A Novel Class Noise Estimation Method and Application in Classification

Lin Gui[1], Qin Lu[2], Ruifeng Xu[1*], Minglei Li[2], Qikang Wei[1]
1.Laboratory of Network Oriented Intelligent Computation,
Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China
2.Department of Computing, the Hong Kong Polytechnic University, Hong Kong, China
guilin.nlp@gmail.com; csluqin@comp.polyu.edu.hk; xuruifeng@hitsz.edu.cn;
csmli@comp.polyu.edu.hk; weiqikang@hotmail.com

ABSTRACT
Noise in class labels of any training set can lead to poor classification results no matter what machine learning method is used. In this paper, we first present the problem of binary classification in the presence of random noise on the class labels, which we call class noise. To model class noise, a class noise rate is normally defined as a small independent probability of the class labels being inverted on the whole set of training data. In this paper, we propose a method to estimate class noise rate at the level of individual samples in real data. Based on the estimation result, we propose two approaches to handle class noise. The first technique is based on modifying a given surrogate loss function. The second technique eliminates class noise by sampling. Furthermore, we prove that the optimal hypothesis on the noisy distribution can approximate the optimal hypothesis on the clean distribution using both approaches. Our methods achieve over 87% accuracy on a synthetic non-separable dataset even when 40% of the labels are inverted. Comparisons to other algorithms show that our methods outperform state-of-the-art approaches on several benchmark datasets in different domains with different noise rates.

Categories and Subject Descriptors
I.5.1 [**PATTERN RECOGNITION**]: Models--- Statistical;

General Terms
Algorithms, Design, Experimentation.

Keywords
Class Noise, Learning with Noise, Noise Elimination.

1. INTRODUCTION
A typical machine learning method uses a classifier learned from a labeled dataset (i.e., the training data) to predict the class labels of new samples (i.e., the testing data). In most of the classification applications, the labels of the training data are assumed correct. However, real-world datasets often contain noise which may occur either in the feature set of the data, referred to as the **attribute noise**, or in the labels of the data, referred to as the **class noise**.

CIKM'15, October 19-23, 2015, Melbourne, VIC, Australia
© 2015 ACM. ISBN 978-1-4503-3794-6/15/10…$15.00
DOI: http://dx.doi.org/10.1145/2806416.2806554

Many studies have focused on dealing with attribute noise since it is quite common in data mining applications and machine learning tasks. Researchers such as [1, 2] have indicated that class noise can potentially be more detrimental than attribute noise. The importance of addressing the class noise problem is not limited to its adverse impact on classification performance [1]. We must point out that class noise is unavoidable in many real world applications including disease prediction in medical applications [3], food labeling for the food industry [4], and manual data labeling in many natural language processing applications [5].

To handle class noise in learning algorithms, the first issue is how to estimate the probability of class noise, referred to as the **class noise rate.** The second issue is how to train a classifier knowing that the training data contains class noise.

Generally speaking, there are two types of strategies to deal with class noise. The first entails learning with noise data and the second is based on noise elimination. In the **learning with noise method,** each training sample is assigned a weight based on an estimated probability of class noise, that is, the class noise rate. The learning algorithm will consider the class noise probability while learning from the original noisy training data [6]. Unfortunately, this method requires a priori knowledge of the class noise rate in the training data.

In the **noise elimination method**, attempts are first made to detect and remove erroneous data from the training set [7, 8, 9]. While this method can reduce the rate of class noise in the training data, it can also lead to the overall reduction of training samples. Because of the reduced training data, this method may lead to even worse results than when it is used for small training sets.

The performance of both methods relies on an appropriate estimation of class noise rate. In this paper, we first propose a novel method to estimate the noise rate of the training data at the level of individual samples and the estimation is done based on the sum of the Rademacher distribution [10] using k-Nearest Neighbors (k-NN) graphs. We then present two general methods to incorporate our proposed class noise estimation method into the two types of class noise handling strategies to reduce the impact of noisy data. Lastly, in the elimination based strategy, we introduce a sampling-based method to select high quality training data rather than to identify low quality data for elimination.

The rest of the paper is organized as follows. Section 2 provides a review of related works and basic definitions. Section 3 presents the problem setup and some background information. Section 4 presents our proposed class noise estimation method. Section 5 describes the methods to incorporate the proposed estimation into learning with noise strategy and the class noise elimination strategy. Section 6 presents performance evaluations based on

both synthetic and real-world public datasets in different domains. Section 7 provides the conclusion and future work.

2. Related work and Basic Definitions

Identifying class noise is an important subject in machine learning. Previous work covers both the theoretical [11] and the application aspects [12, 13] of this topic. In this section, we briefly introduce these works from three perspectives: the source of class noise, the handling of class noise, and the application of class noise.

Class noises exist for different reasons. When used for disease prediction in medical applications [3], training data contains a probability of false positive or negative because data comes from medical experiments. In other words, class noises naturally exist and cannot be avoided. Food labeling for the food industry [4] also faces class noise problem. As shown by [4], because some food has high price than others, such as beef has a higher price than mutton, miss-labeling is an aforethought to achieve more benefit. The manual labeling of data in many natural language processing applications [5] naturally contains class noise because there is always a possibility of inter-annotator inconsistency.

Given an x, being the set of features of a sample, $\tilde{y}$ is the observed label of x; y is the true label of the sample of x and p is the probability to flip the true label into a noise label (thus p is the class noise rate, or **noise rate** for short). For any training algorithm, only x and $\tilde{y}$ can be observed. In general, class noises can be categorized into three different models based the dependency of noise to y and p [14]. The most common model used is the *Noise Completely at Random Model* [11, 15, 16]. In this model, class noise rate of a sample is completely random and independent of the labels and the feature set. Thus, an observed label $\tilde{y}$ is only determined by the true label and class noise rate. The other models assume dependency to either x or y [17-21].

The theoretical discussion about class noise and learning was first proposed by Angluin and Laird in 1988 [11]. In their work, all instances of the labeled data for binary classification have a uniform probability $p \in [0,1/2)$ of being inverted. This is referred to as the random classification noise.

Class noises are typically assumed to be stochastic in algorithms that can handle class noises. The work by [6] assumes a learned noise rate from prior knowledge and every sample is given the same probability, a simple assumption that may not be reasonable in all scenarios. The Cut Edge Weight statistic [8] also requires a prior assumed noise rate. This method uses the prior probability as a hypothesis to test if the training sample satisfies the null hypothesis. The method also requires the neighbors of a training sample to follow the central limit theorem, which is not reasonable because the set of neighbors are too small. Other works simply did not consider noise rate [5, 7, 9, 22].

There are two basic categories of strategies to dealing with class noise in training data: learning with class noise [5, 6, 16] and class noise elimination [7, 8, 9]. The learning with noise strategy approximates a distribution of noiseless training data using the distribution of the original training data with class noise and a priori knowledge about the class noise [6]. The problem, however, is that a priori knowledge of the class noise is not generally available, limiting the applicability of this method. Li [16] use the Kernel Fisher method to estimate the class noise. Then the estimation is used to achieve a robust algorithm to tolerate noise.

The class noise elimination strategy attempts to detect the samples with high noise probabilities and remove them from the training set. There are different methods to detect class noise that can be categorized as classification-based methods and graph-based methods. The classification-based method was first proposed by Brodley in 1999 [7]. He used K-fold cross-validation to split the training set into K subsets, and used any K-1 sets as the training data to classify the remaining data. If the classification result for a sample is different from the original label, that sample is considered class noise and is removed. Zhu et al. proposed a more efficient algorithm using the same ideal and made it suitable for large datasets [22]. Zhu also proposed a cost-sensitive approach based on K-fold cross-validation [1]. Sluba employed a 10-fold cross-validation to detect class noise [9] for their elimination. There are two major problems with the elimination approach. First, some correctly labeled training data can be lost because of potentially inaccurate class noise identification. Second, the number of samples in the training data will be reduced, potentially leading to an adverse effect on the learning algorithm performance. In elimination-based methods, reliable noise estimation crucial and inaccurate estimations can potentially degrade performance compared to using the original noisy dataset.

