Learning on Probabilistic Labels

Peng Peng[*] Raymond Chi-Wing Wong[*] Phillp S. Yu[†]

Abstract

Classification is a fundamental topic in the literature of data mining and all recent hot topics like active learning and transfer learning all rely on the concept of classification. Probabilistic information becomes more prevalent nowadays and can be found easily in many applications like crowdsourcing and pattern recognition. In this paper, we focus on a dataset which contains probabilistic information for classification. Based on this probabilistic dataset, we propose a classifier and give a theoretical bound linking the error rate of our classifier and the number of instances needed for training. Interestingly, we find that our theoretical bound is asymptotically at least no worse than the previously best-known bounds developed based on the traditional dataset. Furthermore, our classifier guarantees a fast rate of convergence compared with traditional classifiers. Experimental results show that our proposed algorithm has a higher accuracy than the traditional algorithm. We believe that this work is influential since it opens a new topic on the probabilistic dataset, allowing researchers to study all topics related to classification like active learning and transfer learning under this new probabilistic setting.

1 Introduction

Classification is very important in a lot of real-life applications. Consider a binary classification. We are given a *training dataset* T containing n *instances* where each instance is associated with an attribute set X denoting its *features* and a target attribute Y. In the binary classification, Y is equal to only two possible values called *labels*, namely 0 and 1. Since Y simply has two possible values, we call this dataset the *clear-cut dataset*. The goal is to find an *accurate classifier* based on T. Some measurements to evaluate the accuracy of a classifier are *precision*, *recall* and *f-measure*.

Motivation: The accuracy of a classifier heavily depends on the *qualities* of the labels in the training dataset T. If the quality of the labels is poor, then it is unlikely that the accuracy of the classifier could be high. In most cases, the label of each instance is given by an expert or a labeler. In the literature of classification including traditional classification and active learning, it is implicitly assumed that the quality of the labels in the training dataset is guaranteed. However, it may not always be satisfied in real-life applications.

In many real-life applications, the label of each instance in a training dataset comes naturally with *probability* because *uncertainty* occurs in its label. When each target variable in the dataset is displayed as the corresponding probability, we call this dataset the *probabilistic dataset*.

Crowdsourcing: In crowdsourcing applications where each label is given by multiple labelers [1], the proportion of labelers giving a particular label corresponds to the probability that the instance has this label.

Medical Diagnosis: In the medical diagnosis application, after a patient undergoes a medical examination, his/her medical report shows some probabilities that s/he suffers from some diseases. For example, paying 15RMB for the electrocardiogram (ECG) test allows the patient to check whether s/he suffers from Coronary Heart Disease with 50%-60% probability [2].

Pattern Recognition: In the pattern recognition application, each instance can be regarded as an image, and attribute set X and attribute Y can be regarded as its features and the class/group it belongs to, respectively. In particular, in astronomy, due to the image resolution problem, we can only determine whether there is a volcano with some probability from an image. If the surface contains no summit visible but with evidence of flanks or circular outline, then a volcano exists with a probability equal to 0.6.

Natural Language Processing: In the natural language processing application, each of the 121 most frequent words is associated with 7.8 meanings on average [3]. In some sentences, some words have ambiguities due to the possibility of different meanings. According to the process of *word sense disambiguation* (WSD) [4] which is a topic still being studied in the literature of natural language processing, some ambiguities are filtered but there are still uncertainties of the meanings for some words. [4] stated that according to the previous results, about 65%-75.2% accuracy can be obtained if we want to estimate the meaning of a word in the context after the process of WSD is involved.

In most cases, the probabilistic dataset is more informative than the clear-cut dataset. Intuitively, since each label is associated with a probability, we can make use of the probabilistic information to find the "best" classifier. However, in the traditional classification on the clear-cut dataset, if the labeler is not sure about the label of a new instance, s/he may give a *single* possible label. Even if the labeler knows the

[*]The Hong Kong University of Science and Technology
[†]University of Illinois at Chicago

probability that the label of a new instance equals a particular value, due to the restriction on the traditional classification model, s/he gives only a single value. Therefore, the original probabilistic information is lost in the clear-cut dataset.

Learning on Probabilistic Label: In this paper, we study the problem called *Learning on Probabilistic Labels (LoPL)*. Specifically, LoPL is defined as follows. Given a probabilistic training dataset T and a new instance, we want to find a classifier, denoted by h, and use h to estimate the *probability* that the content of the target attribute of the new instance is equal to a particular value.

There are two challenges in problem LoPL. The first challenge is that the probability of the label of each instance in the probabilistic dataset may be inaccurate or have a low quality. Apparently, the low-quality probabilities may reduce the accuracy of the classifier. In this paper, we address the above low-quality probabilities issue by considering the existence of "noise" in the probabilities. Thus, we have a *noise-insensitive classifier* to find the probability of the label. In our experimental results, even after we introduced some noise in the training dataset, the accuracy of the classifier is still high (e.g., at least 85% in the "wine-red" dataset when we introduced some noise in our experiment).

The second challenge is whether we can derive a theoretical bound about the relationship between the accuracy/error of a classifier and the number of instances needed in the probabilistic dataset. Interestingly, we derive a theoretical result that the error of the proposed classifier is $\tilde{O}(n^{-\frac{2+\gamma}{4}})$ where n is the total number of instances used for finding the classifier and γ is a noise parameter which is a non-negative real number. Note that a smaller γ means there is more *noise* in the dataset. To the best of our knowledge, this is the *first* work to derive a theoretical bound linking the accuracy of a classifier and the number of instances needed in the probabilistic dataset.

