Received March 28, 2018, accepted May 8, 2018, date of publication May 21, 2018, date of current version June 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2839340

LW-ELM: A Fast and Flexible Cost-Sensitive Learning Framework for Classifying Imbalanced Data

HUALONG YU[1], CHANGYIN SUN[2], XIBEI YANG[1], SHANG ZHENG[1], QI WANG[1], AND XIAOYAN XI[1]
[1]School of Computer, Jiangsu University of Science and Technology, Zhenjiang 212003, China
[2]School of Automation, Southeast University, Nanjing 210096, China

Corresponding author: Hualong Yu (yuhualong@just.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grants 61305058 and 61572242, in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20130471, in part by the China Postdoctoral Science Foundation under Grants 2013M540404 and 2015T80481, in part by the Jiangsu Planned Projects for Postdoctoral Research Funds under Grant 1401037B, and in part by the Qing Lan Project of Jiangsu Province of China.

ABSTRACT Learning from imbalanced data is a challenging task in the fields of machine learning and data mining. As an effective and efficient solution, cost-sensitive learning has been widely adopted to address class imbalance learning (CIL) problems. Weighted extreme learning machine (WELM), which is constructed based on ELM, is a significant member in the cost-sensitive-learning algorithmic family. WELM can effectively deal with CIL problems. However, it has two main drawbacks: 1) it has high time complexity on large-scale data since a large-matrix multiplication operation is required in the solution procedure and 2) it lacks flexibility since it can only tune the training error for each instance and not for each class label. In this paper, we present an alternative to WELM, which is called label-WELM (LW-ELM). Unlike WELM, LW-ELM copes with CIL problems by tuning the training error of each class label. Specifically, the expected output (or training class label) that corresponds to the minority class is augmented, thereby providing stronger tolerance to training errors of the minority-class instances. In this paper, we design two types of weight allocation strategies, both of which are based on the class-imbalance ratio (CIR). In contrast with WELM, LW-ELM is fast and flexible, where fast means that it has low-time complexity and flexible indicates that it can also be used to tackle imbalanced multi-label learning problems, while WELM cannot. The experimental results on binary-class, multiclass, and multi-label data sets with skewed class distributions show the effectiveness and superiority of the proposed LW-ELM algorithm.

INDEX TERMS Class imbalance learning, cost-sensitive learning, extreme learning machine, weighted extreme learning machine, multi-label learning.

I. INTRODUCTION

Many real-world classification problems that have been examined in recent years are represented by highly imbalanced data sets, in which the number of instances from one class is much smaller than that in another. Generally, a problem of this type is called either a class imbalance learning (CIL) problem [1] or a rare event detection problem [2]. For such problem, most traditional classification models tend to provide the biased predictions that focus only on the accuracy of the majority class and neglect the rare events. During the past two decades, CIL problems have attracted attention from many researchers and many learning techniques have been presented. These techniques have also been widely adopted in a variety of CIL applications, including fraud detection [3]–[5], credit card approval [6], [7], software defect prediction [8], [9], network intrusion detection [10], [11], disease diagnosis [12], [13], bioinformatics [14]–[16], industrial manufacturing [17], and environment resource management [18].

For a CIL problem, sampling [19]–[22], cost-sensitive leaning [23]–[28] and ensemble learning [29]–[32] are the most frequently used techniques. In contrast with the other techniques, cost-sensitive learning possesses several advantages: 1) it does not need to modify the distribution of the training instances; 2) it can adaptively determine the optimal position of the classification boundary; and 3) it often has relatively low time complexity because it only modifies the traditional classification algorithm and does not inserts extra steps into the learning procedure. Therefore, we focus on solving CIL problems by cost-sensitive learning in this paper.

With the exception of meta-cost [23], cost-sensitive learning algorithms are generally correlated with a specific classification model. In this paper, extreme learning machine (ELM), which is a popular classification model, is intensively considered. We select ELM because it always has higher or comparable prediction accuracy and generalization ability than back-propagation neural networks (BPNNs) and support vector machine (SVM) [33], [34], fast modeling speed, and can be applied to binary-class, multiclass and multi-label classification problems easily [35], [36].

There is a well-known cost-sensitive learning algorithm that is based on ELM, namely, weighted extreme learning machine (WELM) [25]. WELM regulates the training errors by increasing the penalty factor that corresponds to the minority class instances. It can effectively deal with the class imbalance problem. However, it has two internal drawbacks: 1) when the training set is large, it has high time complexity as the large-matrix multiplication operation is required in the solution procedure and 2) it lacks flexibility as it can only tune the training error of each instance and, thus, cannot be adopted to address multi-label imbalanced learning problems.

To decrease the time complexity and promote the feasibility of WELM, we present a new ELM-based cost-sensitive learning algorithm, which is called label-weighted extreme learning machine (LW-ELM). Instead of tuning the penalty factor of each class, LW-ELM augments the expected output (or class label) that corresponds to the minority class instances, thereby improving the tolerance to training errors of the minority-class instances. LW-ELM is faster than WELM since it does not need to insert a large weight matrix into the optimization procedure; it only needs to provide a weighted assignment for the expected output matrix. Moreover, LW-ELM is more feasible than WELM as it regulates training errors for each class label but not for each instance. Thus, it can be easily extended to solve multi-label imbalance learning problems. In this paper, we design two kinds of weight allocation strategies, both of which are based on the class-imbalance ratio (CIR). The first is more radical and the second more moderate. The experimental results on binary-class, multiclass and multi-label data sets that follow skewed class distributions indicate the effectiveness and superiority of the proposed LW-ELM algorithm.

The remainder of this paper is organized as follows. In Section II, we briefly review the existing CIL techniques in the context of ELM and multi-label learning. Section III introduces two preliminaries for this study: ELM and WELM. In Section IV, we analyze the effectiveness of label weighting and introduce two weight assignment strategies and the proposed LW-ELM algorithmic framework in detail. Finally, we compare the theoretical time complexities of ELM, WELM and LW-ELM. Section V presents the experimental results and the corresponding analysis and discussion. At last, Section VI presents the conclusions of this paper.

II. RELATED WORK

A. CIL TECHNIQUES IN THE CONTEXT OF ELM

In general, three main types of approaches have been applied to CIL problems: sampling [19]–[22], cost-sensitive leaning [23]–[28] and ensemble learning [29]–[32]. Sampling is based on rebalancing class distributions by deleting instances from the majority classes (undersampling) or adding new instances to the minority classes (oversampling). The advantage of the sampling technique lies in its independence from the classification models. The aim of cost-sensitive learning is to bias the existing classifiers towards the minority classes. Therefore, cost-sensitive learning is generally correlated with a specific classification model, such SVM, decision tree, logistic regression or ELM. For the ensemble learning that is associated with CIL problems, sampling or cost-sensitive learning algorithms are integrated into a specific ensemble learning framework, e.g., bagging, boosting or random forest. For reviews and surveys of CIL techniques, interested readers may refer to [1], [2], and [37]–[39].

In the context of ELM, several CIL approaches have been proposed in previous work. Vong *et al.* [40] adopted the random oversampling (ROS) strategy to promote the recognition rate of the level of suspended particulate matter, which follows skewed distributions. Sun *et al.*, [41] integrated the synthetic minority oversampling technique (SMOTE) into a multiple-ELM framework to increase the predictive accuracy of a corporation life cycle. For cost-sensitive learning techniques, Zong *et al.*, [25] proposed a weighted ELM algorithm that is named WELM, which improves the performance on the minority classes by assigning larger penalty factors to the training instances that belong to those minority classes. A similar algorithm, namely, fuzzy ELM (FELM) was independently proposed by Zhang and Ji [42], which regulates the distributions of penalty factors by inserting a fuzzy matrix. However, they failed to provide a unified design rule for the fuzzy matrix. Recently, Zhang and Zhang [43] presented an evolutionary cost-sensitive ELM (ECS-ELM) algorithm. However, the algorithm is specially designed for application to the cost-sensitive but not CIL scenario. Yu *et al.*, [27] proposed a special cost-sensitive ELM algorithm, namely, ODOC-ELM, for coping with CIL problems. The algorithm trains a normal ELM classifier and searches for the optimal combination of decision output compensation thresholds with the aim of minimizing the G-mean metric of the training instances. The WELM algorithm has been integrated with a boosting ensemble learning framework and achieved improved classification performance [44].

B. CIL TECHNIQUES IN THE CONTEXT OF MULTI-LABEL LEARNING

Traditional supervised learning always faces a data set in which each object is associated with a single label that denotes its category. However, in real-world applications, one example might hold multiple labels simultaneously, i.e., be associated with a set of class labels. In general, we call this kind of task a multi-label learning problem [45]. Taking image retrieval as an example, an image might be associated with several categories, such as *grass*, *cloud*, *sky*, *mountain*, *tree* and *lake* (see Fig. 1).

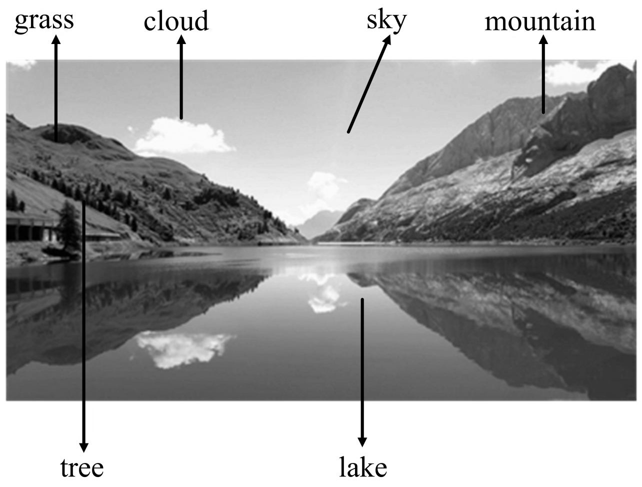


FIGURE 1. An example of multi-label learning instance.