The graph-based method, known as the Cut Edge Weight statistic method, was proposed by Zighed [8]. The principal idea is based on the manifold assumption and a Bernoulli distribution assumption on the consistency of the labels. A similar approach was proposed by Jiang and Zhou, who used a k-NN graph to detect class noise without the need to consider noise rate as probabilities [23].

Problem Setup and Background

Let D denotes the clean distribution with no class noise, and (x_1, y_1), (x_2, y_2),…, (x_n, y_n) denote n training samples from D with true binary label y_i ($y_i = \pm 1, i = 1,2, \ldots, n$). When there are class noises, $\widetilde{D}$ denotes the observed distribution which contains class noise, and the training sample from a noisy distribution $\widetilde{D}$ are $(x_1, \tilde{y}_1)$, $(x_2, \tilde{y}_2)$,…, $(x_n, \tilde{y}_n)$, where the label $\tilde{y}_i$ may be different from the true label y_i. In this paper, we want to estimate noise rate at the level of the individual samples rather than for the whole collection of training data. Thus, we give definition of class noise rate as follows:

Definition.1: Let $(x_i, \tilde{y}_i) \in \widetilde{D}$ be a sample from the noisy distribution, the class noise rate is the probability of the observed label different from the true label of x_i, denoted by $P(\tilde{y}_i \neq y_i | x_i)$.

According this definition, class noise rate is defined on individual data samples. Thus, the first issue we need to address is how to estimate the class noise rate for each training sample. The main challenge, however, is that a learner can only see the noisy data $(x_i, \tilde{y}_i) \in \widetilde{D}$, and there is no a priori knowledge about the class noise rate and the clean distribution D.

Assuming that we can find a reasonably good estimation method, the second issue we need to address is how to make use of the estimated result to train a classifier on the noisy distribution.

Before we address the second issue, we need to know how to measure the performance of a classifier. Generally speaking, the objective of a classifier is to minimize a given loss function on a given set of training data. However, in this paper, the observed training data contains class noise. Thus, the minimized loss function on noisy training data may not be the minimized loss function on the clean data. So our objective is to find the estimation of the class noise rate for each individual sample such that the loss function on the noisy distribution can be modified.

Furthermore, the modified loss function can also minimize the loss on the clean distribution.

In order to address the problem in a formal way, we give some definitions below.

Definition.2. Let $f: X \to \mathbb{R}$ be a real-value decision function, defined as $f(x) = P(y = 1 \mid x) - 1/2$. The **risk** of f for each sample on clean distribution is 0-1 loss given by $R_D(f) = E_{(x,y)\sim D}(1_{sign(f(x))\neq y})$.

Let $l(f(x), y)$ denote a loss function with a real-value prediction, for the clean distribution where $y = \pm 1$ is the true label of x. Let the hat $\tilde{}$ denoted the modified loss function. We can then use $\tilde{l}(f(x), \tilde{y})$ to denote the modified loss function of the noisy distribution because the loss function is defined under the noisy distribution $\tilde{D}$. We can then use the estimation of the class noise rate for each individual sample and modify the loss function on the noisy distribution with an observed label. Three related risks are then defined as follows:

Definition.3.1: the **Expectation of *l*-risk** under the clean distribution D is defined as: $R_{l,D}(f) = E_{(x,y)\sim D}(l(f(x), y))$.

Definition.3.2: the **Expectation of $\tilde{l}$ -risk** under the noisy distribution $\tilde{D}$ is defined as: $R_{\tilde{l},\tilde{D}}(f) = E_{(x,\tilde{y})\sim \tilde{D}}(\tilde{l}(f(x), \tilde{y}))$.

Definition.3.3: the **Empirical $\tilde{l}$-risk** is defined on the training data as: $\hat{R}_{\tilde{l}}(f) = \frac{1}{n}\sum_{i=1}^{n} \tilde{l}(f(x_i), \tilde{y}_i)$.

Note that in the above definitions, the tilted hat $\tilde{}$ means that the noisy label will influence the marked object. The capped hat $\hat{}$ means that the marked object is an estimated result.

3. Class Noise Estimation

In this section, we first introduce our class noise model and the estimation method.

3.1 Class noise modeling

For training data without any priori knowledge on the noise rate, previous works on class noise modeling gives uniform noise rate on the entire training dataset. That is, the class noise rate is defined on the entire dataset as $p \in [0, 1/2)$. In this work, we drop this assumption and propose noise estimation performed on individual data samples.

According to **Definition.1**, class noise rate requires the estimation of the clean distribution $P(y_i|x_i)$ for each individual sample, which is generally impossible because the observed data is noisy. So, in this work, we model the class noise $P(\tilde{y}_i \neq y_i|x_i)$ by looking at the k nearest neighbors of x_i using the k-NN graph on the noisy data by a hypothesis testing method.

For any sample $(x_i, \tilde{y}_i)$ from the training set with the true label y_i, we first assume that the probability of $P(x_i, y_i)$ and $P(x_i)$ are continuous without any noisy label and all samples are independent of each other. Thus, $\forall \varepsilon > 0$, $\exists \delta > 0$, such that $d(x_i, x_j) < \varepsilon$, and $|P(x_i, y_i) - P(x_j, y_j)| < \delta$ and $|P(x_i) - P(x_j)| < \delta$.

Then, we can build a k-NN graph for noisy data $(x_i, \tilde{y}_i)$ to link the top k nearest samples $(x_j, \tilde{y}_j)$ in a graph. So, the distances between $(x_i, \tilde{y}_i)$ to its k nearest samples should also be small enough. Thus, we consider them to share the same probability, $P(y_i|x_i) \approx P(y_j|x_j)$ on the clean distribution.

The problem here is that we neither have clean data nor prior knowledge about the clean distribution. This means that we cannot estimate $P(y_i|x_i)$ for each individual sample. On the other hand, it is reasonable to assume that the true label of an individual sample should be similar to its nearest sample. Let $(x_i, \tilde{y}_i)$ denote the candidate sample whose class noise rate needs to be estimated and the k nearest samples as $(x_j, \tilde{y}_j)$, here j = 1,2,..,k.

We can then define a sign function on the candidate sample $(x_i, \tilde{y}_i)$ and its k nearest samples $(x_j, \tilde{y}_j)$:

$$I_{ij} = \begin{cases} 1, & if\ \tilde{y}_i \neq \tilde{y}_j \\ -1, & if\ \tilde{y}_i = \tilde{y}_j \end{cases} \quad (1)$$

I_{ij} can indicate the difference between an individual sample and its nearest label. Even though, $\tilde{y}_j$ can contain noise with some probability, the noise level of $\tilde{y}_i$ should be similar to that of $\tilde{y}_j$. The idea is that if the label of a candidate sample is different to its nearest sample, it should have a higher probability to be a noisy label. Based on this, we introduce a similarity function here, normalized over [0,1], denoted by $w_{ij} = sim(x_i, x_j)$.

Then, for each candidate sample, we define a statistic factor, S_i, referred to as the sum of noisy similarity, which can be used to measure the total noise level.

Definition.4: For any individual sample $(x_i, \tilde{y}_i) \in \tilde{D}$ and the top k nearest sample $(x_j, \tilde{y}_j)$, j = 1,2,..,k. under Formula (1) and a normalized similarity w_{ij}, the **sum of noisy similarity** is:

$$S_i = \sum_{j=1}^{k} I_{ij}\, w_{ij}. \quad (2)$$

The idea of similarity is only a hypothesis. Since the nearest samples may also contain noise, the sum of noisy similarity S_i is not a constant. Rather, it should follow some distribution. Let us refer to this distribution as the *Sum of Random Noise (SRN)*. For a sample $(x_i, \tilde{y}_i)$, let c denotes the number of k nearest neighbors whose labels are different from $\tilde{y}_i$. Then, the distribution of *SRN* with any specific c is known and shown as the 6 distributions on background in Fig.1. Obviously, the larger c is, the higher *SRN* should be for an individual sample. From the definition of S_i, we know that the distribution of S_i is dependent on $P(I_{ij})$ only. Since we cannot use the observed label to estimate $P(I_{ij})$, and we assume that $P(I_{ij} = 1) = 1/2$ here, which is based on the **Principle of Entropy Maximum (PEM).** That is, when there is no a priori knowledge, the best assumption for label difference is that they share the same probability. Under the principle of entropy maximum, the distribution of S_i is shown as the yellow shadow on the foreground in Fig.1, labeled as *SRN*.