It is worth mentioning that a lot of recent studies [5, 6, 7, 8] can use our theoretical bound on the error of the classifier studied in the paper for their error bound analysis, giving solid theoretical guarantees of their approaches. For example, in [8], the authors proposed a novel SVM model which learns a classifier based on the scenario that only a portion of labels of the instances in the dataset are known. Since the objective function in their optimization algorithm belongs to a regularized empirical risk minimization, which is similar to ours, our technique can also be used to derive the error bound of their proposed SVM model. The difference between the minimization function in our paper and that in their paper is that other than the model complexity, the regularizer in their objective function also includes the difference between the true label proportion and the estimated label proportion, which is more complicated for the analysis. In [5], the authors investigated the conditions of a proper loss function for estimating the posterior probability from partially labeled data. We can naturally plug their strategy into our algorithm for finding a proper loss, and derive the error bounds based on the discovered proper loss. In [6], the authors studied the problem of estimating labels from label proportions from multiple different perspectives, proposing various strategies for solving this problem. Even though they have already analyzed the convergence bounds for their probability estimators, our theoretical analysis is rather different from theirs, which may provide a supportive evidence on their theoretical results. In [7], the authors studied the problem of learning from *group probabilities*, which is referred to as the posterior probabilities for a subset of the whole dataset. Since they did not provide a theoretical guarantee on their estimated probabilities on the whole dataset, our theoretical analysis can be adaptively used for analyzing their problem as well.

Contributions: We have the following contributions. Firstly, to the best of our knowledge, we are the first to provide a theoretical bound on the error of the classifier in the probabilistic dataset. Specifically, the error of our proposed classifier is $O(\frac{1}{n})$ which has the same or at least better complexity bound compared with the bounds in the clear-cut dataset. Secondly, we propose a classifier on probabilistic datasets for problem LoPL by using one of the regression models. Any regression models whose regression function space is convex [9] (e.g., any algorithms in a general class of penalized least square estimators) can also be used for this classifier so that a theoretical bound can be derived. This is an important feature, since it opens the opportunity of leveraging the rich literature of regression models which give a tight theoretical bound for our classifier. In this paper, for illustration, we make use of a non-parametric regression model, Gaussian Process Regression (GPR) [9], for this classifier. Although the *exact* bounds for different regression models are different, we can still obtain the same *complexity* bound of $\tilde{O}(n^{-\frac{2+\gamma}{4}})$. Thirdly, we conducted experiments to show that in general, the accuracy of our classifier is higher than not only the accuracy of the traditional classifier and some other comparable classifiers.

The rest of this paper is organized as follows. We describe the related work in Section 2. We then give our problem definition in Section 3. In Section 4, we introduce our algorithm on the probabilistic training dataset. In Section 5, we present our experimental results. In Section 6, we give a discussion on this paper. Finally, we conclude our work and discuss some future works in Section 7.

2 Related Work

2.1 Classification with Non-Classical Labels

There are many different forms of *non-classical* labels for classification [10, 11, 12, 13]. A non-classical label is the label of an instance in the training dataset which is not just equal to a single value such as 0 and 1.

[10] studied problem LoPL where the training dataset T is probabilistic where each instance is associated with probability. To the best of our knowledge, [10] is the first work studying problem LoPL and proposed an *order-based* classifier. Specifically, [10] first sorts the instances in T in descending order of the probabilities of their labels. Let O be this ordering of these instances. Then, it learns a ranking function $r(\cdot)$ which takes the features of an instance as an input and outputs a score such that the ranking function satisfies two conditions, namely the *order-preserving* condition and the *target-consistent* condition. The order-preserving condition requires that for any two instances I and I' in T, whenever I appears before I' in O, the ranking score of I is greater than that of I'. The target-consistent condition requires that the target attribute of each instance is consistent with the ranking score of this instance. This condition corresponds to the constraint commonly used in the traditional algorithm of support vector machines. Specifically, for any instance I with its features $\mathbf{x}$ and its label y in T, $(r(\mathbf{x}) - b) \cdot y \geq 1$ where b is a real number to be learned.

In [10], $r(\cdot)$ is defined in form of a weighted sum of all features and thus the weights involved in the ranking function are to be learnt. In order to learn the ranking function $r(\cdot)$, [10] proposed the quadratic programming approach. Since in some cases, it is difficult to satisfy all conditions, in this approach, *slack variables* are introduced to each condition so that some conditions can be violated. After the ranking function $r(\cdot)$ is learnt, given a new instance I, [10] estimates the label of I according to the linear discriminator whose weights are equal to the weights used in the ranking function.

However, there are several weaknesses of this approach. Firstly, there is no theoretical bound linking the accuracy of a classifier and the number of instances used in T. However, in this paper, we find such a theoretical bound. Secondly, the order-based approach is not accurate when the dataset is not linearly separable. This is because the order-based approach makes use of the linear discriminator which does not perform well in non-linearly separable datasets. However, our proposed classifier is *non-parametric,* which means that our classifier is accurate in not only linearly separable datasets but also non-linearly separable datasets.

[11] is another study related to ours. Although [11] studied the classification problem from a very different perspective, it is worth mentioning it here. In [11], the authors studied the classification problem on the training dataset with *partial labels*. Specifically, for each instance I in the training dataset, I is associated with not only a set of features but also a set of labels (instead of a single label). [11] assumed that for each instance in this dataset, the set of labels associated with this instance contains one correct label. Under this problem setting, [11] proposed a quadratic programming approach for estimating the label of a new instance. Clearly, [11] is different from us although the set of labels associated to each instance in the training dataset can be thought as uncertainty or probability. However, there is no information about the exact probabilities of all labels in the label set. The best adaption is to set the probability of each label in the set to be $\frac{1}{k}$ where k is the number of labels in a label set. It is expected that the accuracy of a classifier on this dataset (which contains labels with equal probabilities for each label set) is smaller than that of the classifier on the dataset with real probabilities.

