Robust and Efficient Boosting Method Using the Conditional Risk

Zhi Xiao, Zhe Luo, Bo Zhong, and Xin Dang, *Member, IEEE*

***Abstract*— Well known for its simplicity and effectiveness in classification, AdaBoost, however, suffers from overfitting when class-conditional distributions have significant overlap. Moreover, it is very sensitive to noise that appears in the labels. This paper tackles the above limitations simultaneously via optimizing a modified loss function (i.e., the conditional risk). The proposed approach has the following two advantages. First, it is able to directly take into account label uncertainty with an associated label confidence. Second, it introduces a trustworthiness measure on training samples via the Bayesian risk rule, and hence the resulting classifier tends to have finite sample performance that is superior to that of the original AdaBoost when there is a large overlap between class conditional distributions. Theoretical properties of the proposed method are investigated. Extensive experimental results using synthetic data and real-world data sets from UCI machine learning repository are provided. The empirical study shows the high competitiveness of the proposed method in predication accuracy and robustness when compared with the original AdaBoost and several existing robust AdaBoost algorithms.**

***Index Terms*— AdaBoosting, classification, conditional risk, exponential loss, label noise, overfitting, robustness.**

I. Introduction

FOR classification, AdaBoost is well known as a simple but effective boosting algorithm with the goal of constructing a strong classifier by gradually combining weak learners [12], [31], [46]. Its improvement on classification accuracy benefits from the ability of adaptively sampling instances for each base classifier in the training process, more specifically in its reweighting mechanism. It emphasizes the instances that were previously misclassified, and it decreases the importance of those that have been adequately trained. This adaptive scheme, however, causes an overfitting problem for noise data or data from overlapping class distributions [9], [25], [43]. The problem stems from the uncertainty of observed labels. It is usually a great challenge to do classification for the cases with overlapping classes. Also, it is both expensive and difficult to obtain reliable labels [11]. In some applications (such as biomedical data), perfect training labels are almost impossible to obtain. Hence, how to make AdaBoost achieve noise robustness and avoid overfits becomes an important task. The aim of this paper is to construct a modified AdaBoost classification algorithm with a new perspective for tackling those problems.

A. Related Work

Modifications to AdaBoost in dealing with noise data can be summarized into three strategic categories. The first one introduces some robust loss functions as new criteria to be minimized, rather than using the original exponential loss. The second type focuses on modifying the reweighting rule in iterations in order to reduce or eliminate the effects of noisy data or outliers in the training sets. The third approach suggests more modest methods to combine weak learners that take advantage of base classifiers in other ways.

LogitBoost [13] is an outstanding example of a modification of the first strategic category. It uses the negative binomial log-likelihood loss function, which puts relatively less influence on instances with large negative margins[1] in comparison with the exponential loss, and thus LogitBoost is less affected by contaminated data [15]. Based on the concept of robust statistics, Kanamori *et al.* [19] studied loss functions for robust boosting and proposed a transformation of loss functions in order to construct boosting algorithms more robust against outliers. Their usefulness has been confirmed empirically. However, the loss function they utilized was derived without considering efficiency. Onoda [26] proposed a set of algorithms that incorporate a normalization term into the original objective function to prevent from overfitting. Sun *et al.* [35] and Sun *et al.* [36] modified AdaBoost using the regularization method. The approaches of the first category modification mainly differ in the loss functions and optimization techniques that are used. Sometimes, in the pursuit of robustness, it is hard to balance the complexity of a loss function with its computation cost.

In general, modification of a loss function leads to a new reweighting rule for AdaBoost, but some heuristic algorithms directly rebuild their weight updating scheme to avoid skewed distributions of examples in the training set. For instance, Domingo and Watanabe [10] proposed MadaBoost that bounds the weight assigned to every sample by its initial probability.

Manuscript received February 5, 2016; revised September 20, 2016, January 18, 2017, and April 16, 2017; accepted May 24, 2017. Date of publication June 28, 2017; date of current version June 21, 2018. *(Corresponding author: Xin Dang.)*

Z. Xiao is with the Department of Information Management, Chongqing University, Chongqing, China (e-mail: xiaozhi@cqu.edu.cn).

Z. Luo is with the Bank of China, Nanning, China (e-mail: shuilifang_1988@126.com).

B. Zhong is with the Department of Statistics and Actuarial Science, Chongqing University, Chongqing, China (e-mail: zhongbo@cqu.edu.cn).

X. Dang is with the Department of Mathematics, University of Mississippi, Oxford, MS 38677, USA (e-mail: xdang@olemiss.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2711028

[1]Margin is generally defined as $yf(\boldsymbol{x})$; a negative margin implies a misclassification on an instance.

Zhang *et al.* [49] introduced a parameter into the weight updating formula to reduce weight changes in the training process. Servedio [32] provided a new boosting algorithm, SmoothBoost, which produces only smooth distributions of weights but enables generation of a large margin in the final hypothesis. Utkin and Zhuk [40] took the minimax (pessimistic) approach to search the optimal weights at each iteration in order to avoid outliers being heavily sampled in the next iteration.

Since the ensemble classifier in AdaBoost predicts a new instance by a weighted majority voting among weak learners, the classifier that achieves high training accuracy will greatly impact the predictive result because of its large coefficient. This can have a detrimental effect on the generalization error, especially when the training set itself is corrupted [1], [30]. With this in mind, the third strategy seeks to provide a better way to combine weak learners. Schapire and Singer [30] improved boosting in an extended framework where each weak hypothesis produces not only classifications but also confidence scores in order to smooth the predictions. Besides, another method called modest AdaBoost [42] intends to decrease contributions of base learners in a modest way and forces them to work only in their domain.

The algorithms described above mainly focus on some robustifying principle, but they do not consider specific information in the training samples. Many other researches [16], [18], [37] introduced the noise level into the loss function and extended some of the above-mentioned methods. Nevertheless, most of these algorithms do not fundamentally change the fact that misclassified samples are weighted more than they are in the previous stage, though the increment of weights is smaller than that in AdaBoost. Thus, mislabeled data may still hurt the final decision and cause overfitting.

In recent studies, many researchers were inclined to utilize the instance-base method to make AdaBoost robust against label noise or outliers. They evaluated the reliability or usefulness of each sample using statistical methods, and took that information into account. Cao *et al.* [6] suggested a noise-detection-based loss function that teaches AdaBoost to classify each sample into a mostly agreed class rather than using its observed label. Gao and Gao [14] set the weight of suspicious samples in each iteration to zero and eliminated their effects in AdaBoost. Essentially, these two methods use dynamic correcting and deleting techniques in the training process. In [43], the boosting algorithm directly works on a reduced training set whose confusing samples have been removed. Zhang and Zhang [48] considered a local boosting algorithm. Its reweighting rule and the combination of multiple classifiers utilize more local information of the training instances.

For handling label noise, it is natural to delete or correct suspicious instances first and then take the remaining good samples as prototypes for learning tasks. This idea is not just for AdaBoost but is also applicable to general methods in many fields (see [39]). Some approaches aim at constructing a good noise purification mechanism under the framework of different methods, such as ensemble methods [4], [5], [41], KNN or its variants [17], [22], [29], and so on. Data preprocessing technique is a necessary step to improve quality of the prediction models in some cases [28]. However, some correct samples along with some valuable information may be discarded, and in the meantime, some noise samples may be included or some new noise samples may be introduced. This is the limitation of correcting and deleting techniques. To overcome this weakness, Rebbapragada and Brodley [27] tried to use the confidence on the observed label as a weight of each instance during the training process and provided a novel framework for mitigating class noises. They empirically showed that this confidence weighting approach can outperform the discarding approach, but this new method was applied only to tree-based C4.5 classifier. The confidence-labeling technique they utilized fails to be a desirable label correction method. Wang *et al.* [45] and Zhou ıet al. [50] considered and estimated the probability of an instance being from class 1 and used it as a soft label of the instance.

B. Overview of the Proposed Approach

Inspired by instance-base methods and construction of robust algorithms, we propose a novel boosting algorithm based on label confidence, called CB-AdaBoost. The observed label of each instance is treated as uncertain. Not only the correctness but also the degree of correctness of the label is evaluated according to a certain criterion before the training procedure. We introduce the confidence of each instance into the exponential loss function. With such a modification, the misclassified and correctly classified exponential losses are weightily averaged. The weights are their corresponding probabilities represented by the correctness certainty parameter. In this way, the algorithm treats instances differently based on their confidence, and thus, it moderately controls the training intensity for each observation. The modified loss function is indeed the conditional risk or inner risk, which is quite different from an asymmetric loss or a fuzzy loss.

Our method can make a smooth transition between full acceptance and full rejection of a sample label, thereby achieving robustness and efficiency at the same time. In addition, our label-confidence-based learning has no threshold parameter, whereas correcting and deleting techniques have to define a confidence level for suspect instances so that they are relabeled or discarded in the training procedure. We derive theoretical results and also provide empirical evidences to show superior performance of the proposed CB-AdaBoost.

The contributions of this paper are as follows.

1) *A New Loss Function*: We consider the conditional risk so that label uncertainty can be directly dealt with by the concept of label confidence. This new loss function also leads to the consideration of the sign of Bayesian risk rule on each of the sample points at the initialization of the procedure.
2) *A Simple Modification of Adaptive Boosting Algorithm*: Based on the new exponential loss function, AdaBoost has a simple explicit optimization solution at each iteration.
3) Theoretical and empirical justifications for efficiency and robustness of the proposed method.

4) Consistency of the CB-AdaBoost is studied.
5) *Broad adaptivity*: The proposed CB-AdaBoost is suitable for noise data and for class-overlapping data.

C. Outline of this Paper

The remainder of this paper is organized as follows. Section II reviews the original AdaBoost. In Section III, we propose a new AdaBoost algorithm. We discuss in detail assignment of label confidence, the loss function, and the algorithm as well as its ability of adaptive learning in the label-confidence framework. Section IV devotes to a study of the consistency property. In Section V, we illustrate how the proposed algorithm works and investigate its performance through empirical studies of both synthetic and real-world data sets. Finally, this paper concludes with some final remarks in Section VI. A proof of consistency is provided in the Appendix.

II. Review of AdaBoost Algorithm

For binary classification, the main idea of AdaBoost is to produce a strong classifier by combining weak learners. This is obtained through an optimization that minimizes the exponential loss criterion over the training set. Let $\mathcal{L} = \{(\boldsymbol{x}_i, z_i)_{i=1}^n\}$ denote a given training set consisting of n independent training observations, where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T \in \mathbb{R}^p$ and $z_i \in \{1, -1\}$ represent the input attributes and the class label of the ith instance, respectively. The pseudocode of AdaBoost is given in Algorithm 1.

Algorithm 1 AdaBoost Algorithm

`Input`: $\mathcal{L} = \{(\boldsymbol{x}_i, z_i)_{i=1}^n\}$ and the maximum number of base classifiers M.
`Initialize`: For $\forall i$, $w_i^{(1)} = 1/N$, $D_i^{(1)} = w_i^{(1)}/S_1$, where $S_1 = \sum_{i=1}^n w_i^{(1)}$ is the normalization factor.
`For` $m = 1$ `To` M
1 Draw instance from $\mathcal{L}$ with replacement according to the distribution $D_i^{(m)}$ to form a training set $\mathcal{L}_m$;
2 Train $\mathcal{L}_m$ with the base learning algorithm and obtain a weak hypothesis h_m;
3 Compute $\varepsilon_m = \sum_{i:h_m(\boldsymbol{x}_i)\neq z_i}^n D_i^{(m)}$;
4 Let $\beta_m = \frac{1}{2}\ln\left(\frac{1-\varepsilon_m}{\varepsilon_m}\right)$;
`If` $\beta_m < 0$, `then` $M = m - 1$ and `abort loop`.
5 Update $w_i^{(m+1)} = w_i^{(m)} e^{-z_i \beta_m h_m(\boldsymbol{x}_i)}$; $D_i^{(m+1)} = w_i^{(m+1)}/S_{m+1}$ for $\forall i$, where $S_{m+1} = \sum_{i=1}^n w_i^{(m+1)}$;
`End For`
`Output`: $\operatorname{sign}(\sum_{m=1}^M \beta_m h_m(\boldsymbol{x}))$.