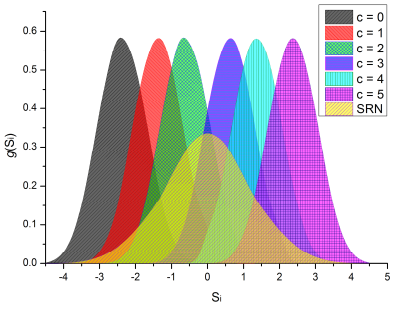


Fig. 1. The Sum of Random Noise (*SRN*) Distribution

Now, we can define the class noise rate for $(x_i, \tilde{y}_i)$ as the probability of S_i being opposite from *SRN*. Because $I_{ij} = 1$

indicates that the sample x_i has different label to its nearest neighbor and the similarity metric is between 0 to 1, the larger S_i is, the higher probably it should be that x_i has a noisy label. So, we should consider only the upper quantile of *SRN* here. Now, we can give the class noise rate under Definition.5.

Definition.5: For any individual sample $(x_i, \tilde{y}_i) \in \widetilde{D}$ and the sum of noisy similarity S_i, the probability of *SRN* noted as P_{SRN}, the **class noise rate** $P(\tilde{y}_i \neq y_i|x_i)$ of $(x_i, \tilde{y}_i)$, is the probability of $1 - P_{SRN}(s \geq S_i)$.

If the principle of entropy maximum reveals a best guessing result, the upper quantile of *SRN* is the probability of the candidate sample being "worse" than a guessing result. So Definition.5 can be presented as a descriptive definition: *the class noise rate of a sample is defined as the probability of the observed label being worse than a guessed label*.

It should be noted that the 6 curves on the background in Fig.1 may easily be mistaken as a normal distribution. In fact, the curve will only approximate the normal distribution when the *k* in the k-NN graph is large enough, usually larger than 25 at least. However, a large *k* will lead to poor accuracy for kNN. [8, 12] actually used normal distribution in their work for noise estimation. Normal distribution is used as a compared estimation method in our experiment in Section 6. We think *SRN* is a more realistic estimation because in practice, *k* is much smaller than 25.

3.2 Class Noise Estimation

In this section, we introduce a method to estimate the class noise rate for individual samples under Definition.5.

We first need to find the estimation of $P(y_i|x_i)$. Since the observed label contains noise, and there is no prior knowledge about the distribution of class noise, the best estimation of $P(y_i = 1|x_i)$, denoted by $\hat{P}(y_i = 1|x_i)$ should minimize the risk of $\hat{P}(y_i = 1|x_i)$:

$$risk(\hat{P}(y_i = 1|x_i)) = \frac{1}{2}(P(y_i = 1|x_i) - \hat{P}(y_i = 1|x_i))^2$$

Since there is no prior distribution about $P(y_i = 1|x_i)$, we assume that $P(y_i = 1|x_i)$ follows a uniform distribution. That is, $P(y_i = 1|x_i)$ is completely random. Therefore,

$$\hat{P}(y_i = 1|x_i) = \underset{p}{\operatorname{argmin}} \int_0^1 \frac{1}{2}(P(y_i = 1|x_i) - p)^2 dP(y_i = 1|x_i)$$

It can easily calculated that $\hat{P}(y_i = 1|x_i) = 1/2$, which means the best assumption for $P(y_i = 1|x_j)$ is $P(y_i = 1|x_i) = P(y_i = -1|x_i)$ when there is no prior knowledge about the clean distribution. This can also be explained by the **Principle of Entropy Maximum**. Under this assumption, the top *k* nearest samples of (x_j, y_j) have the same probability $P(y_i|x_i) \approx P(y_j|x_j)$, which implies that $P(I_{ij} = 1) = P(I_{ij} = -1) = 1/2$. Now, we can use this result to estimate the class noise rate under Definition.5.

Theorem.1.(*main result*) The **estimated class noise rate** of $(x_i, \tilde{y}_i) \in \widetilde{D}$ denoted by $P_c(x_i)$, is:

$$P_c(x_i) \geq 1 - 0.5 \times \exp\left(-\frac{\left(\sum_{j=1}^{k} w_{ij} \cdot I_{ij}\right)^4}{2(\|w_i\|_1 \|w_i\|_2)^2}\right) \quad (3)$$

The proof of Theorem.1 is based on Lemma.1 to be introduced below.

Lemma.1. Let $I_{i1}, I_{i2}, \ldots, I_{ik}$ be independent Bernoulli random variables $(P(I_{ij} = 1) = P(I_{ij} = -1) = 1/2.)$, $w_{i1}, w_{i2}, \ldots, w_{ik}$ be a sequence of real number such that $w_i = (w_{i1}, w_{i2}, \ldots, w_{ik}) \in l_2$ and $t > 0$. We can then conclude that

$$P\left(\sum_{j=1}^{k} I_{ij} w_{ij} > K_{1,2}(w_i, t)\right) \leq e^{-\frac{t^2}{2}}$$

where $K_{1,2}(w_i, t)$ is defined as:

$$K_{1,2}(w_i, t) = \inf\{\|w'_i\|_1 + t\|w''_i\|_2\}$$

Here, $\|\cdot\|_1$ and $\|\cdot\|_2$ are the l_1 and l_2 norms; and $w'_i + w''_i = w_i$. This formula of $K_{1,2}(w_i, t)$ is well known as the *K*-method of real interpolation for Banach Space [24]. The proof of Lemma.1. was given by [10] in details which we will not repeat here.

The issue in is the calculation of $K_{1,2}(w_i, t)$ is quite complex and high cost. Here, we give Lemma.2 below which provides a simple sub-optimal solution of $K_{1,2}(w_i, t)$.

Lemma.2. Let $I_{i1}, I_{i2}, \ldots, I_{ik}$ be independent Bernoulli random variables $(P(I_{ij} = 1) = P(I_{ij} = -1) = 1/2.)$, $w_{i1}, w_{i2}, \ldots, w_{ik}$ be a sequence of real number such that $w_i = (w_{i1}, w_{i2}, \ldots, w_{ik}) \in l_2$ and $t > 0$, we have:

$$P\left(\sum_{j=1}^{k} I_{ij} w_{ij} > \sqrt{t\|w_i\|_1 \|w_i\|_2}\right) \leq e^{-\frac{t^2}{2}}$$

where, $\|\cdot\|_1$ and $\|\cdot\|_2$ are the l_1 and l_2 norms.

Proof: First, let $w'_i = \alpha w_i$ and $w''_i = \beta w_i$, where α and β are two real numbers.

$$\inf\{\|w'_i\|_1 + t\|w''_i\|_2\} = \sqrt{t\|w_i\|_1 \|w_i\|_2}$$

Then, there is a subspace of the interpolation in the Banach space and the optimal result on the subspace is greater than the norm. Thus:

$$\sqrt{t\|w_i\|_1 \|w_i\|_2} \geq K_{1,2}(w_i, t) \quad (4)$$

Assuming the **Probability Density Function** (PDF) of the *SRN* $P(S_i)$ is $g(x)$. Such that

$$P\left(S_i \geq K_{1,2}(w_i, t)\right) = \int_{K_{1,2}(w_i,t)}^{+\infty} g(x)dx$$

based on Formula(4),

$$\int_{K_{1,2}(w_i,t)}^{+\infty} g(x)dx \geq \int_{\sqrt{t\|w_i\|_1\|w_i\|_2}}^{+\infty} g(x)dx$$

So,

$$e^{-\frac{t^2}{2}} \geq P\left(S_i \geq K_{1,2}(w_i, t)\right) \geq P(S_i > \sqrt{t\|w_i\|_1 \|w_i\|_2})$$

This will lead to the result of Lemma 2**.**

According to Definition.4, the estimated class noise rate $P_c(x_i)$ for $(x_i, \tilde{y}_i) \in \widetilde{D}$ is the probability of $(x_i, \tilde{y}_i)$ not following *SRN*. According to Lemma.3, the probability of $(x_i, \tilde{y}_i)$ from *SRN* is less than $\exp\left(-\frac{\left(\sum_{j=1}^{k} w_{ij} \cdot I_{ij}\right)^4}{2(\|w_i\|_1 \|w_i\|_2)^2}\right)$ when $t > 0$.