In previous work, it has been indicated that for multi-label classification data, the CIL problem is generally more serious. That is, for most categories, label 1 is often extremely scarce. Therefore, CIL techniques have also been widely used in multi-label classification problems. Tahir *et al.* [46] proposed the inverse random under sampling (IRUS) algorithm for addressing the CIL problem in multi-label data. In IRUS, the majority class is severely under-sampled multiple times and in each round, the number of instances that belong to the majority class should be guaranteed to be less than that for the minority class. Then, each under-sampled majority set should be integrated with all instances in the minority class to construct the training subset, which is used subsequently to train a classifier. Finally, all classifiers are integrated to make decisions in the form of majority voting. For a multi-label classification problem, this procedure is conducted on each class label. Charte *et al.* [47] discussed the CIL problem in multi-label classification in detail and designed several metrics for measuring the imbalance level in multi-label data sets. Based on these metrics, four types of sampling algorithms, namely, LP-RUS, LP-ROS, ML-RUS, and ML-ROS, were presented. In [48], Charte *et al.* proposed an improved algorithm, namely, ML-SMOTE. The authors further clarified how to determine neighbors and synthesize the label set of the target instance from its neighbors' labels. Zhang *et al.* [49] presented an algorithm named cross-coupling aggregation (COCOA), whose main advantage is that it tries to leverage the exploitation of label correlations and the exploration of class imbalance. In brief, to induce the predictive model on each class label, one binary-class imbalance learner that corresponds to the current label and several multi-class imbalance learners that are coupled with other labels are aggregated for prediction. ELM can be well adapted for multi-label learning without adding extra structures or steps [50]. However, no work has been performed on multi-label CIL problems in the context of ELM.

III. PRELIMINARIES

A. EXTREME LEARNING MACHINE

ELM, which was proposed by Huang *et al.* [33], [34], is a learning algorithm for single-hidden-layer feed-forward neural networks (SLFNs). The main characteristic of ELM that distinguishes it from the conventional learning algorithms of SLFN is the random generation of hidden nodes. Therefore, ELM does not need to iteratively regulate parameters to make them approach the optimal values. Thus, it has faster learning speed and better generalization ability. Previous research has indicated that ELM can produce better or at least similar generalization ability and classification performance compared to SVM and BPNN, but only consumes tenths or hundredths of their training time [33], [34].

Let us consider a classification problem with N training instances for distinguishing m categories. The ith training instance can be represented as $(\mathrm{x}_i, \mathrm{t}_i)$, where x_i is an $n \times 1$-dimensional input vector, while t_i is the corresponding $m \times 1$-dimensional output vector. Suppose there are L hidden nodes in ELM and all weights and biases on these nodes are generated randomly. Then, for instance x_i, its hidden layer output can be represented as a row vector $\mathrm{h}(\mathrm{x}_i) = [\mathrm{h}_1(\mathrm{x}_i), \mathrm{h}_2(\mathrm{x}_i), \ldots, \mathrm{h}_L(\mathrm{x}_i)]$. Therefore, the mathematical model of ELM can be described as

$$\mathrm{H}\beta = \mathrm{T} \tag{1}$$

where $\mathrm{H}=[\mathrm{h}(\mathrm{x}_1), \mathrm{h}(\mathrm{x}_2), \ldots, \mathrm{h}(\mathrm{x}_N)]^{\mathrm{T}}$ is the hidden-layer output matrix over all training instances, $\beta$is the weight matrix of the output layer, and $\mathrm{T}=[\mathrm{t}_1, \mathrm{t}_2, \ldots, \mathrm{t}_N]$ denotes the target matrix (expected output matrix). Obviously, in Eq. (1), only β is unknown, so we can adopt the least-square algorithm to acquire its solution, which can be described as follows:

$$\beta = \mathrm{H}^{\dagger}\mathrm{T} = \begin{cases} \mathrm{H}^{\mathrm{T}}(\mathrm{H}\mathrm{H}^{\mathrm{T}})^{-1}\mathrm{T}, & \text{when } N \le L \\ (\mathrm{H}\mathrm{H}^{\mathrm{T}})^{-1}\mathrm{H}^{\mathrm{T}}\mathrm{T}, & \text{when } N > L \end{cases} \tag{2}$$

where $\mathrm{H}^{\dagger}$ denotes the Moore-Penrose generalized inverse of the hidden-layer output matrix H, which can guarantee that the solution is the least-norm least-square solution of Eq. (1).

We can also train an ELM from the viewpoint of optimization [34]. In the optimization version of ELM, we wish to simultaneously minimize $||\mathrm{H}\beta - \mathrm{T}||^2$ and $||\beta||^2$. Thus, the optimization problem can be described as follows:

$$\begin{aligned} &\text{Minimize: } Lp_{ELM} = \frac{1}{2}||\beta||^2 + C\frac{1}{2}\sum_{i=1}^{N}||\xi_i||^2 \\ &\text{Subject to: } \mathrm{h}(x_i)\beta = \mathrm{t}_i^{\mathrm{T}} - \xi_i^{\mathrm{T}}, \quad i = 1, 2, \ldots, N \end{aligned} \tag{3}$$

where $\xi_i = [\xi_{i,1}, \xi_{i,2}, \ldots, \xi_{i,m}]$ denotes the training error vector of the m output nodes with respect to the training instance x_i, while C is the penalty factor, which represents the tradeoff between the minimization of the training errors and the maximization of the generalization ability. According to the Karush–Kuhn–Tucker (KKT) theorem, the solution of Eq. (3) can be expressed as follows:

$$\beta = \mathrm{H}^{\dagger}\mathrm{T} = \begin{cases} \mathrm{H}^{\mathrm{T}}(\frac{\mathrm{I}}{C}+\mathrm{H}\mathrm{H}^{\mathrm{T}})^{-1}\mathrm{T}, & \text{when } N \leq L \\ (\frac{\mathrm{I}}{C}+\mathrm{H}\mathrm{H}^{\mathrm{T}})^{-1}\mathrm{H}^{\mathrm{T}}\mathrm{T}, & \text{when } N > L \end{cases} \tag{4}$$

B. WEIGHTED EXTREME LEARNING MACHINE

WELM is a cost-sensitive learning version of ELM, which can be regarded as an effective way to cope with the CIL problem [25]. The main strategy of WELM is to assign different penalties to different categories, where the minority class has a larger penalty factor C while the majority class has a smaller C value. Then, it focuses on the training errors of the minority instances, thereby making the classification hyperplane appear at a more impartial position. A weighting matrix W is used to regulate the parameter C for different instances, i.e., Eq. (3) can be rewritten as:

$$\text{Minimize: } Lp_{ELM} = \frac{1}{2}||\beta||^2 + C\frac{1}{2}W\sum_{i=1}^{N}||\xi_i||^2$$
$$\text{Subject to: } \mathrm{h}(x_i)\beta = \mathrm{t}_i^{\mathrm{T}} - \xi_i^{\mathrm{T}}, i = 1, 2, \ldots, N \tag{5}$$

where W is an $N \times N$ diagonal matrix in which each value on the diagonal represents the corresponding regulation weight of parameter C. In [25], the authors provide two weighting strategies, which are described as follows:

$$\mathrm{W}^1\mathrm{ELM} : W_{ii} = 1/\#(\mathrm{t}_i) \tag{6}$$

and

$$\mathrm{W}^2\mathrm{ELM} : W_{ii} = \begin{cases} 0.618/\#(\mathrm{t}_i) & \text{if } \#\,(\mathrm{t}_i) > \mathrm{AVG}(\mathrm{t}_i) \\ 1/\#(\mathrm{t}_i) & \text{if } \#\,(\mathrm{t}_i) \leq \mathrm{AVG}(\mathrm{t}_i) \end{cases} \tag{7}$$

where $\#(\mathrm{t}_i)$ and $\mathrm{AVG}(\mathrm{t}_i)$ denote the number of instances that belong to class t_i and the average number of instances over all classes, respectively. Then, the solution can be described as follows:

$$\beta = \begin{cases} \mathrm{H}^{\mathrm{T}}(\frac{\mathrm{I}}{C} + W\mathrm{H}\mathrm{H}^{\mathrm{T}})^{-1}W\mathrm{T}, & \text{when } N \leq L \\ (\frac{\mathrm{I}}{C} + \mathrm{H}W\mathrm{H}^{\mathrm{T}})^{-1}\mathrm{H}^{\mathrm{T}}W\mathrm{T}, & \text{when } N > L \end{cases} \tag{8}$$

Compared with Eq. (4), an extra weight matrix W has been added into the solution procedure in Eq. (8). Since the size of W is $N \times N$, where N denotes the number of training instances, when N is large enough, the time complexity of determining β in WELM inevitably increases substantially. Therefore, we expect to provide an alternative to WELM that has lower time complexity but the same effect. This is one of the main motivations of this study. The alternative algorithmic framework will be presented in the following section.

IV. METHODS

A. WHY LABEL WEIGHTING?

Before we train an ELM or WELM model, an expected output matrix T should be generated. The matrix T includes m rows and N columns, where m and N denote the numbers of classes and training instances, respectively. T can be represented as:

$$\mathrm{T} = \begin{bmatrix} \mathrm{t}_{11} & \mathrm{t}_{12} & \cdots & \mathrm{t}_{1N} \\ \mathrm{t}_{21} & \mathrm{t}_{22} & \cdots & \mathrm{t}_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{t}_{m1} & \mathrm{t}_{m2} & \cdots & \mathrm{t}_{mN} \end{bmatrix} \tag{9}$$

Each entity t_{ij} in T is either 1 or -1, where 1 indicates that the instance belongs to the corresponding class, and -1 indicates that it does not. Let us revisit Eq. (5). It is not difficult to observe that in WELM, the optimized objective is the tradeoff between the minimization of the weight matrix of the output layer β and the minimization of the weighted training biases. Irrespective of the first optimization term, the weighted training bias for any single instance x_i can be represented as $CW_{ii}||\xi_i||^2$, where $\xi_i^{\mathrm{T}} = \mathrm{t}_i^{\mathrm{T}} - \mathrm{h}(x_i)\beta$. Since the weighted biases of all training instances must be minimized uniformly, it is obvious that the instance with the smaller weight (majority class instance) can incur a higher training bias, whereas the instance with the larger weight (minority class instance) should have a lower training bias.

