Neurocomputing 101 (2013) 229–242

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Weighted extreme learning machine for imbalance learning

Weiwei Zong [a], Guang-Bin Huang [a,*], Yiqiang Chen [b]

[a] School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore
[b] Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100864, China

ARTICLE INFO

Article history:
Received 3 April 2012
Received in revised form
18 July 2012
Accepted 4 August 2012
Communicated by K. Li
Available online 27 September 2012

Keywords:
Extreme learning machine
Imbalanced learning
Single hidden layer feedforward networks
Weighted extreme learning machine

ABSTRACT

Extreme learning machine (ELM) is a competitive machine learning technique, which is simple in theory and fast in implementation. The network types are "generalized" single hidden layer feedforward networks, which are quite diversified in the form of variety in feature mapping functions or kernels. To deal with data with imbalanced class distribution, a weighted ELM is proposed which is able to generalize to balanced data. The proposed method maintains the advantages from original ELM: (1) it is simple in theory and convenient in implementation; (2) a wide type of feature mapping functions or kernels are available for the proposed framework; (3) the proposed method can be applied directly into multiclass classification tasks. In addition, after integrating with the weighting scheme, (1) the weighted ELM is able to deal with data with imbalanced class distribution while maintain the good performance on well balanced data as unweighted ELM; (2) by assigning different weights for each example according to users' needs, the weighted ELM can be generalized to cost sensitive learning.

1. Introduction

Extreme learning machine (ELM) [1–6], as an effective and efficient machine learning technique, has attracted tremendous attention from various fields for recent years. ELM is basically a least-square based learning algorithm for "generalized" single hidden layer feedforward networks (SLFNs), which can be applied as the estimator in regression problem or the classifier for classification tasks.

From algorithmic point of view, ELM is simple in theory in the sense that the hidden nodes are randomly generated and the output weight is analytically determined, so that it is ranked high for easy and fast implementation among machine learning techniques. From optimization point of view, connections between ELM and the popular support vector machines (SVMs) exist mainly in the aspects of problem formulation, network architecture except that solutions of SVMs are suboptimal compared to those of ELM [5,6]. Therefore, ELM essentially provides a unified solution for the "generalized" SLFNs, which include but not limit to support vector network, traditional neural network and regularized network. However, none of the work of ELM has mentioned the problem of imbalanced data distribution [6].

Raw data with imbalanced class distribution can be found almost everywhere, from biomedical application to fraud detection or network intrusion, etc. When classifying data with complex class distribution, the regular learning algorithm has a natural tendency to favor the majority class by assuming balanced class distribution or equal misclassification cost. Typically, to deal with imbalanced data one can attempt to establish balanced data distribution through various sampling methods or algorithmic approaches [7,8]. To re-balance the data distribution, there are mainly two sampling techniques: undersampling approach which removes a fraction of the majority samples and the oversampling approach which duplicates the minority samples. The challenge the oversampling method is facing is to distinguish the informative samples and the redundant samples. And the assumption that the neighborhood of a minority sample share the same label is not always satisfied with different types of data. As the other type of re-balance technique, the algorithmic approach is considered consistent with the sampling approach and is of particular interest in this paper.

A popular algorithmic approach is to assign a different misclassification cost for each particular example [7]. But cases when the misclassification cost matrix is difficult or unnecessary to generate may arise in the real world applications. The most straightforward solution is to automatically generate the misclassification cost matrix in accordance with the class distribution, which usually is in the form of a weight scheme inversely proportional to the number of samples in the class.

Weighted regularized ELM was proposed in the work of Toh [9] and Deng et al. [10]. However, neither of them was targeting the imbalanced data problem. Further, "generalized" feature mapping and kernel node was not considered as in the latest work of Huang et al. [6].

* Corresponding author.
E-mail address: egbhuang@ntu.edu.sg (G.-B. Huang).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.08.010

In this paper, a unified solution of weighted ELM for "generalized" SLFNs is proposed, to tackle binary/multiclass classification tasks, which is robust to both balanced and imbalanced data distribution. The proposed method shares some important features in common with unweighted ELM. They are simple in theory and fast in implementation; the hidden nodes can be wide type of feature mapping or in the form of the kernel; and the classification capability to classify any disjoint region is proved to guarantee the generalization performance. In an attempt to alleviate the bias in performance caused by imbalanced class distribution, an extra weight is assigned to each sample to strengthen the impact of minority class while weaken the relative impact of majority class. In the rest of the paper, we refer to the majority class with negative label and the minority class with positive label.

This paper is organized as follows. Section 2 outlined the related work of unweighted ELM and evaluation metrics used in this paper. The proposed method is presented in Section 3. Section 4 provides the discussions on some issues related to the proposed method. Experiments results are analyzed in Section 5. Section 6 ends this paper with a conclusion and future work.

2. Related work

The proposed weighted extreme learning machine (ELM) is based on the unweighted ELM. This section provides a brief review of unweighted ELM. In addition, the evaluation metrics are of our interest. The description of the evaluation metrics used in the paper can be found in this section as well.

2.1. Unweighted extreme learning machine

Extreme learning machine (ELM) [1,2,11] was originally proposed for the single-hidden layer feedforward *neural* networks and was then extended to the "generalized" single-hidden layer feedforward networks (SLFNs) where the hidden layer need not be neuron alike [3,4]. The perspectives of ELM as a least-square based learning algorithm or solution of an optimization problem actually reflect each other and are presented below.

2.1.1. Algorithmic view

The main feature of ELM that distinguishes from conventional neural network learning algorithms is the random generation of hidden nodes. More precisely, the parameters of the hidden nodes are randomly assigned independent of the training samples and the hidden layer output (with L nodes) can be presented by a row vector $\mathbf{h}(\mathbf{x})=[h_1(\mathbf{x}),\ldots,h_L(\mathbf{x})]$ where $\mathbf{x}$ is the input sample. Given N training samples $(\mathbf{x}_i,t_i)$, the mathematical model of the SLFNs is

$$\mathbf{H}\boldsymbol{\beta}=\mathbf{T} \tag{1}$$

where $\mathbf{H}$ is the hidden layer output matrix, $\boldsymbol{\beta}$ is the output weight and $\mathbf{T}$ is the target vector

$$\mathbf{H}=\begin{bmatrix}\mathbf{h}(\mathbf{x}_1)\\ \vdots\\ \mathbf{h}(\mathbf{x}_N)\end{bmatrix} \tag{2}$$

The least square solution with minimal norm is analytically determined using Moore–Penrose "generalized" inverse $\hat{\mathbf{H}}$ [12,13]:

$$\text{when } N<L:\ \boldsymbol{\beta}=\hat{\mathbf{H}}^{\dagger}\mathbf{T}=\mathbf{H}^T\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}$$
$$\text{when } L<N:\ \boldsymbol{\beta}=\hat{\mathbf{H}}^{\dagger}\mathbf{T}=\left(\frac{\mathbf{I}}{C}+\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{T} \tag{3}$$

As can be seen from the two formulas above, a positive value $1/C$ is added to the diagonal of $\mathbf{H}\mathbf{H}^T$ or $\mathbf{H}^T\mathbf{H}$ in order for better generalization performance [14]. Users are offered two formulas to select from according to the size of training data.

2.1.2. Optimization view

Solution of (1) can also be obtained using the optimization method. Similar to SVMs which aim to minimize the training errors and maximize the marginal distance between two classes, the goal of ELM is the same:

$$\text{Minimize}: \|\mathbf{H}\boldsymbol{\beta}-\mathbf{T}\|^2 \quad \text{and} \quad \|\boldsymbol{\beta}\| \tag{4}$$

Similar to LS-SVM, the optimization problem is mathematically written as

$$\text{Minimize}: \quad L_{P_{ELM}}=\frac{1}{2}\|\boldsymbol{\beta}\|^2+C\frac{1}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|^2$$
$$\text{Subject to}: \quad \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta}=\mathbf{t}_i^T-\boldsymbol{\xi}_i^T, \quad i=1,\ldots,N \tag{5}$$

where $\boldsymbol{\xi}_i=[\xi_{i,1},\ldots,\xi_{i,m}]^T$ is the training error vector of the m output nodes with respect to the training sample $\mathbf{x}_i$. Here C is the regularization parameter to represent the trade-off between the minimization of training errors and the maximization of the marginal distance.

According to Karush–Kuhn–Tucker (KKT) theorem [15], same solution as (3) is obtained.

2.2. Evaluation metrics

Usually overall accuracy is used to measure the effectiveness of a classifier. Unfortunately, in presence of imbalanced data, this metric may fail to provide adequate information about the performance of the classifier. Furthermore, the method is very sensitive to the class distribution and might be misleading in some way [7]. For instance, given a binary classification problem consisting of 1 percent positive class and 99 percent negative class. Any dumb classifier would easily achieve 99 percent accuracy by classifying all the samples as negative. Although 99 percentage of overall accuracy seems quite impressive, the fact that the accuracy for the minority class is actually 0 should not be ignored.

To give more insight into the accuracy obtained within each class in lieu of the accuracy of all the samples, an evaluation metric named G-mean is adopted in this paper. After computing the accuracy within each class, the final result to measure the functionality of a classifier is the geometric mean of those accuracies. Take the binary classifier above for example, the G-mean value would be as low as 0 since the accuracy for minority class is 0. In particular, for binary classification problem, G-mean is the square root of (positive class accuracy $\times$ negative class accuracy), where TP, TN, FP, FN stand for true positive, true negative, false positive and false negative, respectively:

$$\text{G-mean}=\sqrt{\frac{TP}{TP+FN}\times\frac{TN}{TN+FP}} \tag{6}$$

Another interesting tool, receiver operating characteristics (ROC) graph [16], provides a visual illustration of the performance of classifiers on binary datasets, where a classifier corresponds to a point. X-coordinate of the point represents false positives rate (FP_rate) and y-coordinate represents true positives rate (TP_rate), so that classification results for both positive class and negative class are perceivable with a single point. That is to say, the performance exhibited by ROC graph is independent of the class distribution and cost information:

$$TP\_rate=\frac{TP}{\#P}$$
$$FP\_rate=\frac{FP}{\#N}=1-\frac{TN}{\#N}=1-TN\_rate \tag{7}$$

where #*P* is the number of samples in positive class; #*N* for negative class. For instance, the classifier denoted by point *A*(0,1) is a perfect classifier, which is able to correctly classify all the points in both positive class and negative class. Classifier represented by point *B*(0.4,0.7) achieves 70 percent accuracy for positive class and 60 percent accuracy for negative class. Points on the diagonal, such as *D*(0.3,0.3) represent the random classifiers, which provide random guesses about the sample label. However, points below the diagonal, on the other hand, are in no way bad classifiers. In fact, they are consistent to the symmetric points within the upper triangle and can be obtained by reversing the label sign of each sample. Hence, the upper triangle rather than the lower triangle is of our interest. In a word, points close to point *A* in Fig. 1 usually represent classifiers with good performance on both positive and negative classes.

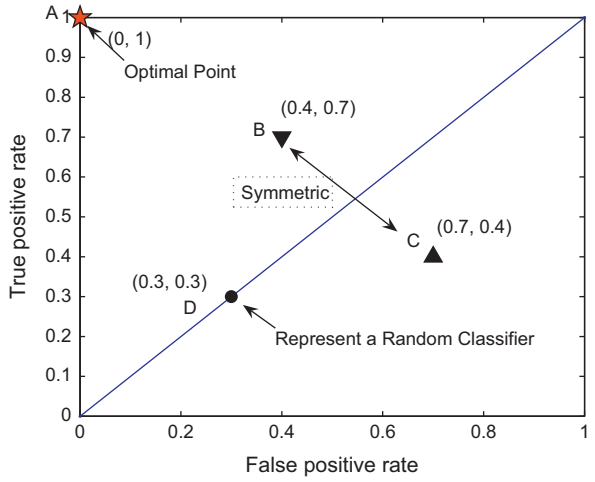

Fig. 1. An example of ROC graph.

3. Weighted ELM

Unweighted extreme learning machine (ELM) with non-kernel or kernel hidden nodes have been demonstrated on various datasets, including face recognition [17,18], protein series [19], etc. The analysis of performance is mainly related to the classification capability of the classifier without considering complex data distribution. This section starts with the analysis of the effect of imbalanced data distribution on the classification performance.