Due to I_{ij} is a Rademacher random variable in our assumption, the probability of $t > 0$ and $t < 0$ are equal. Thus the estimated class noise rate is:

$$P_c(x_i) \geq 1 - 0.5 \times \exp(-\frac{\left(\sum_{j=1}^{k} w_{ij} \cdot I_{ij}\right)^4}{2(\|w_i\|_1 \|w_i\|_2)^2}) \quad \square$$

For noise detection, the lower boundary is more significant since it pertains to the minimum noise rate of a labeled training sample. In practice, we can use

$$P_c(x_i) = 1 - 0.5 \times e^{(-r(x_i)/2)} \tag{5}$$

as the estimation for each training sample. Here,

$$r(x_i) = \frac{\left(\sum_{j=1}^{k} w_{ij} \cdot I_{ij}\right)^4}{(\|w_i\|_1 \|w_i\|_2)^2}$$

$P_c(x_i)$ can either be incorporated in the learning algorithms to weigh the importance of a training sample or be used to identify samples for elimination. It should be noted that $r(x_i)$ is symmetrical on S_i. It means that when S_i is less than 0, the upper quantile of *SRN* needs to be solved by the lower quantile of *SRN* which is introduced by the opposite sign function. In another word, when S_i is less than 0, $P_c(x_i)$ should be:

$$P_c(x_i) = 0.5 \times e^{(-r(x_i)/2)}$$

This inference can be easily proven by the symmetry of $r(x_i)$.

4. Learning in Noisy Data

In this section, we introduce our learning with noise strategy and class noise elimination method.

4.1 Learning with Noise Strategy

Based on the class noise estimation given in Formula (5), we propose a method using the learning with noise strategy. The fundamental idea is to use the estimated class noise rate to weigh a risk-of-loss function. The key is to ensure that the weighted loss function can adequately approximate the loss function for the clean training data. The loss function on the observed distribution is defined as

$$\tilde{l}\left(f(x_i), \tilde{y}_i\right) = \frac{\left(1 - P_c(x_i)\right) l\left(f(x_i), \tilde{y}_i\right) - P_c(x_i) l\left(f(x_i), -\tilde{y}_i\right)}{1 - 2\left(\frac{\sum_i P_c(x_i)}{n}\right)} \tag{6}$$

where the numerator is formed by two parts. The first part is the original loss function with the observed labels weighted by their label correctness probabilities. The second is a penalty for the loss function with an inverted label (i.e., the probability of the observed label is incorrect) weighted by the class noise rate. The denominator is based on the average class noise rate to ensure that the noisy training data's loss function approximates the clean training data's loss function. This loss function is an updated version of the original loss function on noisy data. We are interested to know whether the minimum risk of the proposed loss function on the noisy training data can approximate the minimum risk of the loss function on the clean data. Let us assume that the training data is clean. Then, the risk of the loss function on the clean distribution and noisy distribution are defined according to Definition 3.1 to Definition 3.3.

As mentioned earlier in Section 3, we need to address two issues here. Will the empirical risk $\hat{R}_{\tilde{l}}(f)$ converge to the expected risk $R_{\tilde{l},\tilde{D}}(f)$ under the noisy distribution when n grows? Will the expected risk $R_{\tilde{l},\tilde{D}}(f)$ converge to $R_{l,D}(f)$ under the clean distribution? If the answer to both questions is yes, the empirical risk then converges to the clean distribution. Thus, we can train a classifier for the cleaning distribution by the noisy data without any prior knowledge.

Theorem.2. The empirical $\tilde{l}$-risk on the training data $\hat{R}(f)$ will converge to the risk $R_{\tilde{l},\tilde{D}}(f)$ of the loss function on the noise data with n independent and identically distributed training samples denote as $(x_1, \tilde{y}_1)$, $(x_2, \tilde{y}_2)$,…, $(x_n, \tilde{y}_n) \in \tilde{D}$, $(\tilde{y}_i = \pm 1, i = 1,2, \ldots, n)$ when n grows.

Proof: Given n independent $(x_i, \tilde{y}_i)$, $P_c(x_i)$ denotes the class noise rate estiminate by Formula (5) with the expectation $E(P_c(x_i))$. According to the Chebyshev law of large numbers, $\forall \varepsilon > 0$, $\exists \delta > 0$, and $N > 0$, such that when $n > N$,

$$\left| \sum_i \frac{P_c(x_i)}{n} - E\left(P_c(x_i)\right) \right| \leq \varepsilon$$

is true with a probability of at least $1 - \delta$.

For the same reason, $\forall \varepsilon > 0$, $\exists \delta > 0$, and $N > 0$, when $n > N$,

$$\left| \frac{1}{n} \sum_{i=1}^{n} l(f(x_i), \tilde{y}_i) - E(l(f(x), \tilde{y})) \right| \leq \varepsilon$$

is true with a probability of at least $1 - \delta$.

Then, the risk on $\tilde{D}$ and the empirical $\tilde{l}$-risk on the training data satisfies

$$|\hat{R}(f) - R_{\tilde{l},\tilde{D}}(f)| \leq 2\varepsilon$$

With probability of at least $1 - \delta$. $\square$

In other words, the empirical $\tilde{l}$-risk on the training data $\hat{R}(f)$ will converge to the risk $R_{\tilde{l},\tilde{D}}(f)$ of the loss function on the noise data when the size of training data is large enough.

If we have a perfectly correct estimation of the class noise rate, it is true that

$$E_{\tilde{y}}\left[\tilde{l}(f(x), \tilde{y})\right] = l(f(x), y).$$

Under this assumption, $R_{\tilde{l},\tilde{D}}(f)$ will converge to $R_{l,D}(f)$ under clean distribution. In order to distinguish the perfect estimation result and the estimation by our method, let f_p denote the perfect estimation.

Theorem.3. The minimum result of risk $R_{l,D}(f_p)$ will converge to the minimum risk $R_{l,D}(f)$ on the clean distribution when there is a perfect estimation of class noise rate.

The proof of Theorem.3 is given in detail by [6], which will not repeated here. Based on the proof, the following inequality holds

$$|R_{l,D}\left(\hat{f}_p\right) - R_{l,D}(f_*)| < 2\Re\left(\tilde{l} \circ \mathcal{F}\right) + 2\sqrt{\frac{\log(1/\delta)}{2n}} \tag{7}$$

with a probability of at least $1 - \delta$.

Here, f_* is the minimizer of $R_{l,D}(f)$ on the clean distirbution, $\hat{f}_p$ is the minimizer of $R_{l,D}(f_p)$ on the clean distribution and

$$\Re\left(\tilde{l} \circ \mathcal{F}\right) = \mathbb{E}_{x_i, \tilde{y}_i, \epsilon_i}[\sup_{f \in \mathcal{F}} = \frac{1}{n} \sum_{i=1}^{n} \epsilon_i \tilde{l}(f(x_i), \tilde{y}_i)]$$

where ϵ_i is an independent and identically distributed Rademacher random variable.$\square$

The right side of the inequality in Formula (7) is bound by the richness of the function family $\mathcal{F}$ and the sample size of n. In other words, the class noise estimation method indeed ensures the

optimality of $\hat{f}_p$ with the class noise distribution approximating the risk of the clean distribution.