Some studies [12] allow the labeler to give a value not in the domain of the target attribute to the label of an instance. One example is [12] in which the labeler can give not only all possible values from the domain of the target attribute when s/he is confident to give the label but also a special value called "difficult" when s/he is not confident. This is different from ours since [12] does not consider any probabilistic information in the training dataset.

Recently, multi-label classification has been studied extensively in the literature [13]. In multi-label classification, each instance in the training dataset is associated with a set of labels which means that this instance can belong to all labels in this set at the same time. For example, an image can belong to multiple classes (e.g., sun and sea). Note that this problem is different from [11] which assumes that each instance belongs to *exactly one* label even though each instance is associated with a set of labels.

2.2 Traditional Classification Some studies about classification [14] gave theoretical bounds linking the accuracy of a classifier and the number of instances used in a clear-cut dataset. In [14], the upper bound on the excess error of a classifier on a clear-cut training dataset is $O(\frac{\log n}{n})$ if the dataset follows the *realizability* assumption (i.e., there exists a classifier which can *perfectly* classify any dataset generated from a distribution). With the help of the *chaining* technique, the term $\log n$ on the numerator can be eliminated, and thus the upper bound on the excess error has the order of n equal to -1. In a more general case that the data distribution does not follow the realizability assumption, the error of a classifier becomes $O(\sqrt{\frac{\log n}{n}})$ and thus has the order of n equal to $-\frac{1}{2}$ after the term $\log n$ is removed with the chaining technique. In the literature, introducing a noise condition in a training dataset can bridge these two bounds. One example of a noise condition is Tsybakov Noise Condition [15]. Tsybakov [15] proposed a parameterized condition to model the noise, with which the error of a classifier has the order of n between -1 and $-\frac{1}{2}$, depending on how much noise the dataset has. Intuitively, if the dataset has no noise, then the order is -1. If the dataset has too much noise, then the order is $-\frac{1}{2}$. In general, if the dataset has some noise, the order is a value between -1 and $-\frac{1}{2}$. Our work also studies the error complexity of a classifier with a variation of Tsybakov Noise Condition. The comparison between our result and the existing results is presented in Section 4.

3 Problem Definition

Traditional Setting: Consider a binary classification with two classes 0 and 1. In the *traditional* setting, we are given a training dataset T called a *clear-cut dataset* containing n instances, namely $I_1, I_2, ..., I_n$. Each instance I_i is associated with a feature vector $\mathbf{x}_i$ and a target attribute y_i where $i = 1, 2, ..., n$. Let $\mathcal{X}$ be the set of all possible feature vectors. Note that there are two possible values, namely 0 and 1, in the target attribute. A *classifier* $h(\cdot)$ is defined to be a *hypothesis* or a function which takes a feature vector $\mathbf{x}$ as an input and outputs either 0 or 1. In the following, for clarity, in some cases, we simply denote $h(\cdot)$ by h. In the traditional setting, an accurate classifier is to be found based on T.

In order to give a theoretical analysis on the *accuracy* of a classifier learned from T, following [14, 16, 17, 18, 19], we assume the following process of data generation. Let X be the random variable denoting the feature vector of an instance and Y be the random variable denoting the target attribute of an instance. We assume that all instances are generated according to an underlying joint distribution on two random variables X and Y, denoted by $P(X, Y)$. Given a feature vector $\mathbf{x}$, we denote $\eta(\mathbf{x})$ to be the conditional probability $P(Y = 1|X = \mathbf{x})$, the probability that an instance with its feature $\mathbf{x}$ has its target attribute equal to 1. We also assume that the data points in the dataset are *identically independently distributed (i.i.d.)* when the joint distribution is considered. That is, the generation of an instance is independent of the generation of another instance.

There are many measurements to evaluate the accuracy of a classifier. In this paper, we adopt the *excess error* as a measurement since it was adopted extensively in the literature [14]. Roughly speaking, the excess error of a given classifier corresponds to the "difference" between the *expected error* generated by this classifier and the *expected error* generated by the "best" classifier.

Given a classifier h, the *expected error* of h, denoted by $err(h)$, is defined to be $P_{(\mathbf{x},y)\sim P(X,Y)}(y \neq h(\mathbf{x}))$. The *Bayes classifier*, denoted by h^*, is defined to be the classifier which gives the minimum expected error. Note that $h^* = \mathbb{I}_{\eta(\mathbf{x})\geq 0.5}$. Given a classifier h, the *excess error* of h is defined to be the difference between its expected error and the expected error of h^*. That is, the *excess error* of h, denoted by $E(h)$, is equal to $err(h) - err(h^*)$. Note that $E(h)$ must be non-negative.

Our Setting: In our setting, we are given the *probabilistic dataset* T_f instead of the *clear-cut dataset*. Similar to the traditional setting, T_f contains n instances, namely $I_1, I_2, ..., I_n$. Each instance I_i is associated with a feature vector $\mathbf{x}_i$ and a *fractional score* f_i (instead of a target attribute) where $i = 1, 2, ..., n$. This fractional score is a real number ranging from 0 to 1 and corresponds to the likeliness that this instance belong to class 1. If this score is near to 1, then it is likely that this instance belongs to class 1. If it is near to 0, then it is unlikely that this instance belongs to class 1 and instead, it is likely that this instance belongs to class 0. This score can be obtained by labelers and some statistical information (e.g., the medical statistical history) described in Section 1. Note that if each f_i is equal to either 0 or 1 where $i = 1, 2, ..., n$, then the probabilistic dataset is equivalent to the clear-cut dataset.