3.1. Effect of imbalance on classification performance of ELM

Without loss of generality, consider a binary classification problem with a large amount of samples labeled as negative class and very few samples labeled as positive class. In Fig. 2 samples from majority class are denoted as the red plus sign and those from minority class are denoted as the blue circle. Intuitively speaking, with the advantage in quantity, the majority class tends to push the separating boundary towards the minority side to gain better classification result for itself.

Mathematically, the intuition can be explained referring to (9). There are two terms to minimize: the cumulative training errors for all the samples and the norm of output weight $\boldsymbol{\beta}$, related with a trade-off constant *C*. To minimize $\boldsymbol{\beta}$ is actually to maximize the marginal distance. (1) When *C* is small (see Fig. 2(a)), meaning user is more concerned with maximization of the marginal distance, the separating boundary is supposed to be very close to the minority class in order for maximal marginal class. As a result, minority class might come across a bad generalization performance since their cumulative errors are ignored compared with large amount of majority class samples. (2) When *C* is relatively large (Fig. 2(d)), the boundary is curved as a result of more concern on minimization of training errors. The accuracy of minority class is compromised again if the imbalance degree is high since the boundary is slight to the right side instead of in

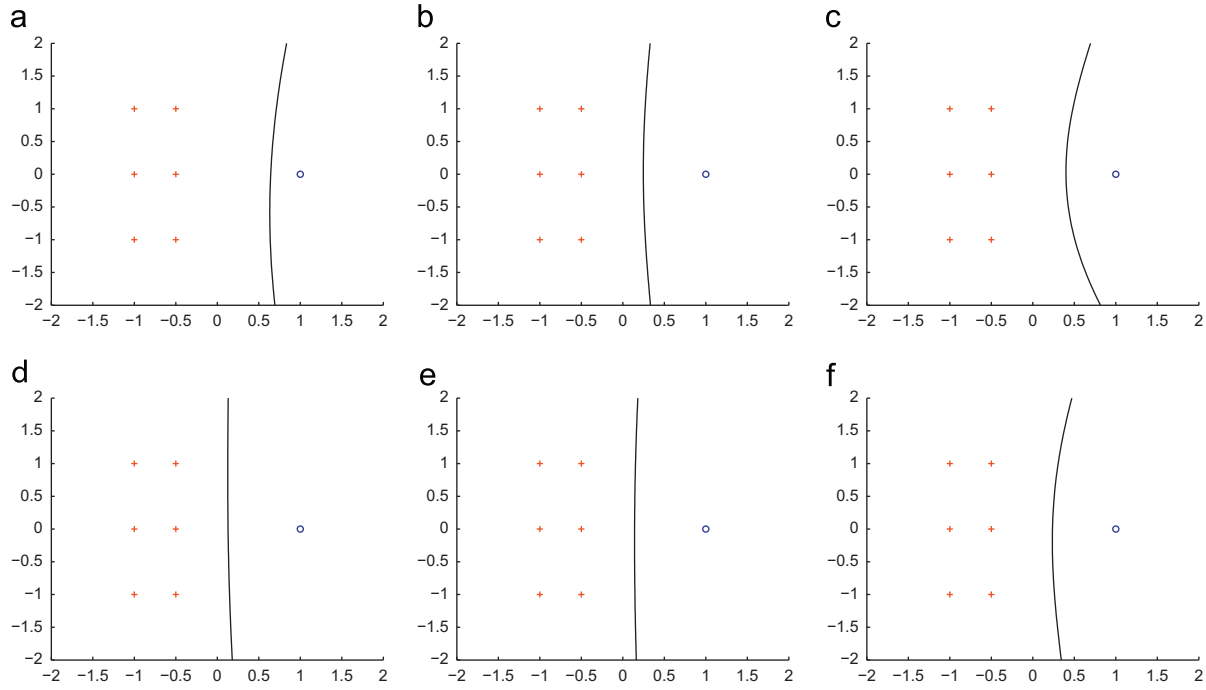

Fig. 2. Effect of imbalanced class distribution on synthetically generated dataset, with three settings of trade-off constant *C*, where the red cross represents the majority class, the blue circle represents the minority class: (a) $C=2^{-5}$ unweighted ELM; (b) $C=2^{0}$ unweighted ELM; (c) $C=2^{5}$ unweighted ELM; (d) $C=2^{-5}$ weighted ELM; (e) $C=2^{0}$ weighted ELM and (f) $C=2^{5}$ weighted ELM. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

the middle. As a matter of fact, the main cause of the performance compromise is that identical focus is given to two classes provided that the imbalance in quantity already exists. Hence, the straightforward solution is to associate higher C to minority class. The second row of Fig. 2 illustrates the movement of separating boundary towards the middle, which is supposed to be the ideal boundary, after trade-off C for minority class is set six times compared to majority class. Therefore, it is necessary to propose ELM with weighted trade-off to cope with imbalanced data distribution.

3.2. Proposed weighted ELM as binary classifier

Given a set of training data $[\mathbf{x}_i,t_i]$, $i=1,\ldots,N$ belonging to two classes, where t_i is either $+1$ or -1 to indicate the positive class or the negative class, we define an $N\times N$ diagonal matrix $\mathbf{W}$ associated with every training sample $\mathbf{x}_i$. Usually if $\mathbf{x}_i$ comes from a minority class (assumed to be positive class), the associated weight W_{ii} is relatively larger than others. How to define the specific weight for each sample will be discussed later. To maximize the marginal distance and to minimize the weighted cumulative error with respect to each sample, we have an optimization problem mathematically written as

$$\text{Minimize}: \|\mathbf{H}\boldsymbol{\beta}-\mathbf{T}\|^2 \quad \text{and} \quad \|\boldsymbol{\beta}\| \tag{8}$$

where $\mathbf{T}=[t_1,\ldots,t_N]$. More precisely,

$$\begin{aligned}\text{Minimize}: \quad & L_{P_{ELM}}=\frac{1}{2}\|\boldsymbol{\beta}\|^2+C\mathbf{W}\frac{1}{2}\sum_{i=1}^{N}\|\xi_i\|^2\\ \text{Subject to}: \quad & \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta}=t_i^T-\xi_i^T, \quad i=1,\ldots,N\end{aligned} \tag{9}$$

To recall, $\mathbf{h}(\mathbf{x}_i)$ is the feature mapping vector in the hidden layer with respect to $\mathbf{x}_i$, and $\boldsymbol{\beta}$ represents the output weight vector connecting the hidden layer and output layer. Since it is a binary classifier, there is only one node in the output layer. ξ_i, the training error of sample $\mathbf{x}_i$, is caused by the difference of the desired output t_i and the actual output $\mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta}$.

According to KKT theorem, the equivalent dual optimization problem with respect to (9) is

$$L_{D_{ELM}}=\frac{1}{2}\|\boldsymbol{\beta}\|^2+C\mathbf{W}\frac{1}{2}\sum_{i=1}^{N}\xi_i^2-\sum_{i=1}^{N}\alpha_i(\mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta}-t_i+\xi_i) \tag{10}$$

where the Lagrange multiplier α_i is the constant factor of sample $\mathbf{x}_i$ in the linear combination to form the final decision function. Further, by making the partial derivatives with respect to variables $(\boldsymbol{\beta},\xi_i,\alpha_i)$ all equal to zero, the KKT optimality conditions are obtained

$$\begin{aligned}&\frac{\partial L_{D_{ELM}}}{\partial\boldsymbol{\beta}}=0\rightarrow\boldsymbol{\beta}=\sum_{i=1}^{N}\alpha_i\mathbf{h}(\mathbf{x}_i)^T=\mathbf{H}^T\boldsymbol{\alpha}\\ &\frac{\partial L_{D_ELM}}{\partial\xi_i}=0\rightarrow\alpha_i=C\mathbf{W}\xi_i, \quad i=1,\ldots,N\\ &\frac{\partial L_{D_ELM}}{\partial\alpha_i}=0\rightarrow\mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta}-t_i+\xi_i=0, \quad i=1,\ldots,N\end{aligned} \tag{11}$$

Similar to (3), two versions of solutions of $\boldsymbol{\beta}$ can be derived from (11) regarding left pseudo-inverse or right pseudo-inverse. When presented data with small size, right pseudo-inverse is recommended because it involves the inverse of an $N\times N$ matrix. Otherwise, left pseudo-inverse is more suitable since it is much easier to compute matrix inversion of size $L\times L$ when L is much smaller than N:

$$\begin{aligned}&\text{when } N \text{ is small}: \boldsymbol{\beta}=\mathbf{H}^T\left(\frac{\mathbf{I}}{C}+\mathbf{W}\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{W}\mathbf{T}\\ &\text{when } N \text{ is large}: \boldsymbol{\beta}=\left(\frac{\mathbf{I}}{C}+\mathbf{H}^T\mathbf{W}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{W}\mathbf{T}\end{aligned} \tag{12}$$

Given a new sample $\mathbf{x}$, the output function of the ELM classifier is obtained from $f(\mathbf{x})=\text{sign}\,\mathbf{h}(\mathbf{x})\boldsymbol{\beta}$:

$$\begin{aligned}&f(\mathbf{x})_{N\times N}=\text{sign}\,\mathbf{h}(\mathbf{x})\mathbf{H}^T\left(\frac{\mathbf{I}}{C}+\mathbf{W}\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{W}\mathbf{T}\\ &f(\mathbf{x})_{L\times L}=\text{sign}\,\mathbf{h}(\mathbf{x})\left(\frac{\mathbf{I}}{C}+\mathbf{H}^T\mathbf{W}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{W}\mathbf{T}\end{aligned} \tag{13}$$

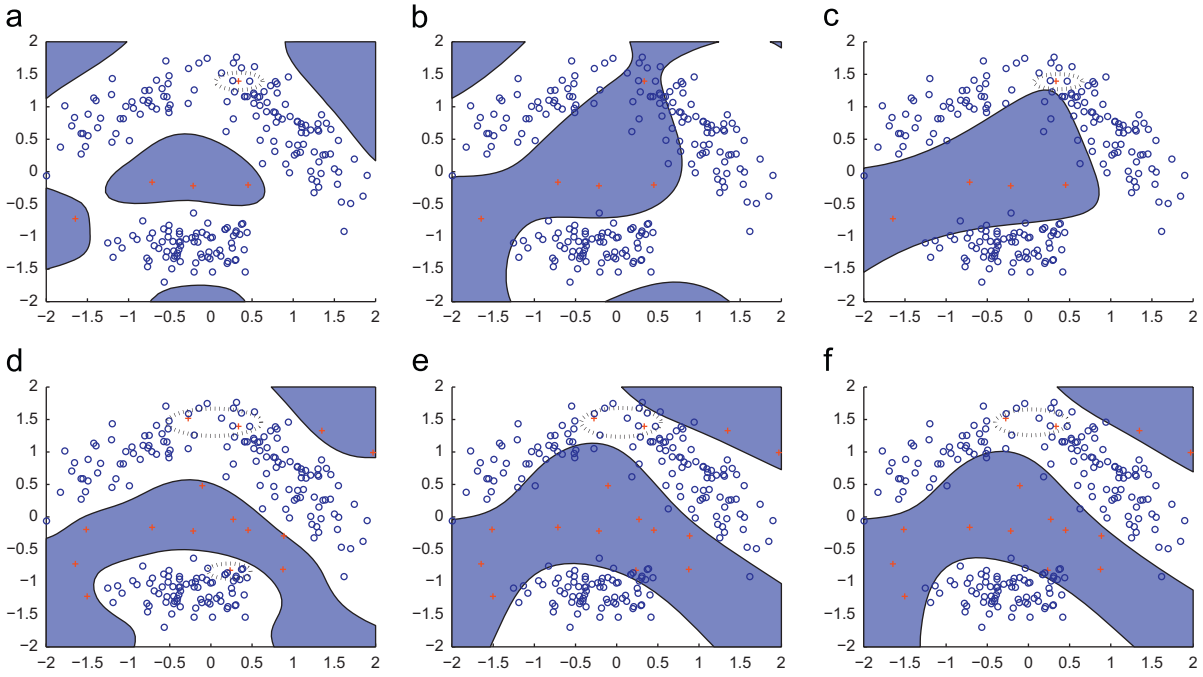

Fig. 3. The decision boundary produced by unweighted ELM and weighted ELM on dataset *banana* with imbalance ratio 1:43 on Row 1 or 3:43 on Row 2: (a) 1:43, unweighted; (b) 1:43, weighted to 1:1; (c) 1:43, weighted to 0.618:1; (d) 3:43, unweighted; (e) 3:43, weighted to 1:1 and (f) 3:43, weighted to 0.618:1. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

