Neural Processing Letters (2020) 52:1927–1944
https://doi.org/10.1007/s11063-020-10282-z

Parameter-Free Extreme Learning Machine for Imbalanced Classification

Li Li[1] · Kaiyi Zhao[1] · Ruizhi Sun[1,2] · Jiangzhang Gan[3] · Gang Yuan[1] · Tong Liu[3]

Published online: 17 June 2020

Abstract
Imbalanced data distribution is a common problem in classification situations, that is the number of samples in different categories varies greatly, thus increasing the classification difficulty. Although many methods have been used for the imbalanced data classification, there are still problems with low classification accuracy in minority class and adding additional parameter settings. In order to increase minority classification accuracy in imbalanced problem, this paper proposes a parameter-free weighting learning mechanism based on extreme learning machine and sample loss values to balance the number of samples in each training step. The proposed method mainly includes two aspects: the sample weight learning process based on the sample losses; the sample selection process and weight update process according to the constraint function and iterations. Experimental results on twelve datasets from the KEEL repository show that the proposed method could achieve more balanced and accurate results than other compared methods in this work.

Keywords Parameter-free · Extreme learning machine · Class imbalance problem · G-mean

1 Introduction

Class imbalance learning (CIL) has attached a lot of attention in classification problems, which generally refers to the number of samples in some classes is significantly more than others [1]. In the CIL, the class that has more samples is called the majority class while the other is called minority class. This poses a difficulty for traditional classification methods for balanced data, as they will be biased towards the majority group and hence show very poor classification accuracy on the minority classes [2]. However, compared to the instances of the majority class, the instances of the minority classes are usually more

✉ Ruizhi Sun
sunruizhi@cau.edu.cn

[1] College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China

[2] Scientific Research Base for Integrated Technologies of Precision Agriculture (Animal Husbandry), The Ministry of Agriculture, Beijing 100083, China

[3] School of Natural and Computational Sciences, Massey University, Auckland 0745, New Zealand

important and more valuable in real applications such as disease diagnosis, fraud detection in banking operations, network intrusion detection etc. [3, 4]. For example, in the identification of fraudulent transactions, there are a small number of fraudulent transactions, that is, the vast majority of transactions are normal, and only a few of them are fraudulent transactions, but it is this very few fraudulent transactions that require more attention and identification [5].

In the past few decades, many techniques have been proposed to alleviate the impact of skewed distribution in each category on the classification result [6]. They are mainly divided into two aspects: from the data point of view, the main method is to modify the collection of samples to balance distributions or remove difficult samples, the commonly used techniques are divided into undersampling and oversampling and corresponding improved methods, where refers to some majority class samples are removed (undersampling) or some minority class samples are added (oversampling) [7]. "Undersampling" of the majority class samples refers to remove some samples in the majority classes so that the number of positive and negative examples is close. The most common one is the random undersampling method, that is, some samples are randomly removed from the majority class to form a sample set, but some important information about the majority class will be missing [8]. The oversampling method is to add some samples to minority classes so that the number of positive and negative examples is close, where the most common one is the random oversampling method, that is, randomly select some samples and then copy the selected samples to generate a new sample set. But the disadvantages of oversampling are obvious, for random oversampling, the complexity of model training is increased due to replicate a small number of samples to expand the data set. On the other hand, it is easy to cause overfitting of the model, which makes the classifier's the generalization performance low [9]. In addition, there are many techniques for solving imbalanced data learning at the algorithm level, that are mainly to directly modify existing learning algorithms to alleviate the bias toward majority samples and adapt them to mining data with skewed distributions, where the most commonly used strategy is based on cost-sensitive learning [9–13]. By assigning a higher cost to the more important minority class, cost-sensitive learning can minimize the global cost associated with mistakes. But the disadvantage of cost-sensitivity is that it's difficult to determine the actual values of cost matrix. In addition, the problem of unbalanced data sets can be considered as a one-class learning or an abnormality detection problem [14–17]. Although these methods can reduce the impact of the data skewed distribution on the classification results, they all have some problems to some extent, so the imbalanced data classification remains a challenging research topic.

In recent years, there is a popular feedforward neural network called Extreme Learning Machine (ELM), different from the traditional artificial neural networks, which based on gradient descent methods that all parameters of the network need to be updated and adjusted when iterating, the merit of ELM is the one-step training process and it can achieve much faster learning speed [18]. Otherwise, the input weights and the number of hidden nodes in ELM are randomly assigned, while the output weights can be analytically determined by solving a linear system [19, 20].

Many improved ELMs have been put forward in the past few years aiming at the imbalanced classification problems. To deal with imbalanced data class distribution, many algorithms based on weighted theory are proposed, where each class of samples is given a weight based on the number of samples in that category [21]. Such as, a weighted extreme learning machine scheme based on neutrosophic set theory was present, which can remove the effect of noise and outliers in the classification stage and yield better classification results [22]. A dissimilarity-based method paying more attention on the discriminative

ability of features in the context of class is proposed to deal with the classification of imbalanced data, it can remove the useless and redundant features by feature selection from the given data set [23]. Lu et al. proposed an improved weighted extreme learning machine (IW-ELM), in which a voting based weighting scheme is introduced when assigning appropriate weights adaptively [24]. Li et al. embedded weighted ELM into a modified AdaBoost framework to face the skewed distribution of data, where the distribution weights are updated separately for training samples from different classes to avoid destroying the distribution weights asymmetry [25]. A technique aimed at handling imbalance data by assigning more weight to minority class and less weight to majority class was proposed in paper, that is, the weight of the sample is closely related to the number of samples. The limitation of this technique lies in that it generates weight according to the class distribution of training data, which leads to the lack of finding optimal weight at which good generalization performance could be achieved [26].

In this paper, a unified framework is proposed to tackle binary and multiple classification tasks, which is effective to both balanced and imbalanced data distribution. The proposed method is mainly divided into three steps: (1) initialize the model and sample weights, and select the samples in each class participating in the next training according to the number of samples in the smallest class; (2) assign a weight for each sample according to the loss value of and select the top-ranked samples in each category to participate in the training; (3) update the weights of the samples participating in the training and select samples from majority class, then train the sample weights until the target value converges. Experimental results on twelve datasets from the KEEL repository show that the proposed method could achieve more balanced results than other compared methods under the *G-mean* metric. The main contributions of this paper are as follows:

1. The proposed method not only balances the number of samples in the training process, but also assigns different weights to each sample according to the loss value of the samples during the training process, and the weight will be updated according to the update of the classifier.
2. Different from the traditional robust classification algorithm [27, 28], this method does not introduce other parameters, but establishes a parameter-free weight determination mechanism, so there is no need to manually determine other parameters.

The remainder of this paper is organized as follows. Section 2 outlined the related work of ELM and its improvements. Section 3 presented the proposed method in this work. Section 4 described the evaluation experiments of the proposed method and its analysis. Section 5 presented our conclusion and put forward the promising directions related to this topic.