In the AdaBoost Algorithm, the current classifier h_m is induced on the weighted sampling data, and the resulting weighted error ε_m is computed. The individual weight of each of the observations is updated for the next iteration. AdaBoost is designed for clean training data—that is, each label z_i is the true label of $\boldsymbol{x}_i$. In this framework, any instance that was previously misclassified has a higher probability to be sampled in the next stage. In this way, the next classifier focuses more on those misclassified instances, and hence, the final ensemble classifier achieves high accuracy. For mislabeled data, however, those observations that were misclassified in the previous step are weighted less, and those correctly classified instances are weighted more than they should. This leads to the next training set $\mathcal{L}_{m+1}$ being seriously corrupted, and those mislabeled data eventually hurt the performance of the ensemble classifier. Therefore, some modifications should be introduced to make AdaBoost insensitive to class noise.

III. Label-Confidence-Based Boosting Algorithm

A. Label Confidence

For the class noise data problem, the observed label y associated with $\boldsymbol{x}$ may be incorrectly assumed due to some random mechanism. For the class overlapping problem, the label y associated with $\boldsymbol{x}$ is a realization of random label from some distribution. In our approach to deal with both problems, we treat the true label Z to be random. Let y (either 1 or -1) be the observed label associated with $\boldsymbol{x}$. We define a parameter γ as the probability of being correctly labeled, that is, $\gamma = P(Z = y|\boldsymbol{x})$ and $P(Z = -y|\boldsymbol{x}) = 1 - \gamma$ for $\gamma \in [0, 1]$. The quantity $|\gamma - (1-\gamma)| = (2\gamma - 1)\operatorname{sign}(2\gamma - 1)$ measures "trustworthiness" of label y and $\operatorname{sign}(2\gamma - 1) = \pm 1$ represents confidence toward correctness or wrongness of the label. Thus, we can use $\operatorname{sign}(2\gamma - 1)y$ as the trusted label with confidence level $|2\gamma - 1|$. For example, for $\gamma = 1$, $|2\gamma - 1| = 1$ and $\operatorname{sign}(2\gamma - 1) = 1$ represent that we are 100% confident about correctness of the label y, while for $\gamma = 0$, $|2\gamma - 1| = 1$ and $\operatorname{sign}(2\gamma - 1) = -1$ represent 100% certainty about the wrongness of y so that $-y$ should be 100% trusted. The label y with $\gamma = 0.5$ is the most unsure or fuzzy case with 0 confidence. It is easy to see that the trusted label $\operatorname{sign}(2\gamma - 1)y$ is exactly the Bayes rule. Let $\eta(\boldsymbol{x}) = P(Z = 1|\boldsymbol{x})$, and hence the Bayes rule is $\operatorname{sign}(2\eta(\boldsymbol{x}) - 1)$, which is equal to $\operatorname{sign}(2\gamma - 1)y$ for both $y = 1$ and $y = -1$.

For given training data $\mathcal{L} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$, let a parameter vector $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_n)$ represent their probabilities of being correctly labeled. That is, the parameter γ_i can be regarded as the confidence of a sample $\boldsymbol{x}_i$ being correctly labeled as y_i. In the following sections, we first introduce the modified loss function based on a given $\boldsymbol{\gamma}$, and then propose the CB-AdaBoost method. At the end of the section, we discuss the estimation of $\boldsymbol{\gamma}$.

B. Conditional-Risk Loss Function

Given a clean training set with correct labels z_i's available, the original AdaBoost minimizes the empirical exponential risk

$$\hat{\text{risk}}(f) = \frac{1}{n}\sum_{i=1}^n \exp(-z_i f(\boldsymbol{x}_i)) \tag{III.1}$$

over all linear combinations of base classifiers in the given space $\mathcal{H}$, assuming that an exhaustive weak learner returns

the best weak hypothesis on every round [13], [31]. Now in class noise data, the true label z_i is unknown. We only observe y_i associated with $\boldsymbol{x}_i$. Based on the assumption, given $\boldsymbol{x}_i$, the probability that Z_i is y_i is γ_i. It is natural to consider the following empirical risk:

$$\hat{R} = \frac{1}{n}\sum_{i=1}^{n}[\gamma_i \exp(-y_i f(\boldsymbol{x}_i)) + (1-\gamma_i)\exp(y_i f(\boldsymbol{x}_i))]. \tag{III.2}$$

That is, we treat the observed label y_i as a fuzzy label with γ_i correctness confidence. In other words, we consider the modified exponential loss function

$$L_\gamma(y, f(\boldsymbol{x})) = \gamma \exp(-yf(\boldsymbol{x})) + (1-\gamma)\exp(yf(\boldsymbol{x})) \tag{III.3}$$

which has a straightforward interpretation. The label y associated with $\boldsymbol{x}$ is trusted with γ confidence and it is corrected as $-y$ with $1-\gamma$ confidence. It is easy to check that the loss (III.3)

$$L_\gamma(y, f(\boldsymbol{x})) = E_{z|\boldsymbol{x}} \exp(-zf(\boldsymbol{x}))$$

which is the inner risk defined in [33]. The reason it is called the inner risk is because the true exponential risk is

$$\begin{aligned} risk(f) &= \mathbb{E}\exp(-zf(\boldsymbol{x})) && \text{(III.4)}\\ &= \mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{z|\boldsymbol{x}}[\exp(-zf(\boldsymbol{x})] \\ &= \mathbb{E}_{\boldsymbol{x}} L_\gamma(y, f(\boldsymbol{x})) && \text{(III.5)} \end{aligned}$$

for $y = \pm 1$. From this perspective, we consider minimizing the empirical inner risk of (III.5), while the original AdaBoost minimizes the empirical risk of (III.4). Steinwart and Christmann [33, Lemma 3.4] showed that the risk can be achieved by minimizing the inner risks, where the expectation is taken with respect to the marginal distribution of $\boldsymbol{x}$, in contrast to (III.4) where the expectation is taken with respect to the joint distribution of $(\boldsymbol{x}, z)$. Clearly, under the scenarios of overlapping class and label noise, the empirical inner risk (III.2) has an advantage over (III.1).

In [2], (III.3) is called the conditional ψ-risk with ψ being the exponential loss function. A classification-calibrated condition on the conditional risk is provided to ensure a pointwise form of Fisher consistency for classification. In other words, if the condition is satisfied, the 0–1 loss can be surrogated by the convex ψ loss in order to make the minimization computationally efficient. The exponential loss is classification calibrated. Our proposed method utilizes a different empirical estimator of the exponential risk. Its consistency follows from the consistency result of AdaBoosting [3] along with consistent estimation of γ. More details are presented in Section IV.

The loss (III.3) is closely related to the asymmetric loss used in the literature (see [24], [44]), but the motivation and goal of the two losses are quite different. The asymmetric loss treats two classes unequally. Two misclassification errors produce different costs. However, the costs or weights do not necessarily sum up to 1. In asymmetric loss, the ratio of two costs is usually used to measure the degree of asymmetry and is often a constant parameter, while in (III.3), it is a function of $\boldsymbol{x}$. Also the loss (III.3) takes a linear combination of the exponential loss at y and $-y$, while the asymmetric loss takes only one.

Indeed, γ in the loss (III.3) is the posterior probability used in [38] for the support vector machine technique. The similarity is that we all use the sign of the Bayes rule as the trusted label. However, we also include the magnitude $|2\gamma - 1|$ in our loss function. We associate the trusted label with a confidence $|2\gamma - 1|$, while in [38], the confidence is always 1. The idea of label confidence is closely related to fuzzy label used in fuzzy support vector machines [21]. The difference is that fuzzy label assigns an importance weight only for the observed label without considering its correctness.

Next, we derive the proposed method based on the modified exponential loss function.

C. Derivation of Our Algorithm

For an additive model

$$f_M(\boldsymbol{x}) = \sum_{m=1}^{M} \beta_m h_m(\boldsymbol{x}) \tag{III.6}$$

where $h_m(\boldsymbol{x}) \in \{-1, 1\}$ is a weak classifier in the mth iteration, β_m is its coefficient, and $f_M(\boldsymbol{x})$ is an ensemble classifier. Our goal is to learn an ensemble classifier with a forward stage-wise estimation procedure by fitting an additive model to minimize the modified loss functions. Let us consider an update from $f_{m-1}(\boldsymbol{x})$ to $f_m(\boldsymbol{x}) = f_{m-1}(\boldsymbol{x}) + \beta_m h_m(\boldsymbol{x})$ by minimizing (III.2). This is an optimization problem to find solutions h_m and β_m, i.e.,

$$\begin{aligned} (\beta_m, h_m) &= \arg\min_{\beta,h} \sum_{i=1}^{n}[\gamma_i \exp(-y_i f_m(\boldsymbol{x}_i)) \\ &\qquad + (1-\gamma_i)\exp(y_i f_m(\boldsymbol{x}_i))] \\ &= \arg\min_{\beta,h} \sum_{i=1}^{n}\left[w_{1i}^{(m)} \exp(-y_i \beta h(\boldsymbol{x}_i)) \right. \\ &\qquad \left. + w_{2i}^{(m)} \exp(y_i \beta h(\boldsymbol{x}_i))\right] \end{aligned} \tag{III.7}$$

where $w_{1i}^{(m)} = \gamma_i e^{-y_i f_{m-1}(\boldsymbol{x}_i)}$ and $w_{2i}^{(m)} = (1-\gamma_i)e^{y_i f_{m-1}(\boldsymbol{x}_i)}$ are independent with h_m and β_m.

As we will show, h_m and β_m can be derived separately in two steps. Let us first optimize the weak hypothesis h_m. The summation in (III.7) can be expressed alternatively as

$$\begin{aligned} &\sum_{i=1}^{n}\left[w_{1i}^{(m)} \exp(-y_i \beta h(\boldsymbol{x}_i)) + w_{2i}^{(m)} \exp(y_i \beta h(\boldsymbol{x}_i))\right] \\ &\quad = \sum_{\{i:h(\boldsymbol{x}_i)=y_i\}}^{n}\left[w_{1i}^{(m)} e^{-\beta} + w_{2i}^{(m)} e^{\beta}\right] \\ &\qquad + \sum_{\{i:h(\boldsymbol{x}_i)\neq y_i\}}^{n}\left[w_{1i}^{(m)} e^{\beta} + w_{2i}^{(m)} e^{-\beta}\right] \\ &\quad = \sum_{i=1}^{n}\left[w_{1i}^{(m)} e^{-\beta} + w_{2i}^{(m)} e^{\beta}\right] \\ &\qquad + (e^{\beta} - e^{-\beta}) \sum_{\{i:h(\boldsymbol{x}_i)\neq y_i\}}^{N}\left[w_{1i}^{(m)} - w_{2i}^{(m)}\right]. \end{aligned}$$

Therefore, for any given value of $\beta > 0$, (III.7) is equivalent to the minimization of

$$h_m = \arg\min_h \sum_{i=1}^{n} \left[w_{1i}^{(m)} - w_{2i}^{(m)}\right] I\{h(\boldsymbol{x}_i) \neq y_i\}. \quad \text{(III.8)}$$

It is worthwhile to mention that the term $(w_{1i}^{(m)} - w_{2i}^{(m)})$ may be negative, and hence it cannot be directly interpreted as the weight of the instance (x_i, y_i) in the training set. According to the analytical solution of h_m, the base classifier is expected to correctly predict $(\boldsymbol{x}_i, y_i)$ in the case of $w_{1i}^{(m)} \geq w_{2i}^{(m)}$ and otherwise misclassify $(\boldsymbol{x}_i, y_i)$. This is equivalent to solving

$$\min_h \sum_{i=1}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right| I\left\{h(\boldsymbol{x}_i) \neq \text{sign}\left(\left[w_{1i}^{(m)} - w_{2i}^{(m)}\right] y_i\right)\right\}. \quad \text{(III.9)}$$

In other words, h_m is actually the one that minimizes the prediction error over the set $\{(\boldsymbol{x}_i, \text{sign}([w_{1i}^{(m)} - w_{2i}^{(m)}]y_i)_{i=1}^n\}$ with each instance weighted $|w_{1i}^{(m)} - w_{2i}^{(m)}|$. In each iteration, we treat $\text{sign}([w_{1i}^{(m)} - w_{2i}^{(m)}]y_i)$ as the label of $\boldsymbol{x}_i$ and $|w_{1i}^{(m)} - w_{2i}^{(m)}|$ as its importance. This provides a theoretical justification of the sampling scheme in our proposed algorithm, which is given later.