Consider an instance with its feature x and its fractional score f. Since its fractional score f is obtained by labelers and some statistical information, it can be regarded as an "observed" version of $\eta(\mathbf{x})$, the *true* probability that this instance belongs to class 1, and thus it may deviate from $\eta(\mathbf{x})$. Following some existing studies [9], we model the deviation with the *Gaussian white noise*. Specifically, the *Gaussian white noise* is represented in the form of Gaussian distribution $\mathcal{N}(0, \sigma^2)$ where σ is a standard deviation of this distribution. With this noise condition, each fractional score f_i in the training dataset follows the distribution of $\mathcal{N}(\eta(\mathbf{x}), \sigma^2)$. Note that it is possible that a value randomly sampled from the distribution $\mathcal{N}(\eta(\mathbf{x}), \sigma^2)$ is out of the range between 0 and 1. In this case, the value can be truncated accordingly. Specifically, if this value is smaller than 0, then it can be assigned to 0. If this value is larger than 1, then it can be assigned to 1. However, in our theoretical analysis, we simply adopt the distribution of $\mathcal{N}(\eta(\mathbf{x}), \sigma^2)$ in order to simplify our theoretical analysis. Considering the truncation method is also possible but yields uninteresting and tedious "boundary" cases.

Problem LoPL: We define our problem called *L̲earning on P̲robabilistic L̲abels (LoPL)* as follows. Given a probabilistic training dataset T_f, we want to learn a classifier h such that whenever we see a new instance I which has the information about its features $\mathbf{x}$ but no information about its label, we can estimate the conditional probability $\eta(\mathbf{x})$ of I, where the estimated probability is denoted by $\hat{\eta}(\mathbf{x})$, and then can give a label (either 0 or 1) to instance I according to $\hat{\eta}(\mathbf{x})$.

Intuitively, when $\hat{\eta}(\mathbf{x})$ is very similar to $\eta(\mathbf{x})$, our classifier performs as good as the optimal achievable classifier (i.e., Bayes classifier). So, the key of solving the proposed problem well is to determine an accurate estimation of $\eta(\mathbf{x})$.

4 Fractional Score-based Classifier

In this section, we employ one of the regression models, namely *Gaussian Process Regression* [9], to obtain an accurate estimation $\hat{\eta}(\mathbf{x})$ of the conditional probability $\eta(\mathbf{x})$ for any $\mathbf{x}$. Any regression models whose regression function space is convex can also be used. We name the classifier h inferred from the estimated conditional probability $\hat{\eta}(\mathbf{x})$ as a *f̲ractional s̲core-based c̲lassifier* (FSC).

After we learn a function which estimates $\eta(\mathbf{x})$ accurately for any $\mathbf{x}$, when we see a new instance I with its features $\mathbf{x}$, FSC can predict the label of I as follows.

If $\hat{\eta}(\mathbf{x}) \geq \frac{1}{2}$, then FSC assigns the label of I to 1. Otherwise, it assigns the label of I to 0.

In the following, we propose to use a non-parametric estimation model called *Gaussian Process Regression (GPR)*.

4.1 Gaussian Process Regression (GPR) Gaussian process is a stochastic process which can be used as our estimator $\hat{\eta}(\mathbf{x})$ in order to estimate the conditional probability $\eta(\mathbf{x})$ for any instance I with its features $\mathbf{x}$. In order to simplify our discussion, we just give the simplest version of Gaussian Process Regression as an example to estimate the fractional score. There are many recent studies about more complicated versions of Gaussian Process Regression focusing on the scalability issue with large datasets. Considering the scalability issue with more complicated versions is an orthogonal issue in this paper. We use Gaussian process regression to find $\hat{\eta}(\mathbf{x})$ and $Var(\mathbf{x})$ given a feature vector $\mathbf{x}$. Since Gaussian process regression is a well-known technique [20, 21], we do not include the detailed expressions in this section. Interested readers can read the appendix of this paper.

4.2 Theoretical Analysis In this section, we give the theoretical result linking the *excess error* of our classifier and the number of instances needed in the training dataset.

In order to give a tight theoretical bound on the error, we want to capture a characteristic of the *distribution* of probabilities, $\eta(\cdot)$, in the training dataset. This can be captured by a noise condition called *Tsybakov's Noise Condition* [15], which has been widely adopted. In the following, we study a variation of Tsybakov's Noise Condition, which leads to a tight error bound in our result.

DEFINITION 1. (NOISE CONDITION) *Given two noise parameters $c > 0$ and $\gamma \geq 0$, for any $t \in (0, 0.5)$,*

$$Pr(\mathbb{E}[|\eta(\mathbf{x}) - \tfrac{1}{2}|] < t) < c \cdot t^{\gamma} \tag{4.1}$$

Both c and γ are the parameters for describing the distribution which the probabilities, $\eta(\mathbf{x})$, in the dataset follow. The noise condition can be explained as follows. Suppose that c and γ are known. Note that $\eta(\mathbf{x})$ is in the range between 0 and 1. If t is near to 0, then the inequality in the condition states that the probability that $\eta(\mathbf{x})$ is close to $\frac{1}{2}$ is very small. If t is near to 0.5, then the inequality means that the probability that $\eta(\mathbf{x})$ is close to either 0 or 1 is very large.

As we described before, c and γ are used to describe the distribution of the probabilities, $\eta(\mathbf{x})$. If c is smaller, then it is less likely that $\eta(\mathbf{x})$ is near to $\frac{1}{2}$. If γ is smaller, then t^{γ} will be larger (when $t \in (0, 0.5)$). In this case, it is more likely that $\eta(\mathbf{x})$ is near to $\frac{1}{2}$ (which can be considered that there is more noise in the dataset). Thus, if c is very small and γ is very large, it is less likely that $\eta(\mathbf{x})$ is near to $\frac{1}{2}$.

THEOREM 4.1. (ERROR BOUND) *Let h be our FSC classifier. Given a confidence parameter $\delta \in [0, 1]$, there exist three constants C_1, C_2 and C_3 which are independent of n and δ such that with probability at least $1 - \delta$,*

$$E(h) \leq 2\sqrt{c}\left(\frac{C_1 + C_2 \ln n + C_3 \ln \frac{1}{\delta}}{n}\right)^{\frac{2+\gamma}{4}}$$

Proof: The proof can be found in the appendix.

