



# A selective boosting technique for pattern classification

De Z. Li<sup>a</sup>, Wilson Wang<sup>b,\*</sup>, Fathy Ismail<sup>c</sup>

<sup>a</sup> Department of Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1

<sup>b</sup> Department of Mechanical Engineering, Lakehead University, Thunder Bay, ON, Canada P7B 5E1

<sup>c</sup> Department of Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1

## ARTICLE INFO

### Article history:

Received 3 August 2014

Received in revised form

19 December 2014

Accepted 21 December 2014

Available online 14 January 2015

### Keywords:

AdaBoost algorithm

Ensemble learning

Pattern classification

Overfitting problem

## ABSTRACT

The classical AdaBoost algorithm is an ensemble of weak learners, and can be used to construct a strong classifier. A weak learner is incorporated into the ensemble at each step, and the classification of the derived ensemble is improved by properly adjusting the weight of each weak learner. The classical AdaBoost algorithm has some limitations; for example, it is sensitive to noisy data, which may impede the generalization capability of the derived classifier and lead to overfitting problem. A selective boosting, sBoost, technique is proposed in this paper to tackle these problems. The proposed sBoost classifier focuses on the generalized classification performance rather than those hard-to-learn samples, and the penalties of hard-to-learn samples are mitigated to the degree associated with their noise level. An error correction method is suggested to detect potential clean samples and prevent them from misclassification to further alleviate the overfitting problem. The effectiveness of the developed sBoost technique is tested by a series of simulation tests. Test results show that the developed sBoost technique can improve the classification accuracy and prevent the overfitting problem effectively.

© 2015 Elsevier B.V. All rights reserved.

## 1. Introduction

AdaBoost, short for Adaptive Boosting, is a machine learning algorithm, which can be used to improve data training performance [1]. AdaBoost is adaptive because subsequent constructed classifiers are modified to help discern samples misclassified by the previous classifiers. It can emphasize misclassified samples by increasing their weights in training data distribution, and then classify them by using appropriate weak learners. By integrating a weak learner at each step, the ensemble of weak learners can better learn how to classify the samples and improve classification accuracy. A weak learner can be any classifier that has moderate classification accuracy. The classical AdaBoost has merits such as solid theoretical basis and ease of implementation, and has been used in many applications such as medical diagnoses [2,3], image retrieval/classification [4,5], and network intrusion detection [6]. However, the classical AdaBoost algorithm has some limitations such as being sensitive to noisy samples, overfitting and slow convergence [7,8].

Many research efforts have been made to improve the AdaBoost [9–11]. To speed up convergence, a Real AdaBoost algorithm was

proposed in [1] to calculate the probability and confidence limits to estimate the degree to which a sample belongs to a specific class. A LogitBoost method [12] employed an additive logistic regression model as cost function to improve the convergence. However, these methods also suffer from the overfitting problem. Overfitting occurs as the number of weak learners increases when noisy samples are present [7]. The classifier does not have an efficient mechanism to learn noisy samples, which causes some potential clean samples to be misclassified. Ratsch et al. [13] proposed a boosting technique with soft margins to deal with noisy samples by computing the aggregated weights of each sample and reducing the penalties for hard-to-learn samples; some hard-to-learn samples, however, may not be noise affected but clean ones, and their penalty should not be reduced. Cao et al. [14] applied the noise detection method to identify noisy samples in the training data and assign them a different class label from their original ones in the binary classification problem. This technique is highly dependent on the noise detection mechanism; less effective noise detection method may lead to poor classifier performance. Gao et al. [15] applied a weighted  $K$ -nearest-neighbor algorithm in a transformed space to identify and discard suspicious noisy samples; but some of these discarded samples may be useful to improve the classification accuracy.

The classical AdaBoost algorithms [1,9–12] focus on the classification of those large weight samples, which may lead to severe misclassification of small weight samples and result in overfitting problems. The overfitting problem could be addressed using some

\* Corresponding author. Tel.: +807 766 7174.

E-mail addresses: [d45li@uwaterloo.ca](mailto:d45li@uwaterloo.ca) (D.Z. Li),

[Wilson.Wang@Lakeheadu.ca](mailto:Wilson.Wang@Lakeheadu.ca) (W. Wang), [fmis@mail@uwaterloo.ca](mailto:fmis@mail@uwaterloo.ca) (F. Ismail).

methods such as [13–15] to some extent; however they cannot effectively process large weight samples based on data characteristics, which may result in low classification accuracy. To effectively reduce the overfitting problem, a new selective Boosting technique, sBoost for short, is proposed in this work to tackle the overfitting problem effectively and improve classification accuracy, under low noise levels (e.g., from no noise to 10% noise). In the proposed sBoost technique, the penalties of samples with large weights are adjusted adaptively to improve the classification performance. For those large weight samples caused by noise, their penalties are mitigated according to the noise degree to reduce the overfitting problem. Some clean samples may carry large weight because of the limitation of weak learners or inconsistency in combining weak learners. Their weights are maintained to prevent misclassification. This proposed sBoost technique is new in the following aspects: (a) a sample weight regulator is proposed to adaptively regulate samples' penalties based on their accumulated weights and noise degrees to reduce overfitting; and (b) an error correction method is proposed to detect potential clean samples in the test data, predict their class label, and prevent them from being misclassified by a classifier suffering overfitting.

The remainder of this paper is organized as follows. The proposed sBoost technique is discussed in Section 2. In Section 3, the effectiveness of the proposed sBoost technique is examined by some benchmark classification problems. Some concluding remarks are drawn in Section 4.

## 2. The proposed sBoost technique

Before the discussion of the proposed sBoost technique, some description is given first for the classical AdaBoost algorithm.