Thus the risk of our estimation method $R_{l,D}(\hat{f})$ satisfy:

$$|R_{l,D}(\hat{f}) - R_{l,D}(f_*)| < |R_{l,D}(\hat{f}) - R_{l,D}(\hat{f}_p)| + |R_{l,D}(\hat{f}_p) - R_{l,D}(f_*)|$$

Because the second item is bound, the risk of our estimate is determined by the first item which is related to the estimation result. This can be interpreted as that when the size of the training set grows, the performance of the classifier trained on the noisy data is determined by the estimation result given in Formula (5).

4.2 Class noise Elimination Strategy

Existing class noise elimination strategies attempt to detect and eliminate noisy samples. In this work, we propose a sampling based method which focuses on selecting only high quality samples to form the training dataset rather than focusing on noisy samples. The quality of a training sample is defined based on its class-noise rate: the lower the class-noise rate, the higher the quality. We refer to this as a sampling-based method. The idea is to use an ensemble learning algorithm as a group of aggregate classifiers, each working on a subset of the training data. Even if a sample is considered noise in one classifier, it may still be used as correct data in other sampling rounds. Generally speaking, an ensemble learning algorithm can use either bagging or boosting (e.g., Adaboost). In this work, we use only the bagging strategy for evaluation.

Each training sample $(x_i, \tilde{y}_i)$ is assigned a weight $p_i = 1 - P_c(x_i)$. We use this weight to select samples in the training dataset. We deploy a bagging algorithm based on the sampling results to obtain a stable classifier on the training data. Any basic classifier can be used in this framework. To ensure the effectiveness of this strategy, we need to confirm that the learning algorithm approximates the learner on a clean distribution.

Let S_i denote the i-th sampling result with $h_i \in \mathcal{H}$ signifying the optimal hypothesis on S_i. Assume an optimal hypothesis on the clean distribution S' exists and is denoted by $h^* \in \mathcal{H}$. According to the Hoeffding boundary from Angluin and Laird [11], the difference between h_i and h^* is bound by:

$$d(h_i, h^*) \leq \sqrt{2\ln\left(\frac{2N}{\delta}\right)/[|S_i|(1-2\eta)]^2} \qquad (8)$$

with a probability of at least $1-\delta$. Here, N is a constant related to the feature space and η is the class-noise rate of S_i. Since the class-noise rate is bound by the result of each sampling, the distance from the optimal hypothesis on S_i to the optimal hypothesis on the clean distribution approaches zero as the size of S_i increases. The bagging strategy allows error correction in the sampling to achieve a superior and more stable result.

4.3 Computational complexity

From the introduction above, we can see that any learning algorithm should be composed by two parts. The first part is for noise estimation, the second part is for training. The training parts are dependent on the learning algorithms. Thus, the complexity discussed here is on the noise estimation part only.

Since the estimation is based on a k-NN graph, usually the computational complexity of kernel based method such as kNN is $O(n^2 d)$. Here, n is the size of the training set and d is the dimension of the feature space. There are some methods to speed up kNN as done in [28, 29] by reducing d for sparse data. However, the computational complexity is still $O(n^2)$. The next step is the estimation. According to Formula (5), the complexity is $O(kd)$. Thus, the computational complex of our estimation method is $O(n^2 d + kd)$. As k is fixed, and not related to data size, complexity is basically $O(n^2 d)$.

5. Performance Evaluation

5.1 Experimental Setup

Our experiments evaluate the estimation performance and show its utility in improving learning performance for both online and offline learning algorithms. In principle, we can use any learning algorithm for this evaluation. For demonstration purpose, we use a simple linear algorithm using perceptron-based online learning to avoid complex parameter tuning. This is referred to as *LiC* (Linear Classification). For offline learning, we use bagged non-linear SVMs with polynomial kernels, referred to as *SaC* (Sampling Classification). The pseudo code for the *LiC* algorithm is given below

Algorithm I: *LiC*

Input:
S: The training dataset with *n* samples;
$(x_i, \tilde{y}_i)$: The training sample, $(x_i, \tilde{y}_i) \in S$
λ: Learning rate for perceptron learning
a_0: Initialized weight of perceptron features
Training:
For each $(x_i, \tilde{y}_i)$ in *S*, build a k-NN graph and calculate $P_c(x_i)$ according to Formula (5)
For *i*=1 to *n*:
Compute the inner product: $\langle a_{i-1}, x_i \rangle$
Update: $a_i \leftarrow a_{i-1} - \lambda \frac{\partial \tilde{l}(\langle a_{i-1}, x_i \rangle, \tilde{y}_i)}{\partial a_{i-1}} x_i$
Output: a_n

For the *SaC* algorithm, the class noise rate estimation given in Formula (5) serves as the weight for each training sample during sampling. The pseudo code of the *SaC* algorithm is given below.

Algorithm II: *SaC*

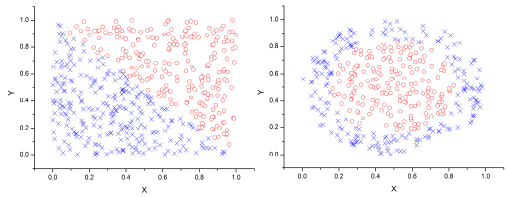
Input:
S: The training dataset with *n* samples;
$(x_i, \tilde{y}_i)$: The training sample, $(x_i, \tilde{y}_i) \in S$
T: Number of iterations
Training:
For each $(x_i, \tilde{y}_i)$ in *S*, built a k-NN graph and calculate $P_c(x_i)$ according to formula (5)
For *j*=1 to *T*:
Sampling based on $1 - P_c(x_i)$ for each $(x_i, \tilde{y}_i)$, the result is S_j
Use the learner on S_j to train the classifier C_j
Output:
An ensemble classifier: $C = \sum_j C_j$ /T

Based on the two algorithms, we conduct two sets of experiments: one for *LiC* and *SaC* on a 2-D synthetic dataset and the other on public datasets with comparisons to other well-known methods.

5.2 Evaluation on Synthetic Data

We prepared one linearly separable dataset and one non-linearly separable set, each has 500 samples, to serve as the clean data as shown in Fig. 2. The corresponding testing datasets have similar distributions. The feature values are real numbers between 0 and 1 and the labels are binary (red circles and blue crosses represent positive and negative labels, respectively). The ratio of positive and negative samples is 1:1. Stochastic class noise is introduced

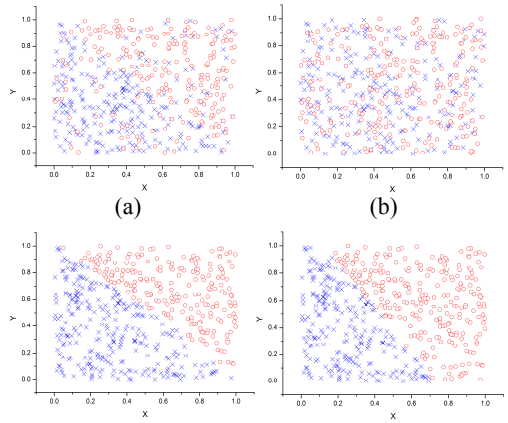
into the clean training data to invert some labels. Experiments on linearly separable data with 20% and 40% noise levels are shown in Fig.3(a) and Fig.3(b) respectively.



(a) (b)

Fig. 2. Linearly(a) and Non-Linearly(b) Separable Data

Fig. 3(c) and 3(d) show the respective result of *LiC* on the testing data and Table 1 shows the accuracy of both *LiC* and the Perceptron algorithm without noise estimation. Comparing Fig. 3(c) and 3(d), we can see that the decrease in accuracy is attributed to the shift of the linear separation boundary as the noise level increases (particularly evident at the bottom right corner). Table.1 indicates that noise estimation can make an obvious difference in performance. This is more apparent at higher noise level of 40%. *LiC* can achieve the improvement of 3.62% compared to the Perceptron algorithm.