Inspired from work of [6] and the definition of a kernel (21), output function in terms of kernel is naturally derived from the $N \times N$ version:

$$f(\mathbf{x})_{kernel} = \text{sign}\ \mathbf{h}(\mathbf{x})\mathbf{H}^T\left(\frac{\mathbf{I}}{C}+\mathbf{W}\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{W}\mathbf{T}$$
$$= \text{sign}\begin{bmatrix}K(\mathbf{x},\mathbf{x}_1)\\ \vdots \\ K(\mathbf{x},\mathbf{x}_N)\end{bmatrix}^T\left(\frac{\mathbf{I}}{C}+\mathbf{W}\mathbf{\Omega}_{ELM}\right)^{-1}\mathbf{W}\mathbf{T} \quad (14)$$

3.3. Proposed weighted ELM as multiclass classifier

One of the pragmatistic advantages of ELM is that a single classifier is capable of multiclass classification task. As far as the network is concerned, the implementation makes use of an SLFN with multiple output nodes. For a given testing sample, the output node with largest decision function value indicates the class label. Assume a set of multiclass training samples $[\mathbf{x}_i,t_i]$, $i=1,\ldots,N$, where $t_i \in [1,m]$. According to the ELM algorithm, each class label is expanded into a label vector of length m. Take $\mathbf{x}_i$ from class two as example, the corresponding label vector is $\mathbf{t}_i = [\underbrace{-1,1,-1,\ldots,-1}_{m}]$.

Subsequently, to minimize the cumulative error vector $\boldsymbol{\xi}_i$ for every sample and to maximize the marginal distance $2/\|\boldsymbol{\beta}\|$, we have

$$\text{Minimize}: \quad L_{P_{ELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\mathbf{W}\frac{1}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|^2$$
$$\text{Subject to}: \quad \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} = \mathbf{t}_i^T - \boldsymbol{\xi}_i^T, \quad i=1,\ldots,N \quad (15)$$

where the label matrix for the whole dataset is differently defined as in binary case $\mathbf{T}=[\mathbf{t}_1,\ldots,\mathbf{t}_N]$ and so is the error vector $\boldsymbol{\xi}_i$ with respect to training sample $\mathbf{x}_i$, $\boldsymbol{\xi}_i=[\xi_1,\ldots,\xi_m]$. The solution to the optimization problem above can be found the same as (27). Therefore, the output function vector $\mathbf{f}(\mathbf{x})=[f_1(\mathbf{x}),\ldots,f_m(\mathbf{x})]$ for a testing sample $\mathbf{x}$ is shown in (16) as the non-kernel format and (14) as the kernel format. To find out the predicted label of $\mathbf{x}$, users just need to refer to a simple equation as below:

$$\text{label}(\mathbf{x}) = \arg\max f_i(\mathbf{x}), \quad i \in \{1,\ldots,m\} \quad (16)$$

Note that another popular machine learning technique support vector machine (SVM) [20,21], originally a binary classifier, fails to be applied to the multiclass problems directly without modification. Usually the multiclass problem is decomposed into binary subproblems, implemented by multiple SVMs. This explains the statement about the advantage of ELM classifiers in the beginning of this section.

3.4. Feature mappings and kernels

The feature mapping in ELM $\mathbf{h}(\mathbf{x})=[h(\mathbf{x})_1,\ldots,h(\mathbf{x})_L]$, also called the hidden layer output vector, actually maps the data from the original data space to the hidden layer space with dimensionality L. As the paramount feature of ELM, the parameters $(\mathbf{a}_i,b_i)$ in the hidden layer node function $h(\mathbf{x})_i = G(\mathbf{a}_i,b_i,\mathbf{x})$ are randomly generated according to any continuous probability distribution. In other words, the hidden layer need to be tuned, in contrast with the conventional neural network learning algorithms.

Theoretically speaking, the hidden layer node function $G(\mathbf{a},b,\mathbf{x})$ includes almost all nonlinear piecewise continuous functions as long as ELM universal approximation capability theorems are satisfied [2,6]. A wide type of feature mapping functions can be considered which include but not limit to

- Sigmoid function
$$G(\mathbf{a},b,\mathbf{x}) = \frac{1}{1+\exp(-(\mathbf{a}\cdot\mathbf{x}+b))} \quad (17)$$
- Gaussian function
$$G(\mathbf{a},b,\mathbf{x}) = \exp(-b\|\mathbf{x}-\mathbf{a}\|^2) \quad (18)$$
- Hardlimit function
$$G(\mathbf{a},b,\mathbf{x}) = \begin{cases}1 & \text{if } \mathbf{a}\cdot\mathbf{x}-b\geq 0\\ 0 & \text{otherwise}\end{cases} \quad (19)$$
- Multiquadrics function
$$G(\mathbf{a},b,\mathbf{x}) = (\|\mathbf{x}-\mathbf{a}\|^2+b^2)^{1/2} \quad (20)$$

On the other hand, it is not necessary that the feature mapping function $\mathbf{h}(\mathbf{x})$ is always known. If $\mathbf{h}(\mathbf{x})$ is unknown to the users, a kernel matrix for ELM as shown in (21) is defined [6] in accordance with Mercer's conditions. The most popular kernel in use is Gaussian kernel $K(\mathbf{u},\mathbf{v})=\exp(-\gamma\|\mathbf{u}-\mathbf{v}\|^2)$, where γ is the kernel parameter.

$$\mathbf{\Omega}_{ELM} = \mathbf{H}\mathbf{H}^T : \Omega_{ELM\,i,j} = h(\mathbf{x}_i)\cdot h(\mathbf{x}_j) = K(\mathbf{x}_i,\mathbf{x}_j) \quad (21)$$

Hence, the output function of an SLFN is kernelized as

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\mathbf{H}^T\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T} = \begin{bmatrix}K(\mathbf{x},\mathbf{x}_1)\\ \vdots \\ K(\mathbf{x},\mathbf{x}_N)\end{bmatrix}^T\left(\frac{\mathbf{I}}{C}+\mathbf{\Omega}_{ELM}\right)^{-1}\mathbf{T} \quad (22)$$

Therefore, with different feature mappings or kernels, the proposed weighted ELM provides a unified solution for the "generalized" SLFNs, including conventional neural networks, support vector networks, regularization networks, polynomial networks, etc.

4. Discussions

The previous section provided a solution for imbalanced data learning by adding a weight matrix in extreme learning machine (ELM) to strengthen the impact of minority class while weaken the impact of majority class. The effect of adding the weight to classification performance improvement is illustrated by a binary classification problem. Issues about how to interpret the effect of the weight and determine the value are discussed as well. Finally, a comparison is conducted between the existing weight ELMs and the proposed method.

4.1. Essence of the difference between unweighted ELM and weighted ELM

From either the algorithmic or optimization point of view, the learning process of the traditional unweighted ELM is actually to search for a mathematical form to approximate the underlying relationship between the feature of the sample and its label. The solution has to maximize the marginal distance between the two classes while minimize the training errors.

In unweighted ELM, the Lagrange multiplier $\alpha_i = C\xi_i, \forall i$ of $\mathbf{x}_i$, which indicates the weight of fraction in decision function, is proportional to the training error ξ_i of that sample. So that those samples provide more information (with larger training error) are emphasized in the training model.

However, a good approximater must give priority to not only those samples with larger errors, but also those samples which is able to characterize the data distribution. Thus the weighted ELM is proposed based on unweighted ELM to provide additional emphasis to the samples which imply the imbalanced class distribution. As seen from the Lagrange multiplier $\alpha_i = C\mathbf{W}\xi_i, \forall i$ of $\mathbf{x}_i$, samples from the minority class is assigned with larger weight W_{ii} such that the information of imbalanced class distribution is well perceived.

4.2. Movement of boundary with/without weight

From the optimization point of view, the essence of an ELM classifier is to find a boundary to separate data from any disjoint two or multiple parts with maximal marginal distance between any two parts, after the data is mapped into the hidden layer space. In imbalanced data environment, the separating boundary is supposed to be pushed toward the side of minority class, which in fact favors the performance of majority class. The difference of unweighted ELM and weighted ELM can be interpreted from the movement of the separating boundary.

Take a binary problem *banana* with two imbalance ratios for example. The separating boundaries produced by ELM using sigmoid feature mapping with/without weight are depicted in Fig. 3. The first row represents training data consisting of five positive samples (red cross) and 215 negative samples (blue circle), with imbalance ratio 1:43. Training data on the second row consists of 15 positive samples and 215 negative samples, with less harsh imbalance ratio 3:43. In addition, to highlight the performance improvement on minority class, misclassified samples from minority class are circled by a black dotted ellipse.

The accuracies for majority class are both 100 percent, both the accuracies for minority class are 80 percent when unweighted ELM is applied (see Fig. 3(a) and (e)). Instead, when applying the weighted ELM, as a result of balancing the impact between minority class and majority class, it is obvious to observe that the boundary is pushed towards the majority class. An unfortunate fact is that the accuracy in minority class is increased in the cost of the decrease in majority class. As can be seen from Fig. 3(b)–(c) and (e)–(f), a certain amount of samples from majority class are misclassified in compromise for better accuracy in minority class, 100 percent for *banana* (1:43) and 86.67 percent for *banana* (3:43). However, the misclassification rate should not exceed 20 percent, which gives an overall better result in terms of geometric mean.

Interestingly, the boundary movement phenomena helps relate the algorithmic approach and sampling approach in imbalanced data learning. In the proposed weighted ELM, after weighting scheme is applied, the separating boundary is pushed from the minority class towards the majority class. As a matter of fact, some amount of samples from majority class are misclassified as a by-product, which on the other hand is interpreted as that some redundant majority samples are removed. Further, another interpretation is that the minority samples around the boundary are duplicated in the neighborhood, which is one type of oversampling technique.

4.3. How to determine the weight

The weight matrix $\mathbf{W} = \mathrm{diag}\{W_{ii}\}$, $i = 1, \ldots, N$ plays an important role in the proposed algorithm. In essence, it determines to what degree of re-balance users are seeking for, and how much further the boundary is pushed towards the majority class.

Users can define W_{ii} for every sample $\mathbf{x}_i$, so that the proposed algorithm actually belongs to the family of cost sensitive learning. For the sake of convenience, we choose a weighting scheme automatically generated from the class information, which is in fact a special case of the cost sensitive learning:

$$\text{Weighting Scheme } \mathbf{W}1: \ W_{ii} = 1/\#(t_i) \tag{23}$$

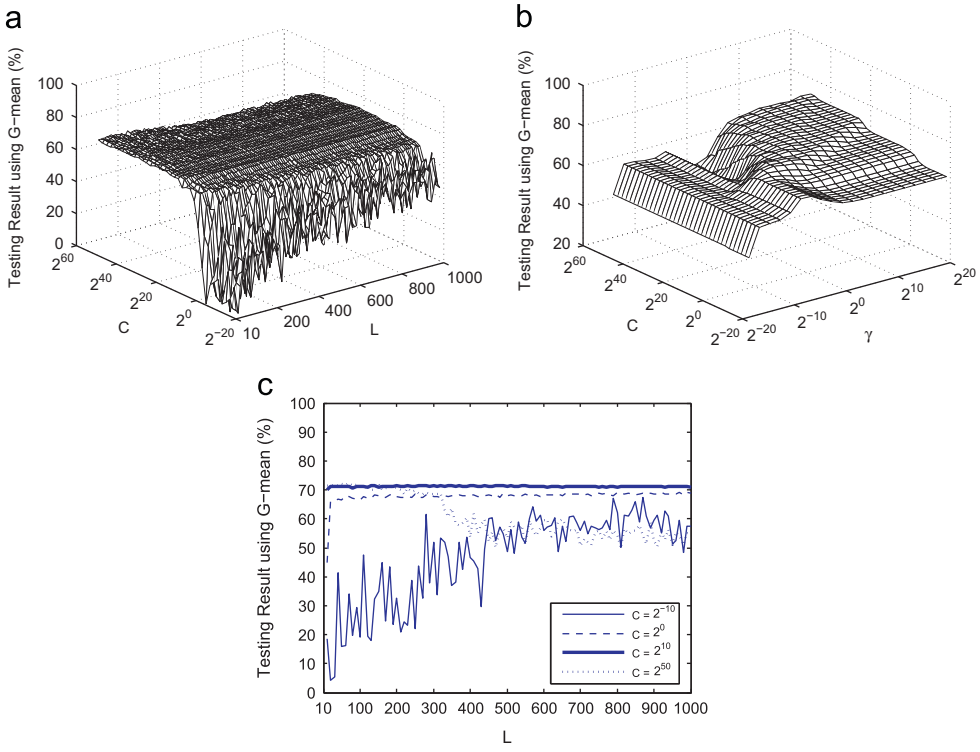

Fig. 4. Display of G-mean distribution of yeast1 with different choices of C and L for Sigmoid node or C and γ for Gaussian kernel: (a) Sigmoid node; (b) Gaussian kernel and (c) Sigmoid node when C varies.

where $\#(t_i)$ is the number of samples belonging to class t_i, $i=1,\ldots,m$.