2 Preliminaries

2.1 Notations

In this paper, we use boldface uppercase letters, boldface lowercase letters, and normal italic letters, respectively, to denote matrices, vectors, and scalars. We donate $\mathbf{X}$ as the feature matrix of samples, its element in the ith row and jth column is denoted as $x_{i,j}$. Besides,

we denote the Frobenius norm of a matrix $\mathbf{X}$ as $\|\mathbf{X}_F\| = \sqrt{\sum_{i,j} \mathbf{x}_{i,j}^2}$. We further denote the diagonal transpose operator, the trace operator, and the inverse of a matrix $\mathbf{X}$ as $\mathbf{X}^{\mathbf{T}}$, $tr(\mathbf{X})$ and $\mathbf{X}^{-1}$, respectively. Besides, we donate the feature data as $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \in \mathrm{R}^{N\times n}$ representing n-dimensional samples, and, we further denote the label matrix as $\mathbf{T} = \{\mathbf{t}_i\}_{i=1}^N \in \mathrm{R}^{N\times m}$ that the label matrix has N m-class label.

2.2 Original ELM

In the past decades, the issues of the imbalanced classification have been discussed and reviewed by ELM and its improvements.This section provides a brief review of ELM and its improvements related to imbalanced data classification. In addition, the evaluation metric is important for the classification problems, the description of the evaluation metrics can be found in this section as well.

For a single-layer feedforward neural network (SLFN), for $\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\right]^T \in R^{N\times n}$ consists N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i = \left[x_{i1}, x_{i2}, \ldots, x_{in}\right]^{\mathrm{T}} \in R^n$, $\mathbf{t}_i = \left[t_{i1}, t_{i2}, \ldots, t_{im}\right]^T \in R^m$. The output model of a SLFN with L hidden layer nodes can be expressed as follows:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \tag{2-1}$$

where $\mathbf{H} = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1) \\ . \\ . \\ . \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix}$, $\mathbf{h}(\mathbf{x}) = \left[\mathrm{h}_1(\mathbf{x}), \mathrm{h}_2(\mathbf{x}), \ldots, \mathrm{h}_L(\mathbf{x})\right]$, $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_L]^{\mathrm{T}}$, $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]^{\mathrm{T}}$.

$\mathbf{H}$ is called the hidden layer output matrix of the neural network, the ith column of $\mathbf{H}$ represents the ith hidden node output with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, while the ith row of $\mathbf{H}$ denotes the ith sample output for L hidden neurons. Considering that regularization term is often added to reduce the influence of ill-conditioned matrix and the problem of over-fitting in ELM, to improve the generalization performance, ELM is generally formulated as the following optimization problem when dealing with classification problems:

$$\begin{aligned} &\min_{\boldsymbol{\beta}} \ \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}\mathrm{C}\sum_{i=1}^{N}\|\xi\|_i^2 \\ &\text{s.t.}, \mathrm{h}\left(\mathbf{x}_i\right)\boldsymbol{\beta} = \mathrm{t}_i^{\mathrm{T}} - \xi_i^{\mathrm{T}}, i = 1, 2, \ldots, N \end{aligned} \tag{2-2}$$

where C is a user-specified parameter and provides a compromise between the minimization of the training errors and the maximization of the marginal distance, ξ_i is the error of fitting. The following formula is equivalent to the above formula:

$$\min_{\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|^2 \tag{2-3}$$

By using the Lagrange multiplier and Karush–Kuhn–Tucker (KKT) theorem[], the dual optimization problem can be obtained. Thus, hidden layer's output weight vector $\boldsymbol{\beta}$ can be derived from Eq. (2-3) regarding left pseudo-inverse or right pseudo-inverse.

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{c} + \mathbf{H}^{\mathrm{T}}\mathbf{H}\right)\mathbf{H}^{\mathrm{T}}\mathbf{T}(L \le N) \tag{2-4}$$

$$\boldsymbol{\beta} = \mathbf{H}^{\mathrm{T}}(\frac{\mathbf{I}}{c} + \mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{T}(L > N) \tag{2-5}$$

2.3 Weighted ELM

Considering the effect of complex data distribution, weighted ELM gives the different weight to samples from the majority and minority classes. Weighted ELM defined an $N \times N$ diagonal matrix $\mathbf{W}$ associated with every training sample $\mathbf{x}_i$, to minimize the weighted cumulative error of samples, the objective function mathematically written as:

$$\begin{aligned} &\min_{\boldsymbol{\beta}} \ \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C\mathbf{W}\sum_{i=1}^{N}\|\xi\|_i^2 \\ &s.t. \ \mathrm{h}\left(\mathbf{x_i}\right)\boldsymbol{\beta} = \mathrm{t}_i^{\mathrm{T}} - \xi_i^{\mathrm{T}}, \ \ i = 1, 2, \ldots, N \end{aligned} \tag{2-6}$$

According to the KKT theorem, the solution of $\boldsymbol{\beta}$ can be obtained as follows:

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{H}^{\mathbf{T}}\mathbf{W}\mathbf{H}\right)^{-1}\mathbf{H}^{\mathbf{T}}\mathbf{W}\mathbf{T}(L \le N) \tag{2-7}$$

$$\boldsymbol{\beta} = \mathbf{H}^{\mathbf{T}}\left(\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{W}\mathbf{H}\mathbf{H}^{\mathbf{T}}\right)^{-1}\mathbf{W}\mathbf{T}(\mathrm{L} > N) \tag{2-8}$$

In the weighted ELM, the author adopted two different weighting schemes:

$$\boldsymbol{W}_{\boldsymbol{minority}} = \frac{1}{\#(t_i^+)} \text{ and } \boldsymbol{W}_{\boldsymbol{majority}} = \frac{1}{\#(t_i^-)}$$

$$\boldsymbol{W}_{\boldsymbol{minority}} = \frac{0.618}{\#(t_i^+)} \text{ and } \boldsymbol{W}_{\boldsymbol{majority}} = \frac{1}{\#(t_i^-)}$$

The weighted ELM provided a solution for imbalanced data classification by adding the weight matrix to emphasize the impact of minority class while weaken the impact of majority class. Samples from the minority class is assigned with larger weight while the majority class is assigned with smaller weight value. In fact, this method of assigning a weight to each sample is also a kind of cost sensitive learning, where the weight setting mechanism can usually be artificially chosen based on the data distribution information.