Let us consider the question from another perspective; if we remove the weighting matrix W but provide stronger tolerance to training biases for the instances that belong to the minority class compared to the examples from the majority class, the CIL problem might be well addressed. This could be implemented by augmenting the expected output that corresponds to the instances in the minority class, i.e., by making some entities in T larger than 1, while leaving the others unchanged. The idea could be intuitively explained by ELM's decision rule. In ELM, although the expected output is restricted to $\{-1,1\}$, the value range of the actual output is generally $(-\infty, +\infty)$ and the actual output is always a continuous value. To make decision, a threshold value of 0 might be required. Then, when and only when an actual output value is larger than 0 will the instance be assigned to the corresponding category. Therefore, when we compare two expected outputs t_i and t_j, if $|\,\mathrm{t}_i| > |\mathrm{t}_j|$, then we say that t_i has stronger tolerance to training bias than t_j as it is farther from the threshold value of 0. That is, if the aim is to minimize the overall training bias, then the instance would tend to be awarded to category i, not j.

In addition to the intuitionistic explanation, we try to explain the concept by the analysis theory that is used in our previous work [27]. Without loss of generality, suppose the imbalanced data set is a binary-class set. Then, in ELM, the expected outputs of the minority class and the majority class are assigned values of 1 and -1, respectively. Considering a small and compact boundary region, there are S majority class instances and 1 minority example, where $S \gg 1$. The words "small and compact" mean that all the instances in that region have similar inputs. Let us describe the feature vector of the minority class instance as $x_0 = (x_0^1, x_0^2, \ldots, x_0^n)$ and the feature vectors of the majority class instances as $x_i = (x_0^1 + \Delta x_i^1, x_0^2 + \Delta x_i^2, \ldots, x_0^n + \Delta x_i^n)$, where $i \in \{1, 2, \ldots, S\}$ and Δx_i^j denotes a small positive or negative deviation for the j^{th} feature component of the i^{th} instance in comparison with that of instance x_0. In addition, we use $\Delta x_i = (\Delta x_i^1, \Delta x_i^2, \ldots, \Delta x_i^n)$ to denote the deviation of the feature vector of the i^{th} instance in comparison with that of the minority class instance x_0. Then, according to Eq. (1), the actual outputs of these instances can be represented as:

$$f(x_i) = \begin{cases} \sum_{i=1}^{L} \beta_i h(x_0), & \text{if } j = 0 \\ \sum_{i=1}^{L} \beta_i h(x_0 + \Delta x_j), & \text{if } j = 1, 2, \ldots, S \end{cases} \tag{10}$$

Based on Eq. (10), $\Delta f(x_i)$, which is the variation between the actual output of the j^{th} majority class instance x_j and that of the minority class instance x_0, can be calculated as follows:

$$\begin{aligned} \Delta f(x_i) &= \sum_{i=1}^{L} \beta_i h(x_0 + \Delta x_j) - \sum_{i=1}^{L} \beta_i h(x_0) \\ &= \sum_{i=1}^{L} \beta_i [h(x_0 + \Delta x_j) - h(x_0)] \end{aligned} \tag{11}$$

In Eq. (11), when $||\beta||$ and Δx_j are both sufficiently small, $\Delta f(x_i)$ can be guaranteed to be a small real number, which is either positive or negative, i.e., two closely adjacent instances have similar actual outputs in ELM. Suppose β has been determined. Then, Q_{sub}, which is the total training bias of the subset, can be represented as:

$$Q_{sub} = [f(x_0) - 1]^2 + \sum_{i=1}^{S} [f(x_0) + \Delta f(x_i) - (-1)]^2 \tag{12}$$

To minimize the training bias of the subset, we should set the following quantity to zero:

$$\frac{\partial Q_{sub}}{\partial f(x_0)} = (2S + 2) f(x_0) + 2 \sum_{i=1}^{S} \Delta f(x_i) + 2S - 2 \tag{13}$$

Then, we can calculate the actual output of the minority instance x_0 as follows:

$$f(x_0) = \frac{1 - S - \sum_{i=1}^{S} \Delta f(x_i)}{1 + S} \tag{14}$$

As we mentioned above, $\sum_{i=1}^{S} \Delta f(x_i)$ is sufficiently small. Thus, its influence can be neglected. Then, it is clear that $f(x_0)$ tends to output a negative value that is far from the threshold of 0 as $S \gg 1$. Furthermore, with the increase of S, $f(x_0)$ gradually approaches -1. That also explains why ELM can be damaged by class-imbalanced distributions.

Next, we analyze the same problem in our label weighting scenario. Suppose the expected output of sample x_0 has been previously augmented P times. Then, Eq. (12) can be rewritten as:

$$Q_{sub} = [f(x_0) - P]^2 + \sum_{i=1}^{S} [f(x_0) + \Delta f(x_i) - (-1)]^2 \tag{15}$$

To minimize the training bias, we have:

$$\frac{\partial Q_{sub}}{\partial f(x_0)} = (2S + 2) f(x_0) + 2 \sum_{i=1}^{S} \Delta f(x_i) + 2S - 2P \tag{16}$$

Then, $f(x_0)$ can be described as follows:

$$f(x_0) = \frac{P - S - \sum_{i=1}^{S} \Delta f(x_i)}{1 + S} \tag{17}$$

If $P = S$, the bias that is caused by the skewed data distributions can be well corrected. This is a powerful theoretical explanation for the effectiveness of the label weighting strategy.

B. WEIGHT ASSIGNMENT

For label weight assignment, we design two rules, both of which are based on the class-imbalance ratio (CIR), which is the ratio between the numbers of instances that belong to the majority class and the minority class. In binary-class and multiclass CIL problems, the expected outputs should be assigned by one of the following two functions:

$$\text{LW}^1\text{ELM: } t_{ij} = \begin{cases} \#\max(C)/\#(C_i), & \text{if } x_j \in C_i \\ -1, & \text{if } x_j \notin C_i \end{cases} \tag{18}$$

and

$$\text{LW}^2\text{ELM: } t_{ij} = \begin{cases} \sqrt[2]{\#\max(C)/\#(C_i)}, & \text{if } x_j \in C_i \\ -1, & \text{if } x_j \notin C_i \end{cases} \tag{19}$$

where $\#(C_i)$ and $\#\max(C)$ denote the number of instances that belong to the i^{th} category and the number of instances in the largest majority class, respectively. When there are fewer instances in one class, each instance that belongs to that class is assigned a larger expected output value. In Eq. (18) and Eq. (19), we also observe that in contrast with LW^{1}ELM, LW^{2}ELM is a more moderate weight assignment rule.

Unlike multiclass classification problems, a multi-label learning problem can be regarded as a combination of multiple binary-class problems. In addition, the expected output matrix T in binary-class or multiclass problems obeys an exclusion principle, namely, that there is only one element that does not equal -1 in each column, in contrast to multi-label problems. Therefore, we modify the weight assignment rules and express them as follows:

$$\text{LW}^1\text{ELM: } t_{ij} = \begin{cases} \sim\#(C_i)/\#(C_i), & \text{if } x_j \in C_i \\ -1, & \text{if } x_j \notin C_i \end{cases} \tag{20}$$

and

$$\text{LW}^2\text{ELM: } t_{ij} = \begin{cases} \sqrt[2]{\sim\#(C_i)/\#(C_i)}, & \text{if } x_j \in C_i \\ -1, & \text{if } x_j \notin C_i \end{cases} \tag{21}$$

where $\sim\#(C_i)$ denotes the number of instances that do not belong to the i^{th} category.

To intuitively show the difference in label weight assignment rules between multiclass problems and multi-label problems, we provide two examples:

Example 1: For a multiclass imbalance problem, with ELM, the expected output matrix is

$$T = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}$$

Then, with LW^{1}ELM and LW^{2}ELM, the expected output matrices are respectively modified as:

$$T^1 = \begin{bmatrix} 3 & -1 & -1 & -1 & -1 & -1 \\ -1 & 3/2 & 3/2 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}$$

and

$$T^2 = \begin{bmatrix} \sqrt{3} & -1 & -1 & -1 & -1 & -1 \\ -1 & \sqrt{3/2} & \sqrt{3/2} & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}$$

Example 2: For a multi-label imbalance problem, with ELM, the expected output matrix is

$$T = \begin{bmatrix} 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & 1 \end{bmatrix}$$

Then, with LW^{1}ELM and LW^{2}ELM, the expected output matrices are respectively modified as:

$$T^1 = \begin{bmatrix} 2 & 2 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 \\ -1 & -1 & 2 & -1 & 2 & -1 \\ -1 & -1 & -1 & -1 & -1 & 5 \end{bmatrix}$$

and

$$T^2 = \begin{bmatrix} \sqrt{2} & \sqrt{2} & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 \\ -1 & -1 & \sqrt{2} & -1 & \sqrt{2} & -1 \\ -1 & -1 & -1 & -1 & -1 & \sqrt{5} \end{bmatrix}$$

C. LW-ELM ALGORITHMIC FRAMEWORK

The detailed procedure of the LW-ELM algorithm is described in Fig. 2. Binary-class, multiclass and multi-label classification problems have been integrated into a uniform algorithmic framework. Their difference is only reflected in step 5∼step 9 in Fig. 2, in which they use different rules to correct the initial expected output matrix T.

D. ANALYSIS OF THE TIME COMPLEXITY

As mentioned above, one of the main aims of this study is to reduce the running time consumption of cost-sensitive learning in the context of ELM. Thus, in this subsection, we will analyze and compare the time complexities of the ELM, WELM and LW-ELM algorithms in detail.

In the testing phase, we have received the hidden weight connecting matrix β, which indicates that the testing times are totally equivalent for the three algorithms. Therefore, we only consider their differences in running time in the training phase. The procedure can be divided into three sequential stages: a preprocessing stage (generating T and/or W), calculation stage I (calculating H) and calculation stage II (calculating β).