(a) (b)

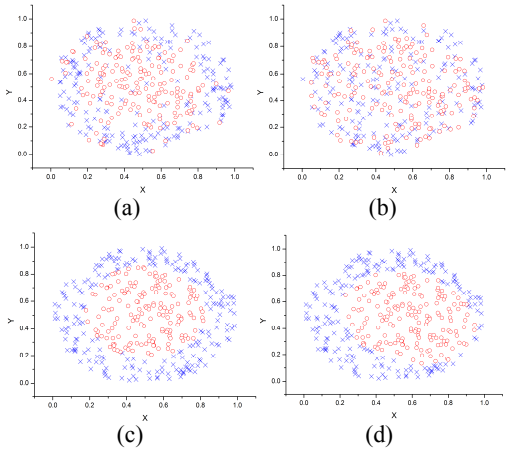
(c) (d)

Fig. 3. Experiment on Linearly Separable Data

Table. 1. Experiment on Linearly Separable Data

Data	Noise rate	Method	Accuracy
(a)	20%	*LiC*	**97.19%**
(a)	20%	Perceptron	96.19%
(b)	40%	*LiC*	**91.78%**
(b)	40%	perceptron	88.57%

Fig. 4(a) and Fig. 4(b) show the non-linearly separable data with 20% and 40% noise levels, respectively, and Fig. 4(c) and Fig. 4(d) show the result of *SaC*. Table 2 shows the accuracy of *SaC* compared to RBF kernel based SVMs. Note that the improvement of 2.54%, when the class noise rate is 20%, is larger than *LiC* because non-linearly separable problem is more sensitive to class noise. However, when the class noise rate is 40%, the improvements of two algorithms are similar.



(a) (b)

(c) (d)

Fig. 4. Experiment on Non-Linearly Separable Data

Table. 2. Experiment on Non-Linearly Separable Data

Data	Noise rate	Method	Accuracy
(a)	20%	*SaC*	94.02%
(a)	20%	SVMs(RBF)	91.69%
(b)	40%	*SaC*	87.00%
(b)	40%	SVMs(RBF)	84.16%

5.3 Evaluation on Public Datasets

In this set of experiments, we evaluate the performance of our proposed methods using seven public datasets for binary classification with different class-noise rates: (1) the LEU [23] set of cancer data; (2) the Splice dataset for DNA sequence splice-junction classification; (3) the UCI Adult dataset collection containing seven subsets (referred to as a1a to a7a according to original naming of the data) of independent training and testing data [26]; (4) the DBWorld e-mails DataSet in English (in short, DB), which consists of 64 e-mails manually collected from DBWorld mailing list as binary labels of either "announces of conferences" or "everything else"; (5) the Farm Ads DataSet in English (in short, FADS), which is collected from text ads found on twelve websites that deal with various farm animal related topics with binary labels based on whether the content owner approves of the ad or not; (6) the Twitter Dataset for Arabic Sentiment Analysis Dataset (in short, TDA) with the class labels being opinion polarity; (7) the Product Reviews from Amazon in three categories, Book, DVD and Music (in short, PRA). The class labels are opinion polarity. The datasets (1) and (2) can be downloaded from the website of LibSVM[1]. The datasets (3) to (6) are from UCI[2] and (7) is from the NLPCC 2013 cross lingual opinion analysis evaluation task[3]. Datasets (4)-(7) are text data. The size, class ratio, types and dimensions of the datasets are listed in Table 3.

Table. 3. Overview of Datasets

Data	Type	Dimension	Training size(+/-)	Testing size(+/-)
LEU	Cancer	7129	38(0.71/0.29)	34(0.59/0.41)
Splice	DNA	60	1000(0.52/0.48)	2175(0.52/0.48)
UCI.a1a	Adult	123	1605(0.37/0.63)	30956(0.24/0.76)
UCI.a2a	Adult	123	2265(0.25/0.75)	30296(0.24/0.76)

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets

[2] http://www.ics.uci.edu

[3] http://tcci.ccf.org.cn/conference/2013/dldoc/evdata03.zip

UCI.a3a	Adult	123	3185(0.24/0.76)	29376(0.24/0.76)
UCI.a4a	Adult	123	4781(0.25/0.75)	27780(0.24/0.76)
UCI.a5a	Adult	123	6414(0.24/0.76)	26147(0.24/0.76)
UCI.a6a	Adult	123	11220(0.24/0.76)	21341(0.24/0.76)
UCI.a7a	Adult	123	16100(0.24/0.76)	16461(0.24/0.76)
DB	English	4698	64(0.45/0.55)	-------
FADS	English	54877	4143(0.51/0.49)	-------
TDA	Arabic	7415	2000(0.50/0.50)	-------
PRA_{Book}	Chinese	74643	4000(0.50/0.50)	-------
PRA_{DVD}	Chinese	74638	4000(0.50/0.50)	-------
PRA_{Music}	Chinese	74638	4000(0.50/0.50)	-------

To introduce the class noise into the training set, we stochastically invert the binary labels of training samples with a probability of 10%, 20%, and 30%, respectively. For the datasets (1) to (3), we train the binary classification algorithm on the noisy data. The algorithm is tuned on a development set before being used on the testing set. We use svm^{light} [4] as the basic classifier for this experiment. We choose two kind of similarity measures, the cosine similarity for text data, and the Euclidean Distance based similarity for all other data. The k in k-NN graph is 7 for *LiC*, *SaC* and CEWS. For the datasets (4) to (7), due to the lack of testing data, the result is based on 5-folds cross-validation.

We compare the performance of *LiC* with two state-of-the-art learning algorithms for noisy data: (1) the widely used open source robust method NHERD [27] and (2) the class noise log loss method, ℓ_{log} [6] which uses the same loss function as *LiC*. We also compare *SaC* with two state-of-the-art noise elimination systems: (3) Identifying Mislabeled Training Data (IMTD) [7] and (4) Cut Edge Weight Statistics (CEWS) [8]. Both ℓ_{log} and CEWS require prior knowledge of the class-noise for the training data. Thus, they are more appropriate for use as benchmarks rather than direct comparison to other algorithms, and should be discussed separately. Table 4 shows the accuracy of the six systems.