2.4 CCR-ELM

An class-specific cost regulation extreme learning machine (CCR-ELM) was presented by [29], it simultaneously considers the effect of the number of the class samples and the dispersion degree of the data, in order to keep the separation boundary to be located

in the ideal position to rebalance the proportion of the classes, CCR-ELM set different parameters to every class, the objective function can be formulated as:

$$\begin{aligned} &\min_{\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}\sum_{d=1}^{D} \mathrm{C}^d \sum_{i=1|t_i=d}^{l_d} \xi_i^2 \\ &\text{s.t.} \mathrm{h}\left(\mathbf{x}_i\right)\boldsymbol{\beta} = \mathrm{t}_i^{\mathrm{T}} - \xi_i^{\mathrm{T}}, \quad i = 1, 2, \ldots, N \end{aligned} \tag{2-9}$$

where C^d represents the regulation cost for misclassification in the performance index specific for dth class and l_d denotes the number of samples of dth class,$\sum_{i=1|t_i=d}^{l_d} \xi_i^2$ is the sum error of dth class. The hidden layer's output weight vector $\boldsymbol{\beta}$ can be derived from Eq. (2-9) regarding left pseudo-inverse or right pseudo-inverse.

$$\boldsymbol{\beta} = \left(\sum_{d=1}^{D} \frac{\mathbf{I}}{C^d} + \mathbf{H}^{\mathbf{T}}\mathbf{H}\right)^{-1} \mathbf{H}^{\mathbf{T}}\mathbf{T}(L \le N) \tag{2-10}$$

$$\boldsymbol{\beta} = \mathbf{H}^{\mathbf{T}}\left(\sum_{d=1}^{D} \frac{\mathbf{I}}{C^d} + \mathbf{H}\mathbf{H}^{\mathbf{T}}\right)^{-1} \mathbf{T}(\mathrm{L} > N) \tag{2-11}$$

By introducing class-specific regulation cost for misclassification of each class in the performance index, CCR-ELM considers not only the effect of the number of class samples, but also the effects of the dispersion degree of the data, thus, it can obtain more satisfactory performance, especially when the kernel based CCR-ELM is used, but the disadvantages of CCR-ELM is obvious, that is, the parameters for CCR-ELM is difficult to determining.

2.5 CS-ELM

A method based on CCR-ELM was proposed to tackle the binary class imbalance problem called CS-ELM [30]. It is different from the CCR-ELM, the CS-ELM does not require assigning any weight to training samples and has the lower computational complexity. The principle of CS-ELM can be abbreviated as follows:

$$\begin{aligned} &\min_{\beta} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}\frac{N_- \times \mathrm{C}}{N_+ + N_-}\left\|\xi_+\right\|^2 + \frac{1}{2}\frac{N_+ \times \mathrm{C}}{N_+ + N_-}\left\|\xi_-\right\|^2 \\ &\text{s.t.} \quad \mathrm{h}\left(\mathbf{x}_i^+\right)\boldsymbol{\beta} = \mathrm{t}_\mathrm{i}^+ - \xi_\mathrm{i}^+, i = 1, 2, \ldots, N_+ \\ &\qquad \mathrm{h}\left(\mathbf{x}_i^-\right)\boldsymbol{\beta} = t_i^- - \xi_i^-, i = 1, 2, \ldots, N_- \end{aligned} \tag{2-12}$$

here $\mathrm{h}\left(\mathbf{x}_i^+\right)$ and $\mathrm{h}\left(\mathbf{x}_i^-\right)$ represents the hidden layer output of positive class and negative class training instances respectively, the t_i^+ and t_i^- denotes the target of positive class and the negative class training samples respectively, while N_+ and N_- denotes the number of positive samples and negative samples. Similar to the above methods,$\boldsymbol{\beta}$ can also be obtained by KKT conditions and differentiation of Eq. (2-12).

$$\boldsymbol{\beta} = \begin{cases} \left[\mathbf{H}_+^{\mathrm{T}}\mathbf{H}_-^{\mathrm{T}}\right]\left(\frac{\mathbf{I}}{C^+} + \frac{\mathbf{I}}{C^-} + \begin{bmatrix} \left(1+\frac{C^+}{C^-}\right)\mathbf{H}_+ \\ \left(1+\frac{C^-}{C^+}\right)\mathbf{H}_- \end{bmatrix} \times \left[\mathbf{H}_+^{\mathrm{T}}\mathbf{H}_-^{\mathrm{T}}\right]\right)^{-1}\begin{bmatrix} \left(\frac{C^+}{C^-}+1\right)\mathbf{T}_+ \\ \left(1+\frac{C^-}{C^+}\right)\mathbf{T}_- \end{bmatrix} (\mathrm{L} > N) \\ \left(\frac{\mathbf{I}}{C^+} + \frac{\mathbf{I}}{C^-} + \left(1+\frac{C^+}{C^-}\right)\mathbf{H}_+^{\mathrm{T}}\mathbf{H}_+ + \left(1+\frac{C^-}{C^+}\right)\mathbf{H}_-^{\mathrm{T}}\mathbf{H}_-\right)\left(\left(1+\frac{C^+}{C^-}\right)\mathbf{H}_+^{\mathrm{T}}\mathbf{T}_+ + \left(1+\frac{C^-}{C^+}\right)\mathbf{H}_-^{\mathrm{T}}\mathbf{T}_-\right) (L \le N) \end{cases} \tag{2-13}$$

The class-specific regularization parameters used for classes can be determined using the class proportion and the value of regularization parameter. The computational complexity of the proposed algorithm is significantly lower than WELM and CCR-ELM, but it can only be used for binary classification.

3 The Proposed Method

In this section, we will introduce a new variant of ELM, designed to handle the imbalanced data classification effectively. Nevertheless, we expect to enhance the outliers insensitiveness of our method, and even hope the weight of outliers to be 0, so a self-weighted imbalanced classification method is proposed. This work does not require to set the weight value of the sample in advance, but determines which samples can participate in training through the sample loss value. In short, the performance of a classifier is mainly affected by the imbalance degree of the sample distributions. Therefore, by adding the constraint of training samples numbers in each class, the impact of the sample number imbalance on the classification result can be weakened. In addition, by assigning weights to samples based on the loss of the sample to the classifier, the accuracy of the classifier can be improved. By continuously updating the sample weights, the purpose of selecting samples and updating the classifier can be achieved.

3.1 Problem Formulation

Considering the different sample sizes of different classes, assuming that the same number of samples are selected in each class during each training, let the training set $\mathbf{X} = \left(\mathbf{x}_1, \ldots, \mathbf{x}_N\right) \in \boldsymbol{R}^N$ is partitioned into b sets according to their target labels, $\boldsymbol{n_j}$ refers to the number of samples in the jth set, there is $\sum_{j=1}^{b} n_j = N$. We select the same number of samples in each category to balance the number of samples. Unlike other weight-based methods, we do not give samples a fixed weight during the training process. Instead, we want the classifier to continuously update the weights of the samples participating in the training process, so we adopt a self-weighting mechanism based on samples loss value, and to avoid adding extra artificial parameters. The mathematical formulation of the proposed method can be written as:

$$\begin{aligned} &\min_{\boldsymbol{\beta},\, \mathbf{V}=[\mathbf{v}_1,\ldots,\mathbf{v}_N]} \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathrm{F}}^2 + \frac{\mathrm{C}}{2}\sum_{i=1}^{N} \mathrm{v}_i \left\|\mathrm{h}\left(\mathrm{x}_i\right)\boldsymbol{\beta} - \mathrm{t}_i\right\|_{\mathrm{F}} \\ &s.t., \left\|\boldsymbol{V}_{I(j)}\right\|_0 = k, \;\; j = 1, 2, \ldots, b \end{aligned} \tag{3-1}$$

where $I(j)=\{i{:}t_i=j\}$ is the index of the sample set with label j, which is $\boldsymbol{V}$ reduced to the indices of $I(j)$. Concretely, the following formula can be got:

$$\begin{aligned} &\min_{\boldsymbol{\beta},\, \mathbf{V}=[\mathbf{v}_1,\ldots,\mathbf{v}_n]} \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathrm{F}}^2 + \frac{\mathrm{C}}{2}\mathbf{V}\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_{\mathrm{F}} \\ &\text{s.t.,} \left\|\mathbf{V}_{\mathrm{I}(j)}\right\|_0 = k, \quad j = 1, 2, \ldots, b \end{aligned} \tag{3-2}$$

where c is a penalty coefficient, a user-defined constant, $\mathrm{v}_i \in [0,1]^{\mathrm{N}}$, $\boldsymbol{V} = \left[\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_N\right]^T$, $\mathbf{H}$ is an $N \times L$ matrix between the input layer and the hidden layer, where L represents the number of nodes in the hidden layer, $\mathbf{T}$ is the actual output of the sample, which is a matrix of $N \times m$, and $\boldsymbol{\beta}$ is the output weight of the hidden layer, which is a matrix of $L \times m$. The value of v_i indicates whether the sample is selected for the next training, if the sample is selected, the value of v_i is 1, otherwise it is 0, k is the number of samples in the smallest category. By introducing the sample weight matrix $\boldsymbol{W}$, the above equation can be equivalent to the following form:

$$\begin{aligned} &\min_{\boldsymbol{\beta},\, \mathbf{V}=[\mathbf{v}_1,\ldots,\mathbf{v}_N]} \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathrm{F}}^2 + \frac{\mathrm{C}}{2}\mathbf{V}\mathbf{W}\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_{\mathbf{F}}^2 \\ &\text{s.t.,} \left\|\mathbf{V}_{\mathrm{I(j)0}}\right\| = k,\ j = 1, 2, \ldots, b \end{aligned} \tag{3-3}$$

where $\mathbf{W}$ is the diagonal matrix, $w_{ii} = \frac{1}{\mathrm{h}(\mathbf{x}_i)\boldsymbol{\beta} - \mathbf{t}_{i2}^2}$, this means that the weight value of a sample depends entirely on its loss to the classifier. It can be known from the definition of $\mathbf{W}$, samples with small values are assigned with larger weight while the samples with large values are assigned with smaller weight. Among them, samples with large loss values can be considered as low confidence samples or outliers [30–32]. In this way, the proposed method do not introduce additional artificial parameters while guaranteeing sensitivity of the classifier to outliers.

3.2 Learning the Value of $\boldsymbol{\beta}$

According to the above introductions, the essence of the problem is still to determine the value of $\boldsymbol{\beta}$. To facilitate the optimization, the formula above can be written as:

$$\begin{aligned} &\min_{\boldsymbol{\beta},\, \mathbf{V}=[\mathbf{v}_1,\ldots,\mathbf{v}_N]} \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathrm{F}}^2 + \frac{\mathrm{C}}{2}\|\mathbf{G}\mathbf{H}\boldsymbol{\beta} - \mathbf{G}\mathbf{T}\|_{\mathbf{F}}^2 \\ &\text{s.t.,} \left\|\mathbf{V}_{\mathrm{I(j)}}\right\|_0 = k,\ j = 1, 2, \ldots, b \end{aligned} \tag{3-4}$$

where $\mathbf{G} = \sqrt{\mathbf{V}\mathbf{W}}$, $\mathbf{V}$, $\mathbf{W}$ are $N \times N$ diagonal matrix respectively, the $\mathbf{W}$ denotes the sample weight, while the $\mathbf{V}$ represents that whether to select a sample. To be more intuitive, let

$$J = \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathrm{F}}^2 + \frac{\mathrm{C}}{2}\|\mathbf{G}\mathbf{H}\boldsymbol{\beta} - \mathbf{G}\mathbf{T}\|_{\mathbf{F}}^2 \tag{3-5}$$

we set the derivative of Eq. (3-5) with respect to $\boldsymbol{\beta}$ to 0, it can obtain the following formula:

$$C(\left(\mathbf{G}\mathbf{H}\right)^{\mathbf{T}}(\mathbf{G}\mathbf{H})\boldsymbol{\beta} - (\mathbf{G}\mathbf{H})^{\mathbf{T}}\mathbf{T}) + \boldsymbol{\beta} = 0 \tag{3-6}$$

after a simple mathematical derivation, the final solution can be got:

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{\mathbf{c}} + (\mathbf{G}\mathbf{H})^{\mathbf{T}}\mathbf{G}\mathbf{H}\right)^{-1}(\mathbf{G}\mathbf{H})^{\mathbf{T}}\mathbf{G}\mathbf{T} \quad (L \le N) \tag{3-7}$$

$$\boldsymbol{\beta} = (\mathbf{GH})^{\mathbf{T}}\left(\frac{\mathbf{I}}{\mathbf{c}} + \mathbf{GH}(\mathbf{GH})^{\mathbf{T}}\right)^{-1}\mathbf{GT} \quad (L > N) \tag{3-8}$$

3.3 Learning the Value of V

To solve the minimization problem, without loss of generality, assume that the loss of the first k samples in each category is smaller than the loss value of other samples in the same category, So there is:

$$\forall i \in \{1, \ldots, N\}, \quad \mathrm{v}_i = \begin{cases} 1, & if\ i \le kb \\ 0, & otherwise \end{cases} \tag{3-9}$$

That is to say, we only choose the best k samples in each category in each training. The proposed method has the following two advantages: on the one hand, the number of samples in each category can be balanced by the constraint function; and on the other hand, the weight values can be used to minimize the impact of poorly performing samples, thereby reducing the influence of noise and outliers and improving the classification accuracy of imbalanced classification.

Based on the above discussion, our algorithm will be presented in Algorithm 2, while the original ELM can be presented in Algorithm 1. To explain the detailed process of the proposed method more intuitively, the flowchart of the proposed method is drawn in Fig. 1. The update of $\boldsymbol{\beta}$ is mutually interacted in successive iterations, while the current $\boldsymbol{\beta}$ and the previous $\mathbf{W}$ jointly influence the learning of the new classifier through selecting new samples.

Algorithm1: ELM for classification

Input: a training dataset $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{t}_i), i = 1,2,...n\}$; the active function $g(.)$; the number of hidden nodes L.
Output: a new classifier.
1: Randomly input the initializing training set and active function.
2: Calculate $\boldsymbol{\beta}$ by Eq. (2-4) or Eq. (2-5).

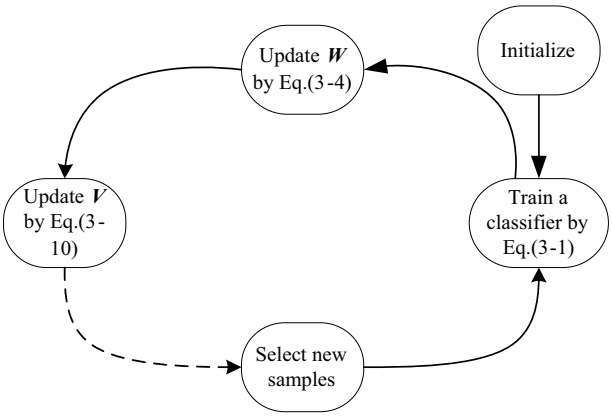

Fig. 1 The flowchart of algorithm

Algorithm2: The proposed method

Input: a training dataset $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{t}_i), i = 1,2, \ldots N\}$; the number of hidden nodes L; the threshold ε.
Output: a new classifier.

