Active Learning From Imbalanced Data: A Solution of Online Weighted Extreme Learning Machine

Hualong Yu, Xibei Yang, Shang Zheng, and Changyin Sun

***Abstract*— It is well known that active learning can simultaneously improve the quality of the classification model and decrease the complexity of training instances. However, several previous studies have indicated that the performance of active learning is easily disrupted by an imbalanced data distribution. Some existing imbalanced active learning approaches also suffer from either low performance or high time consumption. To address these problems, this paper describes an efficient solution based on the extreme learning machine (ELM) classification model, called active online-weighted ELM (AOW-ELM). The main contributions of this paper include: 1) the reasons why active learning can be disrupted by an imbalanced instance distribution and its influencing factors are discussed in detail; 2) the hierarchical clustering technique is adopted to select initially labeled instances in order to avoid the missed cluster effect and cold start phenomenon as much as possible; 3) the weighted ELM (WELM) is selected as the base classifier to guarantee the impartiality of instance selection in the procedure of active learning, and an efficient online updated mode of WELM is deduced in theory; and 4) an early stopping criterion that is similar to but more flexible than the margin exhaustion criterion is presented. The experimental results on 32 binary-class data sets with different imbalance ratios demonstrate that the proposed AOW-ELM algorithm is more effective and efficient than several state-of-the-art active learning algorithms that are specifically designed for the class imbalance scenario.**

***Index Terms*— Active learning, class imbalance, cost-sensitive learning, extreme learning machine (ELM), online learning, stopping criterion.**

I. Introduction

ACTIVE learning is a popular machine learning paradigm and it is frequently deployed in the scenarios when large-scale instances are easily collected, but labeling them is expensive and/or time-consuming [1]. By adopting active learning, a classification model can iteratively interact with human experts to only select those most significant instances for labeling and to further promote its performance as quickly as possible. Therefore, the merits of active learning lie in decreasing both the burden of human experts and the complexity of training instances but acquiring a classification model that delivers superior or comparable performance to the model with labeling all instances.

Past research has accumulated a large number of active learning models, and generally, we have several different taxonomies to organize these models. Based on different ways of entering the unlabeled data, active learning can be divided into pool-based [2], [3] and stream-based models [4]. The former previously collects and prepares all unlabeled instances, while the latter can only visit a batch of newly arrived unlabeled data at each specific time point. According to different numbers of the labeled instances in each round, we have single-mode and batch-mode learning models [5]. As their names indicate, the single-mode model only labels one unlabeled instance on each round, while the batch-mode labels a batch of unlabeled examples once. In addition, we have several different significance measures to rank unlabeled instances, including uncertainty [6], [7], representativeness [8], inconsistency [9], variance [10], and error [11]. Each significance measure has a criterion for evaluating which instances are the most important for improving the performance of the classification model. For example, uncertainty considers the most important unlabeled instance to be the nearest one to the current classification boundary; representativeness considers the unlabeled instance that can represent a new group of instances, e.g., a cluster, to be more important, and inconsistency considers the unlabeled instance that has the most predictive divergence among multiple diverse baseline classifiers to be more significant. In addition, active learning models can also be divided into different categories according to which kind of classifier has been adopted. Some popular classifiers, including naive Bayes [12], k-nearest neighbors [13], decision tree [6], multiple level perceptron (MLP) [14], [15], logistic regression [16], support vector machine (SVM) [17]–[19], and extreme learning machine (ELM) [20], [21], have all been developed to satisfy the requirements of active learning. In the past decade, active learning has also been deployed in a variety of real-world applications, such as video annotation [22], [23], image retrieval [18], [24], text classification [25], [26], remote sensing image annotation [27], speech recognition [28], network intrusion detection [29], and bioinformatics [30].

Manuscript received April 27, 2017; revised March 4, 2018 and June 27, 2018; accepted July 3, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61305058 and Grant 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 Grant 2013M540404 and Grant 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. *(Corresponding author: Hualong Yu.)*

H. Yu, X. Yang, and S. Zheng are with the School of Computer, Jiangsu University of Science and Technology, Zhenjiang 212003, China (e-mail: yuhualong@just.edu.cn; yangxibei@hotmail.com; zhengshang1983@hotmail.com).