According to Theorem 4.1, it is easy to verify that $E(h) = O((\frac{\ln n}{n})^{\frac{2+\gamma}{4}})$. Following the notation convention adopted in the literature since $\ln n = O(n)$, we write $E(h) = \tilde{O}(n^{-\frac{2+\gamma}{4}})$ where the $\tilde{O}$ notation is used to hide the term $\ln n$. We say that the *order* of n for the bound of $E(h)$ is $-\frac{2+\gamma}{4}$. Note that if the order is more negative, then $E(h)$ is smaller and thus h is more accurate. Since $\gamma \geq 0$, it is easy to verify that the order of n ranges from $-\infty$ to $-\frac{1}{2}$.

In general, the error bound $E(h)$ is no worse than the existing error bounds under the realizability setting, the non-realizability setting and the Tysbakov's noise setting based on clear-cut datasets. Under the realizability setting, our error bound is no worse than the best-known error bound [22] (i.e., $\tilde{O}(n^{-1})$) when $\gamma \geq 2$. When $\gamma \geq 2$, the order of n for our bound ranges from $-\infty$ to -1, which is smaller than or equal to -1, the order of n for the best-known bound.

Under the non-realizability setting, our error bound is always no worse than the best-known error bound [22] (i.e., $\tilde{O}(n^{-\frac{1}{2}})$) since the order of n for our error bound is at most $-\frac{1}{2}$, the order of n for the best-known bound.

Under the Tysbakov's noise setting, our error bound is still always no worse than the best-known error bound [22] (i.e., $\tilde{O}(n^{-\frac{\gamma+1}{\gamma+2}})$). Note that the order of n for this best-known bound (i.e., $-\frac{\gamma+1}{\gamma+2}$) ranges from -1 to $-\frac{1}{2}$ and is at least $\frac{2+\gamma}{4}$, the order of n for our error bound. Thus, our error bound is no worse than this best-known error bound.

5 Experiment

5.1 Experimental Setup We conducted experiments on a workstation with 1.60GHz CPU and 32GB RAM. The experiments are conducted on three types of real datasets. The first type of datasets includes a real crowdsoucring dataset. The second type of datasets includes a set of real datasets originally used for regression. The third type of datasets includes a real dataset originally used for classification.

First of all, let us consider a real crowdsoucring dataset for classification, in which the probabilistic labels can be obtained directly. The dataset we used here is the Yahoo!news dataset [23]. Each article in the Yahoo!news dataset corresponds to an instance, of which the feature vector is obtained after some common text preprocessing techniques like the stop words elimination and the TF-IDF transformation. The label of each article could be "politics","business" or "technique". All 895 articles (298 articles from the "politics" category, 298 articles from the "business" category and 299 articles from the "technique" category) in the dataset are crawled by the authors in [23] and the labels are provided by 5 human annotators. Thus, the probabilistic label for each instance is the average score of five 0-1 labels. Since we consider the problem of binary classification in the paper, we generated three binary datasets, namely the "business vs. politics" dataset, the "business vs. technique" dataset and the "politics vs. technique" dataset, by randomly choosing instances such that in each generated binary dataset, half of the instances belong to one category and half of the instances belong to one of the other categories. For instance, in the "business vs. politics" datasets, half of the instances belong to "business" and half of the instances belong to "politics". Thus, we have 3 binary datasets with probabilistic labels (called probabilistic datasets).

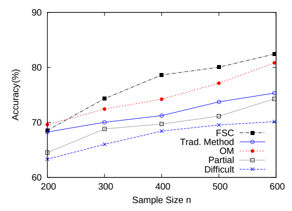
Consider the second type of real datasets. These real datasets originally used for regression are "cadata", "abalone", "wine-red", "wine-white", "solar flare", "auto-mpg", "housing" and "breast-cancer". The first one is from the StatLib Archive [24], while others are from the UCI repository [25]. Note that each real dataset is associated with feature attributes and a target attribute in the regression problem. It is obvious that in our problem setting, the feature attributes originally used for regression corresponds to the feature attributes for our problem. The normalized value of the target attribute of each instance originally used for regression, ranging from 0 to 1, corresponds to the probability that the instance belongs to class 1. Each dataset containing these feature attributes and the probabilities corresponds to the probabilistic dataset T_o without any noise. This can be regarded as the ground-truth dataset in our problem setting. However, as we described in Section 3, we are only given an "observed" version of the probabilistic dataset, says T_f. Thus, we generate T_f by adding a noise value randomly picked from $\mathcal{N}(0, \sigma^2)$ to each probability. Each added value corresponds to a fractional score in T_f. Note that if the added value is greater than 1, we re-set this value to 1. If it is smaller than 0, we re-set this value to 0. In all experiments, we set $\sigma = 0.02$ as the default values if we do not specify its value.

For the third type of datasets, we adopt the movie review dataset [1], which has the same setting as our problem setting. In the movie review dataset, each movie corresponds to an instance where its feature attributes are the vocabularies from the reviews provided by the IMDb users, its fractional score is the average normalized user rating and its target attribute is the sentiment porality of the movie reviews. This dataset was originally used for Movie Review Sentiment Classification. In the dataset, there are 2000 movies where 1000 movies are labeled as "positive" and the remaining 1000 movies are labeled as "negative". Since each movie in the dataset was rated by over 50 thousands users in *IMDb.com* and according to the concept of crowdsourcings, the average rating is closely related to the probability that a person has a positive impression on this movie.