After applying the weighting scheme **W**1, we can roughly imagine the quantity of minority class and majority class is re-balanced into the ratio of $\underbrace{1:1:\cdots:1}_{m}$.

Another proposed weighting (24) is considered to minish the balancing step into the ratio of 0.618:1 (here the author adopts the value of golden standard that represents the perfection in nature) between minority classes and the majority classes. In binary classification, the minority class is assumed to be the positive class and the majority class is assumed to be the negative class. In multiclass classification, the minorities classes refer to those classes whose number is below the average samples per class and the majority classes are those with sample number above the average. As a result, compared to weighting scheme **W**1 the boundary is pushed slightly backwards the minority class so that the misclassification cases in compromise on the majority side is sought of alleviated.

$$\text{Weighting Scheme } \mathbf{W}2 \begin{cases} W_{ii}=0.618/\#(t_i) & \text{if } t_i > \text{AVG}(t_i) \\ W_{ii}=1/\#(t_i) & \text{if } t_i \le \text{AVG}(t_i) \end{cases} \tag{24}$$

4.4. Difference with other weighted ELMs and weighted LS algorithms

In the literature, two weighted ELMs were proposed by Toh [9] and Deng et al. [10], though none is targeting the imbalanced data problem. Moreover, neither of them has mentioned the feasibility of "generalized" feature mappings or kernels.

Deng et al. [10] proposed a weighted regularized ELM (WR-ELM) which is similar to the proposed method (27) in the presence of large training data size. But difference in training procedure, weighting scheme and feature mapping between WR-ELM and the proposed weighted ELM is obvious. (1) Compared with the traditional three-step training procedure of ELM using only one classifier, two classifiers are used in the training phase. The network model is initially obtained by training the data with an *unweighted* ELM classifier. Afterwards, another weighted ELM (25) is applied to the training data to update the model on the basis of the error information obtained from the *unweighted* ELM. (2) The weighting scheme in WR-ELM is defined with respect to the training error (26). More precisely, the weight of each sample is inversely proportional to the training error, so that samples with high training error are suppressed to prevent the effect of outliers. (3) Only sigmoid additive neural node is considered in their work. And WR-ELM mainly focuses on the case when the training data outnumber the network size (when $L<N$):

$$\boldsymbol{\beta}=\left(\frac{\mathbf{I}}{C}+\mathbf{H}^T\mathbf{W}^2\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{W}^2\mathbf{T} \tag{25}$$

where $\mathbf{W}=\text{diag}\{v_1,v_2,\ldots,v_N\}$ and

$$v_i=\begin{cases} 1, & |\xi_i/\hat{s}| \le c_1 \\ \dfrac{c_2-|\xi_i/\hat{s}|}{c_2-c_1}, & c_1 \le |\xi_i/\hat{s}| \le c_2 \\ 10^{-4}, & |\xi_i/\hat{s}| > c_2 \end{cases} \tag{26}$$

The error information after training data with the first *unweighted* ELM includes, training error ξ_i for sample $\mathbf{x}_i$, robust estimate $\hat{s}$ of the standard deviation of ξ_i. c_1 and c_2 are assigned empirical values as 2.5 and 3, respectively.

The total error rate (TER) ELM [9] only defined the binary weighted ELM similar to (27), with weighting scheme same as **W**1. To deal with multiclass tasks, one-against-all (OAA)

Table 1
Specification of binary classification problems.

Datasets	# of Attributes	# of TrainData	# of TestData	Imbalance ratio
ecoli1	7	268	68	0.2947
ecoli2	7	268	68	0.1806
ecoli3	7	268	68	0.1167
ecoli4	7	268	68	0.0635
yeast1	8	1187	297	0.4064
yeast3	8	1187	297	0.1230
yeast4	8	1187	297	0.0349
yeast5	8	1187	297	0.0304
yeast6	8	1187	297	0.0243
yeast-0-5-6-7-9_vs_4	8	422	106	0.1047
yeast-1-2-8-9_vs_7	8	757	188	0.0327
yeast-1-4-5-8_vs_7	8	554	139	0.0453
yeast-1_vs_7	7	367	92	0.0700
yeast-2_vs_4	8	411	103	0.1078
yeast-2_vs_8	8	385	97	0.0434
glass0	9	173	43	0.4786
glass1	9	171	43	0.5405
glass2	9	171	43	0.0823
glass4	9	171	43	0.0621
glass5	9	171	43	0.0427
glass6	9	171	43	0.1554
glass-0-1-2-3_vs_4-5-6	9	171	43	0.3053
glass-0-1-6_vs_2	9	153	39	0.0929
glass-0-1-6_vs_5	9	147	37	0.0500
iris0	4	120	30	0.5000
vowel0	13	790	198	0.1002
new-thyroid1	5	172	43	0.1944
new-thyroid2	5	172	43	0.1944
shuttle-C0_vs_C4	9	1463	366	0.0726
shuttle-C2_vs_C4	9	103	26	0.0404
segment0	19	1846	462	0.1661
vehicle0	18	676	170	0.3075
vehicle1	18	676	170	0.3439
vehicle2	18	676	170	0.3466
vehicle3	18	676	170	0.3333
page-blocks0	10	4377	1095	0.1137
page-blocks1	10	377	95	0.0620
abalone9_18	8	584	147	0.0600
abalone19	8	3339	835	0.0078
wisconsin	9	546	137	0.5380
haberman	3	244	62	0.3556
pima	8	614	154	0.5350

Table 2
Specification of binary and multiclass problems from UCI.

Datasets	# of Attributes	# of Classes	# of TrainData	# of TestData	Imbalance ratio
Adult	123	2	4781	27 780	0.3306
Banana	2	2	400	4900	0.8605
Colon(Gene Sel)	60	2	30	32	0.6667
Leukemia(Gene Sel)	60	2	38	34	0.4074
DNA	180	3	2000	1186	0.4415
Satimage	36	6	4435	2000	0.3871
USPS	256	10	7291	2007	0.4733

Table 3
Performance result of binary problems with imbalance ratio $\in(0,0.2)$.

G-mean	Data(IR)											
	Sigmoid node						Gaussian kernel					
	Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2		Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2	
	(C,L)	Testing result (%)	(C,L)	Testing result (%)	(C,L)	Testing Result(%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)
abalone19(0.0078)	$(2^{42},990)$	47.52	$(2^{6},150)$	**77.19**	$(2^{36},40)$	64.27	$(2^{40},2^{0})$	51.36	$(2^{24},2^{16})$	74.47	$(2^{24},2^{12})$	68.35
yeast6(0.0243)	$(2^{44},350)$	70.77	$(2^{34},20)$	87.77	$(2^{14},900)$	88.29	$(2^{0},2^{-8})$	75.66	$(2^{4},2^{-6})$	89.81	$(2^{2},2^{-6})$	**90.36**
yeast5(0.0304)	$(2^{36},900)$	81.04	$(2^{30},100)$	95.39	$(2^{2},200)$	95.15	$(2^{4},2^{-6})$	87.34	$(2^{4},2^{-8})$	**98.18**	$(2^{-18},2^{-8})$	98.14
yeast-1-2-8-9_vs_7(0.0327)	$(2^{42},880)$	59.23	$(2^{42},20)$	**75.83**	$(2^{50},10)$	73.92	$(2^{48},2^{10})$	60.97	$(2^{4},2^{-2})$	71.41	$(2^{20},2^{10})$	68.98
yeast4(0.0349)	$(2^{42},960)$	65.52	$(2^{6},20)$	**87.29**	$(2^{6},100)$	82.18	$(2^{-6},2^{-16})$	70.47	$(2^{2},2^{-4})$	84.98	$(2^{6},2^{-2})$	83.12
shuttle-C2_vs_C4(0.0404)	$(2^{40},20)$	93.54	$(2^{31},10)$	100	$(2^{24},10)$	100	$(2^{-18},2^{-12})$	94.14	$(2^{-18},2^{0})$	100	$(2^{24},2^{4})$	99.60
glass5(0.0427)	$(2^{20},90)$	90.81	$(2^{10},110)$	95.99	$(2^{10},40)$	96.60	$(2^{18},2^{6})$	93.16	$(2^{10},2^{4})$	96.51	$(2^{14},2^{6})$	**97.26**
yeast-2_vs_8(0.0434)	$(2^{0},290)$	72.83	(2,960)	75.56	$(2^{8},60)$	76.01	$(2^{24},2^{-4})$	77.24	$(2^{14},2^{-4})$	77.89	$(2^{18},2^{-2})$	**78.00**
yeast-1-4-5-8_vs_7(0.0453)	$(2^{44},970)$	61.07	$(2^{10},120)$	67.10	$(2^{42},800)$	64.26	$(2^{24},2^{-4})$	59.89	$(2^{0},2^{-8})$	**69.32**	$(2^{40},2^{2})$	65.68
glass-0-1-6_vs_5(0.0500)	$(2^{18},660)$	92.41	$(2^{6},960)$	98.55	$(2^{6},960)$	**98.70**	$(2^{22},2^{4})$	98.55	$(2^{8},2^{2})$	**98.85**	$(2^{8},2^{2})$	**98.85**
abalone9_18(0.0600)	$(2^{40},150)$	75.29	$(2^{16},70)$	87.99	$(2^{32},20)$	88.72	$(2^{18},2^{0})$	72.71	$(2^{28},2^{14})$	**89.76**	$(2^{14},2^{4})$	86.83
page-blocks1(0.0620)	$(2^{10},440)$	97.78	$(2^{30},30)$	**98.98**	$(2^{44},50)$	**98.98**	$(2^{2},2^{-2})$	97.89	$(2^{20},2^{10})$	98.07	$(2^{28},2^{12})$	98.64
glass4(0.0621)	$(2^{34},30)$	85.72	$(2^{12},120)$	91.34	$(2^{10},140)$	**91.46**	$(2^{-4},2^{-2})$	85.93	$(2^{8},2^{2})$	91.17	$(2^{-2},2^{4})$	91.17
ecoli4(0.0635)	$(2^{22},60)$	91.96	$(2^{6},180)$	97.83	$(2^{6},200)$	95.90	$(2^{46},2^{16})$	94.32	$(2^{6},2^{0})$	**98.24**	$(2^{0},2^{-2})$	97.42
yeast-1_vs_7(0.0700)	$(2^{40},960)$	65.58	$(2^{16},550)$	77.26	$(2^{14},350)$	76.91	$(2^{14},2^{-4})$	64.48	$(2^{20},2^{4})$	**77.72**	$(2^{14},2^{2})$	74.61
shuttle-C0_vs_C4(0.0726)	$(2^{14},10)$	100	$(2^{38},10)$	100	$(2^{28},10)$	100	$(2^{14},2^{0})$	100	$(2^{26},2^{0})$	100	$(2^{26},2^{0})$	100
glass2(0.0823)	$(2^{28},110)$	79.49	$(2^{22},140)$	80.33	$(2^{24},130)$	**83.34**	$(2^{16},2^{-2})$	66.28	$(2^{14},2^{2})$	82.59	$(2^{14},2^{4})$	80.26
glass-0-1-6_vs_2(0.0929)	$(2^{34},150)$	67.78	$(2^{14},380)$	**83.77**	$(2^{22},140)$	83.06	$(2^{50},2^{18})$	63.20	$(2^{32},2^{8})$	**83.59**	$(2^{50},2^{16})$	81.34
vowel0(0.1002)	$(2^{28},110)$	100	$(2^{50},120)$	100	$(2^{40},100)$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100
yeast-0-5-6-7-9_vs_4(0.1047)	$(2^{32},390)$	64.49	$(2^{-2},150)$	81.05	$(2^{14},290)$	80.56	$(2^{-6},2^{-16})$	68.68	$(2^{4},2^{-6})$	**82.21**	$(2^{2},2^{-6})$	80.32
yeast-2_vs_4(0.1078)	$(2^{36},280)$	86.25	$(2^{26},940)$	91.56	$(2^{30},130)$	90.02	$(2^{6},2^{-4})$	85.31	$(2^{32},2^{4})$	**91.88**	$(2^{14},2^{-2})$	90.25
page-blocks0(0.1137)	$(2^{34},830)$	89.92	$(2^{22},750)$	93.21	$(2^{24},820)$	93.40	$(2^{4},2^{-4})$	89.62	$(2^{16},2^{0})$	93.61	$(2^{16},2^{0})$	92.98
ecoli3(0.1167)	$(2^{44},70)$	77.38	$(2^{46},10)$	**90.17**	$(2^{6},30)$	90.00	$(2^{2},2^{-2})$	76.78	$(2^{10},2^{4})$	89.60	$(2^{2},2^{-2})$	**90.14**
yeast3(0.1230)	$(2^{40},100)$	80.75	$(2^{16},700)$	**93.25**	$(2^{16},440)$	91.08	$(2^{44},2^{8})$	84.13	$(2^{48},2^{16})$	**93.11**	$(2^{16},2^{2})$	90.94
glass6(0.1554)	$(2^{46},450)$	94.96	$(2^{44},30)$	95.72	$(2^{26},190)$	**95.90**	$(2^{8},2^{2})$	92.60	$(2^{30},2^{8})$	94.04	$(2^{30},2^{8})$	94.04
segment0(0.1661)	$(2^{8},720)$	99.24	$(2^{18},870)$	99.75	$(2^{22},350)$	99.70	$(2^{4},2^{-4})$	99.57	$(2^{12},2^{0})$	99.77	$(2^{12},2^{0})$	99.82
ecoli2(0.1806)	$(2^{36},60)$	91.17	$(2^{28},40)$	93.91	$(2^{30},20)$	**94.51**	$(2^{-18},2^{-6})$	94.31	$(2^{8},2^{-2})$	94.09	$(2^{8},2^{-2})$	**94.43**
new-thyroid1(0.1944)	$(2^{18},180)$	98.24	$(2^{18},30)$	99.44	$(2^{20},20)$	**99.72**	$(2^{0},2^{-4})$	99.16	$(2^{14},2^{0})$	**99.72**	$(2^{6},2^{0})$	**99.72**
new-thyroid2(0.1944)	$(2^{18},40)$	95.55	$(2^{16},290)$	99.72	$(2^{12},480)$	**100**	$(2^{2},2^{-4})$	99.44	$(2^{12},2^{-4})$	99.72	$(2^{14},2^{0})$	99.72