C. Sun is with the School of Automation, Southeast University, Nanjing 210096, China (e-mail: cysun@seu.edu.cn).

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

Digital Object Identifier 10.1109/TNNLS.2018.2855446

Active learning is undoubtedly effective, but several recent studies have indicated that active learning tends to fail when it is applied to data with a skewed class distribution [25], [26], [31]–[33]. That is, similar to traditional supervised learning, active learning also dares to face class imbalance problem. Several previous studies have tried to address this problem by using different techniques. Zhu and Hovy [25] first noticed this problem and tried to include several sampling techniques in active learning procedure to control the balance between the number of labeled instances in the minority and majority classes. Specifically, they presented three different sampling techniques: random undersampling (RUS), random oversampling (ROS), and bootstrap-based oversampling (BootOS) [25]. The authors indicated that RUS is generally worse than the original active learning algorithm, whereas both ROS and BootOS can increase the performance of learning, although the former tends to be more overfitting than the latter. Bloodgood and Vijay-Shanker [26] took advantage of the idea of cost-sensitive learning, which is another popular class imbalance learning technique, to handle a skewed data distribution during active learning. In particular, cost-sensitive SVM (CS-SVM) was employed as the base learner, empirical costs were assigned according to the prior imbalance ratio, and two traditional stopping criteria, i.e., the minimum error and the maximum confidence, were adopted to find the appropriate stopping condition for active learning. The method is robust and effective; however, it is also more time-consuming because the high time-complexity of training an SVM and no use of online learning. Tomanek and Hahn [31] proposed two methods based on the inconsistency significance measure: balanced-batch active learning (AL-BAB) and active learning with boosted disagreement (AL-BOOD), where the former selects n labeled instances that are class balanced from $5n$ bnew labeled instances on each round of active learning, while the latter modifies the equation of voting entropy to make instance selection focus on the minority class. It is clear that AL-BAB is quite similar to RUS, but it is possibly worse and wastes even more labeled resources than RUS, while AL-BOOD must deploy many diverse base learners (ensemble learning) to calculate the voting entropy of predictive labels, which will inevitably increase the computational burden. Therefore, we did not compare our proposed method with above-mentioned methods in Section V. In addition to the methods mentioned earlier, there has been research on how to treat the class imbalance problem by active learning. Ertekin *et al.* [32], [33] indicated that near the boundary of two different classes, the imbalance ratio is generally much lower than the overall ratio, thus adopting active learning can effectively alleviate the negative effects of imbalanced data distribution. In other words, they consider active learning to be a specific sampling strategy. In addition, a margin exhaustion criterion is proposed as an early stopping criterion to confirm the stopping condition because they selected SVM as a base learner.

To summarize the existing active learning algorithms applied in the scenario of unbalanced data distributions, we found that they suffer from either low classification performance or high time-consumption problems. Therefore, in this paper, we wish to propose an effective and efficient algorithm. The proposed algorithm is named active online-weighted ELM (AOW-ELM), and it should be applied in the pool-based batch-mode active learning scenario with an uncertainty significance measure and ELM classifier. We select ELM as the baseline classifier in active learning based on three observations: 1) it always has better than or at least comparable generality ability and classification performance as do SVM and MLP [34], [35]; 2) it can tremendously save training time compared to other classifiers [36]; and 3) it has an effective strategy for conducting active learning [21]. In AOW-ELM, we first take advantage of the idea of cost-sensitive learning to select the weighted ELM (WELM) [37] as the base learner to address the class imbalance problem existing in the procedure of active learning. Then, we adopt the AL-ELM algorithm [21] presented in our previous paper to construct an active learning framework. Next, we deduce an efficient online learning mode of WELM in theory and design an effective weight update rule. Finally, benefiting from the idea of the margin exhaustion criterion, we present a more flexible and effective early stopping criterion. Moreover, we try to simply discuss why active learning can be disturbed by skewed instance distribution, further investigating the influence of three main distribution factors, including the class imbalance ratio, class overlapping, and small disjunction. Specifically, we suggest adopting the clustering techniques to previously select the initially labeled seed set, and thereby avoid the missed cluster effect and cold start phenomenon as much as possible. Experiments are conducted on 32 binary-class imbalanced data sets, and the results demonstrate that the proposed algorithmic framework is generally more effective and efficient than several state-of-the-art active learning algorithms that were specifically designed for the class imbalance scenario.

The rest of this paper is organized as follows. Section II introduces some priori knowledge related to this paper. In Section III, we construct several representative synthetic data sets with different distributions to analyze the reason why active learning can be destroyed by skewed instance distribution. Section IV presents our proposed algorithmic framework in detail. Section V provides the experimental results and analysis. Finally, Section VI concludes the contributions of this paper and indicates future work.