Suppose the number of training instances is N, the dimension of each training instance is n, the number of class labels is m and the number of hidden nodes is L. In the preprocessing stage, we first scan all instances to count the number of categories to obtain m. Then, we generate an expected output matrix T and assign an appropriate value to each element in T. Therefore, it is not difficult to calculate the time complexity of ELM, which is $O(mN)$. For WELM and LW-ELM, CIL indexes will be calculated. Thus, only a small amount of extra running time is consumed for them. In calculation stage I, all three algorithms perform the same calculation. Hence, they take the same constant running time: O(NnL). However, in calculation stage II, these three algorithms present significant differences in terms of time consumption as both ELM and LW-ELM solve β by Eq. (4), while WELM solves β Eq. (8). The size of H is $L \times N$, the size of T is $m \times N$ and the size of W is $N \times N$. Thus, HH^T takes $O(NL^2)$ time, H^{T}T $O(NLm)$ time, HWH^T$O(N^2L + NL^2)$ time and H^TWT $O(N^2L+NLm)$ time. Meanwhile, since both HH^T and HWH^T are L-order square matrices, their time complexity for the inversion calculation is $O(L^3)$. That means that WELM takes $O(N^2L)$ more time. That is, when the number of training instances N is sufficiently large, we might observe a significant difference between WELM and ELM (LW-ELM) in terms of running time. In addition, with the increase of the number of training instances, this difference will increase.

```
Input
    Θ:        the training set with instances and the corresponding labels
    ψ:        the test set with instances and the corresponding labels
    ActType:  the type of the activation function in ELM
    L:        the number of hidden nodes
    C:        the penalty factor
Training procedure
    1.  calculate the CIR indexes in the training set Θ;
    2.  generate the initial expected output matrix T with labels in Θ;
    3.  for each training instance $x_i$ in Θ
    4.      find the corresponding column $t_{\cdot,i}$ in T;
    5.      if the data set is binary-class or multiclass
    6.          correct $t_{\cdot,i}$ by using Eq. (18) or Eq. (19);
    7.      else if the data set is multi-label
    8.          correct $t_{\cdot,i}$ by using Eq. (20) or Eq. (21);
    9.      end if
    10. end for
    11. generate the hidden-layer parameters randomly;
    12. calculate hidden-layer output matrix H with Θ and the hidden-layer parameters;
    13. calculate $\beta$ with H, T, and C, by using Eq. (4);
Testing procedure
    14. generate the expected output matrix T_TEST with labels in ψ;
    15. calculate hidden-layer output matrix H_TEST with ψ and the hidden-layer parameters;
    16. for each training instance $x_i'$ in ψ
    17.      calculate its actual output $T\_ACTUAL_{\cdot,i} = H\_TEST_i \times \beta$;
    18.      acquire the classification result by comparing $T\_ACTUAL_{\cdot,i}$ with $T\_TEST_{\cdot,i}$;
    19. end for
Output
    the final classification results on test set ψ
```

FIGURE 2. Description of the procedure of the LW-ELM algorithm.

TABLE 1. Binary-class and Multiclass CIL Data sets used in this paper.

Data set	#Attributes	#Instances	#Categories	IR	Minority class	Majority class
wisconsin	9	683	2	1.86	malignant	benign
pima	8	768	2	1.87	1	0
glass1	9	214	2	1.82	non-float processed	remainder
vehicle1	18	846	2	2.90	opel	remainder
abalone9-18	8	731	2	16.40	9	18
abalone19	8	4174	2	129.44	19	1~18
yeast-ME1	8	1484	2	32.73	ME1	remainder
yeast-ME2	8	1484	2	28.10	ME2	remainder
flare-F	11	1066	2	23.79	F	remainder
poker8-9-vs-5	10	2075	2	82.00	8,9	5
wilt	5	4839	2	17.54	1	2
page-blocks5	10	5473	2	46.59	5	1~4
magic	10	19020	2	1.84	1	2
letter-A	16	20000	2	24.35	A	B~Z
balance	4	625	3	5.88	-	-
contraceptive	9	1473	3	1.89	-	-
new-thyroid	5	215	3	4.84	-	-
wine	13	178	3	1.50	-	-
dermatology	34	366	6	5.55	-	-
yeast	8	1484	10	23.15	-	-
penbased	16	1100	10	1.95	-	-
lymphography	18	148	4	40.50	-	-
hayes-roth	4	132	3	1.70	-	-
page-blocks	10	548	5	164.00	-	-

#Attributes denotes the number of attributes, #Instances denotes the number of instances, #Categories denotes the number of classes, IR denotes the class-imbalance ratio, and Minority class and Majority class denote the corresponding original class labels, which have been divided into the binary classes.

To verify our analysis, we tried to track the variation of running time for these three algorithms with the increase in the number of training instances. Without loss of generality, binary-class two-dimensional synthetic data were adopted as the training set, where the majority class satisfies the normal distribution of mean 0.5 and variance 0.2 and the minority class satisfies the normal distribution of mean 0.2 and variance 0.1. CIR was set to 9.0. In addition, we generated 5 training sets with different scales, in which the numbers of instances are 100, 500, 1000, 5000 and 10000. ELM[1] and WELM[2] Matlab codes can be downloaded on the homepage of G.B. Huang in Nanyang Technological University and for all three algorithms, the number of hidden nodes is set to 100. The running time curves of these three algorithms are shown in Fig. 3.

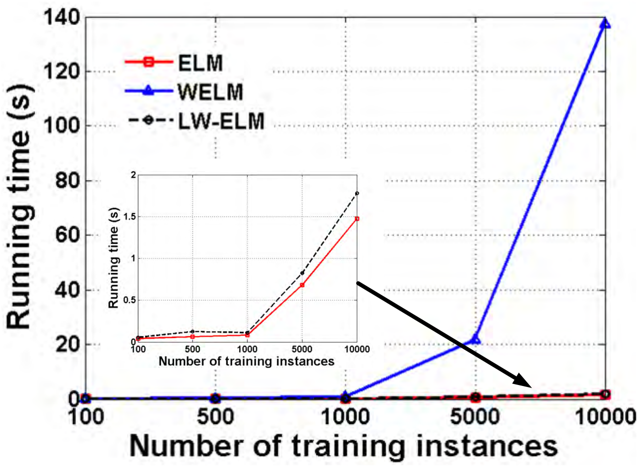


FIGURE 3. Running time curves of ELM, WELM and LW-ELM algorithms.

[1]http://www.ntu.edu.sg/home/egbhuang/elm_random_hidden_nodes.html

[2]http://www.ntu.edu.sg/home/egbhuang/elm_codes.html

In Fig. 3, when the number of training instances is less than a specific value, e.g., 1000, the three algorithms have similar running times, which could be neglected in practical applications. That means that when N is not sufficiently large, the running time is dominated by the other attributes, e.g., L. However, with the continued increase of the number of training instances, N will dominate the running time of each algorithm. At this moment, the time complexities of ELM and LW-ELM increase with N, while that of WELM increases with N^2. The experimental results in Fig. 3 well support our theoretical analysis.

V. EXPERIMENTS AND DISCUSSION

A. DATA SET DESCRIPTION

In our experiments, three types CIL data sets have been collected: binary-class, multiclass and multi-label sets. Specifically, 14 binary-class and 8 multiclass imbalanced data sets were collected from the UCI data repository [51] and 10 multi-label data sets were downloaded from MLC Toolbox [52], which is a Matlab toolbox for multi-label learning. To thoroughly present the trait of each comparison algorithm, the collected data sets were required to contain different numbers of features, numbers of instances and CIL indexes. Table 1-Table 2 present detailed descriptive information about these data sets.

B. EXPERIMENTAL SETTINGS

All experiments were run on a 2.60 GHz Intel(R) Core(TM) i7 6700HQ 8-core CPU with 16 GB RAM using the Matlab 2013a running environment.

On binary-class and multiclass CIL data sets, the LW-ELM algorithm was compared with several benchmark algorithms: ELM [34], W^{1}ELM [25], W^{2}ELM [25], RUS-ELM, ROS-ELM [40] and SMOTE-ELM [41]. Specifically, for all sampling algorithms, the class-imbalance ratio after sampling was required to be 1. In addition, for the SMOTE-ELM algorithm, the parameter for the number of nearest neighbors K has been assigned a default value of 5.

TABLE 2. Multi-label Data sets used in this paper.

Data set	$\|S\|$	$dim(S)$	$L(S)$	$F(S)$	$LCard(S)$	$LDen(S)$	$DL(S)$	$PDL(S)$	IR		
									min	max	avg
CAL500	502	68	124	numeric	25.058	0.202	502	1.000	1.040	24.390	3.846
Emotions	593	72	6	numeric	1.869	0.311	27	0.046	1.247	3.003	2.146
Medical	978	144	14	numeric	1.075	0.077	42	0.043	2.674	43.478	11.236
Enron	1702	50	24	nominal	3.113	0.130	547	0.321	1.000	43.478	5.348
Scene	2407	294	6	numeric	1.074	0.179	15	0.006	3.521	5.618	4.566
Yeast	2417	103	13	numeric	4.233	0.325	189	0.078	1.328	12.500	2.778
Corel5k	5000	499	44	nominal	2.214	0.050	1037	0.207	3.460	50.000	17.857
Rcv1(s1)	6000	472	42	numeric	2.458	0.059	574	0.096	3.344	50.000	15.152
Rcv1(s2)	6000	472	39	numeric	2.170	0.056	489	0.082	3.215	47.619	15.873
Tmc2007	28596	500	15	nominal	2.100	0.140	637	0.022	1.447	34.483	5.848

$|S|$, $dim(S)$, $L(S)$ and $F(S)$ denote the number of instances, number of attributes, number of class labels and the attribute type, respectively. IR denotes the class-imbalance ratio, including the minimal, maximal and average class-imbalance ratios. In addition, several other multi-label statistics are considered; please refer to Ref. [49].

TABLE 3. F-measure metric on Binary-class and Multiclass CIL Data sets.