method was utilized which is obviously complicated than a single classifier. Take an m-class problem for example, m binary TER-ELMs are required where the ith TER-ELM assumes the ith class as positive class and the rest $m-1$ classes as the negative class. Suppose there are m_i^+ number of samples in the positive class, and m_i^- in the negative class. The output of samples from positive class is assigned $y_i^+=(a+b)$, and $y_i^-=(a-b)$ for the negative class, where the threshold a and bias b is specified by users. With the preprocessed training data, the solution of the ith binary classifier is

$$\text{When } N \text{ is small}: \boldsymbol{\beta}_i=\mathbf{H}_i^T\left(\frac{\mathbf{I}}{C}+\mathbf{W}_i\mathbf{H}_i\mathbf{H}_i^T\right)^{-1}\mathbf{W}_i\mathbf{y}_i$$

$$\text{When } N \text{ is large}: \boldsymbol{\beta}_i=\left(\frac{\mathbf{I}}{C}+\mathbf{H}_i^T\mathbf{W}_i\mathbf{H}_i\right)^{-1}\mathbf{H}_i^T\mathbf{W}_i\mathbf{y}_i \quad (27)$$

where $\mathbf{W}_i=\mathbf{W}_i^+ +\mathbf{W}_i^-$ and $\mathbf{W}_i^+=\text{diag}(0,\ldots,0,1/m_i^+,\ldots,1/m_i^+)$, $\mathbf{W}_i^-=\text{diag}(0,\ldots,0,1/m_i^-,\ldots,1/m_i^-)$. The hidden layer output matrix $\mathbf{H}_i$ and the output vector $\mathbf{y}_i$ are in the order of negative class after positive class, that is to say, all the non-jth classes after jth class. An interesting point is that the strategy OAA itself creates an synthetic imbalance between positive and negative classes, which may render the whole data space in a complex class distribution. Furthermore, although TER-ELM was concluded with the same solution as (27), kernels and "generalized" feature mapping was not mentioned.

In [22], a two-class kernel classifier was proposed for imbalanced learning problems, where the weighted regularized orthogonal least square algorithm (ROWLS) was used to train the classifier. Although the proposed weighted ELM (WELM) shares the same regularized least square cost function as work in [22], there are three main aspects that differentiate these two works. (1) ROWLS are kernel based classifiers. But in WELM, the hidden

Table 4
Performance result of binary problems with imbalance ratio $\in(0.2,1)$.

G-mean	Datasets(IR)											
	Sigmoid node						Gaussian kernel					
	Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2		Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2	
	(C,L)	Testing result (%)	(C,L)	Testing result (%)	(C,L)	Testing Result(%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)
ecoli1(0.2947)	$(2^{16},140)$	87.77	$(2^{4},320)$	90.69	$(2^{22},40)$	90.26	$(2^{0},2^{-4})$	88.75	$(2^{10},2^{4})$	**91.04**	$(2^{50},2^{18})$	90.25
glass-0-1-2-3_vs_4-5-6(0.3053)	$(2^{8},140)$	90.67	$(2^{10},290)$	94.68	$(2^{12},100)$	93.14	$(2^{10},2^{-2})$	93.26	$(2^{-18},2^{-4})$	**95.41**	$(2^{-18},2^{-4})$	95.41
vehicle0(0.3075)	$(2^{10},460)$	98.57	$(2^{16},850)$	99.32	$(2^{16},850)$	99.04	$(2^{8},2^{2})$	99.36	$(2^{16},2^{2})$	99.30	$(2^{20},2^{4})$	99.36
adult(0.3306)	$(2^{10},580)$	73.86	$(2^{4},840)$	81.67	$(2^{6},1000)$	80.42	$(2^{5},2^{5})$	73.54	$(2^{10},2^{5})$	81.62	$(2^{20},2^{10})$	80.37
vehicle3(0.3333)	$(2^{8},450)$	78.15	$(2^{14},710)$	85.13	$(2^{14},340)$	84.34	$(2^{16},2^{4})$	72.17	$(2^{2}4,2^{8})$	**92.45**	$(2^{20},2^{6})$	84.90
vehicle1(0.3439)	$(2^{8},570)$	79.29	$(2^{14},450)$	85.30	$(2^{12},450)$	83.44	$(2^{18},2^{8})$	80.60	$(2^{24},2^{8})$	**86.74**	$(2^{26},2^{8})$	85.72
vehicle2(0.3466)	$(2^{12},600)$	98.43	$(2^{16},800)$	**99.12**	$(2^{16},550)$	98.78	$(2^{8},2^{0})$	**99.22**	$(2^{20},2^{2})$	**99.45**	$(2^{22},2^{4})$	99.38
haberman(0.3556)	$(2^{44},910)$	49.16	$(2^{34},10)$	65.11	$(2^{18},90)$	59.26	$(2^{42},2^{0})$	57.23	$(2^{14},2^{4})$	**66.26**	$(2^{32},2^{6})$	60.91
yeast1(0.4064)	$(2^{44},300)$	63.26	$(2^{26},120)$	72.57	$(2^{32},250)$	70.32	$(2^{0},2^{-6})$	65.45	$(2^{10},2^{-4})$	**73.17**	$(2^{10},2^{-4})$	70.33
Leukemia(Gene Sel)(0.4074)	$(2^{10},100)$	100	$(2^{10},100)$	100	$(2^{10},100)$	100	$(2^{-18},2^{-2})$	100	$(2^{-18},2^{-2})$	100	$(2^{-18},2^{-2})$	100
glass0(0.4786)	$(2^{14},950)$	79.61	$(2^{22},800)$	81.17	$(2^{22},710)$	82.62	$(2^{0},2^{-6})$	85.35	$(2^{0},2^{-6})$	**85.65**	$(2^{4},2^{-6})$	**85.59**
iris0(0.5000)	$(2^{-2},10)$	100	$(2^{2},10)$	100	$(2^{2},10)$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100
pima(0.5350)	$(2^{32},30)$	70.10	$(2^{14},20)$	74.74	$(2^{12},40)$	71.51	$(2^{0},2^{2})$	71.16	$(2^{8},2^{2})$	**75.58**	$(2^{10},2^{0})$	72.52
wisconsin(0.5380)	$(2^{34},50)$	96.32	$(2^{34},60)$	97.07	$(2^{10},270)$	96.97	$(2^{-2},2^{2})$	97.18	$(2^{8},2^{4})$	**97.70**	$(2^{6},2^{2})$	97.18
glass1(0.5405)	$(2^{16},440)$	78.36	$(2^{22},900)$	78.31	$(2^{22},240)$	79.32	$(2^{-18},2^{-6})$	77.48	$(2^{2},2^{-4})$	**80.35**	$(2^{-18},2^{-6})$	77.48
Colon(Gene Sel)(0.6667)	$(2^{-12},940)$	85.28	$(2^{-8},220)$	83.12	$(2^{-6},320)$	85.28	$(2^{-18},2^{8})$	85.28	$(2^{6},2^{6})$	83.12	$(2^{-18},2^{8})$	85.28
banana(0.8605)	$(2^{24},50)$	88.98	$(2^{34},50)$	89.13	$(2^{26},90)$	89.04	$(2^{2},2^{0})$	**89.33**	$(2^{12},2^{0})$	89.26	$(2^{8},2^{0})$	88.87

node function $h()$ might be known or unknown. That is to say, to map the original data into the feature space, a large variety of hidden node functions $h()$ or kernel functions (if $h()$ is unknown) are free to select. (2) ROWLS only considered binary classification problems. While WELM is able to tackle multiclass problems directly. (3) The least square solution in WELM is actually an analytical solution, which requires limited and fixed steps. While ROWLS tuned and obtained the solution in an iterative manner, which is time consuming compared with analytical solutions. However, an advantage of ROWLS is that, during iterative training only a subset of data samples are selected to form the regressors (kernel functions), which is particularly beneficial for large scale dataset.

5. Performance evaluation

5.1. Data specification

The weighted extreme learning machine (ELM) was proposed based on unweighted ELM in the previous section to cope with data with a large variety of imbalanced class distribution, represented as in Eqs. (13,14). To verify the theoretical analysis between unweighted ELM and weighted ELM (with two weighting schemes defined for both binary and multiclass classifications), 46 binary datasets and 3 multiclass datasets are tested in the experiments. The result is averaged over 10 runs.