II. Preliminaries

In this section, we present some preliminaries, including the basic flow path of pool-based active learning, ELM, WELM, online sequential ELM, and active learning with ELM (AL-ELM). The proposed core algorithmic model of this paper is presented in Section IV.

A. Flow Path of Pool-Based Active Learning

As mentioned in Section I, according to different ways of entering the unlabeled data, active learning can be divided into two categories: pool based [2], [3] and stream based [4], where the pool-based scenario is more common in real-world applications. In pool-based scenario, all unlabeled instances are previously prepared, and then a fraction of them are randomly

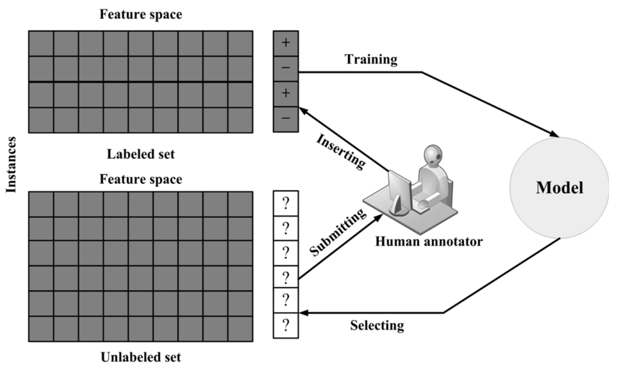

Fig. 1. Flowchart of the pool-based active learning.

extracted and labeled by human experts, further executing active learning iteratively. In other words, the classification model can continuously select the useful instances from the unlabeled pool to promote quality. In this paper, we focus on the pool-based active learning scenario.

The basic flow path of the pool-based active learning is described in Fig. 1. As we can see, the flow path is organized into four parts, and it is a closed loop. In each round, the labeled set is used to train a classification model. Then, the model estimates the significance of each instance in the unlabeled set, ranks them, and selects some of the most significant instances to submit to human annotators, who further provide labels for these instances and then insert them into the training instances to update the labeled set. Active learning repeats the process above until it satisfies a stopping condition. Obviously, in the flow path mentioned above, how to use the classification model to estimate, rank, and extract significant unlabeled instances is the key point because it directly concerns the quality of active learning.

B. Extreme Learning Machine

ELM that was proposed by Huang *et al.* [34], [35] is a specific learning algorithm for single-hidden layer feed-forward neural networks (SLFNs). The main characteristics of ELM that distinguish it from those conventional learning algorithms of SLFN are the random generation of hidden nodes. Therefore, ELM does not need to iteratively adjust the 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 than or at least comparable generality ability and classification performance to SVM and MLP but only consumes tenths or hundredths of training time compared to SVM and MLP [34]–[36].

Let us consider a classification problem with N training instances to distinguishing m categories, and then the ith training instance can be represented as (x_i, t_i), where x_i is an $n \times 1$ input vector and t_i is the corresponding $m \times 1$ output vector. Suppose there are L hidden nodes in ELM and that all weights and biases on these nodes are generated randomly. Then, for the instance x_i, its hidden layer output can be represented as a row vector $h(x_i) = [h_1(x_i), h_2(x_i), \ldots, h_L(x_i)]$ by mapping with an activation function (the most popular sigmoid function is used throughout this paper). The mathematical model of ELM could be described as

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

where $H = [h(x_1), h(x_2), \ldots, h(x_N)]^T$ is the hidden layer output matrix overall training instances, β is the weight matrix of the output layer, and $T = [t_1, t_2, \ldots, t_N]$ denotes the target matrix. Obviously, in (1), only β is unknown, so we can adopt the least-square algorithm to acquire its solution, which can be described as follows:

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

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

We can also train an ELM in the viewpoint of optimization [35]. In the optimization version of ELM, we wish to synchronously minimize $\|H\beta - T\|^2$ and $\|\beta\|^2$, so the question can be described as follows:

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

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, and C is the penalty factor representing the tradeoff between the minimization of training errors and the maximization of generality ability. Obviously, this is a typical quadratic programming problem that can be solved by the Karush–Kuhn–Tucker theorem [38]. The solution for (3) can be described as follows:

$$\beta = H^{\dagger}T = \begin{cases} H^T\left(\dfrac{I}{C} + HH^T\right)^{-1}T, & \text{when } N \le L \\ \left(\dfrac{I}{C} + HH^T\right)^{-1}H^T T, & \text{when } N > L. \end{cases} \tag{4}$$