Next, we optimize β_m. With h_m fixed, β_m minimizes

$$\sum_{i:h_m(\boldsymbol{x}_i)=y_i} \left[w_{1i}^{(m)} e^{-\beta} + w_{2i}^{(m)} e^{\beta}\right] + \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} \left[w_{1i}^{(m)} e^{\beta} + w_{2i}^{(m)} e^{-\beta}\right]. \quad \text{(III.10)}$$

Upon setting the derivative of (III.10) (with respect to β) to zero, we obtain

$$\beta_m = \frac{1}{2} \ln \frac{\sum_{i:h_m(\boldsymbol{x}_i)=y_i} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} w_{2i}^{(m)}}{\sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)=y_i} w_{2i}^{(m)}}. \quad \text{(III.11)}$$

Note that the condition that

$$\sum_{i:h_m(\boldsymbol{x}_i)=y_i} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} w_{2i}^{(m)} > \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)=y_i} w_{2i}^{(m)} \quad \text{(III.12)}$$

should hold in order to ensure the value of β_m is positive.

The approximation on the mth iteration is then updated as

$$f_m(\boldsymbol{x}) = f_{m-1}(\boldsymbol{x}) + \beta_m h_m(\boldsymbol{x}) \quad \text{(III.13)}$$

which leads to the following update of $w_{1i}^{(m)}$ and $w_{2i}^{(m)}$:

$$w_{1i}^{(m+1)} = w_{1i}^{(m)} e^{-y_i \beta_m h_m(\boldsymbol{x}_i)}$$

and

$$w_{2i}^{(m+1)} = w_{2i}^{(m)} e^{y_i \beta_m h_m(\boldsymbol{x}_i)}.$$

By repeating the procedure above, we can derive the iterative process for all rounds $m \geq 2$ until $m = M$ or condition (III.12) is not satisfied. The initial values take $w_{1i}^{(1)} = \gamma_i$ and $w_{2i}^{(1)} = 1 - \gamma_i$. Now we write the procedure into the pseudocode of Algorithm 2.

Algorithm 2 CB-AdaBoost Algorithm

Input: $\mathcal{L} = \{(\boldsymbol{x}_i, y_i)_{i=1}^n\}$, $\boldsymbol{\gamma} = \{(\gamma_i)_{i=1}^n\}$ and M

Initialize: For $\forall i$, $w_{1i}^{(1)} = \gamma_i$, $w_{2i}^{(1)} = 1 - \gamma_i$, $D_i^{(1)} = |w_{1i}^{(1)} - w_{2i}^{(1)}|/S_1$, where $S_1 = \sum_{i=1}^n |w_{1i}^{(1)} - w_{2i}^{(1)}|$

For $m = 1$ To M

1 Relabel all instances in $\mathcal{L}$ to compose a new data set as $\mathcal{L}' = \{(\boldsymbol{x}_i, y_i')_{i=1}^n\}$, where $\boldsymbol{x}_i \in \mathcal{L}$, $y_i' = \text{sign}[(w_{1i}^{(m)} - w_{2i}^{(m)})y_i]$;

2 Draw instance from $\mathcal{L}'$ with replacement according to the distribution $D_i^{(m)}$ to compose a training set $\mathcal{L}_m$;

3 Train $\mathcal{L}_m$ with the base learning algorithm and obtain a weak hypothesis h_m;

4 Let $\beta_m = \frac{1}{2} \ln \frac{\sum_{i:h_m(\boldsymbol{x}_i)=y_i} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} w_{2i}^{(m)}}{\sum_{i:h_m(\boldsymbol{x}_i)\neq y_i} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)=y_i} w_{2i}^{(m)}}$

If $\beta_m < 0$, then $M = m - 1$ and abort loop.

5 Update $w_{1i}^{(m+1)} = w_{1i}^{(m)} e^{-y_i \beta_m h_m(\boldsymbol{x}_i)}$; $w_{2i}^{(m+1)} = w_{2i}^{(m)} e^{y_i \beta_m h_m(\boldsymbol{x}_i)}$; $D_i^{(m+1)} = |w_{1i}^{(m+1)} - w_{2i}^{(m+1)}|/S_{m+1}$ for $\forall i$, where $S_{m+1} = \sum_{i=1}^n |w_{1i}^{(m+1)} - w_{2i}^{(m+1)}|$;

End For

Output: $\text{sign}(\sum_{m=1}^M \beta_m h_m(\boldsymbol{x}))$.

D. Class Noise Mitigation

In this section, we study the effect of label confidence, and we investigate the adaptive ability of CB-AdaBoost in the mitigation of overfitting and class noise from aspects of its reweighting procedure and classifier combination rule.

First, the initialization of distribution shows different initial emphases on training instances between Algorithms 1 and 2. As discussed early, $|\gamma_i - (1 - \gamma_i)|$ actually represents its label certainty, and it is used as the initial weight in Algorithm 2. The conditional risk type of loss function leads to this initialization and the weighting strategy that distinguishes instances based on their own confidences. Consequently, the instances with a high certainty receive a priority to be trained. This makes sense as these instances are usually those identifiable from a statistical standpoint, and thus, they are more valuable in classification. By contrast, Algorithm 1 treats each instance equally at the beginning without considering the reliability on the samples.

Second, we consider $y_i' = \text{sign}(2\gamma_i - 1)y_i$ as the label of $\boldsymbol{x}_i$ in Algorithm 2. Under the mislabeled or class overlapping scenarios, this design makes sense because $\text{sign}(2\gamma_i - 1)$ represents the confidence toward correctness or wrongness of the label y_i. If $\text{sign}(2\gamma_i - 1) = 1$, y_i should be trusted with confidence $|2\gamma_i - 1|$. Nevertheless, if $\text{sign}(2\gamma_i - 1) = -1$, $-y_i$ should be trusted with confidence $|2\gamma_i - 1|$. The original AdaBoost trusts label y_i completely, which is inappropriate under mislabeling and class overlapping. As shown before, the trust label y_i' in CB-AdaBoost has the same sign as the Bayes rule at sample point $\boldsymbol{x}_i$. Intuitively, our method takes more information at the initialization.

Third, we take a detailed look at the weight updating formulas in Algorithm 2 and subsequently obtain the following results on the first reweighting process. We say that an instance $\boldsymbol{x}_i$ is misclassified at the mth iteration if $h_m(\boldsymbol{x}_i) \neq y_i'$, where $y_i' = \text{sign}[(w_{1i}^m - w_{2i}^m)y_i]$; otherwise, it is correctly classified.

Proposition 1: The misclassified instance receives larger weight for the next iteration.

Proof: Two types of misclassification are either $h_m(x_i) \neq y_i$ with $w_{1i}^{(m)} > w_{2i}^{(m)}$ or $h_m(x_i) \neq -y_i$ with $w_{1i}^{(m)} < w_{2i}^{(m)}$. In the first case

$$\begin{aligned}\left|w_{1i}^{(m+1)} - w_{2i}^{(m+1)}\right| &= \left|w_{1i}^{(m)} e^{\beta_m} - w_{2i}^{(m)} e^{-\beta_m}\right| \\ &> \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right|\end{aligned}$$

while in the second case

$$\begin{aligned}\left|w_{1i}^{(m+1)} - w_{2i}^{(m+1)}\right| &= \left|w_{1i}^{(m)} e^{-\beta_m} - w_{2i}^{(m)} e^{\beta_m}\right| \\ &> \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right|.\end{aligned}$$

In both cases, the weight increases.

Proposition 2: If an instance is correctly classified and its certainty is high enough so that $\max\{w_{1i}^{(m)}, w_{2i}^{(m)}\} > e^{\beta_m} \min\{w_{1i}^{(m)}, w_{2i}^{(m)}\}$, then it receives smaller weight at the next iteration.

Proof: We can easily check two cases. For the case of $w_{1i}^{(m)} > w_{2i}^{(m)}$ and $h_m(x_i) = y_i$, when $w_{1i}^{(m)} > e^{\beta_m} w_{2i}^{(m)}$, we have

$$\begin{aligned}\left|w_{1i}^{(m+1)} - w_{2i}^{(m+1)}\right| &= \left|w_{1i}^{(m)} e^{-\beta_m} - w_{2i}^{(m)} e^{\beta_m}\right| \\ &< \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right|.\end{aligned}$$

For the case of $w_{1i}^{(m)} < w_{2i}^{(m)}$ and $h_m(x_i) = -y_i$, if $w_{1i}^{(m)} > e^{\beta_m} w_{2i}^{(m)}$, we have

$$\begin{aligned}\left|w_{1i}^{(m+1)} - w_{2i}^{(m+1)}\right| &= \left|w_{1i}^{(m)} e^{\beta_m} - w_{2i}^{(m)} e^{-\beta_m}\right| \\ &< \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right|.\end{aligned}$$

Propositions 1 and 2 show that on the first important stage, CB-AdaBoost inherits the adaptive learning ability of AdaBoost and has the distinction that it adjusts the distribution of instances according to the current classification with respect to the commonly agreed information. Moreover, the degree of adjustment is managed by the confidence of each sample. For the following iterations, we can imagine the resampling process. The weights of instances with high confidence stay at a high level until most of them are sufficiently learned. After that, their proportion decreases rapidly while the proportion of instances with low confidence increases gradually. As uncertain instances consist of most of the training set, the training process is difficult to continue. On the other hand, once a new classifier becomes no better than a random guess, then an early stop in the iterative process is possible. This is because condition (III.12) no longer holds in that case. Thus, our proposed method effectively prevents the ensemble classifier from overfitting.

Fourth, let us scrutinize the classifier ensemble rule.

Proposition 3: In the framework of Algorithm 2, define ε_m' as the error rate of h_m over its training set $\mathcal{L}_m$ during the mth iteration—that is, $\varepsilon_m' = \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i'}^{n} |w_{1i}^{(m)} - w_{2i}^{(m)}|/S_m$. We then have

$$\beta_m < \frac{1}{2}\ln\left(\frac{1-\varepsilon_m'}{\varepsilon_m'}\right).$$

Proof: We can prove this result by giving an equivalent representation of β_m as

$$\begin{aligned}\beta_m &= \frac{1}{2}\ln\left(\frac{\sum_{i:h_m(\boldsymbol{x}_i)=y_i}^{N} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i}^{n} w_{2i}^{(m)}}{\sum_{i:h_m(\boldsymbol{x}_i)\neq y_i}^{n} w_{1i}^{(m)} + \sum_{i:h_m(\boldsymbol{x}_i)=y_i}^{n} w_{2i}^{(m)}}\right) \\ &= \frac{1}{2}\ln\left(\frac{\sum_{i:h_m(\boldsymbol{x}_i)=y_i'}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right| + c}{\sum_{i:h_m(\boldsymbol{x}_i)\neq y_i'}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right| + c}\right)\end{aligned}$$

where

$$c = \sum_{i:w_{1i}^{(m)} < w_{2i}^{(m)}}^{n} w_{1i}^{(m)} + \sum_{i:w_{1i}^{(m)} > w_{2i}^{(m)}}^{n} w_{2i}^{(m)}.$$

With condition (III.12) being satisfied, we obtain $\sum_{i:h_m(\boldsymbol{x}_i)=y_i'}^{n} |w_{1i}^{(m)} - w_{2i}^{(m)}| > \sum_{i:h_m(\boldsymbol{x}_i)\neq y_i'}^{n} |w_{1i}^{(m)} - w_{2i}^{(m)}|$, which implies

$$\begin{aligned}\frac{1-\varepsilon_m'}{\varepsilon_m'} &= \frac{\sum_{i:h_m(\boldsymbol{x}_i)=y_i'}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right|}{\sum_{i:h_m(\boldsymbol{x}_i)\neq y_i'}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right|} \\ &> \frac{\sum_{i:h_m(\boldsymbol{x}_i)=y_i'}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right| + c}{\sum_{i:h_m(\boldsymbol{x}_i)\neq y_i'}^{n} \left|w_{1i}^{(m)} - w_{2i}^{(m)}\right| + c}.\end{aligned}$$

Thus, the proof of Proposition 3 is complete.