To quantitatively measure the imbalance degree of a dataset, the imbalance ratio (IR) is defined:

$$\text{Binary}: \quad \text{IR}=\frac{\#(+1)}{\#(-1)}$$

$$\text{Multiclass}: \quad \text{IR}=\frac{\text{Min}\#(t_i)}{\text{Max}\#(t_i)}, \quad i=1,\ldots,m \tag{28}$$

Most of the binary datasets with 5-folder cross validations are downloaded online[1] as displayed in Table 1 [23]. As shown in Table 1, the imbalance ratio of the dataset can be as low as 0.0078, and the highest imbalance ratio happens to be 0.5380. Four binary datasets and three multiclass datasets from UCI [24] are displayed in Table 2 where the training and testing parts cannot be shuffled. The imbalance ratio of datasets from UCI varies from 0.3306 to 0.8605.

The attributes of the datasets are normalized into $[-1,1]$.

5.2. Parameter settings

In ELM theory, the feature mapping can be almost any nonlinear continuous piecewise function [2], or even in the form of a kernel [6]. In the experiments, we test the algorithms on Sigmoid additive node and Gaussian kernel, which are popular choices by researchers. There are two parameters to tune for ELMs with sigmoid additive node $G(\mathbf{a},b,\mathbf{x})=1/(1+\exp(-(\mathbf{a}\cdot\mathbf{x}+b)))$: the trade-off constant C and the number of hidden nodes L. A grid search of C on $\{2^{-18}, 2^{-16},\ldots, 2^{48}, 2^{50}\}$ and L on $\{10,20,\ldots,990,1000\}$ is conducted in seek of the optimal result. ELMs with Gaussian Kernel, which is expressed as $K(\mathbf{x},\mathbf{x}_i)=\exp(-\gamma|\mathbf{x}-\mathbf{x}_i|^2)$, consist of two user specified parameters, the trade-off constant C and the kernel parameter γ. C is searched in a wide range $\{2^{-18}, 2^{-16},\ldots, 2^{48},2^{50}\}$ while γ is searched in the range of $\{2^{-18}, 2^{-16},\ldots, 2^{18}, 2^{20}\}$.

The effect of user specified parameters on classification performance is shown in Fig. 4. It is obvious that the performance in terms of G-mean is not sensitive to the number of hidden nodes L in Sigmoid node. When it comes to the Gaussian kernel, it is

[1] http://sci2s.ugr.es/keel/study.php?cod=24

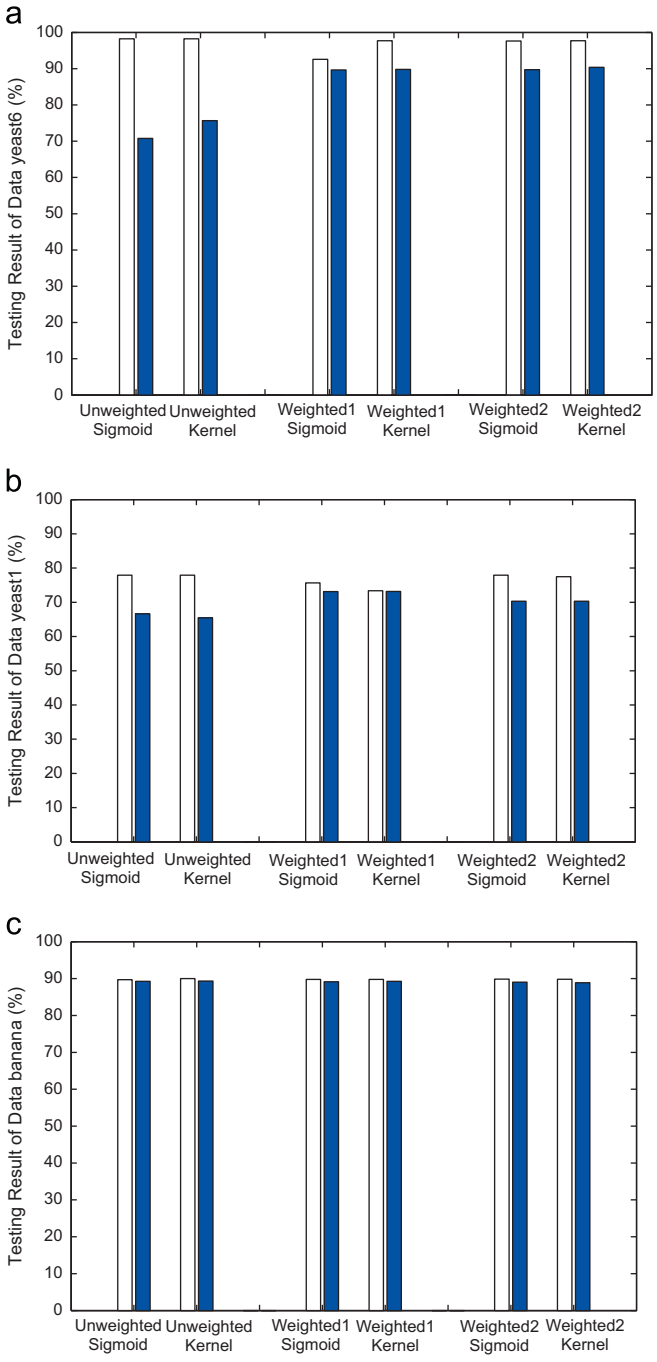

Fig. 5. Display of testing result in terms of accuracy (white bar) and G-mean (blue bar), using unweighted ELM, weighted ELM **W**1 and weighted ELM **W**2, with Sigmoid node or Gaussian kernel: (a) yeast6 with IR 0.0243; (b) yeast1 with IR 0.4064 and (c) banana with IR 0.8605. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

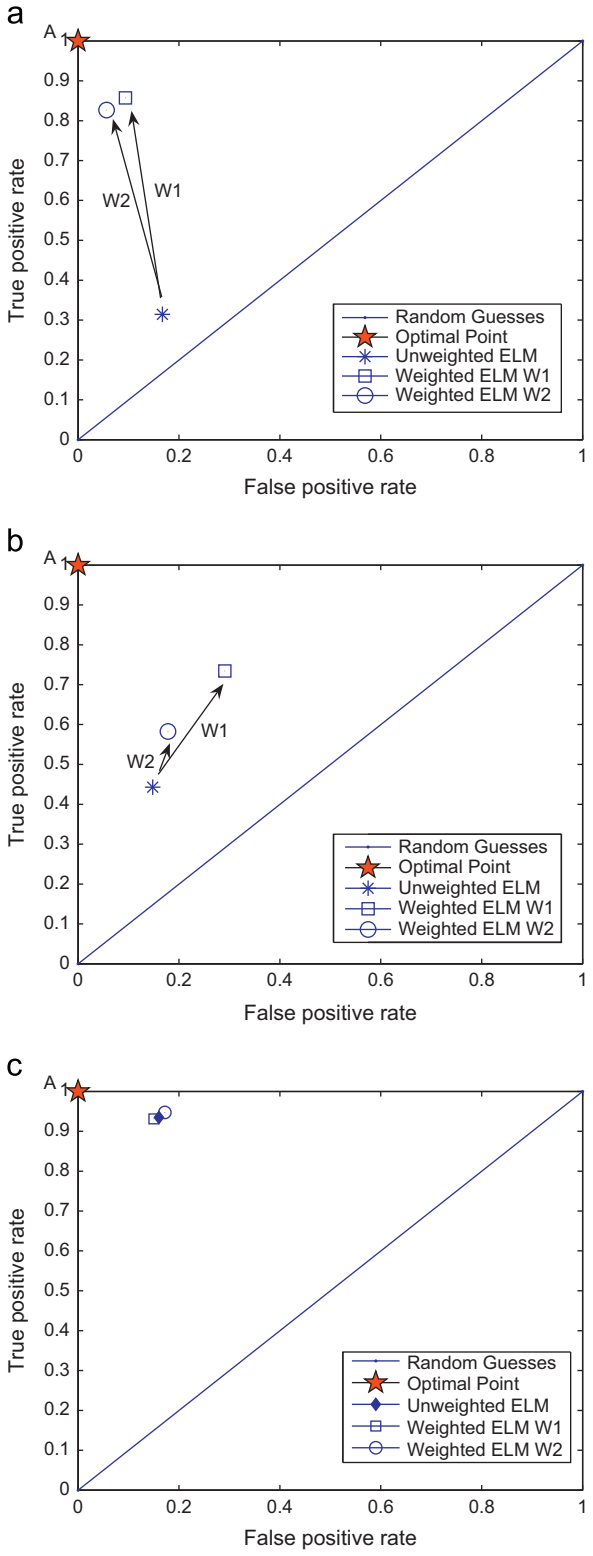

Fig. 6. ROC graph of three classifiers using Sigmoid node on three datasets with different imbalance ratios: (a) yeast6 with IR 0.0243; (b) yeast1 with IR 0.4064 and (c) banana with IR 0.8605.

found that the performance is sensitive to both parameters C and γ, which is consistent with the findings in [5,6].

It can be easily spotted from Fig. 4(c) that when C is small, usually below 2^0 in the experiments, very poor generalization performance is expected. In most of the cases G-mean value drops to as low as 0 when small C is assigned with a network of very few hidden nodes. The reason of the significant performance loss can be traced back to the optimization formula (9). When C is assigned a small value, the model of users' interest is mainly focusing on maximization of the marginal distance, which implies that the cumulative errors of minority class are negligible comparing with the large amount of majority class. An extreme case might happen that all the minority class is classified as the majority class, thus to cause the poor performance in terms of G-mean.

It can be also observed that the performance is decreasing when L is increasing given quite a large C, say 2^{50}, which can

be viewed as an overfitting case. With reasonable value of C, the generalization performance of the proposed method using Sigmoid node is insensitive to number of hidden nodes. Therefore, as the tradition of ELM, for the sake of convenience one can fix L as a large value and tune only C in Sigmoid node without affecting the performance.

5.3. Experiment results

5.3.1. Sigmoid node vs. Gaussian kernel

By comparing Tables 3 and 4 or observed from Fig. 5, the results of ELM with Sigmoid node or Gaussian kernel are quite similar. In a few cases, one is slightly better than another while in some other cases slightly worse. And it is encouraged to explore more efficient nodes or kernels in the future work.

5.3.2. Unweighted ELM vs. weighted ELM

From Fig. 5, it can be seen that, for most datasets, after applying weighting scheme in ELM, G-mean value increases significantly in the cost of relatively slight decreasing in the overall accuracy. That is to say, the separating boundary is moved towards the majority class so that additional minority samples are correctly classified, but slightly more majority samples are misclassified. This trade-off phenomenon is visible especially when imbalance degree is high. And in the case of *banana* (Fig. 5(c)) where the classes are quite balanced, the results of unweighted ELM and weighted ELM are more or less. Therefore, the proposed weighted ELM is applicable to not only the imbalanced datasets, but also the well balanced datasets. In fact, the proposed weighted ELM can be viewed as the general solution that is robust to data with different imbalance degrees.

5.3.3. Weighted scheme 1 vs. weighted scheme 2

Fig. 6 presents the ROC graph for unweighted ELM, weighted ELM with weighting scheme **W**1 and **W**2 on three datasets with imbalance ratio in an ascending order. Each point represents one classifier and provides the accuracy information of both positive class and negative class.

In extremely imbalanced case (Fig. 6(a)), as predicted the true positive rate is greatly increased after applying the weighting scheme in ELM. And the true negative rate is slightly increased as well which happens in some of the cases and the author believes it is closely related to the data complexity. The results of weighted ELM **W**1 and **W**2 are close. However, compared to weighted ELM **W**1, weighted ELM **W**2 has slightly better true negative rate and slightly worse true positive rate, which is in the middle of unweighted ELM and weighted ELM W1 (maybe more close to the latter).

Fig. 6(b) is a typical case of the boundary movement theory analyzed in the last section. The true positive rate is increased in the compromise of decrease in true negative rate for both weighted ELMs. The weighting scheme **W**1 pushes the boundary a little much towards the majority side so that additional minority samples are correctly classified but more majority samples are misclassified.

Fig. 6(c) shows a case with well balanced class distribution, where the results of three classifiers are found quite close. That is to say, the proposed weighted ELM is able to handle well balanced data, whichever weighting scheme might be used.

In a word, observed also from Tables 3 and 4, the weighting scheme **W**2 is supposed to alleviate the compromise in majority class caused by boundary movement, while attempt to maintain the good performance in minority class than unweighted ELM. That is to say, it is the trade-off of unweighted ELM and weighted ELM W1. The choice of weighting schemes are up to the users' requirements.

