this parameterization makes our model flexible, computing posteriors of $\boldsymbol{z}$ and $\boldsymbol{t}$ become intractable, so we cannot estimate the parameters by using an EM algorithm. We will discuss the solution to this problem later.

Finally, we assume that the labels of worker j are generated from the classifier of worker j: $f_{\boldsymbol{\alpha}_j}$:

$$p(\boldsymbol{y}_{ij} \mid \boldsymbol{t}_i, \boldsymbol{z}_i) = \mathrm{Cat}(\boldsymbol{y}_{ij} \mid f_{\boldsymbol{\alpha}_j}(\boldsymbol{t}_i, \boldsymbol{z}_i)), \tag{5}$$

where $f_{\boldsymbol{\alpha}_j}$ is multi-class logistic regression and $\boldsymbol{\alpha}_j$ is the set of its parameters. We use the concatenated vector of $\boldsymbol{t}_i$ and $\boldsymbol{z}_i$ as the input vector of $f_{\boldsymbol{\alpha}_j}$. The graphical model is shown in Figure 1.

Optimization and Inference Model

Here we describe the labeling process in crowdsourcing and then describe the method used for estimating the model parameters. Although our model is very flexible, there are two main problems:

- Our model has two sets of latent variables, $\boldsymbol{z}$ and $\boldsymbol{t}$. Because computing the posteriors of $\boldsymbol{z}$ and $\boldsymbol{t}$ are intractable, it is impossible to estimate the parameters by using an EM algorithm.

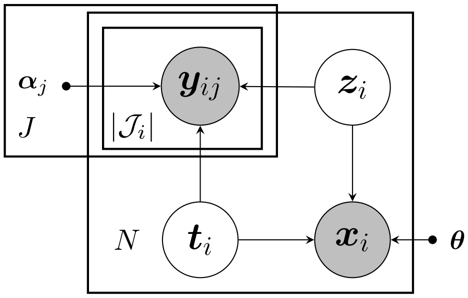


Figure 1: Graphical model of our generative model of labeling process.

- Although our goal is to obtain a classifier, our model does not explicitly include a classifier, and since computing the posteriors of $\boldsymbol{z}$ and $\boldsymbol{t}$ is intractable, we cannot obtain a classifier by simply estimating the parameters of our generative model.

To overcome these problems, we introduce inference model $q(\boldsymbol{t}, \boldsymbol{z} \mid \boldsymbol{x}) = q(\boldsymbol{t} \mid \boldsymbol{x})q(\boldsymbol{z} \mid \boldsymbol{t}, \boldsymbol{x})$ and parameterize it by using a deep neural network. We denote the parameters of this deep neural network as $\boldsymbol{\phi}$. We can use $q(\boldsymbol{t} \mid \boldsymbol{x})$ as a classifier after parameter estimation and also can regard $q(\boldsymbol{z} \mid \boldsymbol{t}, \boldsymbol{x})$ (or of course the network of this distribution) as an encoder. We estimate the parameters of both the generative model and the inference model by minimizing the negative unbiased estimator of the lower bound of the marginal log-likelihood by stochastic gradient descent (Kingma and Welling 2014; Kingma et al. 2014).

The marginal log-likelihood can be decomposed as $\log p(\mathcal{X}_c, \mathcal{Y}_c, \mathcal{X}_u) = \log p(\mathcal{X}_c, \mathcal{Y}_c) + \log p(\mathcal{X}_u)$, where $\log p(\mathcal{X}_c, \mathcal{Y}_c)$ is the marginal log-likelihood of the crowdsourced labeled data, and $\log p(\mathcal{X}_u)$ is that of the unlabeled data. We first consider the lower bound of $\log p(\mathcal{X}_c, \mathcal{Y}_c)$:

$$\begin{aligned} &\log p(\mathcal{X}_c, \mathcal{Y}_c) = \log \int\!\!\int p(\mathcal{X}_c, \mathcal{Y}_c, \mathcal{T}_c, \mathcal{Z}_c) d\boldsymbol{t} d\boldsymbol{z}\\ &\geq \mathbb{E}_{q(\mathcal{T}_c, \mathcal{Z}_c \mid \mathcal{X}_c)}\left[\log \frac{p(\mathcal{X}_c, \mathcal{Y}_c, \mathcal{T}_c, \mathcal{Z}_c)}{q(\mathcal{T}_c, \mathcal{Z}_c \mid \mathcal{X}_c)}\right]\\ &= \sum_{i\in\Lambda_c}\sum_{j\in\mathcal{J}_i} \mathbb{E}_{q(\boldsymbol{t}_i, \boldsymbol{z}_i \mid \boldsymbol{x}_i)}\Big[\log p(\boldsymbol{x}_i, \boldsymbol{y}_{ij} \mid \boldsymbol{t}_i, \boldsymbol{z}_i)\Big]\\ &- \sum_{i\in\Lambda_c} \mathrm{KL}[q(\boldsymbol{t}_i, \boldsymbol{z}_i \mid \boldsymbol{x}_i) || p(\boldsymbol{t}_i, \boldsymbol{z}_i)]\\ &:= -\mathcal{C}(\mathcal{X}_c, \mathcal{Y}_c), \end{aligned}\tag{6}$$

where the second term of Equation (6) is the KL divergence of $q(\boldsymbol{t}_i, \boldsymbol{z}_i \mid \boldsymbol{x}_i)$ from prior $p(\boldsymbol{t}_i, \boldsymbol{z}_i)$. This KL term can be calculated analytically. The first term cannot be computed analytically, however, so we approximate it by using the reparameterization trick of Kingma and Welling (2014) to obtain an unbiased estimator of $\mathcal{C}(\mathcal{X}_c, \mathcal{Y}_c)$, which is differentiable w.r.t. $\boldsymbol{\phi}$ (of course w.r.t. to $\boldsymbol{\alpha}_j$ and $\boldsymbol{\theta}$ as well). This bound is tight when $q(\boldsymbol{t}, \boldsymbol{z} \mid \boldsymbol{x})$ is the true posterior $p(\boldsymbol{t}, \boldsymbol{z} \mid \boldsymbol{x})$.

Similarly, we consider the lower bound of $\log p(\mathcal{X}_u)$:

$$\begin{aligned}\log p(\mathcal{X}_u) &= \log \int\int p(\mathcal{X}_u, \mathcal{T}_u, \mathcal{Z}_u) d\boldsymbol{t} d\boldsymbol{z} \\ &\geq \sum_{i \in \Lambda_u} \mathbb{E}_{q(\boldsymbol{t}_i, \boldsymbol{z}_i | \boldsymbol{x}_i)} \Big[\log p(\boldsymbol{x}_i \mid \boldsymbol{t}_i, \boldsymbol{z}_i) \Big] \\ &- \sum_{i \in \Lambda_u} \mathrm{KL}[q(\boldsymbol{t}_i, \boldsymbol{z}_i \mid \boldsymbol{x}_i) || p(\boldsymbol{t}_i, \boldsymbol{z}_i)] \\ &:= -\mathcal{U}(\mathcal{X}_u).\end{aligned} \tag{7}$$

We can also obtain a differentiable unbiased estimator of $\mathcal{U}(\mathcal{X}_u)$ by using the reparameterization trick.

We can estimate parameters $\boldsymbol{\theta}, \boldsymbol{\alpha}_j$, and $\boldsymbol{\phi}$ by minimizing $\mathcal{C}(\mathcal{X}_c, \mathcal{Y}_c) + \mathcal{U}(\mathcal{X}_u)$ using (mini-batch) stochastic gradient descent. We can use $q(\boldsymbol{t} \mid \boldsymbol{x})$ as a classifier after parameter estimation. However, the output units of the classifier are exchangeable if without any constraint. Therefore, we define the prior of $\boldsymbol{t}$ for crowdsouced data as $p(\boldsymbol{t}_i) = \mathrm{Cat}(\boldsymbol{t}_i \mid \bar{y}_i)$, where $\bar{y}_i = \frac{1}{|\mathcal{J}_i|} \sum_{j \in \mathcal{J}_i} \boldsymbol{y}_{ij}$. For unlabled data, we use uniform prior as described in Equation (2). Although the accuracy of the classifier can be improved by introducing classification loss for labeled data as is done in semi-supervised learning with a VAE (Kingma et al. 2014), the true labels are unknown with our settings. Hence, we introduce pseudo classification loss:

$$-\sum_{i \in \Lambda_c} \log(q(\bar{\boldsymbol{y}}_i \mid \boldsymbol{x}_i)). \tag{8}$$