Data set	ELM	W^1ELM	W^2ELM	RUS-ELM	ROS-ELM	SMOTE-ELM	LW^1ELM	LW^2ELM
wisconsin	.9563±.0032●	.9607±.0025	**.9645±.0020**	.9586±.0036	.9610±.0034	.9578±.0022●	.9626±.0011	.9578±.0011
pima	.6222±.0035●	.6696±.0039	**.6710±.0054**	.6640±.0048	.6591±.0063	.6591±.0105	.6511±.0075	.6386±.0098
glass1	.5436±.0168●	.6317±.0169	.5902±.0099●	.6261±.0144	.6276±.0118	.6396±.0094	**.6466±.0220**	.6190±.0379
vehicle1	.6079±.0163●	.6792±.0081●	.6676±.0072●	.6917±.0062●	.7054±.0072	**.7085±.0118**	.7016±.0115	.6634±.0202
abalone9-18	.3050±.0172●	.4612±.0153●	.3707±.0119●	.4281±.0342●	.4871±.0220●	.5057±.0177●	.3464±.0072	**.6362±.0315**
abalone19	.0000±.0000●	.0483±.0045	.0409±.0026	.0441±.0049	.0464±.0027	.0473±.0024	**.0486±.0013**	.0000±.0000
yeast-ME1	.0089±.0178●	.2618±.0116	.2081±.0019	.2442±.0128	.2656±.0193	.2760±.0148	.2341±.0087	**.2986±.0519**
yeast-ME2	.2694±.0254●	.4302±.0075●	.3914±.0100●	.4073±.0162●	.4779±.0106●	.4855±.0198●	.2696±.0066	**.6402±.0189**
flare-F	.0865±.0317●	.2736±.0080●	.2472±.0062●	.2421±.0021●	.2713±.0178●	.3051±.0135	.2847±.0136	**.3412±.0245**
poker8-9-vs-5	.0000±.0000●	.0691±.0076	.0461±.0078	.0283±.0065●	**.0929±.0111**○	.0910±.0066	.0651±.0072	.0000±.0000
wilt	.0000±.0000●	.2730±.0046●	.2000±.0023●	.4508±.0256○	.5608±.0046○	**.5664±.0049**○	.4096±.0025	.0931±.0067
page-blocks5	.3735±.0177●	.3121±.0046●	.2437±.0048●	.3509±.0121●	.3810±.0087●	.3923±.0025●	.2301±.0033	**.6105±.0073**
magic	.7649±.0019●	.7165±.0008●	.7154±.0006●	.7701±.0013●	.7729±.0011●	.7764±.0012●	**.7813±.0004**	.7766±.0005
letter-A	**.9016±.0009**○	.6853±.0066●	.5846±.0047●	.7546±.0126●	.8041±.0085●	.8077±.0101●	.3973±.0116	.8331±.0042
balance	.6300±.0013●	**.7488±.0098**	.6513±.0070●	.7059±.0131	.7192±.0143	.7120±.0095	.7328±.0067	.6409±.0146
contraceptive	.5060±.0118●	.5070±.0024●	.4509±.0028●	.5109±.0078●	**.5275±.0064**	.5229±.0065	.5263±.0017	.5198±.0049
new-thyroid	.7966±.0287●	.9002±.0147	.7973±.0241●	.8875±.0076●	**.9318±.0097**	.9181±.0194	.9272±.0145	.8681±.0267
wine	**.9879±.0090**	.9803±.0072	.9798±.0102	.9724±.0174	.9873±.0130	.9873±.0130	.9796±.0076	.9839±.0088
dermatology	.9695±.0075	**.9707±.0037**	.9678±.0059	.9339±.0141●	.9610±.0062●	.9585±.0119●	.9583±.0045	.9689±.0101
yeast	**.5083±.0146**	.4657±.0093●	.4292±.0097●	.3727±.0167●	.4878±.0134●	.4868±.0148●	.4367±.0061	.5077±.0078
penbased	.9593±.0044	.9433±.0039●	.9399±.0045●	**.9599±.0049**	.9531±.0052	.9528±.0043●	.9566±.0025	.9583±.0031
lymphography	.5587±.0463●	.6273±.0381	.6000±.0456●	.4190±.0101●	.6196±.0615	.6204±.0303	.4770±.0338	**.6422±.0444**
hayes-roth	.7100±.0203	.7052±.0140	.6106±.0120●	.6633±.0267	.7122±.0268	.6473±.0390●	**.7192±.0158**	.7116±.0123
page-blocks	.5176±.0419●	.5493±.0367●	.5372±.0304●	.4118±.0420●	.5729±.0236●	.5696±.0199●	.4241±.0149	**.6580±.0289**
win/tie/loss	**18/5/1**	**12/12/0**	**17/7/0**	**14/9/1**	**9/13/2**	**11/12/1**	-	-

Results are given in the form of mean±standard deviation, where the best result in each row has been highlighted in boldface. ●/○ indicates whether the best LW-ELM algorithm is statistically superior/inferior to the comparing algorithm on each data set (pairwise *t*-test at 5% significance level).

The two most popular performance evaluation metrics, namely, *F-measure* and *G-mean*, were adopted to compare different algorithms. In particular, *F-m easure* measures the trade-off between *precision* and *recall*, while *G-mean* measures the trade-off among the accuracies that belong to different categories.

On multi-label data sets, the proposed LW-ELM algorithm was compared with several existing multi-label CIL algorithms, namely, ML-ELM [34], ML-ROS [47], ML-SMOTE [48], IRUS [46] and COCOA [49]. All parameters in these algorithms were assigned the default values that were provided by the corresponding references. Two popular metrics, namely, Macro-F and Micro-F, have been adopted for evaluating the performance of each algorithm.

All the algorithms except COCOA use ELM to construct the classification model. Hence, we need to select an activation function and designate two parameters, namely, C and L, in advance. Here, the most popular activation function, namely, the *sigmoid*function, was adopted, while $L$and Cwere determined by grid search by internal 5-fold cross-validation, where $L \in \{10, 20, \ldots, 200\}$ & $C \in \{2^{-4}, 2^{-2}, \ldots, 2^{20}\}$.

In addition, considering the randomness of the experiments, the experimental results might be unstable. Therefore, we use 10 random 5-fold cross-validations to calculate the average result for each algorithm. All experimental results were given in the form of mean ± standard deviation.

C. RESULTS ON BINARY-CLASS AND MULTICLASS IMBALANCED DATA SETS

Table 3 and Table 4 present detailed experimental results on binary-class and multiclass CIL data sets in terms of the *F-measure* and *G-mean* metrics, respectively. The best result on each data set has been highlighted in boldface. Furthermore, to show whether the best LW-ELM algorithm performs significantly better/worse than the comparison algorithms, a pairwise t-test at 5% significance level has been conducted. Accordingly, a win/loss frequency was counted and a marker ●/○ is shown in tables whenever the best LW-ELM algorithm achieves significantly superior/inferior performance on a data set. Otherwise, a tie was declared and no marker was given. The overall win/tie/loss counts across all data sets were summarized in the last row of each table.

TABLE 4. G-mean metric on Binary-class and Multiclass CIL Data sets.

Data set	ELM	W^1ELM	W^2ELM	RUS-ELM	ROS-ELM	SMOTE-ELM	LW^1ELM	LW^2ELM
wisconsin	.9682±.0027●	.9745±.0016	**.9786±.0011**	.9726±.0028	.9739±.0035	.9710±.0019●	.9760±.0014	.9708±.0016
pima	.6987±.0033●	**.7463±.0043**	.7367±.0052	.7415±.0066	.7372±.0052	.7373±.0090	.7305±.0073	.7163±.0071
glass1	.6308±.0122●	.6853±.0214●	.5795±.0169●	.6797±.0142	.6935±.0142	.7017±.0086	**.7199±.0178**	.6964±.0286
vehicle1	.7071±.0112●	.8117±.0077	.8088±.0058●	.8238±.0050	**.8325±.0065**	.8316±.0081	.8265±.0079	.7731±.0144
abalone9-18	.4017±.0210●	.8384±.0137	.8403±.0150	.8306±.0183	.8530±.0121	**.8604±.0109**	.8554±.0099	.7671±.0277
abalone19	.0000±.0000●	.7356±.0403	**.7575±.0260**	.7341±.0404	.7101±.0164	.7119±.0180	.7404±.0155	.0000±.0000
yeast-ME1	.0141±.0283●	.8130±.0159	.8100±.0087	.8189±.0194	.8128±.0246	.8034±.0189	**.8231±.0082**	.4570±.0745
yeast-ME2	.3692±.0296●	.9599±.0007○	.9525±.0004○	.9541±.0035○	**.9649±.0047**○	.9577±.0067○	.9147±.0007	.9250±.0118
flare-F	.1548±.0579●	.8012±.0102	**.8173±.0127**	.8027±.0173	.7792±.0279●	.7907±.0140	.8072±.0145	.5773±.0300
poker8-9-vs-5	.0000±.0000●	.6349±.0684	.6271±.0882	.4797±.1173●	.7020±.0352	.6966±.0420	**.7127±.0289**	.0000±.0000
wilt	.0000±.0000●	.7940±.0050●	.7358±.0039●	.9041±.0102	.9393±.0017○	**.9404±.0017**○	.9114±.0010	.2259±.0182
page-blocks5	.5029±.0140●	.8967±.0063	.8939±.0065	.9106±.0040○	.9168±.0080○	**.9217±.0027**○	.8874±.0058	.8566±.0071
magic	.8033±.0016●	.7797±.0007●	.7807±.0005●	.8233±.0009●	.8254±.0008●	.8278±.0010	**.8287±.0004**	.8187±.0004
letter-A	.9143±.0010●	.9520±.0017○	.9507±.0008○	.9613±.0016○	**.9709±.0011**○	.9689±.0020○	.9174±.0041	.9373±.0008
balance	.0000±.0000●	**.7817±.0231**○	.7616±.0099○	.7596±.0102○	.7753±.0201○	.7676±.0147○	.6449±.0195	.0625±.0806
contraceptive	.4825±.0131●	.5197±.0028●	.4571±.0035●	.5220±.0082	**.5367±.0072**	.5297±.0062	.5302±.0013	.5096±.0061
new-thyroid	.6814±.0670●	.8819±.0137●	.8750±.0179●	.8661±.0159●	.9051±.0106	.8900±.0206●	**.9258±.0205**	.7815±.0685
wine	**.9901±.0068**	.9830±.0046	.9824±.0079	.9755±.0133	.9802±.0049	.9882±.0109	.9819±.0055	.9861±.0065
dermatology	.9694±.0092	**.9708±.0046**	.9681±.0057	.9412±.0106●	.9614±.0077●	.9601±.0100●	.9622±.0052	.9698±.0099
yeast	.0000±.0000	.2316±.0805●	.3586±.0762○	.0878±.0577●	.2881±.0814	**.3952±.0455**○	.2979±.0423	.0000±.0000
penbased	**.9588±.0048**	.9416±.0028●	.9365±.0054●	.9587±.0057	.9516±.0065	.9519±.0048●	.9551±.0026	.9571±.0039
lymphography	.4942±.0654●	**.8854±.0081**	.8445±.0817●	.6087±.0625●	.7293±.1191●	.7902±.0827●	.7890±.0209	.8739±.0775
hayes-roth	.7107±.0160	.7063±.0163	.5874±.0237●	.6604±.0304	.7057±.0238	.6486±.0368●	**.7200±.0177**	.7130±.0170
page-blocks	.1870±.1306●	.5981±.1392●	.6117±.0894●	.4448±.1796●	.5037±.1288●	.4727±.1006●	**.7768±.0583**	.6235±.1272
win/tie/loss	**19/5/0**	**8/13/3**	**10/10/4**	**7/13/4**	**5/15/5**	**7/11/6**	**-**	**-**

Results are given in the form of mean±standard deviation, where the best result in each row has been highlighted in boldface. ●/○ indicates whether the best LW-ELM algorithm is statistically superior/inferior to the comparison algorithm on each data set (pairwise *t*-test at 5% significance level).