1: Input the initializing training set
2: Initializing the output weight $\boldsymbol{\beta_0}$
Initialize $t=1$
Repeat
For $t=1$ to s do
Calculate the **W**
Select the first k samples in each category, and the corresponding v_i value is assigned to 1, otherwise, v_i value is assigned to 0.
Update $\boldsymbol{\beta}$ by Eq.(3-7) or (3-8)
Update **V** by Eq.(3-9)
Update **W** via $w_{ii}=\frac{1}{\|\mathrm{h}(\mathbf{x}_i)\boldsymbol{\beta}-\mathbf{t}_i\|_2^2}$
Update **G** via $\mathbf{G}=\sqrt{\mathbf{VW}}$
Until the$\|J^t - J^{t-1}\|_2^2/|J^t| < \varepsilon$.

3.4 Computational Complexity Analysis

First,we consider the original ELM, once the hidden layer output matrix **H** is determined, there is computational complexity of $O(nLN)$, the computational complexities of matrix multiplication $\mathbf{H^TH}$, $\mathbf{HH^T}$ and $\mathbf{H^TT}$ are equal to $O(L^2 N)$,$O(L N^2)$ and $O(mLN)$ respectively, so the computational complexity of ELM is:

$$O\left(L^3 + L^2N + mLN\right) \quad (\mathrm{L} \leq \mathrm{N})$$
$$O\left(N^3 + 2LN^2 + mLN\right) \quad (\mathrm{L} \leq \mathrm{N})$$

It also can be deduced that the computational complexities of matrix multiplication $\mathbf{H^TWH}$, $\mathbf{WHH^T}$ and $\mathbf{H^TWT}$ are equal to $O(L^2 N+L N^2)$, $O(N^3+ L N^2)$ and $O(L^2 N+mLN)$ respectively. So the computational complexity of $\boldsymbol{\beta}$ in WELM and CCR-ELM can be computed as:

$$O(L^3 + 2LN^2 + L^2N + mLN) \quad (\mathrm{L} \leq \mathrm{N})$$
$$O(2N^3 + 3LN^2 + mLN) \quad (L \leq N)$$

The CS-ELM algorithm divides the data set into two parts based on their labels, and its computational complexity is:

$$O(L^3 + L^2N + mLN) \quad (L \leq N)$$
$$O(N_-^3 + N_+^3 + 2LN_-^2 + 2LN_+^2 + mLN) \quad (L \leq N)$$

It can be known that the computational complexities of proposed method, due to the complexities of matrix multiplication **GH**,**GT** are $O\left(LN^2 + N^3\right), O\left(mLN^2\right)$, the computational complexities of computing inverse of matrix size $N* N$ and $L*L$ are equal to $O(N^3)$ and $O(L^3)$ respectively. So the computational complexities of proposed method is:

$$O(L^3 + N^3 + LN^2 + L^2N + mLN^2) \quad (L \leq N)$$
$$O(2N^3 + 3LN^2 + mLN^2) \quad (L \leq N)$$

From the above analysis, it can be concluded that although the time complexity of the proposed method has increased compared to the original method, the time complexity of all methods is directly proportional to the number of samples or number of hidden neurons, they are $O(L^3)$ and $O(N^3)$, respectively. That is to say, the time complexity of the proposed method in this article will not change by orders of magnitude.

4 Experiment Evaluations

In this section, we will evaluate the classification performance of the proposed method on twelve real-world datasets, the comparative methods include WELM, CS-ELM, CCR-ELM, which are mentioned above. Besides, in the case of imbalanced learning, the overall accuracy is sensitive to the class distribution and may be misleading and not good enough to evaluate the performance of the classifier in imbalanced classification problem [29, 33]. Thus, an evaluation matrix called *G-mean* specially for measuring the performance of classifiers with imbalanced data was proposed, which can overcome the draw-backs of the above overall accuracy, it is defined as: $\text{G-mean} = \sqrt{sensitivity * specificity} = \sqrt{\frac{TP}{TP+FN}} \times \sqrt{\frac{TN}{TN+FP}}$, where *TP, TN, FP, FN* stands for true positive, true negative, false positive and false negative, respectively. Thus, the value of *G-mean* is adopted in this paper to demonstrate the performance of the proposed method. Considering the input weights of proposed method are randomly chosen, so we ran the algorithm 30 times, the final result is from the average of 30 results. The data set is separated into two nonoverlapped subsets: one for training and the other for validation. The optimal number of hidden units is selected as the one which results in the lowest validation error.

4.1 Data Description

Twelve real datasets from KEEL data set repository (https://sci2s.ugr.es/keel/imbalanced.php) are tested in our experiments. All of them are publicly available and fully labeled with each sample belonging to only one class. The datasets used for experimentation have different degrees of class imbalance. The commonly used value measuring the data imbalance degree is IR (imbalance rate), which is defined as follows:

$$IR = \frac{\#minoritysamples}{\#majoritysamples}$$

Here, #represent "number of". The details of the datasets used are summarized in Table 1, where the second column lists the number of samples, the third column represents the number of features, the fourth and fifth columns are the number of samples in minority class and majority class, while the last column is imbalance rate of datasets (IR), the values of IR are in the range from 0 to 1. As shown in Table 1, the imbalance ratio of the dataset can be low as 0.0590, while the highest imbalance ratio happens to be 0.5495.

Table 1 Description of datasets

Dataset	Instances	Features	Mix-class	Max-class	IR
Glass1	214	9	76	138	0.5495
Pima	768	8	268	500	0.5356
Glass0	214	9	70	144	0.4861
Yeast1	1484	8	429	1055	0.4066
Haberman	306	3	81	225	0.3731
Vehicle1	846	18	217	629	0.3448
Ecoil1	336	7	77	259	0.2973
New-thyroid1	215	5	35	180	0.1945
Glass6	214	9	29	185	0.1567
Ecoil3	336	7	36	300	0.12
Pageblosks0	472	10	28	444	0.0646
Abalone9vs18	731	8	41	690	0.0590

4.2 Experimental Setting

In this paper, all the experiments are carried out in Python 3.6 environment running in i7 processor 2.7 GHz CPU and 8 G RAM. In our experiments, attributes in each dataset are normalized in the range $[-1, 1]$. And similar data processing methods can be also found in the works [34, 35]. The sigmoid function: $G(\boldsymbol{a}, b, \mathbf{x}) = 1/(1 + \exp(-(\boldsymbol{a} \cdot \mathbf{x})))$ is selected as the active function of the hidden layer in all algorithms. The hidden nodes parameters a and b are randomly chosen from the range of $[-1, 1]$ based on the uniform sampling distribution probability.

The proposed method consists of the initialization phase and the process of gradual selecting samples and assigning weights to samples gradually. In the initialization phase, all the samples are selected from the given training set to calculate the output weight, that is, the weights of all the samples are the same, then in the subsequent training process, the constraint function will determine the number of samples and uptade their weights, it can determine which samples are added to the following training process according to the loss value based on the last classifier. In the experiments, there are only two parameters to tune for the proposed method with sigmoid additive node: trade-off constant C and the number of hidden nodes L. A grid search of C on $\{ 2^{-10}, 2^{-8}, \ldots, 2^{8}, 2^{10} \}$ and L on $\{10,20,\ldots,190,200\}$ is conducted in seek of the optimal result.