The result is our final objective function:

$$\mathcal{J} = \mathcal{C}(\mathcal{X}_c, \mathcal{Y}_c) + \mathcal{U}(\mathcal{X}_u) - \alpha \cdot \sum_{i \in \Lambda_c} \log(q(\bar{\boldsymbol{y}}_i \mid \boldsymbol{x}_i)), \tag{9}$$

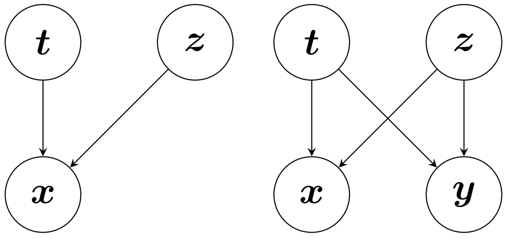
where α is a weight hyper parameter between generative loss $\mathcal{C}(\mathcal{X}_c, \mathcal{Y}_c) + \mathcal{U}(\mathcal{X}_u)$ and pseudo discriminative (classification) loss $-\sum_{i \in \Lambda_c} \log(q(\bar{\boldsymbol{y}}_i \mid \boldsymbol{x}_i))$.

4 Related Work

Our work is based on two lines of research: crowdsourcing, especially for inferring the true labels and learning from crowds, and deep generative models, especially VAEs.

Dawid and Skene (1979) proposed a model for aggregating diagnoses from multiple doctors, and theirs was the first work addressing the inference of true labels from multiple noisy labels. They assumed that a noisy label from a worker depends on the (unknown) true label and the worker's ability. Whitehill et al. (2009) extended this DS model by explicitly modeling the difficulty of each instance. Oyama et al. (2013) extended the DS model further by using self-reported confidence scores from the workers.

Welinder et al. (2010) assumed that each data point has latent features and that the workers observe the noisy version of latent features. They proposed a model for inferring the true labels. Their model and ours are similar in that they introduce the use of latent features but different in that (i) our model is for (semi-supervised) learning from crowds; (ii) the latent features do not depend on the true label in our model but do so in their model; (iii) the inputs to each worker's classifier are the latent features and the true label in our model but only the latent features are input in their model. Therefore, our model reflects the assumption that a worker's label strongly depends on the true label and not only on the true label. (iv) We model the distribution of the raw feature vector $\boldsymbol{x}$, which improves the quality of the latent features; (v) Welinder et al. explicitly modeled the competency of each worker while explicitly modeling the difficulty of each instance and worker competency remains for future work in our model.



(a) (b)

Figure 2: (a) M2 generative model and (b) proposed generative model.

Raykar et al. (2010) also extended the DS model for learning from crowds. In their model, true labels are generated from the classifier, and this approach has been commonly used in most studies on learning from crowds (Bi et al. 2014; Kajino, Tsuboi, and Kashima 2012; Rodrigues, Pereira, and Ribeiro 2013; Yan et al. 2010a; 2010b). On the other hand, the true labels generates data $\boldsymbol{x}$ in our generative model. Yan et al. (2010b) assumed that the workers error rates depend on the data. Kajino, Tsuboi, and Kashima (2012) proposed a model in which workers provide the labels on the basis of their personal classifier (we call this the PC model). There is a base classifier and the parameters of the personal classifiers are generated from a Normal distribution for which the mean is the parameter of the base classifier. As in our model, workers provide the labels in accordance with their personal classifier. However, in the PC model, the input to the personal classifiers is raw feature $\boldsymbol{x}$ while in our model latent features $\boldsymbol{z}$ and (unknown) true label $\boldsymbol{t}$ are input.

The study by Yan et al. (2010a) was the first to address semi-supervised learning from crowds. They used unlabeled data for the prior distribution of the classifier parameters by using a method based on the graph Laplacian. Although the proposed prior distribution can be used in many other models for learning from crowds, the suitable similarity function between data points depends on data and task and designing it may be difficult. The advantage of our model is that it leverages unlabeled data not only for learning a classifier but also for estimating the latent features and data distribution while the Yan et al. method can leverage unlabeled data only for learning a classifier.