From the results in Table 3 and Table 4, we observe the following:

1) On binary-class and multiclass CIL data sets, both sampling and cost-sensitive learning could promote the quality of the classification model to some extent. This conclusion can be drawn by comparing the results between any CIL algorithm and the original ELM algorithm. On some simple data sets, e.g., wisconsin, wine and penbased, the improvement might not be presented clearly; this phenomenon has been well explained and analyzed in several previous works [27], [53]. That is, the harmfulness of a class-imbalanced distribution on traditional classification models is correlated with multiple complex factors, including the class-imbalance ratio, class overlap, small disjunctions and the ratio of noise. However, on some extremely difficult data sets, e.g., abalone19, poker8-9-vs-5 and wilt, several CIL algorithms have improved the classification performance substantially.
2) In terms of F-measure (Table 3), our proposed LW-ELM algorithm significantly outperforms the comparison algorithms in 75% (ELM), 50% (W^1ELM), 70.84% (W^2ELM), 58.33% (RUS-ELM), 37.5% (ROS-ELM) and 45.83% (SMOTE-ELM) of cases and is only significantly inferior to several other algorithms once or twice. These results indicate that LW-ELM is capable of achieving a good balance between predictive exactness (precision) and completeness (recall) in handling CIL data. Specifically, the LW-ELM algorithm has achieved the best F-measure on 11 data sets, including 4 by LW^1ELM and 7 by LW^2ELM.
3) In terms of G-mean, the LW-ELM algorithm significantly outperforms the comparison algorithms in 79.16% (ELM), 33.33% (W^1ELM), 41.67% (W^2ELM), 29.17% (RUS-ELM), 20.83% (ROS-ELM) and 29.17% (SMOTE-ELM) of cases but has been outperformed in 0% (ELM), 12.5% (W^1ELM), 16.67% (W^2ELM), 16.67% (RUS-ELM), 20.83% (ROS-ELM) and 25% (SMOTE-ELM) of cases. Therefore, compared with ELM and W^2ELM, our proposed LW-ELM performs significantly better. However, compared with several other algorithms, it could only produce comparable performance. Specifically, the LW-ELM algorithm only obtains the best result on 7 data sets, which is the same as the WELM algorithm. Therefore, in terms of running time, the LW-ELM algorithm is a successful alternative to the WELM algorithm, but the LW-ELM algorithm does not outperform WELM in terms of classification performance.
4) We also note an interesting phenomenon that LW^1ELM tends to yield a better G-mean metric, while LW^2ELM is more appropriate for obtaining a high F-measure metric. We believe that this phenomenon is correlated with the weighting strategies that are adopted by these two algorithms, as LW^1ELM uses a radical weighting strategy, which can promote the accuracy of the minority class by sacrificing that of the majority class to a large extent, while LW^2ELM is more moderate and is helpful for finding the best tradeoff between precision and recall. That is, it is recommended for the user to select an appropriate version according to his or her practical requirements.
5) In addition, on highly skewed data sets, e.g., abalone19 and poker8-9-vs-5, it is easier to achieve excellent classification performance using the LW^1ELM algorithm, while LW^2ELM generally lacks stability on these data sets. In other words, the highly imbalanced data requires augmenting the instance weight that belongs to the minority category to a large extent.

D. RESULTS ON MULTI-LABEL DATA SETS

Table 5 and Table 6 show the results of seven comparison algorithms on 10 multi-label data sets, in terms of Macro_F

TABLE 5. Macro_F metric on multi-label data sets.

Data set	ML-ELM	ML-ROS	ML-SMOTE	IRUS	COCOA	LW^{1}ELM	LW^{2}ELM
CAL500	.0789±.0029●	.1189±.0033●	.1662±.0030●	**.2245±.0010**	.2055±.0024	.2121±.0027	.1302±.0042
Emotions	.3927±.0191●	.4033±.0305●	.4784±.0311●	.5872±.0053○	**.6223±.0051**○	.5326±.0130	.4717±.0204
Medical	.5505±.0064●	.6452±.0052	.6527±.0046	**.6816±.0021**○	.6569±.0059	.5810±.0012	.6472±.0046
Enron	.1836±.0017●	.2420±.0014●	.2331±.0019●	.2778±.0006	.2103±.0058●	.2336±.0011	**.2843±.0079**
Scene	.6079±.0075●	.6483±.0130●	.6502±.0067●	.7148±.0022	**.7267±.0049**	.5445±.0031	.7261±.0035
Yeast	.3509±.0017●	.3981±.0036●	.4205±.0030●	**.4787±.0014**○	.4513±.0048	.4464±.0025	.4355±.0041
Corel5k	.0379±.0116●	.0891±.0296●	.0792±.0219●	.0981±.0139●	.1132±.0014●	.1479±.0217	**.1822±.0322**
Rcv1(s1)	.0310±.0070●	.0707±.0061●	.1203±.0064●	.1956±.0071●	.2398±.0052	.0959±.0043	**.2568±.0057**
Rcv1(s2)	.0724±.0063●	.1920±.0038●	.2422±.0132	.1727±.0059●	**.2612±.0034**	.1430±.0037	.2520±.0033
Tmc2007	.3865±.0023●	.4182±.0030●	.4196±.0035●	.5743±.0003	**.6359±.0035**○	.4731±.0003	.6180±.0011
win/tie/loss	10/0/0	9/1/0	8/2/0	3/4/3	2/6/2	-	-

Results are given in the form of mean±standard deviation, where the best result in each row has been highlighted in boldface. ●/○ indicates whether the best LW-ELM algorithm is statistically superior/inferior to the comparing algorithm on each data set (pairwise *t*-test at 5% significance level).

TABLE 6. MIcro_F metric on multi-label data sets.

Data set	ML-ELM	ML-ROS	ML-SMOTE	IRUS	COCOA	LW^{1}ELM	LW^{2}ELM
CAL500	.3325±.0020	.3339±.0044	.3322±.0025	.2892±.0012●	**.4515±.0026**○	.2597±.0028	.3467±.0054
Emotions	.4260±.0202●	.4360±.0295●	.4611±.0290●	.5943±.0052○	**.6300±.0044**○	.5391±.0128	.4916±.0191
Medical	.6261±.0097○	.6221±.0071○	.6271±.0086○	.2130±.0046●	**.8180±.0049**○	.1136±.0009	.4707±.0037
Enron	.4836±.0048	.4801±.0043	.4793±.0065	.2405±.0020●	**.5231±.0044**○	.1714±.0014	.4847±.0043
Scene	.6063±.0073●	.6381±.0129●	.6498±.0065●	.6977±.0021●	.7182±.0053	.5424±.0031	**.7197±.0032**
Yeast	.6306±.0018●	.6461±.0025●	.6786±.0017●	.5782±.0015●	.6494±.0019●	.5942±.0025	**.7098±.0017**
Corel5k	.0410±.0107●	.0782±.0232●	.0691±.0177●	.1488±.0346●	.1864±.0022●	.1792±.0199	**.2360±.0428**
Rcv1(s1)	.1707±.0025●	.1999±.0018●	.2266±.0015●	.2334±.0024●	.4007±.0028	.1025±.0004	**.4268±.0013**
Rcv1(s2)	.1700±.0024●	.2687±.0024●	.2403±.0033●	.2158±.0009●	**.4173±.0021**○	.1828±.0012	.4076±.0022
Tmc2007	.6136±.0076●	.6408±.0081●	.6700±.0040●	.6475±.0093●	.7325±.0015	.5208±.0074	**.7357±.0111**
win/tie/loss	7/2/1	7/2/1	7/2/1	9/0/1	2/3/5	-	-

Results are given in the form of mean±standard deviation, where the best result in each row has been highlighted in boldface. ●/○ indicates whether the best LW-ELM algorithm is statistically superior/inferior to the comparison algorithm on each data set (pairwise *t*-test at 5% significance level).

and Micro_F, respectively. Pairwise *t*-test at 5% significance level has also been conducted to determine the significant difference between any algorithm and the best LW-ELM algorithm.

LW-ELM significantly outperforms the ML-ELM, ML-ROS and ML-SMOTE algorithms in terms of both the Macro_F and Micro_F metrics. LW-ELM only performs worse than those three algorithms on the Medical data set in terms of Micro_F. Thus, LW-ELM significantly outperforms those three algorithms. For the two other algorithms, namely, IRUS and COCOA, we observe that in terms of the Macro_F metric, LW-ELM has presented similar performance to those two algorithms, whereas in terms of the Micro_F metric, LW-ELM performs obviously better than IRUS and worse than COCOA. To clarify the reason the IRUS and COCOA achieve better or at least similar performance to our proposed LW-ELM algorithm, we must investigate the intrinsic mechanism that is hidden behind these two algorithms. Ensemble learning, which takes advantage of the diversity of multiple base classification models, generally improves the performance of the classification model. Therefore, both IRUS and COCOA profit from ensemble learning. Additionally, in addition to ensemble learning, COCOA considers the label correlations, which might provide important hidden information for improving the quality of multi-label classification models.

Compared with LW^{1}ELM, LW^{2}ELM performs significantly better, except on several data sets with low average class-imbalance ratios, e.g., CAL500, Emotions and Yeast. We believe the reason lies in that regardless of whether Macro_F or Micro_F deviates in terms of F-measure, as in our analysis in the previous subsection, LW^{2}ELM is more appropriate for producing high F-measure than LW^{1}ELM. Accordingly, we recommend LW^{2}ELM for multi-label CIL scenarios.

E. COMPARISON OF RUNNING TIME

We should also consider the computation times of various comparison algorithms. The average running times of various algorithms are summarized in Table 7 (binary-class and multiclass data sets) and Table 8 (multi-label data sets).