4.3 Experimental Results

The accuracy of the classifiers under *G-mean* value is shown in Table 2, The results of the comparison algorithms can be found from the article [36], similarly, the proposed method is conducted on the same training sets and verified on the same testing datasets. And, to verify the sensitivity of the proposed method to the above two parameters, Fig. 2 is plotted. In addition, the impact of parameters C on classification performance is shown in Fig. 3.

It can be observed from Table 2, in twelve data sets, the proposed method performs best on nine of them, especially on the *Haberman,Vehicle1* and *Abalone9vs18* data sets, the accuracy of them can reach 0.6975, 0.9280, 0.9219, respectively, while on other three data sets, the *G-mean* value is lower than the comparison algorithms. Although in *Glass0*,

Table 2 The G-mean value of data sets (Bold fonts indicate the best result)

Dataset	Proposed	WELM	CCR-ELM	CS-ELM
Glass1	**0.7998**	0.7831	0.7607	0.7964
Pima	**0.7681**	0.7474	0.7099	0.7573
Glass0	0.8392	0.8117	**0.8856**	0.8070
Yeast1	**0.7455**	0.7257	0.7170	0.7298
Haberman	**0.6975**	0.6511	0.4981	0.6571
Vehicle1	**0.9280**	0.8530	0.7960	0.8612
Ecoil1	0.9163	0.9069	0.8906	**0.9173**
New-thyroid1	**0.9950**	0.9944	0.9924	0.9944
Glass6	0.9518	0.9572	0.9129	**0.9578**
Ecoil3	**0.9176**	0.9017	0.9145	0.8986
Pageblocks0	**0.9594**	0.9340	0.9089	0.9338
Abalone9vs18	**0.9219**	0.8872	0.7622	0.9199

Ecoil1 and *Glass6* data sets, the *G-mean* value of the proposed method is not the highest, it is not the worst. So the overall performance of the proposed method is better than comparison methods in terms of *G-mean* value.

In order to test the influence of parameter C and number of hidden nodes L, the Fig. 2 is presented as follows. It can be easily known form Fig. 2 that when C is small, usually below 2^0 in the experiments, better generalization performance is expected. In most of cases *G-mean* value drops very quickly when big C is assigned with a network of very few hidden nodes. It can be also observed that the performance is decreasing when C is increasing given quite a large L. For the *Glass0, Yeast1,* and *Abalone9vs18,* the *G-mean* value fluctuates greatly with the change of C and L, that is, the two data sets are more sensitive to parameters. However, for the data sets *Glass1, Ecoil1* and *New-thyroid1*, the *G-mean* value is relatively stable and does not fluctuate greatly with the parameters changing.

Here in Fig. 3, to more clearly observe the change of *G-mean* value with the user-specified parameter C, a grid search of C on $\{2^{-10}, 2^{-8}, \ldots, 2^8, 2^{10}\}$ is conducted to evaluate the performance of proposed method. It can be seen more clearly that when the value of C is near 0, its *G-mean* value performs best and gradually stabilizes, that is, *G-mean* is not sensitive to the C value. Combining the analysis of Fig. 2, it is also explained that the *G-mean* value is more susceptible to the number of hidden layer nodes L from the side, so it is important to choose the appropriate number of hidden layer nodes in the proposed method.

In order to reflect the convergence of the algorithm, let the number of iterations of the program be 50, the following Fig. 4 is plotted, which demonstrates the convergence of the proposed method to optimize the objective function values of Eq. (3-3).

From the Fig. 4, we can observe that the objective function values of the proposed algorithm sharply decreased in the first several iterations and then began to stable quickly, and the objective function converges within 20 iterations for most datasets.

5 Conclusions

By considering the information of the imbalanced distribution of sample classes, the proposed method is able to deal with imbalanced data classification problem. The method is simple and easy to implement and it can achieve comparable and better

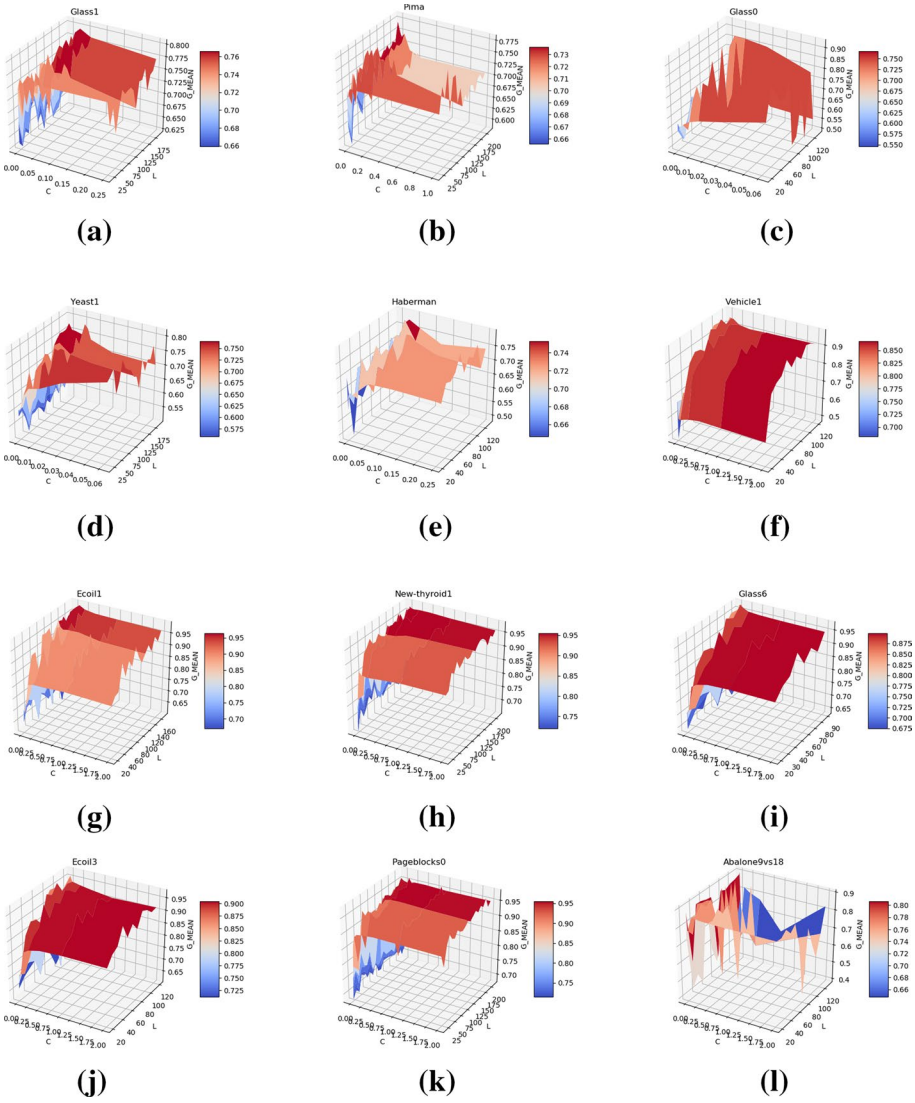

Fig. 2 G-mean of proposed method with different L and C: **a** Glass1; **b** Pima; **c** Glass0; **d** Yeast1; **e** Haberman; **f** Vehicle1; **g** Ecoil1; **h** New-thyroid1; **i** Glass6; **j** Ecoil3; **k** Pageblocks0; **l** Abalone9vs18

generalization performance compared to the conventional machine learning techniques. This work provided a solution for imbalanced data learning by adding a parameter-free weight matrix and adding the constraint of sample size based on the original ELM to balance the impact of sample weights. Compared with traditional improvement of ELM based on weighting schemes, the proposed method does not require to set weights before training, which eliminates the setting process of parameters, instead, it continuously