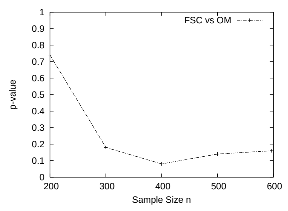
We implemented our proposed classifier called *FSC*. We compared it with four other comparative classifiers, namely (1) the *traditional method* (*Trad. Method*), (2) the *order-based method* (*OM*) [10], (3) the *partial method* (*Partial*) [11] and (4) the *difficult method* (*Difficult*) [12]. The first method is the traditional approach while the last three methods were discussed in Section 2. (1) The traditional method was implemented by the support vector machine (SVM). Note that since this traditional method is based on the clear-cut dataset, according to T_o, we generated the corresponding clear-cut dataset T_c by setting the target attribute value of each instance to 0/1 labels probabilistically according to the probabilities in T_o. (2) The order-based method is based on T_f. (3) The partial method is based on a dataset where each target attribute may contain multiple labels. We generated this dataset from T_f by assigning the target attributes of all instances with their scores in the range in $[0.3, 0.7]$ (which can be regarded as an uncertain region) to $\{0, 1\}$, those with their scores in the range in $[0, 0.3)$ to $\{0\}$ and those with their scores in the range in $(0.7, 1]$ to $\{1\}$. Then, we set the probability of the label of each instance to be $1/k$ where k is the number of labels in the target attribute of this instance. After that, according to these probabilities, we can use the partial method for classification. (4) The difficult method is based on a dataset containing 0, 1, and "difficult" tags. We generated this dataset from T_f by labeling all instances with their scores in the range in $[0.3, 0.7]$ to "difficult", those with their scores in the range in $[0, 0.3)$ to 0 and those with their scores in the range in $(0.7, 1]$ to 1. Note that this method is originally designed for multiple labelers. If the label of an instance given by a labeler is "difficult", its label will be estimated by other labelers who give its label as either 0 or 1. Since our problem is for a single labeler and no other labelers are involved, we adapted the difficult method by removing all instances with "difficult" labels and trained a classifier based on the remaining instances.

We performed a 10-fold *cross-validation* for these experiments. In particular, the training set was randomly partitioned into 10 pieces, each of which was held out for testing in one of the ten folds, while the remaining nine pieces were collected for training. We evaluated a classifier in terms of its average accuracy on the held-out test set. We repeated

[1]http://www.cs.cornell.edu/people/pabo/movie-review-data/

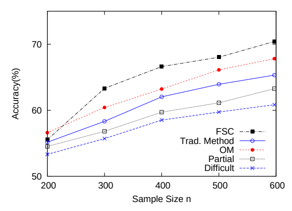


(a) The average accuracy

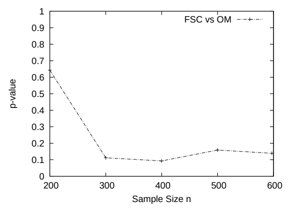


(b) The p-value of paired t-test: FSC vs OM

Figure 1: Performance of different classifiers on the “business vs. politics” dataset from the Yahoo!news dataset



(a) The average accuracy



(b) The p-value of paired t-test: FSC vs OM

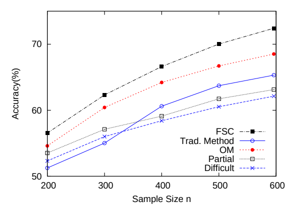
Figure 2: Performance of different classifiers on the “business vs. technique” dataset from the Yahoo!news dataset

the cross-validation four times for each classification algorithm, so each reported value is an average of 40 results, and the residual error for each point in the figure is the standard deviation of the corresponding 40 results. For the last experiment, we performed a 10-fold cross-validation from the sampled training set, and also repeated the cross-validation four times for each training dataset.

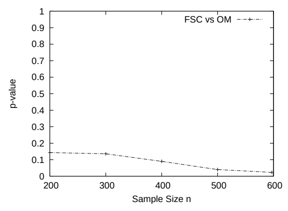
5.2 Experimental Results

5.2.1 Results for the First Type Datasets In this part, we compared our FSC algorithm with the four competitive algorithms in the three binary dataset from the Yahoo!news dataset.

Comparison Among All Algorithms: We compare the accuracies of the algorithms when we vary the number of instances in each binary dataset with probabilities. The results on the “business vs. politics” binary dataset, the “business vs. technique” binary dataset and the “politics vs. technique” dataset are shown in Figure 1(a), Figure 2(a) and Figure 3(a), respectively. FSC performs the best among all algorithms in all these datasets. In particular, FSC has a higher accuracy than the *order-based* method (OM) since the order-based method does not perform well in non-linear separable datasets like the datasets used in the experiment, which is usually the case. Compared with the traditional classification algorithm (i.e., SVM), FSC always has a higher accuracy in all cases. The accuracy of the *difficult* method is not that desirable in most cases, even worse than the traditional method, since only a single labeler is involved. FSC also outperforms the *partial* method which has no information about the true fractional scores/probabilities. Besides, since the accuracies of OM is closest to the accuracies of FSC among all other baseline algorithms in most cases, we compute the p-value of the paired t-test, denoted by “FSC vs OM” to compare the significance of the “difference” between the accuracy of FSC and the accuracy of OM for each Yahoo!news dataset. The p-value is a measurement in the paired t-test, ranging from 0 to 1. When it is close to 0, we say that FSC is better than OM in a statistical significance. The results on these datasets can be found in Figure 1(b), Figure 2(b) and Figure 3(b). In the figures, we can see that the p-values in the figure are close to 0 in most cases, which indicates that FSC is statistically much better than OM.