According to Table 7, the LW-ELM algorithm generally consumes slightly more running time than both the ELM and RUS-ELM algorithms but has obviously lower time complexity than the ROS-ELM, SMOTE-ELM and WELM algorithms. This phenomenon is clearer on large-sample data sets, e.g., abalone19, wilt, page-blocks5, magic and letter-A. This phenomenon occurs because ROS-ELM increases the number of training instances, SMOTE-ELM requires neighborhood calculations to be frequently performed, and WELM must deal with the problem of large-matrix multiplication, which inevitably increase the training time consumption. The results are consistent with our theoretical analysis. On several large-sample data sets, the proposed LW-ELM algorithm can reduce the running time by several hundred times compared

TABLE 7. Running time (seconds) of various comparison algorithms on binary-class and multiclass data sets.

Data set	ELM	W^1ELM	W^2ELM	RUS-ELM	ROS-ELM	SMOTE-ELM	LW^1ELM	LW^2ELM
wisconsin	0.2527	0.4805	0.3650	0.1061	0.2246	2.8205	0.0998	0.1498
pima	0.1404	0.4742	0.4992	0.1373	0.2090	3.3259	0.1622	0.1622
glass1	.0.0874	0.0718	0.0967	0.0499	0.0374	0.5427	0.0593	0.0624
vehicle1	0.0967	0.5616	0.4493	0.1591	0.3214	4.1964	0.0874	0.1872
abalone9-18	0.1654	0.4462	0.3931	0.0780	0.2902	1.5101	0.1373	0.1498
abalone19	0.6926	15.7873	15.7436	0.1622	2.9640	4.2276	0.7519	0.7831
yeast-ME1	0.2122	1.7753	1.7878	0.0655	0.7644	2.6333	0.3557	0.2464
yeast-ME2	0.2246	1.6692	1.6755	0.1092	0.7145	2.4211	0.2839	0.3120
flare-F	0.2184	0.8050	0.8424	0.0250	0.5335	2.0811	0.2465	0.1092
poker8-9-vs-5	0.3214	3.1294	3.1450	0.0374	1.2418	2.7706	0.3650	0.3245
wilt	0.6926	19.1413	19.4221	0.2246	2.6801	14.0775	0.8018	0.8798
page-blocks5	0.7488	25.1973	25.8556	0.2059	3.9531	10.0371	1.0015	0.9953
magic	2.5865	715.5672	713.0057	3.6660	8.9857	743.0172	2.9921	3.2074
letter-A	2.7799	820.1191	818.9491	0.9142	24.1864	172.9021	2.9609	3.2230
balance	0.1373	0.2496	0.2652	0.0499	0.2028	2.8143	0.0842	0.1061
contraceptive	0.2028	1.3478	1.4602	0.1997	0.3432	7.5910	0.1841	0.1498
new-thyroid	0.0312	0.1248	0.0624	0.0250	0.0374	0.5366	0.0250	0.0125
wine	0.0686	0.0624	0.1248	0.0468	0.1092	0.5741	0.0624	0.0780
dermatology	0.1248	0.0905	0.2090	0.0998	0.2153	1.5756	0.0998	0.1373
yeast	0.4805	1.7379	1.6099	0.4056	1.2386	7.6035	0.4306	0.4493
penbased	0.4056	1.0670	1.0733	0.3869	0.4368	3.4632	0.3650	0.3869
lymphography	0.0406	0.0811	0.0250	0.0749	0.0499	0.2995	0.0374	0.0437
hayes-roth	0.0250	0.0125	0.0499	0.0593	0.0374	0.2215	0.0374	0.0125
page-blocks	0.1123	0.2496	0.2153	0.0874	0.4742	1.7410	0.0624	0.1342

TABLE 8. Running time (seconds) of various comparison algorithms on multi-label data sets.

Data set	ML-ELM	ML-ROS	ML-SMOTE	IRUS	COCOA	LW^1ELM	LW^2ELM
CAL500	0.0324	0.2989	0.5245	40.2717	5.6813	0.0384	0.0384
Emotions	0.0156	0.0452	0.0867	2.1625	0.5666	0.0215	0.0209
Medical	0.0721	0.3173	0.2998	57.9088	0.9928	0.0721	0.0705
Enron	0.1223	0.8823	0.9388	84.4249	8.8927	0.0883	0.0980
Scene	0.0402	0.2640	0.5139	6.5511	3.2804	0.0546	0.0452
Yeast	0.0390	0.5772	0.5816	12.0383	3.6819	0.0415	0.0365
Corel5k	0.6000	1.7020	1.8319	1183.4000	9.9432	0.8127	0.8159
Rcv1(s1)	0.6608	3.9412	3.9116	809.4733	23.7880	0.6705	0.6979
Rcv1(s2)	0.6649	3.6582	3.8005	867.7908	26.6537	0.7382	0.7423
Tmc2007	1.3862	85.4514	92.2652	365.0199	32.3617	1.4661	1.4187

with the WELM algorithm. That means that LW-ELM could be an efficient alternative to WELM.

In Table 8, two interesting phenomena are observed: First, as an undersampling algorithm, IRUS requires much longer training time than the two oversampling algorithms, namely, ML-ROS and ML-SMOTE. It is correlated with the ensemble mechanism that is adopted by IRUS, which extremely undersamples the majority class to train a set of base classifiers. The number of base classifiers that should be constructed depends on multiple factors, such as the number of class labels, the number of instances and the average class-imbalance ratio. Therefore, it is not difficult to explain why IRUS consumes more training time on Corel5k data sets than that on Rcv1(s1), Rcv1(s2) and Tmc2007 data sets. Second, as another ensemble learning algorithm, COCOA requires substantially less running time but achieves significantly better performance than IRUS. It is easy to explain this phenomenon as follows: COCOA integrates significantly fewer base classifiers than IRUS but leverages exploitation of label correlations and exploration of class imbalance, while IRUS only considers class imbalance.

The LW-ELM algorithm still has very low time-complexity, which is only slightly higher than that of ML-ELM. On large-scale data sets with many class labels, LW-ELM could reduce the running time by hundreds or even thousands of times compared with the more time-consuming algorithms, e.g., IRUS. Meanwhile, LW-ELM has achieved excellent classification performance on most multi-label data sets. Hence, it is a good choice for dealing with multi-label class imbalance problems.

VI. CONCLUSIONS

In this paper, a fast and feasible cost-sensitive learning algorithm, namely, label-weighted extreme learning machine (LW-ELM), is proposed. As an alternative to the traditional weighted extreme learning machine (WELM) algorithm, LW-ELM has two main advantages: 1) it achieves substantially faster training speed on large-scale data sets by eliminating a large-matrix multiplication operation, and 2) it can be used to cope with the class-imbalance problem in multi-label data by regulating the weight of each class label. Specifically, for different purposes, two diverse label weight assignment strategies have been proposed, of which one is drastic and the other is moderate. We tried to verify the effectiveness and feasibility of LW-ELM theoretically and empirically evaluated the LW-ELM algorithm by comparing it to six other algorithms on 24 binary-class and multiclass CIL data sets and to five algorithms on 10 multi-label data sets. LW-ELM achieves the best performance of all compared algorithms on single-label data sets and better performance than all algorithms except COCOA on multi-label data sets. It has significantly lower time complexity than the other algorithms, which increases its practicability in real-world applications.

In future work, the effectiveness of LW-ELM will be further evaluated in additional real-world application scenarios. Additionally, the integration of LW-ELM into the Boosting ensemble learning and online learning frameworks will be investigated.

REFERENCES

[1] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[2] H. Guo, Y. Li, J. Shang, M. Gu, Y. Huang, and B. Gong, "Learning from class-imbalanced data: Review of methods and applications," *Expert Syst. Appl.*, vol. 73, pp. 220–239, Dec. 2017.

[3] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, 2013.

[4] A. Zakaryazad and E. Duman, "A profit-driven artificial neural network (ANN) with applications to fraud detection and direct marketing," *Neurocomputing*, vol. 175, pp. 121–131, Jan. 2016.

[5] M. Kirlidog and C. Asuk, "A fraud detection approach with data mining in health insurance," *Procedia-Soc. Behav. Sci.*, vol. 62, pp. 989–994, Oct. 2012.

[6] J. A. Sanz, D. Bernardo, F. Herrera, H. Bustince, and H. Hagras, "A compact evolutionary interval-valued fuzzy rule-based classification system for the modeling and prediction of real-world financial applications with imbalanced data," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 4, pp. 973–990, Aug. 2015.

[7] H. Li and M.-L. Wong, "Financial fraud detection by using Grammar-based multi-objective genetic programming with ensemble learning," in *Proc. IEEE Congr. Evol. Comput.*, May 2015, pp. 1113–1120.

[8] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *IEEE Trans. Rel.*, vol. 62, no. 2, pp. 434–443, Jun. 2013.

[9] D. Rodriguez, I. Herraiz, R. Harrison, J. Dolado, and J. C. Riquelme, "Preliminary comparison of techniques for dealing with imbalance in software defect prediction," in *Proc. 18th Int. Conf. Eval. Assessment Softw. Eng.*, 2014, p. 43.

[10] V. Engen, J. Vincent, and K. Phalp, "Enhancing network based intrusion detection for imbalanced data," *Int. J. Knowl.-Based Intell. Eng. Syst.*, vol. 12, nos. 5–6, pp. 357–367, 2008.

[11] S. Hajian, J. Domingo-Ferrer, and A. Martinez-Balleste, "Discrimination prevention in data mining for intrusion and crime detection," in *Proc. IEEE Symp. Comput. Intell. Cyber Secur.*, Apr. 2011, pp. 47–54.

[12] R. Dubey, J. Zhou, Y. Wang, P. M. Thompson, J. Ye, and Alzheimer's Disease Neuroimaging Initiative, "Analysis of sampling techniques for imbalanced data: An n=648 ADNI study," *NeuroImage*, vol. 87, pp. 220–241, Feb. 2014.

[13] T.-H. Cheng and P. J.-H. Hu, "A data-driven approach to manage the length of stay for appendectomy patients," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 39, no. 6, pp. 1339–1347, Nov. 2009.

[14] H. Yu and J. Ni, "An improved ensemble learning method for classifying high-dimensional and imbalanced biomedicine data," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 11, no. 4, pp. 657–666, Jul. 2014.