C. Weighted Extreme Learning Machine

WELM that can be regarded as a cost-sensitive learning version of ELM is an effective way to handle the imbalanced data [37]. Similar to CS-SVM, the main idea of WELM is to assign different penalties for different categories, where the minority class has a larger penalty factor C, while the majority class has a smaller C value. Then, WELM focuses on the training errors of the minority instances, making a classification hyperplane emerge in a more impartial position. A weighted matrix W is used to regulate the parameter C for different instances, i.e., (3) can be rewritten as

$$\begin{aligned} &\text{min: } Lp_{\text{ELM}} = \frac{1}{2}\|\beta\|^2 + C\frac{1}{2}W\sum_{i=1}^{N}\|\xi_i\|^2 \\ &\text{s.t: } h(x_i)\beta = t_i^T - \xi_i^T, \quad i = 1, 2, \ldots, N \end{aligned} \tag{5}$$

where W is a $N \times N$ diagonal matrix in which each value existing on the diagonal represents the corresponding regulation weight of parameter C. Zong *et al.* [37] provided two different weighting strategies that are described as follows:

$$\text{WELM1}: W_{ii} = 1/\#(t_i) \tag{6}$$

and

$$\text{WELM2}: W_{ii} = \begin{cases} 0.618/\#(t_i) & \text{if } \#(t_i) > \text{AVG}(t_i) \\ 1/\#(t_i) & \text{if } \#(t_i) \le \text{AVG}(t_i) \end{cases} \tag{7}$$

where $\#(t_i)$, average (AVG)(t_i), and 0.618 denote the number of instances belonging to the class t_i, the average number of instances over all classes, and the value of the golden standard, respectively. Compared with WELM2, WELM1 is more practical and popular. Then, the solution can be described as follows:

$$\beta = \begin{cases} H^T\left(\dfrac{I}{C} + WHH^T\right)^{-1} WT, & \text{when } N \le L \\ \left(\dfrac{I}{C} + HWH^T\right)^{-1} H^T WT, & \text{when } N > L. \end{cases} \tag{8}$$

D. Online Sequential Extreme Learning Machine

In 2006, Liang *et al.* [39] proposed an online sequential learning mode of ELM named oversampling (OS)-ELM. OS-ELM adopts extended recursive least squares for training with sequentially received data that can be either received in the mode of one by one or chunk by chunk. Based on their derivation, the update rule of the output layer weight matrix β can be represented as

$$\beta^{(k+1)} = \beta^{(k)} + P_{k+1}H_{k+1}^T(T_{k+1} - H_{k+1}\beta^{(k)}) \tag{9}$$

where H_{k+1} and T_{k+1} correspond to the hidden layer output matrix and the target matrix of the new observations in the $(k+1)$th chunks, while $\beta^{(k)}$ and $\beta^{(k+1)}$ denote the output layer weight matrix β after receiving the kth and $(k+1)$th chunks, respectively. As for P_{k+1}, it can be calculated by the following formula:

$$P_{k+1} = P_k - P_k H_{k+1}^T(I + H_{k+1}P_k H_{k+1}^T)^{-1} H_{k+1}P_k \tag{10}$$

that is, P_k can also be iteratively updated, and the initial P_0 can be represented as

$$P_0 = (H_0^T H_0)^{-1} \tag{11}$$

where H_0 indicates the premier hidden layer output matrix. Let us return to (9), as H_{k+1}, T_{k+1}, and P_{k+1} all can be calculated only by the newly received instances. Thus, the output layer weight matrix β can be adjusted to adapt both old and new instances, but it does not need to be recalculated with all of the training examples. It is not difficult to observe that the main merit of the OS-ELM algorithm is to decrease the time-complexity to a large extent when we can only acquire the training instances dynamically.