It turns out that β_m calculated in our modified algorithm does not take into account the full value of the odd ratio for each hypothesis. In fact, it is smaller than that calculated in AdaBoost, so our algorithm combines base classifiers and updates instance weights modestly. This effectively avoids the situation where some hypotheses dominated by substantial classification noise are exaggerated by their large coefficients in the final classifier.

We have studied the CB-AdaBoost algorithm in detail and compared its advantages with the original one. Next, we discuss the remaining issue of how to estimate label confidence.

E. Assignment of Label Confidence

In most cases, since it is difficult to track the data collection process and identify where corruptions will most likely occur, we evaluate the confidence on labels according to the statistical characteristics of the data itself. In this regard, [27] suggested a pairwise expectation maximization method to compute confidence of labels. Cao *et al.* [6] applied KNN to detect suspicions examples. However, a direct application of these methods may not be efficient for data sets whose noise level is high. We believe that a cleaner data set can make a better confidence estimation. Therefore, before confidence assignment, a noise filter shall be introduced to eliminate very suspicious instances so that we are able to extract more reliable statistical characteristics from the remaining data.

First, a noise filter scans over the original data set. Using a similarity measure between instances to find a neighborhood of each instance, one can compute the agreement rate for its label from its neighbors. The instances with an agreement rate below a certain threshold are eliminated. The above process can be repeated several times since some suspect instances may be exposed later when their neighborhood changes. In our experiment, the threshold is set to 0.07 at the beginning with an increment of 0.07 in each subsequent round. The process is repeated three times, and the final cut-off value for the agreement rate is 0.21 so that the sample size does not decrease much. In the meantime, distributional information of the sample is kept relatively intact. Once a filtered data set, denoted by $\mathcal{L}_{red}$, is obtained, two methods can be used to compute label confidence.

If the noise level ε over the training labels is known or can be estimated, we can represent the frequency of observations with label y as follows:

$$\begin{aligned} P(Y=y) &= P(Y=y, Z=y) + P(Y=y, Z=-y) \\ &= (1-\varepsilon)P(Z=y) + \varepsilon P(Z=-y) \end{aligned}$$

where the noise level $\varepsilon = P(Y=y|Z=-y) = P(Y=-y|Z=y)$. This representation explains two sources for the composition of label y: correctly labeled instances belonging to true class y and mislabeled instances belonging to true class $-y$. Then $P(Z=y) = (P(Y=y)-\varepsilon)/(1-2\varepsilon)$, and utilizing the Bayesian formula, we assess the confidence as follows:

$$\begin{aligned} \gamma &= P(Z=y|\boldsymbol{x}) = \frac{P(Z=y)f(\boldsymbol{x}|Z=y)}{f(\boldsymbol{x})} \\ &= \frac{P(Z=y)f(\boldsymbol{x}|Z=y)}{f(\boldsymbol{x}|Z=y)P(Z=y) + f(\boldsymbol{x}|Z=-y)P(Z=-y)} \\ &= \frac{(P(Y=y)-\varepsilon)f(\boldsymbol{x}|Z=y)}{(P(Y=y)-\varepsilon)f(\boldsymbol{x}|Z=y) + \varepsilon f(\boldsymbol{x}|Z=-y)}. \end{aligned}$$

With conditional distribution type known, $f(\boldsymbol{x}|Z=y)$ and $f(\boldsymbol{x}|Z=-y)$ can be estimated under $\mathcal{L}_{red}$ while $P(Y=y)$ is directly set to be the sample proportion of class y in $\mathcal{L}$.

The second method does not need to assume the noise level. KNN is recalled to assign confidence on each label. Based on $\mathcal{L}_{red}$, the label agreement rate of each instance among its nearest neighbors can act as its confidence. Thus, the confidence probability of an example $(\boldsymbol{x}, y)$ in $\mathcal{L}$ is computed as follows:

$$P(Z=y|\boldsymbol{x}) = \frac{1}{K}\sum_{j=1}^{K}\sum_{\boldsymbol{x}_j \in \mathcal{N}(\boldsymbol{x})} I(y_j = y) \qquad \text{(III.14)}$$

where $\mathcal{N}(\boldsymbol{x})$ represents the set containing K nearest neighbors of $\boldsymbol{x}$ from $\mathcal{L}_{red}$. In our experiment, $K=5$ is used.

In the simulation of Section V, we will evaluate the quality of confidence assigned by these two methods. In practice, however, the Bayesian method is usually infeasible since the noise level is unknown.

F. Relationship to Previous Work

Note that our modified algorithm reduces to AdaBoost if we set the confidence on each label to one. The greater the confidence on each instance, the less CB-AdaBoost differs from AdaBoost in terms of the weight updating and base classifiers, as well as their coefficients in successive iterations. Rebbapragada and Brodley [27] proposed instance weighting via confidence in order to mitigate class noise. They attempted to assign confidence on instance label such that incorrect labels receive lower confidences. We share a similar opinion in dealing with noise data, but instance weighting via confidence itself seems to be a discarding technique rather than a correcting technique. That is, a low confidence implies an attempt to eliminate the example, while a high confidence implies keeping it. By contrast, our algorithm considers both the correctly labeled and mislabeled probability for an instance. Therefore, the loss function

$$L_\gamma(y, f(x)) = \gamma e^{-yf(x)} + (1-\gamma)e^{yf(x)}$$

explains the attitude toward an instance: delete it with γ and correct it by $1-\gamma$. In other words, our algorithm can be viewed as a composition technique of discarding and correcting. For the same reason, our algorithm differs from those proposed in [14] and [6]. In their discussions, they suggested heuristic algorithms to delete or revise suspicious examples during iterations in order to improve the accuracy of AdaBoost for mislabeled data. In our algorithm, the suspicious labels are similarly revised, which is a consequence of minimizing the modified loss function (III.2). The trusted label at each sample point is the sign of the Bayes rule and is associated with a confidence level.

Other closely related work includes [45] and [50]. Both consider the same confidence level of $\boldsymbol{x}_i$ as $p_i = p(z_i = 1|\boldsymbol{x}_i)$, whereas our approach takes advantage of the observed label y_i by considering $\gamma_i = P(z_i = y_i|\boldsymbol{x}_i)$. We evaluate confidence of the observed label y_i, while they assess confidence of the positive label $+1$. In [50], the initial weight $|2p_i - 1|$ is very similar to our choice, but our reweighting and classifier combination rules are different. Reference [45] has a similar combination rule as ours, but the initial weights are different.

IV. Consistency of CB-AdaBoosting

In this section, we study consistency of the proposed CB-AdaBoosting method with label confidences estimated by KNN approach. Several authors have shown that the original and modified versions of AdaBoost are consistent. For example, Zhang and Yu [47] considered a general "boosting" with a step size restriction. Lugosi and Vayatis [23] proved the consistency of regularized boosting methods. Bartlett and Traskin [3] studied the stopping rule of the traditional AdaBoost that guarantees its consistency. In our algorithm, we use the exponential loss function. We just use a different empirical version of the exponential risk. This enables us to adopt the stopping strategy used in [3] with a consistency result on the nearest neighborhood method (see [8], [34]) to show that the proposed CB-AdaBoost is Bayes-risk consistent.

We use notation similar to [3]. Let $(\boldsymbol{X}, Z)$ be a pair of random values in $\mathbb{R}^p \times \{-1, 1\}$ with the joint distribution $P_{X,Z}$ and the marginal distribution of $\boldsymbol{X}$ being P_X. The training sample data $\mathcal{L}_n = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ are available, having the

same distribution as $(\boldsymbol{X}, Z)$. The mislabel problem can be treated as the case $P_{X,Z}$ being a contamination distribution. The CB-AdaBoost produces a classifier $g_n = \text{sign}(f_n) : \mathbb{R}^p \to \{-1, 1\}$ based on this sample $\mathcal{L}_n$. The misclassification probability is given by

$$L(g_n) = P(g_n(\boldsymbol{X}) \neq Z|\mathcal{L}_n).$$

Our goal is to prove that $L(g_n)$ approaches the Bayes risk

$$L^* = \inf_f L(f) = \mathbb{E}(\min(\eta(\boldsymbol{X}), 1 - \eta(\boldsymbol{X})))$$

as $n \to \infty$, where the infimum is taken over all measurable classifiers and where $\eta(\boldsymbol{X})$ is the conditional probability $\eta(\boldsymbol{X}) = P(Z = 1|\boldsymbol{X})$.

Assume that $\mathcal{H}$ is the set of all linear combinations of base classifiers and has a finite VC dimension. The proposed CB-AdaBoost finds a combination f in $\mathcal{H}$ that minimizes

$$R_{n,k_n}(f) = \frac{1}{n}\sum_{i=1}^{n}[\hat{\gamma}_i \exp(-y_i f(\boldsymbol{x}_i)) + (1 - \hat{\gamma}_i)\exp(y_i f(\boldsymbol{x}_i))]$$

where $\hat{\gamma}_i$ is a K-NN estimator of $\gamma_i = P(Z = y_i|\boldsymbol{x}_i)$. That is

$$\hat{\gamma}_i = \frac{1}{k_n}\sum_{j=1}^{k_n}\sum_{\boldsymbol{x}_j \in \mathcal{N}(\boldsymbol{x}_i)} I(y_j = y_i)$$

where $\mathcal{N}(\boldsymbol{x}_i)$ denotes the set containing k_n nearest neighbors of $\boldsymbol{x}_i$. We denote

$$\bar{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n}[\gamma_i \exp(-y_i f(\boldsymbol{x}_i)) + (1 - \gamma_i)\exp(y_i f(\boldsymbol{x}_i))]$$

and the true exponential risk as

$$R(f) = \mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{z|\boldsymbol{x}} \exp(-Zf(\boldsymbol{X})) = \mathbb{E}\exp(-Zf(\boldsymbol{X}).$$

We first prove that the CB-Adaboost is consistent with the exponential risk. Then, by [2], its 0–1 risk also approaches the Bayes risk L^*, since the exponential loss is classification calibrated.

We shall denote the convex hull of $\mathcal{H}$ scaled by $\lambda \geq 0$ as

$$\mathcal{F}_\lambda = \{f|f = \sum_{i=1}^{n}\beta_i h_i, n \in \mathcal{N} \cup \{0\}, \quad \sum_{i=1}^{n}\beta_i = \lambda, h_i \in \mathcal{H}\}$$

and the set of t-combinations, $t \in \mathcal{N}$, of functions in $\mathcal{H}$ is denoted by

$$\mathcal{F}^t = \{f|f = \sum_{i=1}^{t}\beta_i h_i, \beta_i \in \mathbb{R}, h_i \in \mathcal{H}\}.$$

Define the truncated function $\pi_l(\cdot)$ to be

$$\pi_l(x) = xI(x \in [-l, l]) + l\text{sign}(x)$$

where $I(x)$ is the indicator function. The set of truncated functions is $\pi_l \circ \mathcal{F} = \{\tilde{f}|\tilde{f} = \pi_l(f), f \in \mathcal{F}\}$ and the set of classifiers based on a class $\mathcal{F}$ is denoted by $g \circ \mathcal{F} = \{\tilde{f}|\tilde{f} = g(f), f \in \mathcal{F}\}$.

Based on the stopping strategy of [3] and the universal consistency of nearest neighbor function estimate of [8], we have the following proposition.