[15] L. Song, D. Li, X. Zeng, Y. Wu, L. Guo, and Q. Zou, "nDNA-prot: Identification of DNA-binding proteins based on unbalanced classification," *BMC Bioinf.*, vol. 15, no. 1, p. 298, 2014.

[16] R. Blagus and L. Lusa, "SMOTE for high-dimensional class-imbalanced data," *BMC Bioinf.*, vol. 14, no. 1, p. 106, 2013.

[17] S. Cateni, V. Colia, and M. Vannucci, "A method for resampling imbalanced datasets in binary classification tasks for real-world problems," *Neurocomputing*, vol. 135, pp. 32–41, Jul. 2014.

[18] W.-Z. Lu and D. Wang, "Ground-level ozone prediction by support vector machine approach with a cost-sensitive classification scheme," *Sci. Total Environ.*, vol. 395, nos. 2–3, pp. 109–116, 2008.

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, 2002.

[20] E. Ramentol, Y. Caballero, R. Bello, and F. Herrera, "SMOTE-RSB*: A hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory," *Knowl. Inf. Syst.*, vol. 33, no. 2, pp. 245–265, 2012.

[21] S. Chen, H. He, and E. A. Garcia, "RAMOboost: Ranked minority oversampling in boosting," *IEEE Trans. Neural Netw.*, vol. 21, no. 10, pp. 1624–1642, Oct. 2010.

[22] H. Yu, J. Ni, and J. Zhao, "ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data," *Neurocomputing*, vol. 101, pp. 309–318, Feb. 2013.

[23] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognit.*, vol. 40, no. 12, pp. 3358–3378, 2007.

[24] C. L. Castro and A. P. Braga, "Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 6, pp. 888–899, Jun. 2013.

[25] W. Zong, G.-B. Huang, and L. Chen, "Weighted extreme learning machine for imbalance learning," *Neurocomputing*, vol. 101, pp. 229–242, Feb. 2013.

[26] S. Datta and S. Das, "Near-Bayesian support vector machines for imbalanced data classification with equal or unequal misclassification costs," *Neural Netw.*, vol. 70, pp. 39–52, Oct. 2015.

[27] H. Yu, C. Sun, X. Yang, W. Yang, J. Shen, and Y. Qi, "ODOC-ELM: Optimal decision outputs compensation-based extreme learning machine for classifying imbalanced data," *Knowl.-Based Syst.*, vol. 92, pp. 55–70, Jan. 2016.

[28] V. López, S. del Río, J. M. Benítez, and F. Herrera, "Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data," *Fuzzy Sets Syst.*, vol. 258, pp. 5–38, Jan. 2015.

[29] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 4, pp. 463–484, Jul. 2012.

[30] S. Wang, L. Minku, and X. Yao, "Resampling-based ensemble methods for online class imbalance learning," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1356–1368, May 2015.

[31] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 1, pp. 185–197, Jan. 2010.

[32] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 63–77, Jan. 2006.

[33] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.

[34] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.

[35] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: A review," *Neural Netw.*, vol. 61, pp. 32–48, Jan. 2015.

[36] C. Deng, G. Huang, J. Xu, and J. Tang, "Extreme learning machines: New trends and applications," *Sci. China Inf. Sci.*, vol. 58, no. 2, pp. 1–16, 2015.

[37] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Prog. Artif. Intell.*, vol. 5, no. 4, pp. 221–232, 2016.

[38] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 23, no. 4, pp. 687–719, 2009.

[39] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Inf. Sci.*, vol. 250, pp. 113–141, Nov. 2013.

[40] C.-M. Vong, W.-F. Ip, P.-K. Wong, and C.-C. Chiu, "Predicting minority class for suspended particulate matters level by extreme learning machine," *Neurocomputing*, vol. 128, pp. 136–144, Mar. 2014.

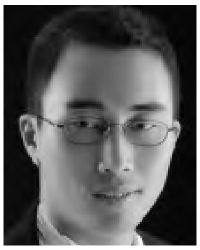
[41] S.-J. Sun, C. Chang, and M.-F. Hsu, "Multiple extreme learning machines for a two-class imbalance corporate life cycle prediction," *Knowl.-Based Syst.*, vol. 39, pp. 214–223, Feb. 2013.

[42] W. B. Zhang and H. B. Ji, "Fuzzy extreme learning machine for classification," *Electron. Lett.*, vol. 49, no. 7, pp. 448–450, Mar. 2013.

[43] L. Zhang and D. Zhang, "Evolutionary cost-sensitive extreme learning machine," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 12, pp. 3045–3060, Dec. 2017, doi: 10.1109/TNNLS.2016.2607757.

[44] K. Li, X. Kong, Z. Lu, L. Wenyin, and J. Yin, "Boosting weighted elm for imbalanced learning," *Neurocomputing*, vol. 128, pp. 15–21, Mar. 2014.

[45] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.
[46] M. A. Tahir, J. Kittler, and F. Yan, "Inverse random under sampling for class imbalance problem and its application to multi-label classification," *Pattern Recognit.*, vol. 45, no. 10, pp. 3738–3750, 2012.
[47] F. Charte, A. J. Rivera, M. J. del Jesus, and F. Herrera, "Addressing imbalance in multilabel classification: Measures and random resampling algorithms," *Neurocomputing*, vol. 163, pp. 3–16, Sep. 2015.
[48] F. Charte, A. J. Rivera, M. J. del Jesus, and F. Herrera, "MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation," *Knowl.-Based Syst.*, vol. 89, pp. 385–397, Nov. 2015.
[49] M.-L. Zhang, Y.-K. Li, and X.-Y. Liu, "Towards class-imbalance aware multi-label learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2015, pp. 4041–4047.
[50] X. Sun, J. Xu, C. Jiang, J. Feng, S.-S. Chen, and F. He, "Extreme learning machine for multi-label classification," *Entropy*, vol. 18, no. 6, p. 225, 2016.
[51] C. Blake, E. Keogh, and C. J. Merz, "UCI repository of machine learning databases," Dept. Inf. Comput. Sci., Univ. California, Irvine, CA, USA, Tech. Rep. 213, 1998. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html
[52] K. Kimura, L. Sun, and M. Kudo. (2017). "MLC toolbox: A MATLAB/OCTAVE library for multi-label classification." [Online]. Available: https://arxiv.org/abs/1704.02592
[53] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–450, 2002.



HUALONG YU was born in Harbin, China, in 1982. He received the B.S. degree from Heilongjiang University, Harbin, in 2005, and the M.S. and Ph.D. degrees from Harbin Engineering University, Harbin, in 2008 and 2010, respectively, all in computer science.

Since 2010, he has been an Associate Professor with the School of Computer, Jiangsu University of Science and Technology, Zhenjiang, China. From 2013 to 2017, he was a Post-Doctoral Fellow with the School of Automation, Southeast University, Nanjing, China. Since 2017, he has been a Senior Visiting Scholar with the Faculty of Information Technology, Monash University, Melbourne, Australia. He has authored or co-authored over 60 journal and conference papers and six monographs. His research interests include machine learning, data mining, and bioinformatics.

Dr. Yu is the member of the ACM, China Computer Federation, and the Youth Committee of the Chinese Association of Automation. He is an Active Reviewer for over 20 high-quality international journals, including the IEEE TNNLS, TCYB, TKDE, and TCBB and a member in the organizing committee of several international conferences.

CHANGYIN SUN received the B.S. degree from the Department of Mathematics, Sichuan University, Chengdu, China, in 1996, and the M.S. and Ph.D. degrees in electrical engineering from Southeast University, Nanjing, China, in 2001 and 2003, respectively.

He was a Post-Doctoral Fellow with the Department of Computer Science, The Chinese University of Hong Kong, Hong Kong, in 2004. From 2006 to 2007, he was a Professor with the School of Electrical Engineering, Hohai University, Nanjing. Since 2007, he has been a Professor with the School of Automation, Southeast University. His current research interests include intelligent control, neural networks, support vector machine, data-driven control, theory and design of intelligent control systems, optimization algorithms, and pattern recognition.

Dr. Sun has been an Associate Editor of *Neural Processing Letters* since 2008, *Recent Parents of Computer Science* since 2008, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS since 2010, and the *International Journal of Swarm Intelligence Research* since 2010. He has been also involved in organizing several major international conferences.

XIBEI YANG received the B.S. degree in computer sciences from Xuzhou Normal University, Xuzhou, China, in 2002, the M.S. degree in computer applications from the Jiangsu University of Science and Technology, Zhenjiang, China, in 2006, and the Ph.D. degree in Pattern Recognition and Intelligence System from the Nanjing University of Science and Technology, Nanjing, China, in 2010.

Since 2010, he has been an Associate Professor with the School of Computer Science and Engineering, Jiangsu University of Science and Technology. He has published over 100 research articles on the professional journals and conferences. His research interests include granular computing and rough set theory.

Dr. Yang is the reviewer for over 10 high-quality international journals and a member in the organizing committee of several international conferences.

SHANG ZHENG received the B.S. degree in software engineering from Northeast Normal University, Changchun, China, in 2006, the M.S. degree in computer science from Jilin University, Changchun, China, in 2009, and the Ph.D. degree in software engineering from De Montfort University, Leicester, U.K., in 2014.

Since 2015, he has been a Lecturer with the School of Computer Science and Engineering, Jiangsu University of Science and Technology, Zhenjiang, China. He has authored or co-authored over 10 top journal and conference papers. His research interests include software repository mining, software maintenance, software evolution, and intelligent computation.

Dr. Zheng is a member of the ACM and the China Computer Federation. He is also a reviewer for several major software engineering conferences.

QI WANG received the M.S. degree in electrical and computer engineering from Sungkyunkwan University, South Korea, in 2011, and the Ph.D. degree in information and communications technology from University of Trento, Italy, in 2015. He was a Visiting Scholar at North Carolina State University from 2013 to 2014.

He is currently a Lecturer with the School of Computer, Jiangsu University of Science and Technology, Zhenjiang, China. His research interests include wireless sensor network, wireless communications in smart grid, architecture reliability of smart grid with renewable energy system, and fast power charging station for electric vehicles.

XIAOYAN XI received the B.S. degree in science of law from the Southwest University of Political Science and Law, Chongqing, China, in 2012. She is currently pursuing the master's in computer science from the Jiangsu University of Science and Technology, Zhenjiang, China. Her research interests include machine learning and data mining.