E. Active Learning With Extreme Learning Machine

In our previous work, we proposed an active learning algorithm based on ELM and named it AL-ELM [21]. We found that the actual outputs of ELM can reflect the uncertainty level of instances, i.e., their classification confidences. Specifically, we also proved that there is an approximate mapping relationship between the actual outputs of ELM and the posterior probabilities in the Bayes classifier. The more approximate the actual outputs on different output nodes are, the more approximate the posterior probabilities belonging to the different classes are, and the more uncertain and important the corresponding instance is for modeling the classifier, too. As mentioned in Section I, labeling the most uncertain instances, i.e., the instances that are the closest to the current classification hyperplane, may provide a maximum promotion for the quality of the classification model. To convert the nonprobabilistic outputs in ELM into probabilistic outputs, we use the sigmoid function as follows:

$$P(t_i|f_i(x)) = \frac{1}{1 + \exp(-f_i(x))}, \quad i = 1, 2, \ldots, m \tag{12}$$

where $t_i = 1$ means that category i happens, $f_i(x)$ denotes the actual output of the ith output node with respect to the instance x. For the binary-class problem, the accumulated probabilistic value on two output nodes strictly equals 1. For the multiclass problem, however, the sum of the converted probabilistic values is always larger than 1, thus we also present a normalized strategy as follows:

$$P'(t_i|f_i(x)) = \frac{P(t_i|f_i(x))}{\sum_{k=1}^{m} P(t_k|f_k(x))}, \quad i = 1, 2, \ldots, m. \tag{13}$$

In AL-ELM, we select the most uncertain instances, i.e., the instances with the most approximate actual outputs in two different output nodes, to a label on each round. In this paper, we inherit the uncertainty estimation and selection strategy in AL-ELM.

TABLE I
DISTRIBUTIONS OF THE SIX SYNTHETIC DATA SETS

Dataset	Distribution: number of instances
D1	Maj: *G*(0.5,0.5): 1000; Min: *G*(0.7,0.5): 500
D2	Maj: *G*(0.5,0.5): 1000; Min: *G*(0.7,0.5): 100
D3	Maj: *G*(0.5,0.5): 1000; Min: *G*(0.7,0.5): 10
D4	Maj: *G*(0.5,0.5): 1000; Min: *G*(0.7,0.5): 5
D5	Maj: *G*(0.5,0.5): 1000; Min: *G*(1.0,0.5): 100
D6	Maj: *G*(0.5,0.5): 1000; Min: *G*(0.8,0.5): 60 / *G*(0.5,0.8): 30 / *G*(0.2,0.2): 10

Maj denotes the majority class, Min denotes the minority class, *G*(*a*, *b*) denotes the instances satisfying Gaussian distribution with respect to the mean values *a* and *b* on attribute 1 and 2, respectively. For each distribution, the variance identically equals to 0.1.

III. INVESTIGATION INTO THE DISTRIBUTION

To clarify why traditional active learning algorithms are easily destroyed by a skewed data distribution, we explore it from the perspective of data distribution. Here, we generate six synthetic 2-D binary-class imbalanced data sets with considering three different data distribution factors: class imbalance ratio, class overlapping, and small disjunction. The distributions of these six data sets are summarized in Table I.

In Table I, we can observe that data sets $D1$–$D4$ satisfy the same distribution but have different class imbalance ratios, from 2:1 to 200:1. Although data set $D5$ has the same class imbalance ratio as $D2$ does, they satisfy different distributions,