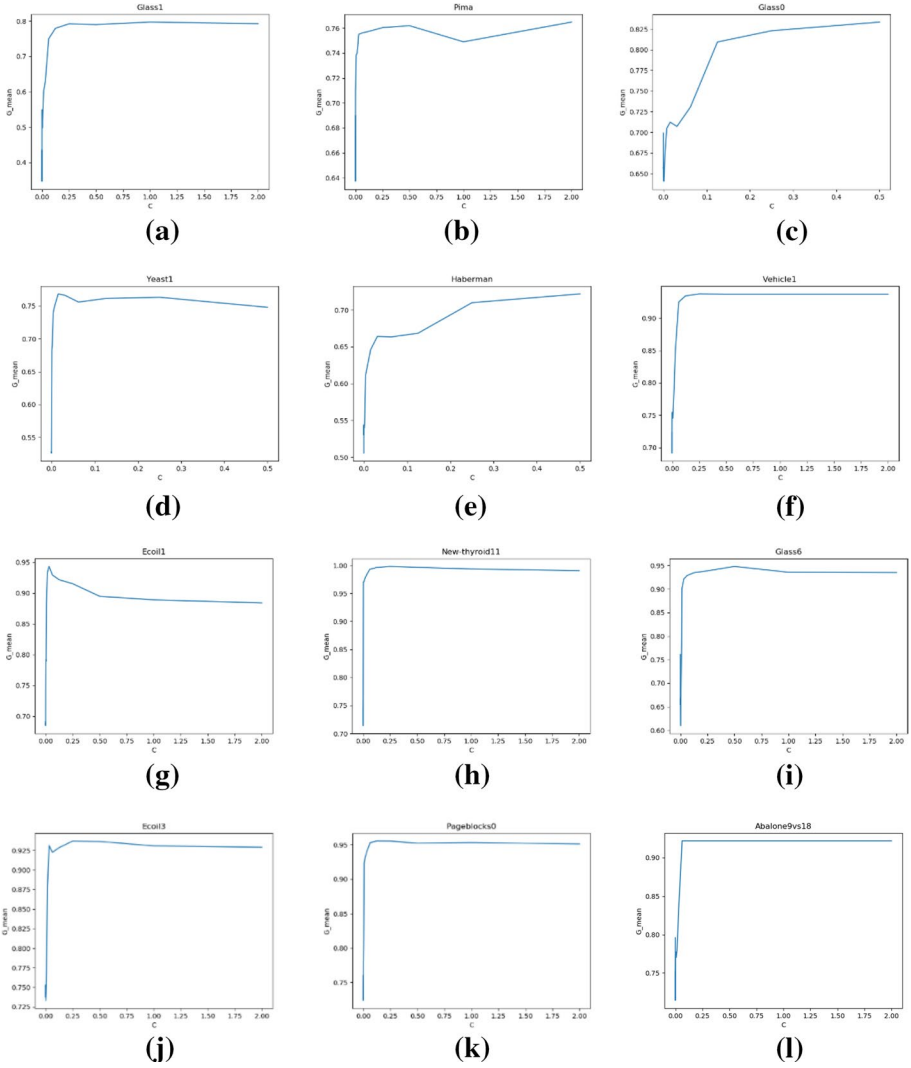

Fig. 3 G-mean of proposed method with different C: **a** Glass1; **b** Pima; **c** Glass0; **d** Yeast1; **e** Haberman; **f** Vehicle1; **g** Ecoil1; **h** New-thyroid1; **i** Glass6; **j** Ecoil3; **k** Pageblocks0; **l** Abalone9vs18

selects samples with high confidence based on the sample loss value and assigns the sample weights that can strengthen the impact of minority class while weaken the impact of majority class. In general, the proposed method in most cases can achieve the top performance among the compared state-of-the-art classification method. However,

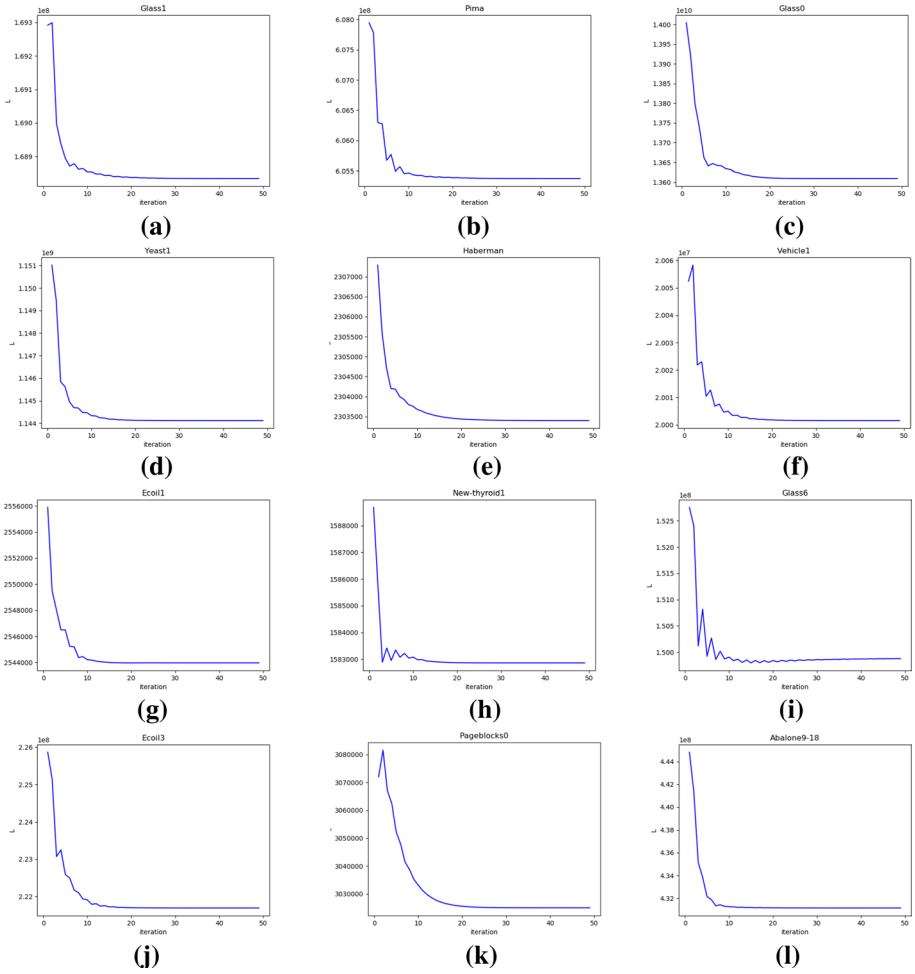

Fig. 4 The convergence of proposed method: **a** Glass1; **b** Pima; **c** Glass0; **d** Yeast1; **e** Haberman; **f** Vehicle1; **g** Ecoil1; **h** New-thyroid1; **i** Glass6; **j** Ecoil3; **k** Pageblocks0; **l** Abalone9vs18

we find that the proposed method achieves the unsatisfactory results in computing complexities, which is due to the fact that it needs to find the optimal weights of samples through iterations. So there is still improvement room in the proposed method, to reduce time complexity based on this work is also one of our future goals.