Kingma and Welling (2014) proposed a stochastic inference and learning algorithm that scales to large datasets and that works even for a generative model with intractable posteriors of latent variables. In their algorithm, an inference model, which approximates the true posteriors, is introduced, and the parameters of the generative and inference models are estimated simultaneously by minimizing a negative differentiable unbiased estimator of the lower bound. They also proposed a VAE that is a generative model for data. The joint distribution of the VAE is $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z})$, and the parameters of $p(\boldsymbol{x} \mid \boldsymbol{z})$ and the inference model $q(\boldsymbol{z} \mid \boldsymbol{x})$ are computed from $\boldsymbol{z}$ and $\boldsymbol{x}$ by using deep neural networks.

Kingma et al. (2014) extended the VAE for semi-supervised learning (they called the original version VAE M1 and the extended version VAE M2). Figure 2 shows simplified diagrams of the M2 generative model and our proposed generative model. Because $p(\boldsymbol{z}),\ p(\boldsymbol{t})$, and $p(\boldsymbol{x} \mid \boldsymbol{t}, \boldsymbol{z})$ in our model are the same as in the M2 model, our model can be regarded as an extension of the M2 model to include the labeling process. Several extensions of the VAE have been studied (Maaløe et al. 2016; Sønderby et al. 2016) that can be applied to our model.

A few studies investigated deep learning methods for inferring true labels. Yin et al. (2017) proposed a VAE-based model in which the distributions of noisy labels and true labels $p(\boldsymbol{t},\ \boldsymbol{y})$ are modeled like a VAE. They called their model a Label-Aware Autoencoder (LAA). They extended the basic version (LAA-B) by introducing object ambiguity (LAA-O) and latent aspects as latent variables (LAA-L). Although the LAA-L version is similar to our model, there are several differences: (i) our model is for (semi-supervised) learning from crowds; (ii) our model uses the raw feature $\boldsymbol{x}$, which improves the quality of the learned latent features (or aspects).

Gaunt, Borsa, and Bachrach (2016) proposed a method that uses two neural networks. The first one takes the features of the worker and data point and predicts the probability that the workers label is correct. The second one takes the soft answers computed using the output of the first network and infers the true label. The inputs to the first network are computed using the outputs of the second network. The advantage of this method is that true labels can be inferred using completely different sets of labels for the workers and data once the network is trained. The disadvantages are that the true labels are required for training the networks and that it is difficult to handle the case in which all workers do not label all data.

5 Experiments

To assess the effectiveness of the proposed model, we compared with four existing models, including a baseline model, on the MNIST dataset with simulated workers and the Rotten Tomatoes movie review dataset with multiple AMT workers.

Datasets

MNIST. We used MNIST (LeCun et al. 1998) as a benchmark dataset. For semi-supervised learning from crowds, we split the 50,000 training data points between a crowdsourced labeled set $\mathcal{X}_c, N_c = 100$ and an unlabeled set $\mathcal{X}_u, N_u = 49,900$. In MNIST, $\boldsymbol{x} \in [0,1]^{784}$ and $K = 10$. We tested using two types of labeling processes:

1. PC labeling
   Each worker had a personal classifier (a logistic regression model) and used it for labeling. The classifier was learned using 100 data points randomly sampled from $\mathcal{X}_u$ (in this phase, we used $\mathcal{X}_u$ as the labeled dataset). Although each worker had a personal classifier, this labeling process is not exactly the same as in the PC model because we did not assume that there is a base classifier.
2. DS labeling
   Each worker had a conditional probability of producing a label given a (an unknown) true label. If the conditional probabilities were randomly set, a non-realistic probability can be generated. Hence, the personal classifiers (logistic regression models) were first introduced and learned, and then the conditional probabilities were calculated using the personal classifiers and a validation set with true labels.

There were 20 workers, and all workers labeled $\forall \boldsymbol{x} \in \mathcal{X}_c$ in both the PC and the DS labeling. For evaluation, we used 10,000 test data points. .