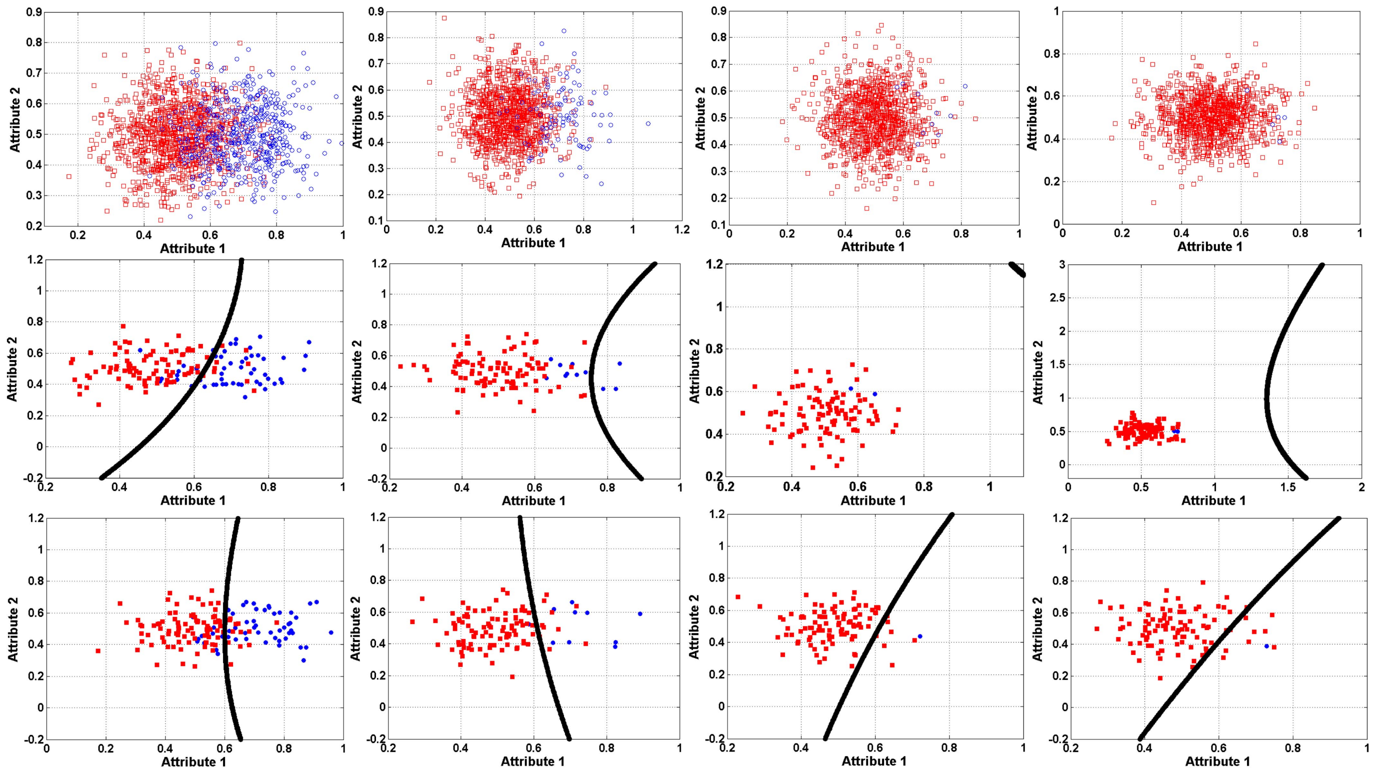

Fig. 2. First row: the original distribution of the first four data sets. Second row: classification hyperplane produced by ELM ($L = 100$, $C = 2^{10}$) on 10% randomly extracted initial labeled instances. Third row: classification hyperplane produced by WELM ($L = 100$, $C = 2^{10}$) on 10% randomly extracted initial labeled instances. Each column denotes one data set ($D1$–$D4$). □ and ■ represent the unlabeled and labeled instances of the majority class, while ○ and ● denote the unlabeled and labeled instances belonging to the minority class, respectively (from left to right).

i.e., a different class overlapping. For $D6$, there are three within-class subclusters in the minority class, and they contain a different number of instances.

First, we investigate whether the class imbalance ratio is related to the instance selection of active learning. We randomly extract 10% of instances as the initially labeled examples on $D1$–$D4$. Then, we train an ELM classifier and a WELM classifier on these data. The original distribution of these data sets, the classification hyperplanes produced by ELM and WELM ($L = 100$, $C = 2^{10}$) on the randomly selected 10% instances can be observed in Fig. 2.

From the second row in Fig. 2, it is not difficult to observe a clear tendency that the classification hyperplane generated by ELM would be increasingly pushed toward the minority class in synchronization with the increase of the class imbalance ratio, especially on $D3$ and $D4$ whose class imbalance ratios are more than 100:1, all instances in the minority class would be misclassified. In fact, it is related to the training rule adopted by the regular classifiers, i.e., to minimize the overall training errors. To minimize the overall training errors, the regular classifiers focus on the accuracy of the majority class but neglect the minority class because there is no difference between misclassifying a majority instance and misclassifying a minority sample.

Some previous work has indicated that almost all conventional regular classifiers could be destroyed by data with a highly imbalanced ratio [40]–[43]. Compared with ELM, WELM proportionally increases the penalty to training errors of the minority class instances, thus making the generated hyperplane move closer to the real classification boundary (see the third row in Fig. 2). In the procedure of active learning, the classification hyperplane produced by the ELM is always far from the real boundary and the labeling instances surrounding this hyperplane will be meaningless and waste human resources. While WELM can generally produce a hyperplane nearer to the real classification boundary, further guaranteeing anyone labeled instance is important enough. Therefore, we call weighting a balance control strategy that can synchronously guarantee the impartiality of unlabeled instances selection and the effectiveness of the classification model in the procedure of active learning. Based on this observation, it is not difficult to understand why we select WELM but not ELM as the baseline classification model to address the imbalanced active learning issue. In fact, it is expected to acquire an excellent enough classification model with as few labeled instances as possible.