Proposition 4: Assume that $V = d_{VC}(\mathcal{H}) < \infty$ and that $\mathcal{H}$ is dense in the sense of $\lim_{\lambda\to\infty}\inf_{f\in\mathcal{F}_\lambda} R(f) = R^*$. Further assume $k_n \to \infty$, $k_n/n \to 0$ and $t_n = n^{1-a}$ for $a \in (0, 1)$. Then the CB-AdaBoost stopped at step t_n returns a sequence of classifiers almost surely satisfying $L(g(f_n)) \to L^*$.

The proposition states the strong consistency of the proposed CB-AdaBoost method if it stops at $t_n = n^{1-a}$ and the size of neighbors for estimating label confidence $k_n \to \infty$ but $k_n/n \to 0$. A proof of Proposition 4 is given in the Appendix.

V. Experiments

To begin, we run three experiments to investigate performance of our proposed algorithm on synthetic data. The first one examines the quality of assigned label confidence, since it has a great impact on the effectiveness of the proposed method. The second explores the advantages of the proposed algorithm over other commonly used methods in dealing with noise data. The third experiment demonstrates significant differences of weights between the proposed algorithm and the original AdaBoost method. We generate random samples from two scenarios with increasing levels of label noise.

Norm: Two classes of data are sampled from bivariate normal distributions $N((0, 0)^T, I)$ and $N((2, 2)^T, I)$, respectively.

Sine: Random vectors $\boldsymbol{x}_i = (x_{i1}, x_{i2})^T$ uniformly distributed on $[-3, 3] \times [-3, 3]$ are simulated, and their labels are assigned according to the conditional probability $P(z = y|\boldsymbol{x}_i) = e^{yg(\boldsymbol{x}_i)}/(e^{yg(\boldsymbol{x}_i)} + e^{-yg(\boldsymbol{x}_i)})$, where $y \in \{1, -1\}$ and $g(\boldsymbol{x}_i) = ((x_{i2} - 3\sin x_{i1}))/2$.

Data sets consist of 50/500 training observations and 10 000 testing instances. We introduce mislabeled data by randomly choosing training instances and reversing their labels.

We then carry out the experiment on real data sets from the UCI repository [20]. Seventeen data sets of different sizes with different numbers of input variables are used to compare performance of the proposed algorithm with some existing robust boosting methods.

We set the number of iterations M to be 200 for all ensemble classifiers. The base classifier used in the AdaBoost and CB-AdaBoost is the classification stump, the simplest one-level decision tree.

A. Assessing the Quality of Label Confidence

It is expected that the label confidence of clean sample instances shall be high, while for mislabeled instances, the confidence should be low. In this experiment, we examine two assignment methods previously introduced in Section III-E by assessing the quality of their label confidence results. We use the Bayesian method on the Normal data in which the noise level is known to be 0%, 10%, and 20%, respectively. The KNN method is used on the Sine data. The number of nearest neighbors (K) used in KNN is selected from the range 3–15, and is set to 5 for consideration of balance between accuracy and computation efficiency.

Table I reports the average and standard deviation of confidences on clean and mislabeled samples. The averages and standard deviations are calculated through 30 repetitions.

TABLE I

AVERAGE AND STANDARD DEVIATION OF THE CONFIDENCES FOR CLEAN AND MISLABELED SAMPLES IN TWO DATA SETS WITH DIFFERENT NOISE LEVELS

			Noise Level		
			10%	20%	30%
Normal	$n = 50$	Clean	0.8919 ± 0.2068	0.8693 ± 0.2267	0.8201 ± 0.2519
		Mislabeled	0.0581 ± 0.0616	0.1795 ± 0.2265	0.4459 ± 0.4018
	$n = 500$	Clean	0.9172 ± 0.2011	0.8547 ± 0.1978	0.7145 ± 0.1514
		Mislabeled	0.0850 ± 0.1790	0.1446 ± 0.1843	0.2742 ± 0.1499
Sine	$n = 50$	Clean	0.8551 ± 0.2503	0.8503 ± 0.2720	0.7142 ± 0.3556
		Mislabeled	0.1833 ± 0.2905	0.3888 ± 0.4047	0.4661 ± 0.3999
	$n = 500$	Clean	0.8731 ± 0.2639	0.8543 ± 0.2675	0.8451 ± 0.2844
		Mislabeled	0.2870 ± 0.3832	0.4142 ± 0.4195	0.4958 ± 0.4257

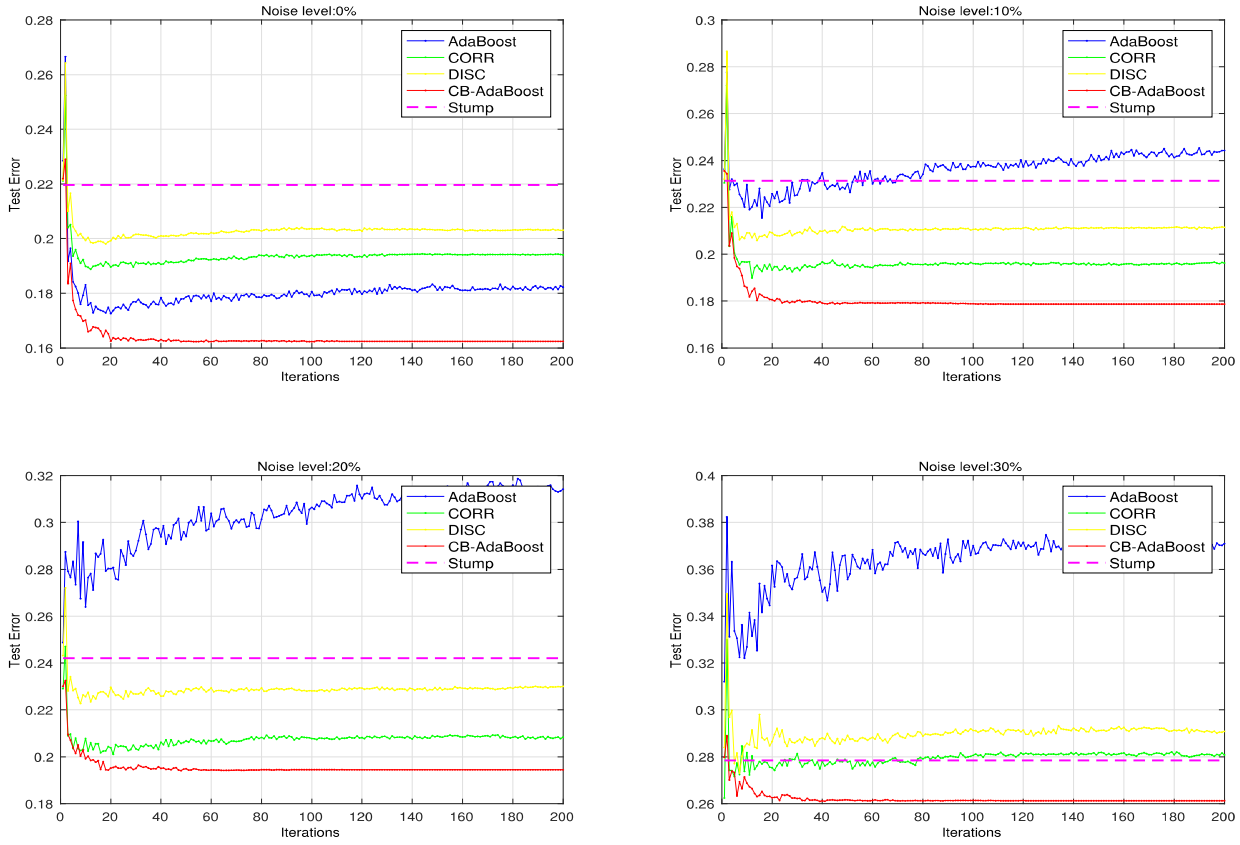

Fig. 1. Testing errors of each method under different noise levels (0%, 10%, 20%, and 30%) as the number of iterations increases.

As expected, there exists a significant separation between two types of samples on confidences. On average, clean labels achieve a higher degree of confidence than corrupted ones. For example, under 10% contamination of normal samples of size $n = 500$, confidence for clean sample is 0.9172 compared with 0.0855 for mislabeled sample. For the small size $n = 50$, the confidence difference is also significant with 0.8919 for clean sample and 0.0581 for mislabeled sample. As the noise level increases, the difference of label confidences between clean data and mislabeled data becomes smaller. This phenomenon mentioned in [27] is understandable because the certainty decreases in high noise data and because assignment methods tend to be more conservative than they are in low noise data.

B. Comparisons With Discarding and Correcting Methods

We compare the efficiency of the label-confidence-based learning with the discarding and correcting techniques. For the latter two, a threshold on confidence is prespecified to define suspect samples. We consider four types of classifiers: 1) AdaBoost; 2) AdaBoost working on the data with suspected samples having been discarded (DISC); 3) AdaBoost working on the original training set but suspected labels having been corrected (CORR); and 4) CB-AdaBoost. We repeat the procedure 30 times and record the test errors of the four classifiers.

Fig. 1 illustrates how the average test error changes as the number of iterations increases for different classifiers based on the training set of size 50. The threshold is set to 0.5

TABLE II

AVERAGE AND STANDARD DEVIATION OF TESTING ERRORS OF EACH METHOD UNDER DIFFERENT NOISE LEVELS. DISCARDING AND CORRECTING METHODS USE 0.20, 0.50, AND 0.80 AS THE CONFIDENCE THRESHOLDS. THE SMALLEST ERRORS ARE SHOWN IN BOLD

Data	Level	n	AdaBoost	DISC20	DISC50	DISC80	CORR20	CORR50	CORR80	CB-AdaBoost
Normal	0%	50	.1453±.0302	.1407±.0308	.1510±.0410	.1719±.0410	.1457±.0537	.1413±.0429	.1484±.0272	**.1070±.0168**
		500	.0942±.0040	.0863±.0038	.0848±.0037	.0899±.0074	.0857±.0037	.0833±.0026	.0888±.0044	**.0809±.0032**
	10%	50	.1979±.0419	.1447±.0464	.1562±.0452	.1779±.0544	.1410±.0458	.1468±.0441	.1482±.0627	**.1128±.0269**
		500	.1296±.0108	.0901±.0044	.0881±.0049	.0949±.0078	.0895±.0042	.0863±.0044	.0921±.0052	**.0835±.0048**
	20%	50	.2749±.0576	.1678±.0418	.1769±.0528	.2041±.0490	.1651±.0453	.1782±.0530	.1900±.0577	**.1390±.0366**
		500	.1742±.0183	.0967±.0069	.0882±.0047	.1048±.0142	.0953±.0070	.0857±.0047	.1099±.0164	**.0849±.0050**
	30%	50	.3446±.0391	.2602±.0771	.2450±.0872	.3270±.1381	.2679±.0782	.2497±.1169	.2994±.0882	**.2375±.1195**
		500	.2474±.0298	.1457±.0205	.1015±.0134	.6049±.4305	.1410±.0220	**.1014±.0131**	.2397±.0699	.1028±.0173
Sine	0%	50	.2305±.0221	.2271±.0272	.2358±.0508	.2420±.0281	.2306±.0316	.2307±.0330	.2402±.0228	**.2139±.0188**
		500	.1934±.0074	.1850±.0071	.1859±.0090	.1871±.0076	.1851±.0073	.1861±.0083	.1891±.0087	**.1834±.0067**
	10%	50	.2872±.0303	.2497±.0450	.2430±.0343	.2469±.0353	.2408±.0310	.2428±.0328	.2566±.0339	**.2318±.0299**
		500	.2242±.0086	.1961±.0089	.1934±.0083	.1902±.0081	.1954±.0099	.1926±.0085	.1931±.0100	**.1887±.0098**
	20%	50	.3247±.0433	.2782±.0406	.2761±.0395	.2848±.0717	.2754±.0432	.2786±.0494	.2978±.0558	**.2672±.0540**
		500	.2641±.0162	.2295±.0135	.2236±.0135	.2166±.0147	.2250±.0129	.2217±.0140	.2275±.0135	**.2096±.0168**
	30%	50	.4017±.0545	.3349±.0711	.3433±.0822	.3448±.0793	.3338±.0709	.3320±.0766	.3497±.0687	**.3258±.0811**
		500	.3166±.0310	.2676±.0292	.2671±.0270	.2576±.0264	.2659±.0267	.2661±.0285	.2654±.0290	**.2264±.0278**

TABLE III

SUMMARIES OF DATA SETS

Datasets	Instances	Input variables	Original classes	Datasets	Instances	Input variables	Original classes
Breast-Cancer	683	9	2	Wine	178	13	3
Wpbc	194	33	2	Haberman	306	3	2
Wdbc	569	30	2	Vehicle	846	18	4
Pima	768	8	2	Banknote	1372	4	2
Aust	690	14	2	Cardiotocography	2126	21	3
Heart	270	13	2	Waveform	5000	21	3
Glass	214	9	6	Urban Land Cover	181	147	2
Seeds	210	7	3	Musk	6598	168	2
Ecoli	336	7	8				

for DISC and CORR methods. AdaBoost greatly improves the prediction accuracy of Stump (a simple one-level decision tree) in the clean data, but its ability in boosting is limited when the training set is corrupted, especially at high noise levels where it performs even worse than a single stump. This demonstrates that AdaBoost is indeed very sensitive to noise. It also suffers from overfitting at 0% noise level if the number of iterations becomes large. With preprocessing techniques (CORR or DISC), AdaBoost acts well at the beginning, but its accuracy decreases as a large number of base learners accumulate. Compared with the above methods, our proposed algorithm shows better performance in clean data and better robustness against noise. Moreover, it tactically avoids overfitting by ceasing the learning process at an early iteration (as early as 40).