5.3.4. Statistical test result

To further compare the results between ELM with Sigmoid hidden nodes or Gaussian kernel, with weighting scheme 1 or weighting scheme 2, the pairwise t-test is used as shown in Table 6. Usually a small p-value indicates the fact that, one algorithm is significant better than the other one. The common threshold is set as 0.05 or 0.01. Therefore, it is obvious that the performance of original ELM in terms of G-mean is significantly improved after weighting scheme is applied.

5.3.5. Data complexity vs. imbalance ratio

In the literature of imbalanced learning [7,25], an interesting point was mentioned again and again: the data complexity plays an more important role in the performance than the imbalance ratio. That is to say, the effect of imbalance ratio on easily separable data distribution is minor. On the contrary, complex data distribution may render the classification performance easily affected by the imbalance ratio.

Although the topic of data complexity is out of the scope of this paper, this point of view is verified with the proposed weighted ELM on datasets with wide range of imbalance ratio. Datasets *glass-0-1-6_vs_5*, *page-blocks1*, *shuttle-C0_vs_C4*, *glass6*, *segment0*, *vehicle0*, *vehicle2*, *glass0*, *iris0*, *pima*, *wisconsin* and *banana* are typical datasets with minor difference with/without weighting. But the significant improvement in terms of G-mean for the rest datasets are observed after the proposed weighting scheme is applied.

Table 5
Performance result of multiclass problems.

	Datasets (IR)											
	Sigmoid node						Gaussian kernel					
	Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2		Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2	
	(C,L)	Testing result (%)	(C,L)	Testing result (%)	(C,L)	Testing result (%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)
G-mean												
DNA(0.4415)	$(2^{-6},980)$	94.14	$(2^{2},940)$	94.10	$(2^{2},1000)$	93.93	$(2^{6},2^{6})$	96.10	$(2^{16},2^{6})$	96.10	$(2^{16},2^{6})$	96.10
Satimage(0.3871)	$(2^{8},780)$	87.64	$(2^{36},980)$	89.56	$(2^{14},840)$	89.03	$(2^{5},2^{0})$	89.34	$(2^{10},2^{0})$	**90.62**	$(2^{12},2^{-1})$	89.24
USPS(0.4733)	$(2^{28},950)$	96.39	$(2^{28},950)$	96.36	$(2^{42},750)$	96.28	$(2^{5},2^{8})$	**98.64**	$(2^{15},2^{7})$	98.33	$(2^{15},2^{7})$	98.33
Accuracy												
DNA(0.4415)	$(2^{-6},640)$	94.35	$(2^{4},960)$	93.09	$(2^{4},1000)$	94.10	$(2^{6},2^{6})$	96.29	$(2^{20},2^{15})$	96.04	$(2^{16},2^{6})$	96.29
Satimage(0.3871)	$(2^{8},780)$	90.45	$(2^{36},980)$	90.00	$(2^{14},900)$	90.40	$(2^{4},2^{-2})$	92.35	$(2^{10},2^{0})$	91.25	$(2^{12},2^{0})$	91.65
USPS(0.4733)	$(2^{28},950)$	97.11	$(2^{28},950)$	96.96	$(2^{22},1000)$	96.76	$(2^{4},2^{8})$	98.90	$(2^{16},2^{7})$	98.60	$(2^{16},2^{7})$	98.60

5.3.6. Multiclass imbalanced learning

Finally we would like to note that, so far although main focus in this paper is given to the binary classification, multiclass imbalanced learning is of equal importance. Therefore, the weighted ELM is naturally extended to multiclass problems with two weighting schemes as in binary problems. A couple of multiclass datasets from UCI are tested in the experiments, see Table 5. Two weighting schemes **W**1 (23) and **W**2 (24) are proposed for multiclass problems as well.

The same observations of binary classifiers apply to multiclass classifiers. The proposed weighted ELM is verified to be efficient on the multiclass classification problems as well.

5.3.7. Performance result in terms of accuracy

Performance result in terms of accuracy of all the binary problems and multiclass problems is shown in Tables 7, 8 and 5 for interested readers.

Table 6
p-Value of the *t*-test for pairwise comparisons between the algorithms over binary problems.

p-Value	Sigmoid	Sigmoid+W1	Sigmoid+W2	Gaussian	Gaussian+W1
Sigmoid					
Sigmoid+W1	8.8054e−008				
Sigmoid+W2	1.1444e−008	0.0306			
Gaussian	**0.0674**	1.2105e−006	1.5216e−006		
Gaussian+W1	8.0126e−009	**0.0478**	6.9523e−004	1.7090e−007	
Gaussian+W2	1.6047e−008	**0.0688**	**0.4570**	2.5454e−007	1.6146e−004

Table 7
Performance result in terms of accuracy for binary problems with imbalance ratio $\in(0,0.2)$.

Accuracy	Data											
	Sigmoid node						Gaussian kernel					
	Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2		Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2	
	(C,L)	Testing result (%)	(C,L)	Testing result (%)	(C,L)	Testing Result(%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)
abalone19	$(2^{-18},10)$	99.23	$(2^{40},980)$	93.56	$(2^{-18},10)$	99.23	$(2^{-16},2^{-4})$	99.23	$(2^{-16},2^{-16})$	98.80	$(2^{-16},2^{4})$	99.23
yeast6	$(2^{16},400)$	98.25	$(2^{32},820)$	92.59	$(2^{-18},20)$	97.64	$(2^{0},2^{-4})$	98.25	$(2^{18},2^{-8})$	97.71	$(2^{18},2^{-8})$	97.71
yeast5	$(2^{20},860)$	97.91	$(2^{-12},40)$	96.63	$(2^{2},60)$	97.51	$(2^{-2},2^{-6})$	98.59	$(2^{14},2^{-8})$	98.38	$(2^{14},2^{-8})$	98.38
yeast-1-2-8-9_vs_7	$(2^{16},210)$	97.05	$(2^{-12},120)$	88.26	$(2^{0},710)$	96.94	$(2^{4},2^{-4})$	97.04	$(2^{12},2^{-8})$	95.67	$(2^{-18},2^{-2})$	96.83
yeast4	$(2^{26},40)$	96.90	$(2^{42},700)$	90.43	$(2^{-8},960)$	96.63	$(2^{-2},2^{-4})$	96.97	$(2^{14},2^{-8})$	96.16	$(2^{0},2^{0})$	96.63
shuttle-C2_vs_C4	$(2^{32},100)$	99.23	$(2^{31},10)$	100	$(2^{8},10)$	100	$(2^{-18},2^{-12})$	99.23	$(2^{-18},2^{0})$	100	$(2^{-18},2^{-12})$	99.23
glass5	$(2^{12},90)$	98.14	$(2^{18},160)$	98.14	$(2^{16},220)$	98.14	$(2^{4},2^{0})$	98.61	$(2^{20},2^{0})$	98.14	$(2^{12},2^{0})$	98.14
yeast-2_vs_8	$(2^{-2},160)$	97.92	$(2^{4},60)$	98.13	$(2^{6},120)$	98.13	$(2^{0},2^{0})$	97.92	$(2^{2},2^{-2})$	97.93	$(2^{4},2^{-2})$	98.13
yeast-1-4-5-8_vs_7	$(2^{-18},10)$	95.67	$(2^{-6},50)$	89.91	$(2^{-18},10)$	95.67	$(2^{0},2^{-4})$	95.82	$(2^{12},2^{-8})$	94.81	$(2^{-18},2^{-2})$	95.67
glass-0-1-6_vs_5	$(2^{34},30)$	98.36	$(2^{10},80)$	97.84	$(2^{22},560)$	98.36	$(2^{-18},2^{-12})$	97.30	$(2^{14},2^{0})$	97.84	$(2^{16},2^{0})$	97.84
abalone9_18	$(2^{12},640)$	96.58	$(2^{-18},80)$	93.68	$(2^{50},10)$	95.73	$(2^{4},2^{-4})$	96.75	$(2^{14},2^{-4})$	95.72	$(2^{16},2^{-4})$	95.89
page-blocks1	$(2^{10},440)$	99.57	$(2^{16},300)$	98.94	$(2^{18},140)$	98.94	$(2^{2},2^{-2})$	99.79	$(2^{10},2^{-2})$	99.79	$(2^{10},2^{-2})$	99.79
glass4	$(2^{8},60)$	96.74	$(2^{30},20)$	95.81	$(2^{24},30)$	96.74	$(2^{-4},2^{-2})$	96.74	$(2^{10},2^{-2})$	96.28	$(2^{10},2^{-2})$	96.28
ecoli4	$(2^{38},30)$	99.11	$(2^{28},90)$	97.32	$(2^{2},110)$	98.21	$(2^{-2},2^{-4})$	98.81	$(2^{4},2^{-10})$	97.92	$(2^{-18},2^{-10})$	97.92
yeast-1_vs_7	$(2^{26},100)$	95.42	$(2^{-16},90)$	92.81	$(2^{0},780)$	94.34	$(2^{12},2^{0})$	94.99	$(2^{14},2^{-8})$	92.16	$(2^{2},2^{0})$	93.90
shuttle-C0_vs_C4	$(2^{14},10)$	100	$(2^{38},10)$	100	$(2^{28},10)$	100	$(2^{14},2^{0})$	100	$(2^{26},2^{0})$	100	$(2^{26},2^{0})$	100
glass2	$(2^{-18},10)$	92.06	$(2^{18},630)$	87.81	$(2^{-18},10)$	92.06	$(2^{-18},2^{-18})$	92.06	$(2^{-18},2^{-18})$	92.06	$(2^{-18},2^{-18})$	92.06
glass-0-1-6_vs_2	$(2^{-18},10)$	91.16	$(2^{14},560)$	87.49	$(2^{-18},10)$	91.16	$(2^{-18},2^{-18})$	91.16	$(2^{-18},2^{-18})$	91.16	$(2^{-18},2^{-18})$	91.16
vowel0	$(2^{38},100)$	100	$(2^{40},100)$	100	$(2^{46},90)$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100
yeast-0-5-6-7-9_vs_4	$(2^{12},130)$	93.37	$(2^{22},80)$	87.30	$(2^{12},130)$	92.05	$(2^{18},2^{6})$	93.37	$(2^{16},2^{-8})$	90.91	$(2^{-4},2^{-2})$	92.05
yeast-2_vs_4	$(2^{40},40)$	96.31	$(2^{30},200)$	96.11	$(2^{0},880)$	96.89	$(2^{2},2^{-2})$	96.50	$(2^{12},2^{-8})$	95.72	$(2^{2},2^{-2})$	95.92
page-blocks0	$(2^{30},890)$	96.86	$(2^{38},500)$	95.45	$(2^{38},610)$	96.05	$(2^{16},2^{0})$	96.97	$(2^{16},2^{-8})$	95.98	$(2^{16},2^{-4})$	96.16
ecoli3	$(2^{26},20)$	93.15	$(2^{30},10)$	89.28	$(2^{24},20)$	91.37	$(2^{-6},2^{-4})$	93.74	$(2^{10},2^{-4})$	91.08	$(2^{10},2^{-4})$	91.68
yeast3	$(2^{16},220)$	94.61	$(2^{24},160)$	92.99	$(2^{2},720)$	94.68	$(2^{0},2^{-2})$	94.88	$(2^{10},2^{-8})$	94.14	$(2^{-2},2^{-4})$	95.22
glass6	$(2^{42},50)$	97.67	$(2^{50},50)$	97.67	$(2^{26},30)$	97.67	$(2^{8},2^{2})$	97.67	$(2^{16},2^{6})$	97.67	$(2^{12},2^{4})$	97.67
segment0	$(2^{8},720)$	99.78	$(2^{16},820)$	99.83	$(2^{18},260)$	99.78	$(2^{4},2^{0})$	99.74	$(2^{48},2^{20})$	99.87	$(2^{44},2^{20})$	99.91
ecoli2	$(2^{18},40)$	95.54	$(2^{24},180)$	94.94	$(2^{16},170)$	95.84	$(2^{-18},2^{-6})$	97.03	$(2^{8},2^{-6})$	95.54	$(2^{8},2^{-2})$	95.83
new-thyroid1	$(2^{14},170)$	99.07	$(2^{18},30)$	99.04	$(2^{16},80)$	100	$(2^{10},2^{-4})$	99.07	$(2^{14},2^{0})$	99.54	$(2^{6},2^{0})$	99.54
new-thyroid2	$(2^{18},40)$	98.61	$(2^{16},290)$	99.54	$(2^{12},400)$	100	$(2^{2},2^{-4})$	99.07	$(2^{12},2^{-4})$	99.54	$(2^{14},2^{0})$	99.54

Table 8
Performance result in terms of accuracy for binary problems with imbalance ratio $\in(0.2,1)$.