Table. 4. Performance of the Six Systems

Data	Noise rate	Learning with noise			Noise elimination		
		LiC	NHERD	ℓ_{log}*	***SaC***	IMTD	CEWS*
LEU	10%	87.88	81.62	**90.91**	**73.53**	58.82	**73.53**
	20%	**90.91**	76.62	84.85	**70.59**	58.82	67.65
	30%	**78.79**	58.68	57.58	58.82	58.82	58.82
Splice	10%	**84.67**	72.14	83.99	**85.17**	83.42	83.15
	20%	**83.05**	66.63	83.02	**83.34**	82.91	82.39
	30%	**79.53**	61.14	78.31	79.54	**80.83**	73.54
UCI.a1a	10%	**83.40**	81.23	83.33	83.25	**83.40**	82.86
	20%	81.40	79.30	**81.85**	82.78	79.79	**82.95**
	30%	**77.30**	74.78	77.10	**81.42**	78.39	81.31
UCI.a2a	10%	**84.24**	82.25	83.92	**84.14**	83.79	83.71
	20%	**83.16**	79.34	82.72	**83.12**	82.32	81.54
	30%	**80.66**	74.13	76.22	**81.81**	76.58	81.05
UCI.a3a	10%	**84.17**	82.79	83.93	83.54	**83.96**	83.07
	20%	**83.49**	81.14	82.23	**83.36**	78.02	82.83
	30%	**81.02**	77.52	80.23	**81.96**	79.28	81.35
UCI.a4a	10%	**84.28**	83.48	84.05	83.53	**83.95**	83.37
	20%	**84.10**	82.34	82.90	83.42	82.70	**83.51**
	30%	**81.81**	80.05	81.54	**82.46**	78.09	82.31
UCI.a5a	10%	**84.45**	83.57	83.78	**84.05**	84.03	83.87
	20%	**84.20**	82.88	83.06	83.47	**83.55**	83.23
	30%	79.20	**80.42**	74.71	**82.32**	80.30	82.22
UCI.a6a	10%	**84.77**	83.97	83.20	84.31	83.83	**84.44**
	20%	**83.38**	82.45	80.37	**83.49**	78.00	81.48
	30%	**81.83**	80.22	77.83	**82.21**	78.00	78.00
UCI.a7a	10%	**85.65**	84.58	82.93	**84.66**	84.45	83.14
	20%	**84.27**	82.87	80.43	**83.83**	78.17	81.65
	30%	**83.49**	80.28	78.67	**82.71**	79.33	80.86
DB	10%	**91.66**	**91.66**	87.98	**91.66**	66.67	73.53
	20%	**91.66**	75.00	84.50	58.33	58.33	**67.65**
	30%	**81.82**	58.33	80.52	58.33	58.33	58.33
FADS	10%	82.81	86.44	**90.91**	**90.28**	75.15	60.74
	20%	**84.38**	81.27	81.82	**86.07**	73.47	53.78
	30%	**85.58**	78.42	63.64	**82.59**	71.07	49.58
TDA	10%	**84.06**	82.92	82.35	**84.84**	71.88	47.68
	20%	**79.66**	76.81	77.70	**80.19**	65.28	45.97
	30%	**77.94**	68.70	76.23	**77.75**	58.92	47.19
PRA_{Book}	10%	77.72	75.65	**78.02**	**78.42**	63.38	64.36
	20%	75.55	74.31	**76.29**	**75.22**	61.65	56.97
	30%	**77.03**	69.52	76.97	**73.86**	59.43	70.04
PRA_{DVD}	10%	**80.42**	79.21	79.55	**81.44**	69.24	80.20
	20%	**79.43**	74.33	79.30	**80.32**	73.10	69.86
	30%	**78.96**	70.16	77.55	**76.71**	62.39	71.98
PRA_{Music}	10%	78.83	71.39	**79.35**	**77.84**	72.16	52.06
	20%	**79.84**	77.27	76.25	**79.25**	68.30	52.45
	30%	**75.23**	69.78	72.13	**74.10**	71.26	70.75
Macro Average	10%	83.62	81.53	**83.88**	**83.38**	76.54	74.65
	20%	**83.23**	78.17	81.15	**79.79**	73.63	71.59
	30%	**80.01**	72.14	75.28	**77.11**	71.40	71.16
Macro Average		**82.39**	77.28	80.10	**80.09**	73.85	72.47
Micro Average	10%	**83.63**	82.23	83.02	**83.72**	80.16	77.66
	20%	**82.70**	80.51	80.53	**82.59**	76.59	75.05
	30%	**80.99**	77.04	76.67	**80.77**	75.04	75.62
Micro Average		**82.44**	79.92	80.07	**82.36**	77.26	76.11

Comparison to Other Methods

Table 4 shows that our proposed algorithms *LiC* and *SaC* outperform the other the algorithms in most of the cases. In terms of both the macro average and the micro average which is weighted over the size of the different class labels, our algorithms clearly outperform all other methods. Even though the class-noise rate of the training data is provided to ℓ_{log} and CEWS, they only outperform *LiC* and *SaC* for the 10% and 20% noise levels of two and three datasets, respectively. This is because ℓ_{log} and CEWS require that all samples have the same class-noise rate for the probability weighting.

Fort the small training set, such as LEU, which has only 37 training samples, and DB, which has only 64 samples for training, the size effect is quite prominent. Intuitively, the noisy data elimination based method should not work well in these data. Results show that when the class noise rate increase to 20% and 30%, the high percentage of noise obviously have a big effect on the training data. Most methods have a big performance decrease for these two datasets, but *LiC* performs well and shows a

[4] http://svmlight.joachims.org

significant advantage over other methods. This is because in the small training set, the size of training data is more important than the quality. *LiC* keeps the size of the training data. Due to the class noise estimation, *LiC* can also show a higher performance than ℓ_{log} even though ℓ_{log} also has the same size of training data.

When the training set grows, such as Splice which has 1,000 samples or TDA which has 2,000 samples, the quality of training samples becomes much more important. For these datasets, both *LiC* and *SaC* show better performance than the other methods. When the training set size becomes larger, the advantages of our methods become even more obviously for UCI Adult with 16,100 samples and the review text from Amazon with 4,000 samples. When the noise level is at the 30% level, *LiC* and *SaC* can achieve a 5% higher micro average accuracy than other methods.

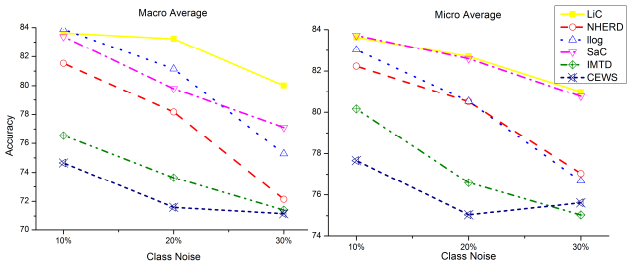


Fig. 5. Comparison between six methods

Generally speaking, both *LiC* and *SaC* show a more stable and more accurate performance in most cases as shown in Fig.5. For the cases which *LiC* and *SaC* cannot achieve the best performance, they are at least as competitive as the other methods. Thus, our proposed method is the overall best performer.

Comparison between *LiC* and *SaC*

We now compare the performance of the two proposed algorithms *LiC* and *SaC*. *LiC* is a learning with noise algorithm. Therefore, it should be more sensitive to the percentage of class noise in the training sample. On the other hand, *SaC* is a class noise elimination based method. It should be more sensitive to the size of training data. Take the small training dataset LEU as an example, it has only 37 samples. Thus any elimination on the training data will lead to degradation of performance. In this dataset, *LiC* is definitely a better choice than *SaC*. In general, we are more interested in the suitability of the two proposed strategies with respect to training data size. To eliminate the extremely small training datasets, we use "small" to means the size of training data at around 1,000 or so. For training set size at around 100 or so, we called them as "tiny training set".

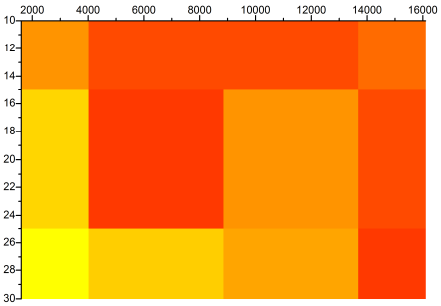


Fig. 6. Comparison between *LiC*(*Red*) and *SaC*(*Yellow*)

To examine the relationship between training set size and algorithm performance, we further analyze performance in more details for the UCI Adult dataset because the subsets are ordered by increasing size and all are from the same distribution and feature space. Fig. 6 shows the comparison of *LiC*(*Red*) to *SaC*(*Yellow*) for different dataset sizes and noise rates using color spectrum. The different shades indicate the relative suitability of the two algorithms. Here, the abscissa is the training set size and the ordinate is the class noise rate. We observe the following facts:

1. *LiC* is inefficient for small training sets(Indicated by pure yellow) and *SaC* is suitable for small data size.
2. Irrespective of the training size, *LiC* outperforms *SaC* for low class-noise (the color is more red than yellow). But when noise increase, *SaC* performs better (more yellow than red).
3. When the training size is relatively large, both *LiC* and *SaC* show similar performance and both are stable.

For text classification, because n-gram models are used to extract features, the feature space can be quite sparse and the elimination of noisy data may reduce the performance of classifier sharply. So, for tiny training sets such as DB, the size effect is very prominent. Note that *LiC* achieves an acceptable performance for DB even when the class-noise rate is 30%, whereas, all noise elimination based algorithms including *SaC* show very poor performance at higher noise levels. It is similar to the comparison on LEU dataset. Fig. 7 shows the performance for text classification which tend to have sparseness problem in the feature space. Here, the abscissa is the training set size and the ordinate is the class-accuracy.