Table II provides test errors for correcting and discarding methods under different thresholds of 0.2, 0.5, and 0.8, denoted by DISC20, DISC50, and DISC80 or CORR20, CORR50, and CORR80, respectively. We now see that CB-AdaBoost's performance is superior in 15 out of 16 cases. The only exception is the normal data under 30% noise level for $n = 500$, where CORR50 and DISC50 perform better. The advantage of CB-AdaBoost over the others is more significant for smaller sizes than for larger sizes. Neither the correcting nor discarding method at one threshold performs uniformly better than other thresholds. This makes their practice use difficult with reasonable confidence thresholds. It is worthwhile to mention that CB-AdaBoost uniformly outperforms AdaBoost even for the case without mislabels. This is because there is overlapping between the two classes and because the proposed loss function considers true risks that may help the classification achieve a better performance, with a test error close to the theoretically minimum error, namely, Bayes error.

C. Reweighting

This experiment illustrates reweighting differences between the original AdaBoost and the proposed one. Fig. 2 plots how the average weights of different groups of instances change as the number of iterations increases. First, we consider two groups: mislabeled instances and clean-labeled instances. Their mean weights are plotted in Fig. 2(a). As the learning process continues, the mean weight of noise data in AdaBoost (mis-AdaBoost, the top red curve) rapidly rises and stays at a level much higher than that of CB-AdaBoost (mis-CB, the middle red curve). If the iterations cannot be stopped in time, the weak classifiers trained by heavily weighted noise data become unreliable. By contrast, our proposed method does not place too much weight on noisy examples.

Fig. 2(b) illustrates groups divided by the certainty degree (higher than 0.7 or not). The plot clearly demonstrates the features of the weighting rule in CB-AdaBoost. Instances with high certainty are initialized more and their average weights

TABLE IV
AVERAGE AND STANDARD DEVIATION OF TESTING ERRORS OF EACH CLASSIFIER. THE BOLDFACE REPRESENTS THE SMALLEST TESTING ERROR

Dataset	Stump	β-Boosting	MadaBoost	LogitBoost	AdaBoost	CB-AdaBoost
10% Noise Level						
Breast-Cancer	.0792±.0173	.0481±.0117	.0652±.0150	**.0470±.0141**	.0850±.0180	.0495±.0113
Wpbc	.2945±.0581	.3048±.0540	.3024±.0420	.3069±.1374	.3093±.0470	**.2670±.0383**
Wdbc	.1008±.0148	.0786±.0190	.1098±.0209	.0719±.0170	.1043±.0228	**.0589±.0166**
Pima	.2845±.0327	.2696±.0197	.2845±.0210	**.2424±.0156**	.3036±.0296	.2555±.0213
Aust	.2663±.1537	**.1478±.0173**	.1845±.0188	.1724±.1570	.2016±.0278	.1694±.0621
Heart	.2998±.0477	.2842±.0836	.2649±.0403	.2183±.0535	.2719±.0363	**.2015±.0288**
Glass	.3318±.0513	.2673±.0466	.2717±.0522	**.2417±.0476**	.2785±.0498	.2667±.0602
Seeds	.3286±.0819	.1263±.0316	.1422±.0424	.1137±.0378	.1400±.0463	**.1070±.0258**
Ecoli	.1460±.0290	.0879±.0244	.1175±.0308	.0683±.0196	.1222±.0246	**.0589±.0212**
Wine	.1030±.0424	.0839±.0361	.1011±.0413	.0846±.0376	.1034±.0373	**.0472±.0256**
Harberman	.2767±.0354	.2950±.0330	.3187±.0327	.2693±.0310	.3429±.0418	**.2641±.0307**
Vehicle	.2586±.0211	.0621±.0145	.0742±.0163	**.0574±.0138**	.1012±.0162	.0652±.0158
Banknote	.1598±.0138	.0201±.0084	.0218±.0074	.0332±.0180	.0418±.0115	**.0116±.0093**
Cardiotocography	.1611±.0199	**.0780±.0079**	.0786±.0078	.0861±.0125	.1024±.0111	.0917±.0104
Waveform	.2322±.0114	.1147±.0064	.1138±.0047	.1289±.0105	.1350±.0083	**.1114±.0059**
Urban Land Cover	.0524±.0285	.1535±.2326	.0941±.0334	.0883±.0340	.0835±.0341	**.0333±.0151**
Musk	.3384±.0620	.2577±.0297	.2536±.0340	.2612±.0403	**.2524±.0337**	.2762±.0366
20% Noise Level						
Breast-Cancer	.0986±.0241	.0589±.0163	.0875±.0175	**.0543±.0169**	.1193±.0223	.0562±.0184
Wpbc	**.2883±.0556**	.3443±.0407	.3505±.0528	.3265±.1384	.3485±.0515	.2921±.0438
Wdbc	.0987±.0189	.1280±.0247	.1860±.0261	.0878±.0217	.1801±.0305	**.0743±.0216**
Pima	.3084±.0519	.3023±.0251	.3283±.0295	**.2618±.0170**	.3510±.0343	.2751±.0255
Aust	.2946±.1741	**.1648±.0215**	.2270±.0240	.1774±.0822	.2654±.0252	.1728±.0604
Heart	.3136±.0825	.2721±.0467	.3094±.0512	.2896±.1506	.3156±.0530	**.2222±.0356**
Glass	.3380±.0558	.3150±.0520	.3174±.0545	.3171±.1366	.3044±.0515	**.2897±.0528**
Seeds	.3527±.0859	.2048±.0360	.2283±.0519	.1740±.0571	.2394±.0451	**.1222±.0346**
Ecoli	.1581±.0423	.1492±.0405	.1935±.0454	.0909±.0332	.2006±.0401	**.0839±.0280**
Wine	.1221±.0640	.1648±.0447	.2034±.0428	.1281±.0555	.2052±.0492	**.0861±.0513**
Haberman	.2972±.0854	.3440±.0480	.3564±.0456	.2756±.0347	.3793±.0362	**.2706±.0242**
Vehicle	.2578±.0338	.1039±.0225	.1295±.0225	.0875±.0246	.1716±.0273	**.0828±.0172**
Banknote	.1597±.0156	**.0294±.0118**	.0460±.0173	.0587±.0184	.0841±.0216	.0376±.0145
Cardiotocography	.1653±.0197	**.0993±.0128**	.1080±.0104	.1027±.087	.1537±.0164	.1046±.0148
Waveform	.2325±.0142	.1273±.0086	.1347±.0079	.1457±.0099	.1719±.0115	**.1239±.0096**
Urban Land Cover	.1077±.0949	.2516± .2110	.2084±.0471	.1996±.0574	.2205±.0468	**.0802±.0589**
Musk	.3508±.0829	.3064±.0381	.3132± .0358	**.2957±.0461**	.3265± .0460	.3234±.0308
30% Noise Level						
Breast-Cancer	.0978±.0306	.0983±.0293	.1418±.0285	**.0736±.0250**	.1835±.0406	.0805±.0302
Wpbc	.3794±.1418	.4041±.0636	.3973±.0660	.3835±.0820	.4076±.0527	**.3543±.0834**
Wdbc	**.1187±.0360**	.2339±.0388	.2919±.0408	.1262±.0375	.2953±.0383	.1209±.0403
Pima	.3139±.0592	.3322±.0348	.3574±.0280	**.2868±.0350**	.3930±.0337	.2924±.0340
Aust	.2947±.1552	.2479±.1500	.2989±.0308	.2310±.1642	.3326±.0357	**.2014±.0704**
Heart	.3133±.0685	.3637±.1473	.3630±.0536	.2630±.0449	.3723±.0538	**.2477±.0422**
Glass	.3910±.0921	.3679±.0619	.3757±.0554	.3617±.0677	.3816±.0613	**.3542±.0788**
Seeds	.3622±.0839	.2876±.0683	.3321±.0629	.2559±.0862	.3371±.0572	**.2029±.0632**
Ecoli	.2022±.0689	.2351±.0528	.2796±.0646	.1512±.0629	.3022±.0527	**.1359±.0472**
Wine	.2307±.1092	.2625±.0565	.2884±.0467	.2015±.0714	.2805±.0633	**.1528±.0758**
Haberman	.3266±.0990	.3752±.0498	.3983±.0475	**.3089±.0591**	.4157±.0512	.3205±.0767
Vehicle	.2610±.0260	.1829±.0308	.2212±.0274	.1516±.0574	.2644±.0334	**.1357±.0420**
Banknote	.1678±.0189	.0825±.0238	.1152±.0238	.0921±.0390	.1657±.0273	**.0720±.0221**
Cardiotocography	.1837±.0317	.1388±.0192	.1614±.0219	.1272±.0235	.2326±.0254	.1218±.0237
Waveform	.2433±.0221	.1561±.0110	.1723±.0111	.1639±.0143	.2287±.0143	**.1410±.0106**
Urban Land Cover	.1758±.1046	.2868±.0513	.2956±.0705	.2674±.0565	.2938±.0682	**.1524±.0746**
Musk	.4214±.0710	.3633±.0316	.3800±.0471	.3639±.1274	.3821±.0295	**.3472±.0464**

decline after being fully trained, whereas the average weights of the others increase and remain at high values until the iteration stops. However, this adaptive ability is not present in AdaBoost.