Accuracy	Data											
	Sigmoid node						Gaussian kernel					
	Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2		Unweighted ELM		Weighted ELM **W**1		Weighted ELM **W**2	
	(C,L)	Testing result (%)	(C,L)	Testing result (%)	(C,L)	Testing Result(%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)	(C,γ)	Testing result (%)
ecoli1	$(2^{14},190)$	92.57	$(2^{32},580)$	89.89	$(2^{26},20)$	91.37	$(2^{6},2^{-2})$	92.57	$(2^{14},2^{-2})$	90.19	$(2^{12},2^{-2})$	91.38
glass-0-1-2-3_vs_4-5-6	$(2^{8},140)$	94.86	$(2^{10},160)$	96.27	$(2^{12},70)$	96.27	$(2^{16},2^{0})$	95.33	$(2^{-18},2^{-4})$	96.25	$(2^{-18},2^{-4})$	96.25
vehicle0	$(2^{8},610)$	99.53	$(2^{16},540)$	99.41	$(2^{16},920)$	99.53	$(2^{12},2^{4})$	99.53	$(2^{42},2^{12})$	99.29	$(2^{32},2^{8})$	99.41
adult	$(2^{-6},960)$	84.66	$(2^{-8},200)$	82.10	$(2^{6},460)$	83.41	$(2^{0},2^{5})$	84.79	$(2^{15},2^{0})$	81.47	$(2^{10},2^{5})$	83.54
vehicle3	$(2^{6},440)$	86.41	$(2^{16},730)$	85.22	$(2^{16},480)$	85.22	$(2^{10},2^{4})$	86.05	$(2^{24},2^{6})$	84.28	$(2^{24},2^{8})$	85.11
vehicle1	$(2^{6},850)$	86.76	$(2^{16},520)$	85.58	$(2^{14},660)$	86.41	$(2^{18},2^{8})$	86.52	$(2^{18},2^{4})$	85.34	$(2^{26},2^{8})$	85.81
vehicle2	$(2^{8},710)$	99.41	$(2^{16},690)$	99.53	$(2^{20},450)$	99.53	$(2^{14},2^{4})$	99.41	$(2^{20},2^{2})$	99.41	$(2^{22},2^{4})$	99.53
haberman	$(2^{8},80)$	75.49	$(2^{-14},430)$	75.16	$(2^{4},180)$	76.79	$(2^{12},2^{6})$	75.49	$(2^{12},2^{6})$	72.87	$(2^{14},2^{10})$	75.81
yeast1	$(2^{14},160)$	77.90	$(2^{-10},340)$	75.67	$(2^{22},420)$	77.90	$(2^{4},2^{0})$	77.90	$(2^{12},2^{-4})$	73.38	$(2^{14},2^{0})$	77.49
Leukemia(Gene Sel)	$(2^{28},10)$	100	$(2^{-2},10)$	100	$(2^{-14},10)$	100	$(2^{-18},2^{-2})$	100	$(2^{-18},2^{-2})$	100	$(2^{-18},2^{-2})$	100
glass0	$(2^{12},960)$	83.18	$(2^{24},990)$	79.45	$(2^{20},980)$	82.25	$(2^{0},2^{-6})$	85.98	$(2^{4},2^{-10})$	85.50	$(2^{8},2^{-6})$	85.52
iris0	$(2^{-2},10)$	100	$(2^{4},10)$	100	$(2^{2},10)$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100	$(2^{-18},2^{-10})$	100
glass1	$(2^{14},700)$	79.03	$(2^{22},300)$	78.99	$(2^{22},240)$	79.96	$(2^{-18},2^{-6})$	81.82	$(2^{-18},2^{-12})$	81.77	$(2^{-18},2^{-6})$	81.82
pima	$(2^{0},480)$	78.25	$(2^{38},20)$	76.56	$(2^{42},30)$	78.25	$(2^{10},2^{8})$	77.99	$(2^{8},2^{2})$	76.04	$(2^{18},2^{6})$	77.60
wisconsin	$(2^{10},50)$	97.66	$(2^{32},60)$	97.66	$(2^{42},30)$	93.37	$(2^{-2},2^{2})$	97.22	$(2^{8},2^{4})$	97.51	$(2^{6},2^{2})$	97.22
colon	$(2^{28},210)$	93.75	$(2^{20},120)$	90.63	$(2^{34},280)$	90.63	$(2^{4},2^{16})$	87.50	$(2^{8},2^{14})$	81.25	$(2^{8},2^{16})$	87.50
Colon(Gene Sel)	$(2^{-14},440)$	93.75	$(2^{-4},10)$	87.50	$(2^{-8},330)$	93.75	$(2^{-18},2^{8})$	87.50	$(2^{6},2^{6})$	84.38	$(2^{-18},2^{8})$	87.50
banana	$(2^{16},500)$	89.71	$(2^{32},160)$	89.80	$(2^{30},60)$	89.84	$(2^{2},2^{0})$	90.00	$(2^{12},2^{0})$	89.80	$(2^{6},2^{-2})$	89.82

6. Conclusions

This paper proposes a weighted ELM based on original unweighted ELM for binary and multiclass classification tasks. To benefit from original unweighted ELM, the proposed method is simple in theory and implementation; wide types of feature mapping or kernels are available as options; comparable or better generalization performance is achievable compared to the conventional machine learning techniques. On the other hand, by incorporating the information of imbalance class distribution, the proposed method is able to deal with data with imbalanced class distribution. Two weighting schemes are tested in the experiments on datasets with various imbalance ratios, which demonstrate the better performance of weighted ELM compared with unweighted ELM. The future work may include applying the proposed weighted ELM into the real world applications, with large variety in class distribution.

Acknowledgments

Y. Chen's work is supported in part by Natural Science Foundation of China under Grant No. 61173066 and Beijing Natural Science Foundation under Grant No. 4112056.

References

[1] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1–3) (2006) 489–501.
[2] G.-B. Huang, L. Chen, C.-K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, IEEE Trans. Neural Networks 17 (4) (2006) 879–892.
[3] G.-B. Huang, L. Chen, Convex incremental extreme learning machine, Neurocomputing 70 (2007) 3056–3062.
[4] G.-B. Huang, L. Chen, Enhanced random search based incremental extreme learning machine, Neurocomputing 71 (2008) 3460–3468.
[5] G.-B. Huang, X. Ding, H. Zhou, Optimization method based extreme learning machine for classification, Neurocomputing 74 (2010) 155–163.
[6] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multi-class classification, IEEE Trans. Syst. Man Cybern., 42 (2) (2012) 513–529.
[7] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[8] R. Akbani, S. Kwek, N. Japkowicz, Applying support vector machines to imbalanced datasets, in: Proceedings of the 15th European Conference on Machine Learning (ECML), Pisa, Italy, September 20–24, 2004, pp. 39–50.
[9] K.-A. Toh, Deterministic neural classification, Neural Comput. 20 (6) (2008) 1565–1595.
[10] W. Deng, Q. Zheng, L. Chen, Regularized extreme learning machine, in: IEEE Symposium on Computational Intelligence and Data Mining (CIDM 09), March 30–April 2, 2009, pp. 389–395.
[11] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: a new learning scheme of feedforward neural networks, in: Proceedings of International Joint Conference on Neural Networks (IJCNN 2004), vol. 2, Budapest, Hungary, July 25–29, 2004, pp. 985–990.
[12] C.R. Rao, S.K. Mitra, Generalized Inverse of Matrices and its Applications, John Wiley & Sons, Inc., New York, 1971.
[13] D. Serre, Matrices: Theory and Applications, Springer-Verlag, New York, Inc., 2002.
[14] A.E. Hoerl, R.W. Kennard, Ridge regression: biased estimation for nonorthogonal problems, Technometrics 12 (1) (1970) 55–67.
[15] R. Fletcher, Practical Methods of Optimization: Constrained Optimization, vol. 2, Wiley, New York, 1981.
[16] T. Fawcett, An introduction to roc analysis, Pattern Recognition Lett. 27 (June) (2006) 861–874.
[17] W. Zong, G.-B. Huang, Face recognition based on extreme learning machine, Neurocomputing (May) .
[18] W. Zong, H. Zhou, G.-B. Huang, Z. Lin, Face recognition based on kernelized extreme learning machine, in: Proceedings of the Second International Conference on Autonomous and Intelligent Systems, AIS'11, Springer-Verlag, 2011, pp. 263–272.
[19] Y. Lan, Y.C. Soh, G.-B. Huang, Extreme learning machine based bacterial protein subcellular localization prediction, in: Proceedings of the International Joint Conference on Neural Networks, Hong Kong, China, June 1–6, 2008, pp. 1859–1863.
[20] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.

[21] J.A.K. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Process. Lett. 9 (3) (1999) 293–300.
[22] X. Hong, S. Chen, C.J. Harris, A kernel-based two-class classifier for imbalanced data sets, IEEE Trans. Neural Networks 18 (1) (2007) 28–41.
[23] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Multiple-Valued Logic Soft Comput. 17 (2–3) (2011) 255–287.
[24] A. Frank, A. Asuncion, UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences, 2010. 〈http://archive.ics.uci.edu/ml〉.
[25] J. Yuan, J. Li, B. Zhang, Learning concepts from large scale imbalanced data sets using support cluster machines, in: Proceedings of the 14th Annual ACM International Conference on Multimedia, Santa Barbara, California, USA, October 23–27, 2006, pp. 441–450.

Weiwei Zong received the B.Eng. degree from Central South University, China, in 2008. She is currently a Ph.D. student with School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Her research interests include neural networks, extreme learning machines, support vector machines and pattern recognition.

Guang-Bin Huang (M'98-SM'04) received the B.Sc degree in applied mathematics and M.Eng degree in computer engineering from Northeastern University, Shenyang, China, in 1991 and 1994, respectively, and the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore, in 1999. During undergraduate period, he also concurrently studied in the Applied Mathematics Department and Wireless Communication Department, Northeastern University, China. From June 1998 to May 2001, he was a Research Fellow with Singapore Institute of Manufacturing Technology (formerly known as Gintic Institute of Manufacturing Technology), where he has led/implemented several key industrial projects (e.g., Chief Designer and Technical Leader of Singapore Changi Airport Cargo Terminal Upgrading Project, etc). Since May 2001, he has been an Assistant Professor and Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His current research interests include machine learning, computational intelligence, and extreme learning machines. Dr. Huang serves as an Associate Editor of Neurocomputing and the IEEE Transactions on Systems, Man, and Cybernetics—Part B.

Yiqiang Chen is a professor and Vice Director of pervasive computing research center of Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS). He received the BASc and MA degrees from the Xiangtan University in 1996 and 1999, respectively, and the PhD degree from the ICT, CAS in 2002. He was a visiting scholar researcher in the Department of Computer Science at the Hong Kong University of Science and Technology (HKUST) and Nanyang Technological University (NTU) in 2004 and 2010 respectively. His research interests include pervasive computing and intelligent human computer interaction. He is a member of IEEE, ACM and CCF. He has published about 90 papers on International Journal IEEE TKDE, IEEE TCSVT, IJCI, IEEE TNN, IJIIS, PR and NeuroComputing etc, also International conference ACM Multimedia, IJCAI, Ubicomp, PerCom, Siggraph Asia, ICME, PCM etc. He has more than 30 patents and 16 have get authorization. One of his works named Chinese Sign Language Synthesis system now has been used in 300 deaf middle schools all around china. And also has been used in 2008 Beijing Olympic Games. He was award the National Science and Technology Award (Second Level) the top award in China Science area on 2004 and has been selected as new star scientist of Beijing on 2005.