Rotten Tomatoes. For sentiment polarity classification, Pang and Lee (2005) introduced a sentence polarity dataset consisting of 10,428 movie review snippets from Rotten Tomatoes. Each snippet was labeled with its source reviews label: positive or negative. Rodrigues, Pereira, and Ribeiro (2013) provided a crowdsourced labeled version of this dataset, for which 203 workers labeled 4,999 snippets on the AMT platform, and there were 27,746 crowdsourced labels. We used these 4,999 snippets as $\mathcal{X}_c$. Rodrigues, Pereira, and Ribeiro also provided feature vectors $\boldsymbol{x} \in \mathbb{R}^{1200}$ that were obtained by applying latent semantic analysis to bag-of-words feature vectors, and we used these feature vectors. We split the remaining 5429 snippets into 5 folds, and compared the accuracies using 5-fold cross-validation. Unlike general cross-validation, we used folds for training as unlabeled data.

Methods Compared

We compared our model with four other models.

Baseline Model

- Majority Voting Method
  A majority voting model was used as the baseline model. First the true labels were estimated by majority voting, and then a general supervised learning method was trained using these estimated labels. We used a logistic regression (MV-LR) model and a multi-layer perceptron (MV-MLP) model.

Remaining Models

- Personal Classifier (PC)
  This is the model proposed by Kajino, Tsuboi, and Kashima (2012). We used the limited-memory BFGS (Broyden-Fletcher-Goldfarb-Shanno) algorithm to optimize the personal classifiers.
- Personal Classifier with Graph Prior (PC-GP)
  This model combines the PC model with the graph prior proposed by Yan et al. (2010a). We used the graph prior as

Table 1: Experimental results for the MNIST dataset.

	PC labeling	DS labeling
Proposed	**0.9646**	**0.8647**
MV-LR	0.7162	0.6705
MV-MLP	0.7128	0.6547
PC	0.7756	0.6063
PC-GP	0.7754	0.6055
M2	0.9421	0.8445

the prior of the parameters of the base classifier. The similarity between two data points was defined as the Gaussian kernel: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$.

- M2
  This is the model proposed by Kingma et al. (2014). Because this model is not for learning from crowds and requires true labels, we used $\bar{\boldsymbol{y}}_i = \frac{1}{|\mathcal{J}_i|} \sum_{j \in \mathcal{J}_i} \boldsymbol{y}_{ij}$ as labels.

In all experiments, the architectures of our model and the M2 model were almost the same. We used batch normalization (Ioffe and Szegedy 2015) for all hidden layers of the neural networks except the decoder for the Rotten Tomatoes dataset. For estimation (optimization) of the parameters, we used the Adam stochastic optimization method (Kingma and Ba 2015). The mini-batch size was 200, and half the data points in each mini-batch were labeled examples. For the MNIST dataset, we defined the classifier, encoder, and decoder as MLPs with one hidden layer with 600 units, and d_z was 100. We set the learning rate to 3e-4. Because $\boldsymbol{x} \in [0, 1]^{784}$ for the MNIST dataset, we binarized the feature vectors and defined $p(\boldsymbol{x} \mid \boldsymbol{t}, \boldsymbol{z})$ as a Bernoulli distribution. For the Rotten Tomatoes dataset, we defined the encoder and the decoder as MLPs with two hidden layers and the classifier as an MLP with one hidden layer with 500 units. d_z was also 100. We set the learning rate to 1e-6. We set the exponential decay rate for the 1st and 2nd moment to default values. The architecture of the MV-MLP was same as the architecture of the classifier of the proposed model and the M2 model. We reparameterized α, which is a weight hyper parameter between generative loss and pseudo classification loss of proposed model, as $\alpha = \beta \cdot \frac{N_c + N_u}{N_c}$ like Maaløe et al. (2016). We set $\beta = 1$ on the MNIST dataset and $\beta = 10$ on the Rotten Tomatoes dataset.

Results

Table 1 compares the accuracies by different models on the MNIST dataset. The proposed model archived the best performance in both the PC labeling and the DS labeling. As we saw in the results of the PC and the PC-GP on the MNIST dataset, the GP prior based on the Gaussian kernel (Yan et al. 2010a) were not effective on the MNIST dataset. Appropriate similarity functions vary with data, and designing an appropriate similarity function is difficult in general. On the other hand, the proposed model and the M2 model, especially the proposed model, leveraged unlabeled data effectively in all settings as shown in Table 1. Moreover, Figure 3 shows that the proposed model were outperforming the M2 model during 20000 epochs and converged faster than the M2 model. Hence, modeling the labeling process of workers were also effective because the proposed model can be regarded as the extension of the M2 model by modeling it, as we have shown in Figure 2. In all models, especially PC and PC-GP, the accuracies on the DS labeling were lower than that on the PC labeling. This result is reasonable because the labeling process of the PC labeling is almost same as that modeled by the PC. We also evaluated the proposed model without using unlabeled data $\mathcal{X}_u$ in both PC labeling and DS labeling settings. The accuracies were 0.7023 and 0.5563 respectively, which were lower than the accuracies using unlabeled data. This shows that the proposed model effectively used the unlabeled data to improve the accuracy.