D. Real Data Sets

In addition, we conducted experiments on 17 real data sets available from the UCI repository [20]. Since we focus on the two-class problem, the classes of several multiclass data sets are combined into two classes. If the class variable is nominal, class 1 is treated as the positive class, and the remaining classes are treated as the negative class. If the class variable is ordinal, we merge the classes with similar properties. For example, in the Cadiotocography data, the Suspect class and the Pathologic class are combined as the positive class and Normal as the negative class. For the Urban Land Cover data

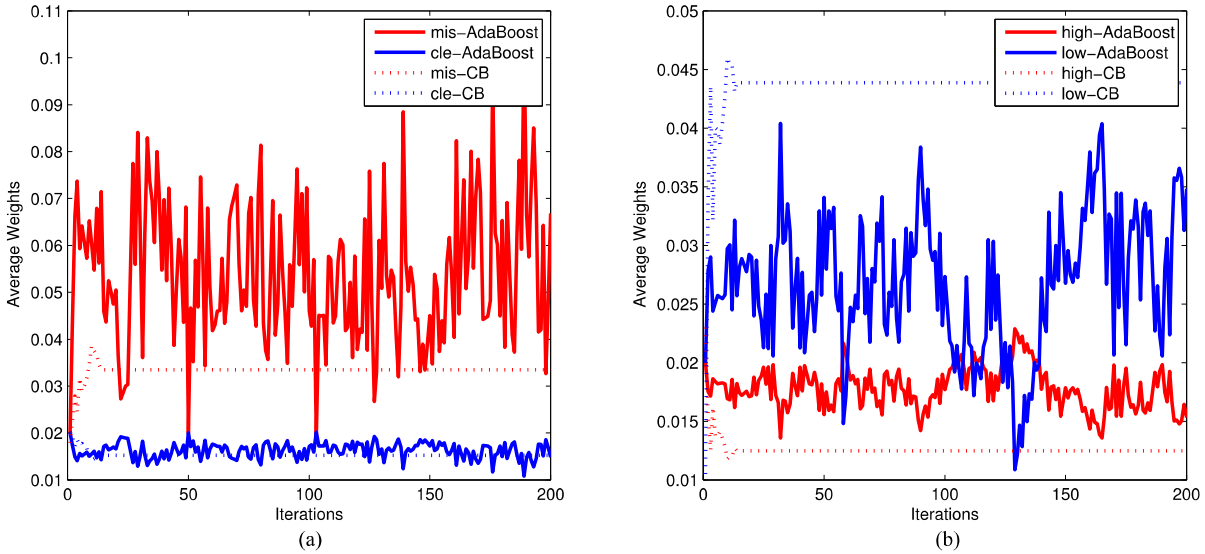

Fig. 2. Average weights of different types of instances during the learning process in the original AdaBoost and CB-AdaBoost. (a) Weights of mislabeled instances and clean-labeled instances. (b) Weights of groups with high and low label confidence.

TABLE V

RELATIVE FREQUENCIES THAT CB-ADABOOST WINS OVER 17 DATA SETS IN PAIRWISE COMPARISONS WITH OTHER ALGORITHMS. "***," "**," AND "*" ARE USED IF CB-ADABOOST IS STATISTICALLY BETTER WITH 1%, 5%, AND 10% SIGNIFICANCE LEVEL, RESPECTIVELY

Noise level	Stump	β-Boosting	MadaBoost	LogitBoost	AdaBoost
10%	17/17***	12/17*	15/17***	11/17	16/17***
20%	16/17***	13/17**	16/17***	13/17**	17/17***
30%	16/17***	17/17***	17/17***	14/17***	17/17***

set, we combine the training and test set, and any instances with missing values are removed. Table III summarizes the main characteristics of all data sets. For each data set, half of the instances are randomly selected as the training set and the remaining are used for testing; 10%, 20%, and 30% mislabels are introduced in the training data by randomly choosing training instances and reversing their labels. For comparison, we consider another boosting method known as LogitBoost, in addition to the two modified AdaBoost algorithms known as MadaBoost [10] and β-Boosting [49],[2] all of which are robust against noise data. The procedure is repeated 30 times, and we take the average of the 30 test errors for each classifier as a measurement of its performance.

According to Table IV, CB-AdaBoost performs better than the original AdaBoost for all cases except for one that is the case of Musk under 10% noise level. It also greatly improves the accuracy of the stump (i.e., the base classifier). β-Boosting, MadaBoost, and LogitBoost methods show robustness to mislabeled data. They outperform AdaBoost for most cases, especially which LogitBoost achieves a lower test error than the other two. However, like AdaBoost, they suffer from overfitting because they cannot stop iterations due to their weight distributions. This problem is overcome by CB-AdaBoost, and as a result, the win–lose numbers of the proposed algorithm when compared with the robust three algorithms are 42–9, 48–3, and 38–13 respectively. We conducted the sign test based on counts of wins, losses, and ties [7] in order to quantify the significance of the proposed method. Table V lists the frequency and significance level that CB-AdaBoost wins each of other algorithms on 17 data sets at each noise level. This demonstrates the effectiveness and advantages of CB-AdaBoost in handling mislabeled data.

[2]The original paper did not name the method. We name it β-Boosting after the β parameter added to the algorithm, as suggested by a reviewer.

VI. CONCLUSION

In this paper, we have provided a label-confidence-based boosting method that is sufficiently immune to the label noise and overfitting problems. With the assignment of confidence, our proposed algorithm distinguishes between clean and contaminated instances. In addition, the values of confidence on instances represent different levels of judgments on their label reliability. Under the guidance of confident instances, CB-AdaBoost is able to minimize the loss function over the training set under the conditional risk function. Moreover, in CB-AdaBoost, explicit solutions for weak learners and their coefficients on each stage can be easily obtained and applied practically. In comparisons with some common noise handling techniques and other robust algorithms, CB-AdaBoost does a better job of tackling problems of class overlapping and mislabeling.

The proposed method has some limitations.

1) The computational complexity of the proposed CB-AdaBoost is $O(n^2d)$, where n is the sample size and d is dimension. This is because we need to compute or estimate label confidence of each

instance, and the KNN method for label confidence evaluation has the computation complexity $O(n^2d)$. The remaining process of the CB-AdaBoost is $O(n^{2-a}d)$ with $a \in (0,1)$. Collectively, this yields an overall computational complexity $O(n^2d)$, which may be prohibitive for large-scale applications.

2) As currently formulated, the proposed method cannot directly handle categorical or symbolic features. A similarity metric on those type of features needs to be introduced to define neighbors for label confidence assignment.

Continuation of this paper could take several directions.

1) A general framework of optimization strategy based on the conditional risk deserves a deeper understanding and further development.
2) In this paper, KNN is used to estimate the confidence of each instance. Theoretically, the number of neighbors shall go to infinity with a speed slower than the sample size to ensure strong consistency of KNN estimator. In practice, however, a small number of neighbors seem to be sufficient. Perhaps a proof of consistence exists without the conditions of k_n. It will be interesting to study the impact of parameter k and discuss a proper selection on the number of neighborhoods in practice. For example, the cross-validation method for choosing k deserves further investigation. In fact, the problem of how to design a good criterion for confidence assignment is still open. Other methods are needed to produce high-quality confidences, especially when categorical features are involved.
3) CB-AdaBoost outperforms the AdaBoost for class overlapping problems, and thus it is promising to extend CB-AdaBoost for multiple class classification problems and other applications such as image or object recognition.

APPENDIX

Proof of Proposition 4: Let $\{\bar{f}_n\}_{n=1}^{\infty}$ be a sequence of reference functions such that $R(\bar{f}_n) \to R^*$. We shall prove that there exist nonnegative sequences $t_n \to \infty$, $\xi_n \to \infty$, $k_n \to \infty$, and $k_n/n \to 0$ such that the following conditions are satisfied.

Uniform Convergence of t_n-combinations

$$\sup_{f \in \pi_{\xi_n} \circ \mathcal{F}^{t_n}} |R(f) - \bar{R}_n(f)| \xrightarrow{a.s.} 0. \tag{VII.1}$$

Empirical convergence for the sequence $\{\bar{f}_n\}$

$$\bar{R}_n(\bar{f}_n) - R(\bar{f}_n) \xrightarrow{a.s.} 0. \tag{VII.2}$$

Convergence of the KNN estimates

$$R_{n,k_n}(\bar{f}_n) - \bar{R}_n(\bar{f}_n) \xrightarrow{a.s.} 0. \tag{VII.3}$$

Algorithm convergence of t_n-combinations

$$R_{n,k_n}(f_{t_n}) - R_{n,k_n}(\bar{f}_n) \xrightarrow{a.s} 0. \tag{VII.4}$$

Since $\bar{R}_n(f)$ is an empirical exponential risk, a proof of (VII.1) follows exactly the same lines of [3, Lemma 4] with the Lipschitz constant $L_\xi = (e^{\xi} - e^{-\xi})/(2\xi)$ and $M_\xi = e^{\xi}$. Then for any $\delta > 0$, with probability at least $1 - \delta$

$$\sup_{f \in \pi_\xi \circ \mathcal{F}^t} |R(f) - \bar{R}_n(f)| \le c\xi L_\xi \sqrt{\frac{(V+1)(t+1)\log_2[2(t+1)/\ln 2]}{n}} + M_\xi \sqrt{\frac{1/\delta}{2n}} \tag{VII.5}$$

where $V = d_{VC}(\mathcal{H})$ and $c = 24\int_0^1 (\ln(8e/\epsilon^2))^{1/2} d\epsilon$. We can take $t = n^{1-a}$ and $\xi = \kappa \ln n$ with $\kappa > 0$, $a \in (0,1)$, and $2\kappa - a < 0$ so that the right side of inequality (VII.5) converges to 0, and in the meantime, $\sum_{n=1}^{\infty} \delta_n < \infty$. Hence, an application of the Borel–Cantelli lemma ensures the almost surely convergence of (VII.1).

Applying [3, Th. 8], we have the result of (VII.4), in which the reference sequence $\bar{f}_n \in \mathcal{F}_{\lambda_n}$ with $\lambda_n = \kappa_1 \ln n$ where $\kappa_1 \in (0, 1/2)$.

Equation (VII.2) can be proved by Hoeffding's inequality if the range of $\bar{f}_n$ is restricted to the interval $[-\lambda_n, \lambda_n]$. That is

$$P(\bar{R}_n(\bar{f}_n) - R(\bar{f}_n) \ge \epsilon_n) \le \exp(-2n\epsilon_n^2/M_{\lambda_n}^2) := \delta_n$$

where $M_{\lambda_n} = e^{\lambda_n} - e^{-\lambda_n}$. Let $\lambda_n = \kappa_1 \ln n$ with $\kappa_1 \in (0, 1/2)$. Letting $\epsilon_n \to 0$, we still have $\sum_{n=1}^{\infty} \delta_n < \infty$, and hence convergence in probability 1 of (VII.2) holds.

By the result of [8, Th. 1], for each KNN estimate $\hat{\gamma}_i$ with $k_n \to \infty$ and $k_n/n \to 0$, we have

$$P(2|\hat{\gamma}_i - \gamma_i| > \epsilon_n) \le \exp\big[-n\epsilon_n^2/\big(8N_p^2\big)\big]$$

where the constant N_p is the minimal number of cones centered at the origin of angle $\pi/6$ that cover $\mathbb{R}^p$. Then with the restriction of $\bar{f}_n$ in $[-\lambda_n, \lambda_n]$, we have

$$P(|R_{n,k_n}(\bar{f}_n) - \bar{R}_n(\bar{f}_n)| > \epsilon_n) < \exp\big[-n\epsilon_n^2/\big(2M_{\lambda_n}^2 N_d^2\big) + \ln n\big] := \delta_n.$$

Again, a choice of $\lambda_n = \kappa_1 \ln n$ with $\kappa_1 \in (0, 1/2)$ guarantees $\sum \delta_n < \infty$ when $\epsilon_n = o(1)$, and hence (VII.3) holds.

Now we are ready to prove Proposition 4. For almost every outcome ω on the probability space, we can define sequences $\epsilon_{n,i}(\omega) \to 0$ for $i = 1, \ldots, 5$ so that for almost all ω, the following inequalities are true:

$$\begin{aligned}
R(\pi_{\xi_n}(f_{t_n})) &\le \bar{R}_n(\pi_{\xi_n}(f_{t_n})) + \epsilon_{n,1}(\omega) \quad \text{by (VII.1)} \\
&\le R_{n,k_n}(\pi_{\xi_n}(f_{t_n})) + \epsilon_{n,2}^*(\omega) && \text{(VII.6)} \\
&\le R_{n,k_n}(f_{t_n}) + e^{-\xi_n} + \epsilon_{n,2}^*(\omega) && \text{(VII.7)} \\
&\le R_{n,k_n}(\bar{f}_n) + e^{-\xi_n} + \epsilon_{n,3}^*(\omega) \quad \text{by (VII.4)} \\
&\le \bar{R}_n(\bar{f}_n) + e^{-\xi_n} + \epsilon_{n,4}^*(\omega) \quad \text{by (VII.3)} \\
&\le R(\bar{f}_n) + e^{-\xi_n} + \epsilon_{n,5}^*(\omega) \quad \text{by (VII.2)} && \text{(VII.8)}
\end{aligned}$$

where $\epsilon_{n,k}^*(\omega) = \sum_{j=1}^{k} \epsilon_{n,j}(\omega)$. Inequality (VII.6) follows similarly as (VII.3) with $\xi_n = \kappa_1 \ln n$, where $\kappa_1 \in (0, 1/2)$. Inequality (VII.7) follows from the facts that $e^{\pi_{\xi_n}(x)} < e^{x} + e^{-\xi_n}$ and $e^{-\pi_{\xi_n}(x)} < e^{-x} + e^{-\xi_n}$. Then with $t_n = n^{1-a}$, $\xi_n = \kappa \ln n$ $(a > 0, \kappa > 0, 2\kappa < a)$ and (VII.8), by choice of the sequence $\{\bar{f}_n\} \in \mathcal{F}_{\lambda_n}$ with $\lambda_n = \kappa_1 \log n, \kappa_1 \in (0, 1/2)$, we have $R(\bar{f}_n) \to R^*$ and $R(\pi_{\xi_n}(f_{t_n})) \to R^*$ a.s.