### 2.1. Classical AdaBoost algorithm

Consider a binary classification problem with training data samples  $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ , where the class label is  $y_i \in \{-1, +1\}$  and  $N$  is the number of samples in the training data set. If  $T$  is the maximum number of weak learners, the AdaBoost algorithm takes the following processes:

- Initialize the distribution of training data set  $L_1(i) = 1/N$ ,  $i = 1, 2, \dots, N$ .
- Train the weak learner  $h_t \in \{-1, +1\}$  using training data set with distribution  $L_t$ .
- Compute the sum of weights of misclassified samples  $\varphi_t = \sum_{i=1}^N L_t(i) [h_t(x_i) \neq y_i]$ . The logic operator is defined as  $[[I]] = \begin{cases} 1 & I \neq 0 \\ 0 & I = 0 \end{cases}$ .
- Calculate the weight of the weak learner  $h_t$ ,  $\alpha_t = (1/2) \ln(1 - \varphi_t / \varphi_t)$ .
- Update the weights of the training samples,  $L_{t+1}(i) = (L_t(i) \exp(-\alpha_t h_t(x_i) y_i) / J_t)$ , where  $J_t = \sum_{i=1}^N L_t(i) \exp(-y_i \alpha_t h_t(x_i))$ .
- Repeat step (b) to step (e) with  $t = 1: T$ .
- Normalize the weight of weak learner  $h_t$ ,  $\alpha'_t = (\alpha_t / \sum_{t=1}^T \alpha_t)$ .

If  $f(x_i) = \sum_{t=1}^T \alpha'_t h_t(x_i)$ , and  $H(x_i) = \text{sign}[f(x_i)]$  is the predicted class label, the margin of a sample  $(x_i, y_i)$  can be given as

$$\hat{\rho}_t(x_i) = \frac{y_i \sum_{t=1}^T \alpha_t h_t(x_i)}{\sum_{t=1}^T \alpha_t} \quad (1)$$

**Theorem 1.** (Schapire et al. [16]): Given the weighted training error  $\varphi_t = \sum_{i=1}^N L_t(i) [h_t(x_i) \neq y_i]$ , for any  $\delta$ , the probability that  $\hat{\rho}_t(x_i) \leq \delta$

can be given as,

$$\mathbf{P}[\hat{\rho}_t(x_i) \leq \delta] \leq 2^T \prod_{t=1}^T \sqrt{\varphi_t^{1-\delta} (1 - \varphi_t)^{1+\delta}} \quad (2)$$

When  $\delta = 0$ ,  $\mathbf{P}[\hat{\rho}_t(x_i) \leq 0]$  represents the probability that a sample is misclassified, which should be minimized.

### 2.2. Sample weight regulator in the proposed sBoost technique

The accumulated weight of a sample is an effective indicator, which determines the degree this sample contributes to potential overfitting. Thus it is used to properly adjust the penalty of this sample in order to reduce overfitting.

Let  $\beta_t$  be the weight of the weak learner  $h_t$  in sBoost, the margin can be defined in the same way as in Eq. (1)

$$\rho_t(x_i) = \frac{y_i \sum_{t=1}^T \beta_t h_t(x_i)}{\sum_{t=1}^T \beta_t} \quad (3)$$

The accumulated weight of sample  $i$  is computed as

$$\kappa_t(i) = \frac{\exp(\rho_t(x_i))}{\sum_{i=1}^N \exp(\rho_t(x_i))} \quad (4)$$

$$\xi_t(i) = \sum_{j=1}^t (\beta_j \kappa_j(i)) \quad (5)$$

$$\psi_t(i) = \frac{\xi_t(i)}{\max \{\xi_t(i)\}} \beta_t \quad (6)$$

Since some noisy samples may cause overfitting, they need to be processed properly in the training process. In the proposed sample weight regulator, the  $k$ -nearest-neighbor method is adopted to identify noisy samples and evaluate their noise degrees. Given  $K$  nearest samples of a target sample in the training data set, if  $p$  samples of these  $K$  samples have the same class label as this target sample ( $0 \leq p \leq K$ ), the noise degree of this sample will be estimated by

$$\theta_i = 1 - \frac{p}{K} \quad (7)$$

where the noise degree is independent of time step  $t$ .

The proposed sample weight regulator is given by

$$\varpi_t(i) = \psi_t(i) \theta_i = \frac{\xi_t(i)}{\max \{\xi_t(i)\}} \beta_t \theta_i \quad (8)$$

It can be proven that  $0 \leq \varpi_t(i) \leq \beta_t$ . Accordingly, the update of the  $i^{\text{th}}$  sample weight at step  $t$  can be formulated as:

$$L_{t+1}(i) = \frac{L_t(i) \exp(-\beta_t h_t(x_i) y_i - \varpi_t(i))}{J'_t} = \frac{1}{N} \frac{\exp\left\{-\sum_{t=1}^T [y_i \beta_t h_t(x_i) + \varpi_t(i)]\right\}}{\prod_{t=1}^T J'_t} \quad (9)$$

where  $J'_t$  is the cost function of sBoost.

By using the sample weight regulator  $\varpi_t(i)$ , the noisy samples will be processed according to their noise degree. For each sample, the maximum penalty reduction is determined by its accumulated weight, which will be lessened by the sample's noise degree. Consequently, the less noisy samples will not have over-reduced penalties to decrease their chance of being correctly classified, and the penalties of noisy samples can be adjusted to reduce overfitting.

Download English Version:

<https://daneshyari.com/en/article/406250>

Download Persian Version:

<https://daneshyari.com/article/406250>

[Daneshyari.com](https://daneshyari.com)