Acknowledgements This article was funded by the National Study Abroad Fund.

References

1. Longadge R, Dongre S (2013) Class imbalance problem in data mining review. arXiv:13051707
2. Ganganwar V (2012) An overview of classification algorithms for imbalanced datasets. Int J Emerg Technol Adv Eng 2(4):42–47
3. Li L, Sun R, Cai S, Zhao K, Zhang Q (2019) A review of improved extreme learning machine methods for data stream classification. Multimed Tools Appl 78(23):33375–33400
4. Srimuang W, Intarasothonchun S (2015) Classification model of network intrusion using weighted extreme learning machine. In: 2015 12th international joint conference on computer science and software engineering (JCSSE), 2015. IEEE, pp 190–194
5. Wei W, Li J, Cao L, Ou Y, Chen J (2013) Effective detection of sophisticated online banking fraud on extremely imbalanced data. World Wide Web 16(4):449–475
6. Haixiang G, Yijing L, Shang J, Mingyun G, Yuanyue H, Bing G (2017) Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl 73:220–239
7. Burnaev E, Erofeev P, Papanov A (2015) Influence of resampling on accuracy of imbalanced classification. In: Eighth international conference on machine vision (ICMV 2015), 2015. International Society for Optics and Photonics, p 987521
8. Charte F, Rivera AJ, del Jesus MJ, Herrera F (2015) Addressing imbalance in multilabel classification: measures and random resampling algorithms. Neurocomputing 163:3–16
9. Wang S, Minku LL, Yao X (2014) Resampling-based ensemble methods for online class imbalance learning. IEEE Trans Knowl Data Eng 27(5):1356–1368
10. Zhu X, Yang J, Zhang C, Zhang S (2019) Efficient utilization of missing data in cost-sensitive learning. IEEE Trans Knowl Data Eng. https://doi.org/10.1109/TKDE.2019.2956530
11. Zheng E, Zhang C, Liu X, Lu H, Sun J (2013) Cost-sensitive extreme learning machine. In: International conference on advanced data mining and applications, 2013. Springer, pp 478–488
12. Wang X, Liu X, Japkowicz N, Matwin S (2013) Resampling and cost-sensitive methods for imbalanced multi-instance learning. In: 2013 IEEE 13th international conference on data mining workshops, 2013. IEEE, pp 808–816
13. Qian Y, Liang Y, Li M, Feng G, Shi X (2014) A resampling ensemble algorithm for classification of imbalance problems. Neurocomputing 143:57–67
14. Zhu X, Zhu Y, Zheng W (2019) Spectral rotation for deep one-step clustering. Pattern Recognit 105:107175
15. Zhu X, Zhang S, He W, Hu R, Lei C, Zhu P (2018) One-step multi-view spectral clustering. IEEE Trans Knowl Data Eng 31(10):2022–2034
16. Ling CX, Sheng VS (2008) Cost-sensitive learning and the class imbalance problem. In: Encyclopedia of machine learning, vol 2011, pp 231–235
17. Ren Y, Zhao P, Xu Z, Yao D (2017) Balanced self-paced learning with feature corruption. In: 2017 international joint conference on neural networks (IJCNN), 2017. IEEE, pp 2064–2071
18. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1–3):489–501
19. Huang G, Huang GB, Song S, You K (2015) Trends in extreme learning machines: a review. Neural Netw 61:32–48. https://doi.org/10.1016/j.neunet.2014.10.001
20. Alade OA, Selamat A, Sallehuddin R (2017) A review of advances in extreme learning machine techniques and its applications. In: International conference of reliable information and communication technology, 2017. Springer, pp 885–895
21. Zong W, Huang G-B, Chen Y (2013) Weighted extreme learning machine for imbalance learning. Neurocomputing 101:229–242
22. Akbulut Y, Şengür A, Guo Y, Smarandache F (2017) A novel neutrosophic weighted extreme learning machine for imbalanced data set. Symmetry 9(8):142
23. Zhang X, Song Q, Wang G, Zhang K, He L, Jia X (2015) A dissimilarity-based imbalance data classification algorithm. Appl Intell 42(3):544–565
24. Lu C, Ke H, Zhang G, Mei Y, Xu H (2019) An improved weighted extreme learning machine for imbalanced data classification. Memetic Comput 11(1):27–34
25. Li K, Kong X, Lu Z, Wenyin L, Yin J (2014) Boosting weighted ELM for imbalanced learning. Neurocomputing 128:15–21
26. Raghuwanshi BS, Shukla S (2018) Class-specific kernelized extreme learning machine for binary class imbalance learning. Appl Soft Comput 73:1026–1038
27. Hu R, Zhu X, Zhu Y, Gan J (2019) Robust SVM with adaptive graph learning. World Wide Web 23:1–24

28. Zhu X, Gan J, Lu G, Li J, Zhang S (2019) Spectral clustering via half-quadratic optimization. World Wide Web 23:1–20
29. Xiao W, Zhang J, Li Y, Zhang S, Yang W (2017) Class-specific cost regulation extreme learning machine for imbalanced classification. Neurocomputing 261:70–82
30. Gan J, Wen G, Yu H, Zheng W, Lei C (2018) Supervised feature selection by self-paced learning regression. Pattern Recogn Lett. https://doi.org/10.1016/j.patrec.2018.08.029
31. Zhu X, Li X, Zhang S, Xu Z, Yu L, Wang C (2017) Graph PCA hashing for similarity search. IEEE Trans Multimed 19(9):2033–2044
32. Zheng W, Zhu X, Wen G, Zhu Y, Yu H, Gan J (2018) Unsupervised feature selection by self-paced learning regularization. Pattern Recognit Lett
33. Huang C, Li Y, Change Loy C, Tang X (2016) Learning deep representation for imbalanced classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016. pp 5375–5384
34. Cao J, Lin Z, Huang G-B, Liu N (2012) Voting based extreme learning machine. Inf Sci 185(1):66–77
35. Wang T, Cao J, Lai X, Chen B (2018) Deep weighted extreme learning machine. Cognit Comput 10(6):890–907
36. Raghuwanshi BS, Shukla S (2018) Class-specific extreme learning machine for handling binary class imbalance problem. Neural Netw 105:206–217

Parameter-Free Extreme Learning Machine for Imbalanced Classification

Li, L

2020-12