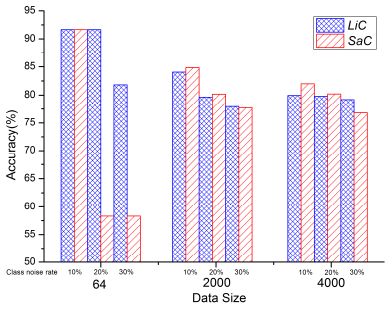


Fig.7 Comparison between *LiC* and *SaC* on text data

We observe the following from Fig. 7:

1. For the tiny training set, *SaC* is not acceptable and *LiC* achieves reasonably good performance.
2. For the small training set, *SaC* outperforms *LiC* when the class noise rate is low. But when noise rate increases, the gap between the two methods narrows.
3. Irrespective of the training size, *SaC* outperforms *LiC* for low class-noise rates. But when noise rate increases, *LiC* performs better.
4. When the training size is relatively large, *LiC* and *SaC* show similarly performance.

The performance of *SaC* is better than *LiC* when the class noise is low. The reason behind is that SVMs shows a higher performance than perceptron based algorithm when the feature space is large and sparse. So *SaC* is more suitable algorithm text classification.

In conclusion, *LiC* is more robust on data size and *SaC* is more robust on class noise rate. When the training set is relatively large, both algorithms perform similarly well and converge to a

consistent result. This is proven through both the theoretical boundary analysis and the experimental results.

6. Conclusion and Future Work

In this paper, we present a novel class noise estimation method. We apply our method to both the learning with noise strategy and the class noise elimination strategy with calculated theoretic bounds for proof of accuracy. Both algorithms are competitive with state-of-the-art techniques and show superior performance on real datasets. We analyze the algorithm performance on different training dataset sizes and class-noise rates. Results conform to the learning theorem provided in Equation (7). In future work, we will investigate noise handling in semi-supervised tasks such as semi-supervised classification, transductive transfer learning, and also look into the domain adaptation problem. We will also consider different noise rate for different classes since label noise rates are often class-conditional in practice.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 61370165, 61203378), National 863 Program of China 2015AA015405, Natural Science Foundation of Guangdong Province S2013010014475, Shenzhen Development and Reform Commission Grant No.[2014]1507, Shenzhen Peacock Plan Research Grant KQCX20140521144507925, Baidu Collaborate Research Funding, and Hong Kong Polytechnic University Project Z0EP.

7. REFERENCES

[1] Zhu, X., and Wu, X "Class noise vs. attribute noise: A quantitative study." In Artificial Intelligence Review. 22(3): 177-210, 2004.

[2] Sáez, J. A., Galar, M, Luengo, J, and Herrera, F. "Analyzing the presence of noise in multi-class problems: alleviating its influence with the One-vs-One decomposition." In Knowledge and Information Systems. 38(1): 179-206, 2014.

[3] Joseph, L., Gyorkos, T. W., and Coupal, L.. "Bayesian estimation of disease prevalence and the parameters of diagnostic tests in the absence of a gold standard." In American Journal of Epidemiology. 3: 263-272, 1995.

[4] Cawthorn, D. M., Steinman, H. A., and Hoffman, L. C.. "A High Incidence of Species Substitution and Mislabelling Detected in Meat Products Sold in South Africa." In Food Control. 32(2): 440-449, 2013.

[5] Beigman, E. and Klebanov, B. B.. "Learning with Annotation Noise". In Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, 280–287, 2009.

[6] Natarajan, N., Dhillon, I. S., and Ravikumar, P.. "Learning with Noisy Labels". In Proceeding of Advances in Neural Information Processing Systems. 2013.

[7] Brodley, C. E., and Friedl, M. A.. "Identifying mislabeled training data." In Journal of Artificial Intelligence Research. 11: 131-167, 1999.

[8] Zighed, D.A., Lallich, S., Muhlenbach, F.. "A Statistical Approach to Class Separability". In Applied Stochastic Models in Business and Industry, Wiley-Blackwell, 21 (2): 187-197, 2005.

[9] Sluban, B., Gamberger, D., and Lavrac, N.. "Advances in Class Noise Detection." In Proceeding of European Conference on Artificial Intelligence, 1105-1106. 2010.

[10] Montgomery-Smith, S. J. "The distribution of Rademacher Sums." In Proceeding of the American Mathematical Society. 109(2): 517-522, 1990.

[11] Angluin, D., and D.Laird, P. "Learning from Noisy Examples." In Machine Learning 2(4): 343-370, 1988

[12] Zhang, M. L., and Zhou, Z. H.. "CoTrade: Confident Co-Training with Data Editing." In Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions. 41(6): 1612-1626, 2011.

[13] Gui, L., Xu, R. F., Lu, Q., et al. "Cross-lingual Opinion Analysis via Negative Transfer Detection." In Proceedings of the 52th Annual Meeting of the ACL. 860-865, 2014.

[14] Frénay, B., and Verleysen, M.. "Classification in the Presence of Label Noise: a Survey". In IEEE Transactions on Neural Networks and Learning Systems, Vol. 25, 5, 2014.

[15] Heskes, T. "The use of being Stubborn and Introspective," In Studies in Cognitive Systems. 2000.

[16] Li, Y., Wessels, L. F. A., Ridder, D., and Reinders, M. J. T.. "Classification in the presence of class noise using a probabilistic Kernel Fisher method". In Pattern Recognition, Volume 40, Issue 12, December 2007, Pages 3349–3357.

[17] Scott, C., Blanchard. G., and Handy, G.. "Classification with Asymmetric Label Noise: Consistency and Maximal Denoising". In Journal of Machine Learning Research: Workshop and Conference Proceedings vol 30 (2013) 1–23

[18] Lawrence, N. D., and Schölkopf, B.. "Estimating a Kernel Fisher Discriminant in the Presence of Label Noise," In Proceeding of International Conference on Machine Learning. 306–313, 2001.

[19] Perez, C. J., Giron, F. J., Martin, J., Ruiz, M., and Rojano, C.. "Misclassified Multinomial Data: A Bayesian Approach," Revista De La Real Academia De Ciencias Exactas Físicas Y Naturales Serie A Matemáticas, vol. 101, no. 1, 71–80, 2007.

[20] Klebanov, B. B., and Beigman, E.. "From Annotator Agreement to Noise Models," In Computational. Linguistics, vol. 35, no. 4, 495–503, 2009..

[21] Kolcz, A., and Cormack, G. V.. "Genre-based Decomposition of Email Class Noise," In Proceeding of 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 427–436, 2009.

[22] Zhu, X., Wu, X., and Chen, Q. J.. "Eliminating Class Noise in Large Datasets." In Proceeding of International Conference on Machine Learning, vol. 3, 920-927. 2003.

[23] Jiang, Y., and Zhou, Z. H.. "Editing Training Data for k-NN Classifiers with Neural Network Ensemble." In Advances in Neural Networks, 356-361. Springer Berlin Heidelberg, 2004.

[24] Bennett, C., and Sharpley, M.. "Interpolation of Operators". Vol. 129. Academic press, 1988.

[25] Golub, T. R., Donna K. S., Pablo Tamayo, C. H., Michelle G., Jill, P. M., Hilary C.. "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring." Science. 286(5439): 531-537, 1999.

[26] Platt, J. C. "Fast Training of Support Vector Machines using Sequential Minimal Optimization". In Advances in Kernel Methods - Support Vector Learning, Cambridge, MA, 1998.

[27] Crammer, K., and Lee, D.. "Learning via Gaussian Herding." In Proceeding of Advances in Neural Information Processing Systems, 451-459. 2010.

[28] Cui, B., Ooi, B. C., Su, J., and Tan, K. L. "Contorting high dimensional data for efficient main memory KNN processing". In Proceeding of International Conference on Management of Data - SIGMOD , pp. 479-490, 2003.

[29] Hui, J., Ooi, B.C., Shen, H., Yu, C., Zhou, A.: An adaptive and efficient dimensionality reduction algorithm for high-dimensional indexing. In: Proc. 19th ICDE Conference, p. 87 2003.