Then, we investigate whether the size of the class overlapping, i.e., the margin size between two different categories, would influence the instance selection of active learning or not. Fig. 3 presents the original distribution, the hyperplanes produced by ELM and WELM on randomly extracted 10% labeled instances from the $D5$ data set. Data set $D5$ has the same class imbalance ratio (10:1) as $D2$ does, but they hold different sizes of class overlapping. Comparing the second

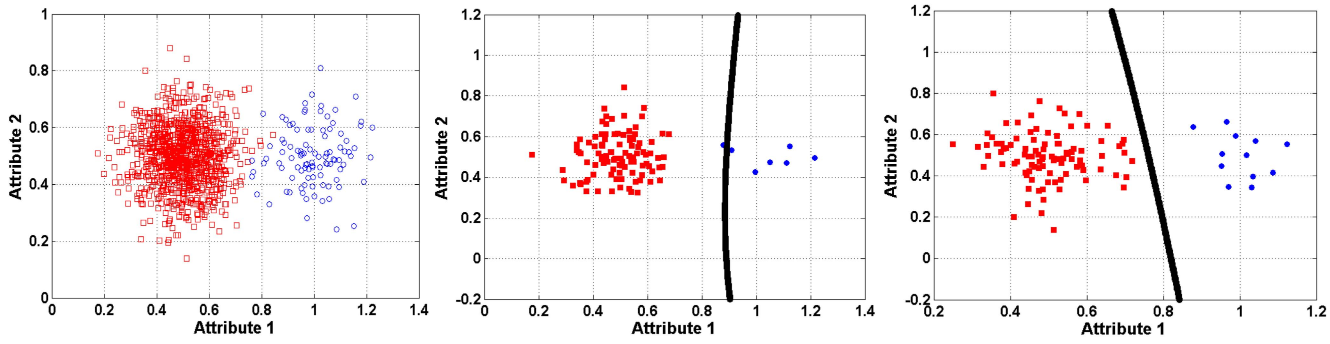

Fig. 3. Original distribution, classification hyperplane produced by ELM ($L = 100, C = 2^{10}$) and WELM ($L = 100, C = 2^{10}$) on 10% randomly extracted initial labeled instances of $D5$ data set, respectively. □ and ■ represent the unlabeled and labeled instances of the majority class, while ○ and ● denote the unlabeled and labeled instances belonging to the minority class, respectively.

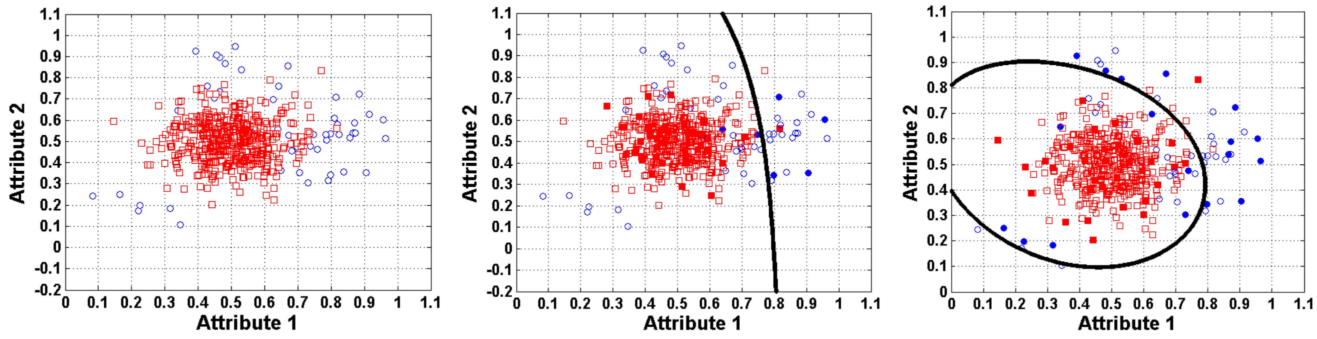

Fig. 4. Distribution of the unlabeled set, the classification hyperplane produced by ELM ($L = 100, C = 2^{10}$) on 10% randomly extracted labeled instances, and the classification hyperplane produced by ELM ($L = 100, C = 2^{10}$) on 10% representative initially labeled instances on $D6$ data set, respectively. □ and ■ represent the unlabeled and labeled instances of the majority class, while ○ and ● denote the unlabeled and labeled instances belonging to the minority class, respectively.