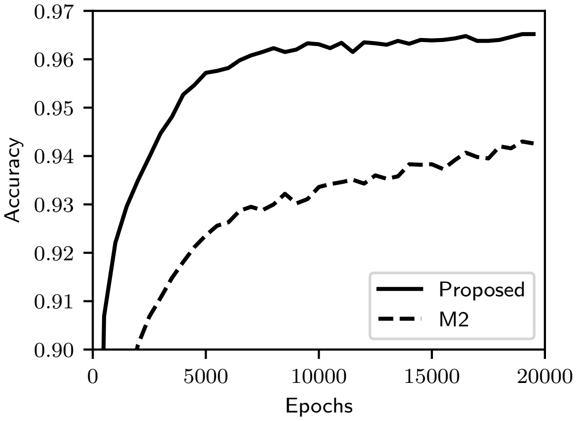


Figure 3: Accuracy for the PC labeling evaluated every 100 epochs between the proposed model and the M2 model.

Table 2: Experimental results for the Rotten Tomatoes dataset. The accuracies of the MVhard (this model is same as the MV-LR), the MVsoft, the Raykar, the Rayker w/prior and the MA-LR are taken from Rodrigues, Pereira, and Ribeiro (2013).

Model	Accuracy	Model	Accuracy
Proposed	**0.7290**	MVhard	0.7027
MV-MLP	0.6979	MVsoft	0.7165
PC	0.7261	Raykar	0.4867
PC-GP	0.7263	Raykar w/prior	0.7078
M2	0.7096	MA-LR	0.7240

Table 2 also compares the accuracies by different models on the Rotten Tomatoes dataset. We trained the DNN based models five times using different random seeds and evaluated them by calculating the mean accuracy. The accuracies of the MVhard (this model is same as the MV-LR), the MVsoft, the Raykar, the Rayker w/prior and the MA-LR are taken from Rodrigues, Pereira, and Ribeiro (2013). The MVsoft model is a logistic regression model using $\bar{\boldsymbol{y}}_i$ as the label. The Rayker model is the model proposed by Raykar et al. (2010), and the Rayker w/prior model is the Raykar model with the prior for the ability of workers. The MA-LR model is the model proposed by Rodrigues, Pereira, and

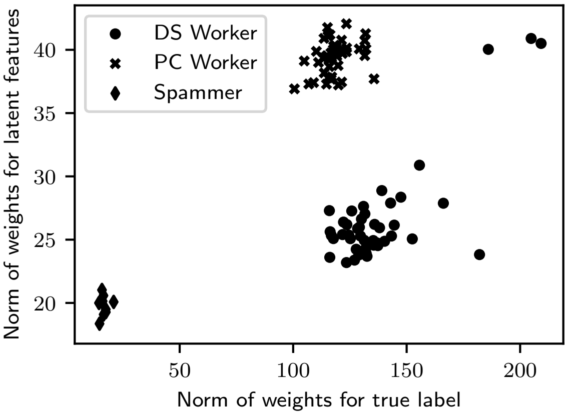


Figure 4: Estimated $\|\boldsymbol{\alpha}^{(t)}\|_1$ and $\|\boldsymbol{\alpha}^{(z)}\|_1$ of 45 DS workers, 45 PC workers and 10 spammers.

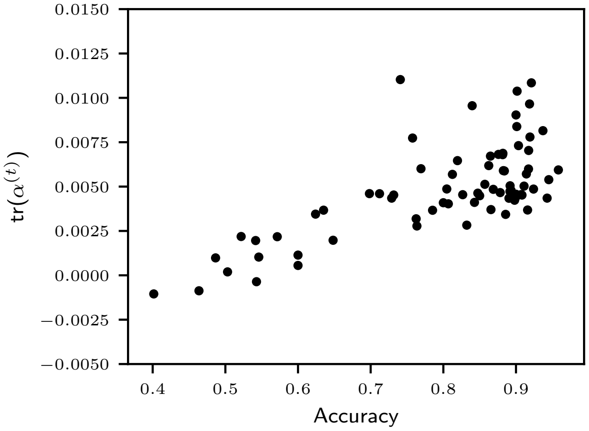


Figure 5: Estimated $\mathrm{diag}(\boldsymbol{\alpha}^{(t)})$ and the accuracy of each worker who labels more than 100 data.