(a) The average accuracy



(b) The p-value of paired t-test: FSC vs OM

Figure 3: Performance of different classifiers on the “politics vs. technique” dataset from the Yahoo!news dataset

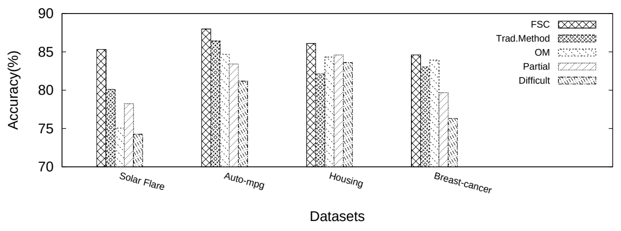
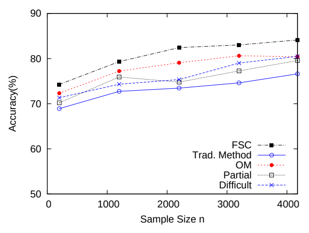


Figure 4: Accuracies of different algorithms on different datasets

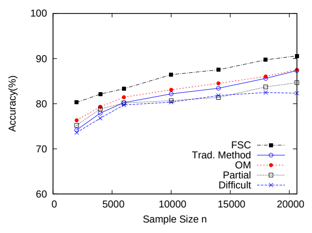
5.2.2 Results for the Second Type Datasets Comparison Among All Algorithms: We compared our FSC algorithm with other algorithms in the second type datasets. Figure 4 shows this result in four second-type datasets, namely “solar flare”, “auot-mpg”, “housing” and “breast-cancer”. Similarly, FSC outperforms the other competitive algorithms in general.

In order to give more experiments on different datasets, in the following, we conducted experiments on other datasets, namely “abalone”, “cadata”, “wine-red” and “wine-white”, to study the effect of different factors.

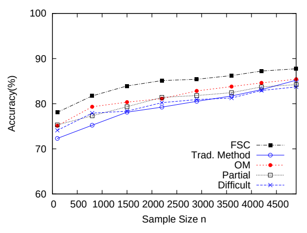
Effect of Sample Size: In Figure 5, we study how the sample size affects the accuracies of all algorithms. In this figure, the accuracies of all algorithms increase with the sample size n. Similarly, our FSC classifier performs the best.



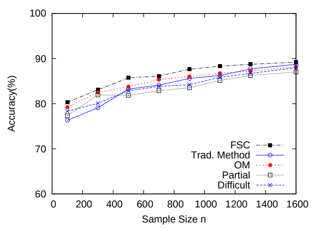
(a) abalone



(b) cadata

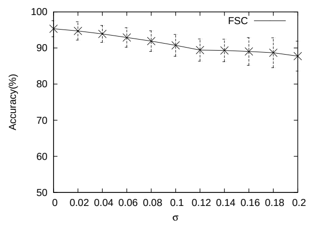


(c) wine-white

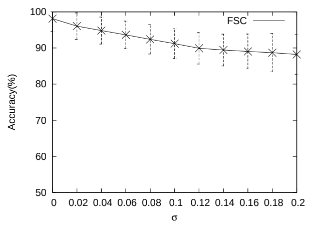


(d) wine-red

Figure 5: Effect of sample size n on the accuracies of different classifiers

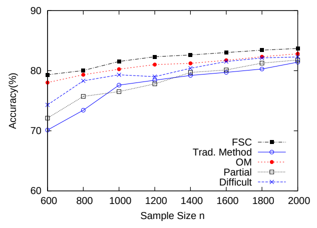


(a) wine-white

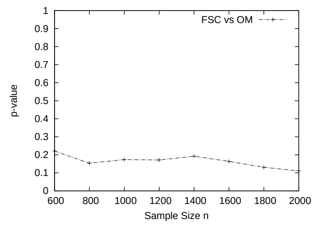


(b) wine-red

Figure 6: Effect of σ on the accuracy of our FSC classifier



(a) probabilistic data



(b) clear-cut data

Figure 7: Performance of different classifiers on the movie review dataset with the paired t-test: FSC vs OM

Effect of Noise: We want to study the accuracy of our FSC classifier when some noise is introduced. For the interest of space, we show the results on only two datasets, namely wine-white and wine-red. Figure 6 shows that the accuracy of FSC is still high even when σ, denoting the standard deviation of the Gaussian noise introduced to the training set, increases. In Figure 6, when σ increases from 0 (i.e., no error) to 0.2, the accuracy of FSC decreases. However, we can see that the accuracy of FSC will not be affected too much when some noise exists in the dataset.

5.2.3 Results for the Third Type Dataset

For the third type of dataset (i.e., movie review dataset), we conducted the experiment which is the same as the one conduced on the Yahoo!news dataset. Specifically, we vary the number of movie instances in the training dataset. Each time we randomly pick half of the movie instances whose reviews are "positive" and the remaining half are the movie instances labeled "negative". Through Figure 7(a), we can see that FSC performs much better than other five algorithms in terms of accuracy. Note that OM still performs slightly better than the other four algorithms. Thus, we continued to compute the p-value of the paired t-test, denoted by "FSC vs OM", comparing the statistical significance on the results of FSC and OM shown in Figure 7(a). Figure 7(b) shows that FSC is better than OM since the p-values are close to 0.

Conclusion: FSC has the greatest accuracy among all algorithms in most cases. FSC has a better accuracy than not only the traditional method but also the order-based method studying problem LoPL. Besides, we also compared FSC with other two methods which are related to ours, namely the partial method and the difficult method. The accuracy of these two approaches are lower than that of FSC. Besides, the accuracy of FSC is not affected too much even if we introduce some noise.

6 Discussion

In Section 3, we assume that the "observed" fractional score of each instance follows a normal distribution with mean equal to the true fractional score of this instance. We call this assumption the *good-quality assumption*. Under this assumption, we say that each label has a *good quality*. Even though we consider that the "observed" fractional score can deviate from the true fractional score with a normal distribution, the "observed" fractional score which is given by a human, a labeler, may be *over-estimated* or *under-estimated*. In other words, in these cases, the "observed" fractional score of each instance may have the mean not equal to the true fractional score. However, we have the following two reasons to support that our good-quality assumption makes sense in real-life applications.