By [2, Th. 3], $L(g(\pi_{\xi_n}(f_{t_n}))) \xrightarrow{a.s} L^*$. Since for $\xi_n > 0$, we have $g(\pi_{\xi_n}(f_{t_n})) = g(f_{t_n})$, it follows that:

$$L(g(f_{t_n})) \xrightarrow{a.s} L^*.$$

Hence, the proposed CB AdaBoosting procedure is consistent if stopped after t_n steps.

ACKNOWLEDGMENT

The authors would like to thank Y. Chen for discussing the conditional risk.

REFERENCES

[1] H. Allende-Cid *et al.*, "Robust alternating adaboost," in *Progress in Pattern Recognition, Image Analysis and Applications*. Berlin, Germany: Springer, 2007, pp. 427–436.

[2] P. L. Bartlett, M. Jordan, and J. D. McAuliffe, "Convexity, classification, and risk bounds," *J. Amer. Statist. Assoc.*, vol. 101, pp. 138–156, Apr. 2006.

[3] P. L. Bartlett and M. Traskin, "AdaBoost is consistent," *J. Mach. Learn. Res.*, vol. 8, no. 1, pp. 2347–2368, 2007.

[4] C. E. Brodley and M. A. Friedl, "Identifying and eliminating mislabeled training instances," in *Proc. AAAI/IAAI*, vol. 1. 1996, pp. 799–805.

[5] C. E. Brodley and M. A. Friedl, "Identifying mislabeled training data," *J. Artif. Intell. Res.*, vol. 11, no. 1, pp. 131–167, 1999.

[6] J. Cao, S. Kwong, and R. Wang, "A noise-detection based AdaBoost algorithm for mislabeled data," *Pattern Recognit.*, vol. 45, no. 12, pp. 4451–4465, 2012.

[7] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[8] L. Devroye, L. Gyorfi, A. Krzyzak, and G. Lugosi, "On the strong universal consistency of nearest neighbor regression function estimates," *Ann. Statist.*, vol. 22, no. 3, pp. 1371–1385, 1994.

[9] T. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Mach. Learn.*, vol. 40, no. 2, pp. 139–157, 2000.

[10] C. Domingo and O. Watanabe, "MadaBoost: A modification of AdaBoost," in *Proc. COLT*, 2000, pp. 180–189.

[11] B. Frénay and A. Kabán, "A comprehensive introduction to label noise," in *Proc. Eur. Symp. Artif. Neural Netw. Comput. Intell. Mach. Learn. (ESANN)*, Bruges, Belgium, 2014, pp. 1–10.

[12] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. 13th Int. Conf. Mach. Learn. (ICML)*, 1996, pp. 148–156.

[13] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Ann. Statist.*, vol. 28, no. 2, pp. 337–374, 2000.

[14] Y. Gao and F. Gao, "Edited AdaBoost by weighted kNN," *Neurocomputing*, vol. 73, pp. 3079–3088, Apr. 2010.

[15] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: Data mining," in *Inference and Prediction*. New York, NY, USA: Springer, 2001.

[16] K. Hayashi, "A simple extension of boosting for asymmetric mislabeled data," *Statist. Probabil. Lett.*, vol. 82, no. 2, pp. 348–356, 2012.

[17] Y. Jiang and Z. Zhou, "Editing training data for kNN classifiers with neural network ensemble," in *Proc. Adv. Neural Netw. (ISNN)*, 2004, pp. 356–361.

[18] T. Kanamori, T. Takenouchi, and S. Eguchi, "The most robust loss function for boosting," in *Proc. Neural Inf. Process. (ICONIP)*, 2004, pp. 496–501.

[19] T. Kanamori, T. Takenouchi, and S. Eguchi, "Robust loss functions for boosting," *Neural Comput.*, vol. 19, no. 8, pp. 2183–2244, 2007.

[20] M. Lichman, *UCI Machine Learning Repository*. Irvine, CA, USA: Univ. California, School of Information and Computer Science, 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[21] C. Lin and S. Wang, "Fuzzy support vector machine," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 464–471, Feb. 2002.

[22] H. Liu and S. Zhang, "Noisy data elimination using mutual k-nearest neighbor for classification mining," *J. Syst. Softw.*, vol. 85, no. 5, pp. 1067–1074, 2012.

[23] G. Lugosi and N. Vayatis, "On the Bayes-risk consistency of regularized boosting methods," *Ann. Statist.*, vol. 32, no. 1, pp. 30–55, 2004.

[24] H. Masnadi-Shirazi and N. Vasconcelos, "Cost-senstive boosting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 294–309, Feb. 2011.

[25] P. Melville, N. Shah, L. Mihalkova, and R. Mooney, "Experiments on ensembles with missing and noisy data," in *Proc. 5th Int. Workshop Multi Classifier Syst.*, 2004, pp. 293–302.

[26] T. Onoda, "Overfitting of boosting and regularized boosting algorithms," *Electron. Commun. Jpn.*, vol. 90, no. 9, pp. 69–78, 2007.

[27] U. Rebbapragada and C. Brodley, *Class Noise Mitigation Through Instance Weighting* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2007, p. 788.

[28] J. A. Sáez, J. Luengo, and F. Herrera, "Predicting noise filtering efficacy with data complexity measures for nearest neighbor classification," *Pattern Recognit.*, vol. 46, no. 1, pp. 355–364, 2013.

[29] J. S. Sánchez, R. Barandela, A. I. Marqués, R. Alejo, and J. Badenas, "Analysis of new techniques to obtain quality training sets," *Pattern Recognit. Lett.*, vol. 24, no. 7, pp. 1015–1022, 2003.

[30] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Mach. Learn.*, vol. 37, no. 3, pp. 297–336, 1999.

[31] R. E. Schapire and Y. Freund, *Boosting: Foundations and Algorithms*. Cambridge, MA, USA: MIT Press, 2012.

[32] R. A. Servedio, "Smooth boosting and learning with malicious noise," *J. Mach. Learn. Res.*, vol. 4, pp. 633–648, Sep. 2003.

[33] I. Steinwart and A. Christmann, *Support Vector Machines*. New York, NY, USA: Springer, 2008.

[34] C. J. Stone, "Consistent nonparametric regression," *Ann. Statist.*, vol. 5, no. 4, pp. 595–620, 1977.

[35] Y. Sun, J. Li, and W. Hager, "Two new regularized AdaBoost algorithms," in *Proc. ICMLA*, 2004, p. 11.

[36] Y. Sun, S. Todorovic, and J. Li, "Reducing the overfitting of AdaBoost by controlling its data distribution skewness," *Int. J. Pattern Recognit.*, vol. 20, no. 7, pp. 1093–1116, 2006.

[37] T. Takenouchi and S. Eguchi, "Robustifying AdaBoost by adding the naive error rate," *Neural Comput.*, vol. 16, no. 4, pp. 767–787, 2004.

[38] Q. Tao, G. Wu, F. Wang, and J. Wang, "Posterior probability support vector machines for unbalanced data," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1561–1573, Jun. 2015.

[39] J. Thongkam, G. Xu, Y. Zhang, and F. Huang, "Toward breast cancer survivability prediction models through improving training space," *Expert Syst. Appl.*, vol. 36, pp. 12200–12209, Apr. 2009.

[40] L. Utkin and Y. Zhuk, "Robust boosting classification models with local sets of probability distributions," *Knowl.-Based Syst.*, vol. 61, pp. 59–75, May 2014.

[41] S. Verbaeten and A. Van Assche, "Ensemble methods for noise elimination in classification problems," in *Multiple Classifier Systems*. Berlin, Germany: Springer, 2003, pp. 317–325.

[42] A. Vezhnevets and V. Vezhnevets, "Modest AdaBoost-teaching adaBoost to generalize better," *Graphicon*, vol. 12, no. 5, pp. 987–997, 2005.

[43] A. Vezhnevets and O. Barinova, *Avoiding Boosting Overfitting by Removing Confusing Samples* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2007, pp. 430–441.

[44] P. Wang, C. H. Shen, N. Barnes, and H. Zheng, "Fast and robust object detection using asymmetric totally corrective boosting," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 1, pp. 33–46, Jan. 2012.

[45] W. Wang, Y. Wang, F. Chen, and A. Sowmya, "A weakly supervised approach for object detection based on soft-label boosting," in *Proc. IEEE Workshop Appl. Comput. Vis.*, Sep. 2013, pp. 331–338.

[46] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.

[47] T. Zhang and B. Yu, "Boosting with early stopping: Convergence and consistency," *Ann. Statist.*, vol. 33, no. 4, pp. 1538–1579, 2005.

[48] C. X. Zhang and J. S. Zhang, "A local boosting algorithm for solving classification problems," *Comput. Statist. Data Anal.*, vol. 52, no. 4, pp. 1928–1941, 2008.

[49] C. X. Zhang, J. S. Zhang, and G. Y. Zhang, "An efficient modified boosting method for solving classification problems," *J. Comput. Appl. Math.*, vol. 214, no. 2, pp. 381–392, 2008.

[50] D. Zhou, B. Quost, and V. Fremont, "Soft label based semi-supervised boosting for classification and object recognition," in *Proc. 13th Int. Conf. Control Autom., Robot. Vis. (ICARCV)*, Dec. 2014, pp. 1062–1067.

Zhi Xiao is currently a Professor and the Chair with the Information Management Department, Chongqing University, Chongqing, China. He has been the Principal Investigator of 50 funding projects. He has authored five textbooks and more than 100 scientific papers including journals such as *Knowledge-Based Systems*, *Expert Systems with Application*, *Applied Mathematical Modeling*, *Journal of Computational and Applied Mathematics*, and *Computers & Mathematics with Applications*. His current research interests include operational optimization, Statistics, forecasting, information intelligence analysis, data mining, soft sets, and interdisciplinary big data analysis.

Prof. Xiao serves as the Vice Executive Director with the China Information Economics Association, an Executive Officer with the National Statistical Society of China, and the Vice President of the Chongqing Statistical Society.

Zhe Luo received the master's degree in probability and mathematical statistics from Chongqing University, Chongqing, China.

He is currently an Assistant Manager with the Bank of China, Nanning, China. His current research interests include statistical decision, pattern recognition, cluster analysis, and Monte Carlo simulations.

Bo Zhong is currently a Professor with the Statistics and Actuary Department, Chongqing University, Chongqing, China, where she is the Director of the Graduate Mathematics Courses Program. She leads 30 funding projects, including ten from national agents. She has authored eight textbooks and 70 scientific papers in journals, including *Expert Systems with Application and Knowledge-Based Systems* and so on. Her current research interests include soft set, soft computation, rough set, statistical learning, and reliability analysis in power systems.

Xin Dang (M'17) received the Ph.D. degree in statistics from the University of Texas, Dallas, TX, USA, in 2005.

She is currently an Associate Professor with the Department of Mathematics, University of Mississippi, Oxford, MS, USA. Her current research interests include robust and nonparametric statistics, statistical and numerical computing, multivariate data analysis, data depth and application, bioinformatics, machine learning, and robust procedure computation.

Dr. Dang is a member of the Institute of Mathematical Statistics, the American Statistical Association, the International Chinese Statistical Association, and the International Neural Network Society.