Ribeiro (2013). Proposed model also outperformed other models on the Rotten Tomatoes dataset. The accuracy of the proposed model without using unlabeled data was 0.7262 and this shows that the proposed model utilized the unlabeled data. We used same dataset in this paper for comparison but the size of the dataset was not very large and we could not use many unlabeled data, which may resulted in small difference in accuracy among different models. Hence, we would like to make larger dataset for semi-supervised learning from crowds and experiment in the future.

The parameters of each worker's multi-class logistic regression, $\boldsymbol{\alpha}_j$, have the information about each worker. $\boldsymbol{\alpha}_j$ consists of the weights for the true label and for latent features. We denote them as $\boldsymbol{\alpha}_j^{(t)} \in \mathbb{R}^{K \times K}$ and $\boldsymbol{\alpha}_j^{(z)} \in \mathbb{R}^{K \times d_z}$. Because labels given by workers in DS labeling (DS workers) depend on only the true labels, $\|\boldsymbol{\alpha}^{(z)}\|_1$ of DS workers is expected to be lower than that of PC workers. We trained the proposed model on the MNIST dataset with 100 simulated workers consisting of 45 DS workers, 45 PC workers, and 10 spammers who labeled randomly (we set $N_c = 10,000$, $N_u = 40,000$, and each worker labeled 2000 data points), and then we compared $\|\boldsymbol{\alpha}^{(t)}\|_1$ and $\|\boldsymbol{\alpha}^{(z)}\|_1$ among workers. The results are shown in Figure 4. DS workers, PC workers, and spammers are respectively clustered. $\|\boldsymbol{\alpha}^{(z)}\|_1$ of DS workers were lower than that of PC workers although that of a few (three) DS workers were high. Both $\|\boldsymbol{\alpha}^{(t)}\|_1$ and $\|\boldsymbol{\alpha}^{(z)}\|_1$ of spammers were lower than that of DS workers and PC workers because spammers' labels depend nether true labels nor latent features. This result indicates that the proposed model can learn the characteristic of workers.

Diagonal elements of $\boldsymbol{\alpha}^{(t)}$, $\mathrm{diag}(\boldsymbol{\alpha}^{(t)})$, can be regarded as the weights for the probability of labeling correctly. Hence, $\mathrm{diag}(\boldsymbol{\alpha}^{(t)})$ of accurate workers is expected to be higher than that of inaccurate workers. We trained the proposed model on the Rotten Tomatoes dataset and then calculated the $\mathrm{tr}(\boldsymbol{\alpha}^{(t)})$ of each worker. We also calculated the accuracy of each worker because Rodrigues, Pereira, and Ribeiro (2013) provided the true labels of 4,999 crowdsoureced labeled data, and then compared $\mathrm{tr}(\boldsymbol{\alpha}^{(t)})$ and the accuracy among workers. Figure 5 shows the result for 73 workers who labeled more than 100 data. The accuracy and $\mathrm{tr}(\boldsymbol{\alpha}^{(t)})$ showed positive correlation (the Pearson correlation coefficient $= 0.704$). This result also indicates that the proposed model can learn the characteristic of workers. However, we cannot completely predict the accuracy of the worker only from $\mathrm{diag}(\boldsymbol{\alpha}^{(t)})$ because the worker's labels depend not only on the true labels but also on the latent features in the proposed model. Therefore, our future work includes predicting the accuracy of workers accurately by taking into account the latent features.

6 Conclusion and Future Work

In this paper, we proposed a novel generative model for semi-supervised learning from crowds. In our model, latent features are introduced, and the workers' labels depend on the true labels and the latent features. We also introduced the distribution of data for leveraging unlabeled data. Because the distribution of data can be complicated, we use a deep neural network for representing the distribution of data. Our model can thus be regarded as a kind of deep generative model, especially as an extension of the M2 model (Kingma et al. 2014). Our experimental results showed that the proposed model outperformed other existing models including a baseline model.