The first reason is that it is reasonable to assume that the quality of each label in the probabilistic training dataset is good because most existing studies about traditional classification have the same assumption on the *quality* of the clear-cut training dataset. Most existing studies about the traditional classification assumes that the qualities of the labels of most instances are correct and thus the quality of the dataset is considered as good. Some studies [26, 27] consider that the quality of the clear-cut training dataset is *not very* good, which means that the labels of *some* instances are incorrect and can be regarded as the non-realizability setting, but they also assume that the labels of *most* instances are correct and thus the quality of the dataset is rather good.

The second reason is that the "observed" fractional score of each instance can be obtained from external statistical data sources and thus the quality of the dataset can be considered as good. In Section 1, we discussed a lot of applications which come with statistical data sources. For example, the medical diagnosis application can have the statistical sources about the likeliness of suffering from a disease. The pattern recognition application can have the scientistic statistics of determining the chance of the volcano existence.

7 Conclusion and Future Work

In this paper, we studied an interesting classification on the probabilistic dataset, which has a lot of real-life applications. We showed the theoretical error bound is at least no worse than the best-known error bound in the traditional classification problem in the asymptotic sense. Finally, we conducted experiments showing that our proposed classifier is better than other comparable classifiers in terms of accuracy.

There are a lot of interesting future works. Firstly, studying active learning, online learning and transfer learning in the probabilistic dataset is a possible direction. Secondly, considering the setting of semi-supervised learning is also possible since here, we focused on supervised learning.

Acknowledgements: The research of Peng Peng and Raymond Chi-Wing Wong is supported by grant FS-GRF13EG27. The research of Philip S. Yu is supported in part by US NSF through grants CNS-1115234, DBI-0960443, and OISE-1129076, and US Department of Army through grant W911NF-12-1-0066.

References

[1] K. Crammer, M. Kearns, and J. Wortman, "Learning from multiple sources," *The Journal of Machine Learning Research*, vol. 9, pp. 1757–1774, 2008.

[2] J. Schwartz, R. Johnson, F. Aepfelbacher, J. Parker, L. Chen, R. Azar, R. Parker, and P. Danias, "Sensitivity, specificity and accuracy of stress spect myocardial perfusion imaging for detection of coronary artery disease in the distribution of first-order branch vessels, using an anatomical matching of angiographic and perfusion data," *Nuclear medicine communications*, vol. 24, no. 5, p. 543, 2003.

[3] E. Agirre and P. Edmonds, *Word Sense Disambiguation: Algorithms and Applications*. Springer, 2006.

[4] G. Paab and F. Reichartz, "Exploiting Semantic Constraints for Estimating Supersenses with CRFs," in *SDM*, 2009.

[5] J. Cid-Sueiro, "Proper losses for learning from partial labels," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1574–1582.

[6] N. Quadrianto, A. J. Smola, T. S. Caetano, and Q. V. Le, "Estimating labels from label proportions," *The Journal of Machine Learning Research*, vol. 10, pp. 2349–2374, 2009.

[7] S. Rueping, "Svm classifier estimation from group probabilities," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 911–918.

[8] F. X. Yu, D. Liu, S. Kumar, T. Jebara, and S.-F. Chang, "propotion-svm for learning with label proportions," *arXiv preprint arXiv:1306.0886*, 2013.

[9] C. Rasmussen, "Gaussian processes in machine learning," *Advanced Lectures on Machine Learning*, pp. 63–71, 2004.

[10] Q. Nguyen, H. Valizadegan, and M. Hauskrecht, "Learning classification with auxiliary probabilistic information," in *International Conference on Data Mining (ICDM)*. IEEE, 2011.

[11] N. Nguyen and R. Caruana, "Classification with partial labels," in *KDD*. ACM, 2008, pp. 551–559.

[12] B. C. Wallace, K. Small, C. E. Brodley, and T. A. Trikalinos, "Who should label what? instance allocation in multiple expert active learning." in *SDM*. SIAM / Omnipress, 2011, pp. 176–187.

[13] G. Tsoumakas and I. Katakis, "Multi-label classification," *International Journal of Data Warehousing & Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[14] M. Anthony and P. L. Bartlett, *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

[15] A. Tsybakov, "Optimal aggregation of classifiers in statistical learning," *The Annals of Statistics*, vol. 32, no. 1, pp. 135–166, 2004.

[16] M. Balcan, A. Beygelzimer, and J. Langford, "Agnostic active learning," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 65–72.

[17] A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang, "Agnostic active learning without constraints," *Arxiv preprint arXiv:1006.2588*, 2010.

[18] S. Dasgupta, "Coarse sample complexity bounds for active learning," *NIPS*, 2006.

[19] S. Hanneke, "A bound on the label complexity of agnostic active learning," in *ICML*, 2007.

[20] M. Deisenroth, C. Rasmussen, and J. Peters, "Gaussian process dynamic programming," *Neurocomputing*, vol. 72, no. 7-9, pp. 1508–1524, 2009.

[21] M. Lázaro-Gredilla and M. Titsias, "Variational heteroscedastic gaussian process regression," in *28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 841–848.

[22] P. Massart and É. Nédélec, "Risk bounds for statistical learning," *The Annals of Statistics*, vol. 34, no. 5, pp. 2326–2366, 2006.

[23] S. Xie, W. Fan, and S. Y. Philip, "An iterative and reweighting framework for rejection and uncertainty resolution in crowdsourcing," in *SDM*, 2012.

[24] "Statlib," 2012. [Online]. Available: http://lib.stat.cmu.edu/datasets/

[25] A. Frank and A. Asuncion, "UCI machine learning repository," 2010.

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

[27] V. Sheng, F. Provost, and P. Ipeirotis, "Get another label? improving data quality and data mining using multiple, noisy labelers," in *KDD*, 2008.