The Rotten Tomatoes dataset we used is not very large, so future work includes using a larger dataset. While several previous studies on inferring the true labels and learning from crowds have explicitly modeled the difficulty of each instance, we have not. Hence, future work also includes extending our model to include modeling the difficulty of each instance. Although we used crowdsourced labeled data and unlabeled data in semi-supervised learning from crowds, a smaller amount of data labeled by experts may also be available in some cases. Therefore, we plan to extend our model to handle this situation.

7 Acknowledgements

This work was partially supported by JSPS KAKENHI Grant Numbers JP15H02782 and JP15H05711.

References

Bi, W.; Wang, L.; Kwok, J. T.; and Tu, Z. 2014. Learning to Predict from Crowdsourced Data. In *UAI*, 82–91.

Dawid, A. P., and Skene, A. M. 1979. Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm. *Applied statistics* 20–28.

Dayan, P.; Hinton, G. E.; Neal, R. M.; and Zemel, R. S. 1995. The Helmholtz Machine. *Neural computation* 7(5):889–904.

Gaunt, A.; Borsa, D.; and Bachrach, Y. 2016. Training Deep Neural Nets to Aggregate Crowdsourced Responses. In *UAI*, 242–251.

Ioffe, S., and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *ICML*, 448–456.

Kajino, H.; Tsuboi, Y.; and Kashima, H. 2012. A Convex Formulation for Learning from Crowds. In *AAAI*, 73–79.

Kingma, D., and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Kingma, D. P., and Welling, M. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

Kingma, D. P.; Mohamed, S.; Rezende, D. J.; and Welling, M. 2014. Semi-supervised Learning with Deep Generative Models. In *NIPS*, 3581–3589.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Maaløe, L.; Sønderby, C. K.; Sønderby, S. K.; and Winther, O. 2016. Auxiliary Deep Generative Models. In *ICLR*.

Oyama, S.; Baba, Y.; Sakurai, Y.; and Kashima, H. 2013. Accurate Integration of Crowdsourced Labels Using Workers' Self-reported Confidence Scores. In *IJCAI*, 2554–2560.

Pang, B., and Lee, L. 2005. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with respect to Rating Scales. In *ACL*, 115–124. Association for Computational Linguistics.

Raykar, V. C.; Yu, S.; Zhao, L. H.; Valadez, G. H.; Florin, C.; Bogoni, L.; and Moy, L. 2010. Learning from Crowds. *Journal of Machine Learning Research* 11(Apr):1297–1322.

Rodrigues, F.; Pereira, F.; and Ribeiro, B. 2013. Learning from Multiple Annotators: Distinguishing Good from Random Labelers. *Pattern Recognition Letters* 34(12):1428–1436.

Sønderby, C. K.; Raiko, T.; Maaløe, L.; Sønderby, S. K.; and Winther, O. 2016. Ladder Variational Autoencoders. In *NIPS*, 3738–3746.

Welinder, P.; Branson, S.; Perona, P.; and Belongie, S. J. 2010. The Multidimensional Wisdom of Crowds. In *NIPS*, 2424–2432.

Whitehill, J.; Wu, T.-f.; Bergsma, J.; Movellan, J. R.; and Ruvolo, P. L. 2009. Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *NIPS*, 2035–2043.

Yan, Y.; Rosales, R.; Fung, G.; and Dy, J. 2010a. Modeling Multiple Annotator Expertise in the Semi-Supervised Learning Scenario. In *UAI*, 674–682.

Yan, Y.; Rosales, R.; Fung, G.; Schmidt, M. W.; Valadez, G. H.; Bogoni, L.; Moy, L.; and Dy, J. G. 2010b. Modeling Annotator Expertise: Learning When Everybody Knows a bit of Something. In *AISTATS*, 932–939.

Yi, J.; Jin, R.; Jain, S.; Yang, T.; and Jain, A. K. 2012. Semi-Crowdsourced Clustering: Generalizing Crowd Labeling by Robust Distance Metric Learning. In *NIPS*, 1772–1780.

Yin, L.; Han, J.; Zhang, W.; and Yu, Y. 2017. Aggregating Crowd Wisdoms with Label-aware Autoencoders. In *IJCAI*, 1325–1332.