column in Fig. 2 with Fig. 3, we observe that on the large margin class imbalance data set, the regular classifier, e.g., ELM, generally produces a hyperplane that is nearer to the majority class than that produced on the small margin data set. That is, the harmfulness will decrease in synchronization with the increase of margin size. However, it is still harmful to regular classifiers, so adopting a balance control strategy is still necessary (compare the second subgraph with the third subgraph in Fig. 3).

Finally, we investigate the influence of a more complex data distribution factor, i.e., within-class subclusters that are also called small disjunction. There are several different subconcepts or subdistributions in one class, and these subconcepts or subdistributions have a different number of instances. On the skewed data set with small disjunctions, we face both the problem of "between-class imbalance" and the problem of "within-class imbalance" [44]. Taking the $D6$ data set as an example, there are three within-class clusters with an imbalance ratio of 6:3:1 in the minority class. Then, a problem will appear, i.e., if none of the instances in the small subclusters are selected into the initially labeled set (seed set), then it is possible to permanently misclassify all instances belonging to these clusters. The problem is called the missed cluster effect [45]. To address this problem, we adopt the hierarchical clustering technique to subtly explore the distribution structure of the collected unlabeled instances, further precisely selecting and labeling those representative examples to construct the initial seed set. Specifically, after hierarchical clustering is finished, the instance that is closest to the centroid of each cluster is extracted to label, as these instances are considered the representative instances. Fig. 4 presents the procedure of seed sets generation by randomly extracting labeled instances and selecting representative samples with the hierarchical clustering technique, and the classification hyperplanes produced by the ELM on these two seed sets. To simulate the actual procedure of active learning, half of the instances in the $D6$ data set are randomly extracted to construct the initially collected unlabeled set, and then 10% of the examples in the unlabeled set is extracted randomly and selected carefully by the clustering technique.

Fig. 4 indicates that if we randomly extract some instances to construct the initially labeled seed set, some small subclusters may be missed and may even be permanently misclassified because they are far from the initial classification hyperplane. That means, these instances might not be labeled permanently during active learning. Hierarchical clustering, however, is helpful for finding those representative instances that can better describe the original data distribution. In addition, this technique is useful for avoiding the cold start phenomenon that often appears in the scenario of highly skewed data distribution. Cold start means that none of the minority class instances are selected into the seed set, resulting in the impracticability of active learning [46]. Fig. 4 shows that the selection of initially labeled examples based on the clustering technique explores all small distribution structures, and further extracts more instances belonging to the minority class, thus it can be used to effectively reduce the probability of a cold start. In this paper, all experiments are based on the seed set generated by the hierarchical clustering technique, and the number of subclusters equals the number of the initially labeled instances.

The analysis mentioned above demonstrates that on imbalanced distribution data, the performance of active learning can be influenced by a combination of multiple factors and is not only related to class imbalance ratio. Therefore, it is necessary to conduct a balance control strategy during active learning,

IV. AOW-ELM Algorithm Model

A. Online Sequential Weighted Extreme Learning Machine

As we know, active learning conducts an iterative procedure, i.e., adding one or a batch of new labeled instances into the labeled set each round. Undoubtedly, it would be quite time-consuming to retrain the classification model on each round. Therefore, it is necessary to adopt an online learning algorithm to implement active learning. Mirza *et al.* [47] presented an online sequential WELM algorithm to address the incremental class imbalance learning problem. However, the algorithm is based only on the least-squares version and not on the optimization version of ELM, thereby losing control of the generalization ability of ELM. In addition, the iterative weight tuning and ranking mechanism adopted in this algorithm increase the unnecessary time expenditure. In [48], a kernel-based imbalance online learning algorithm was presented. The algorithm is effective in addressing imbalanced online learning with concept drift. However, it has a limitation for application in our active learning scenario, i.e., its instance weight is correlated with the number of training instances belonging to each class acquired to date, which means that the new instances' weights will gradually decrease, while the old instances could be highlighted. Obviously, it cannot satisfy our application requirements. To avoid these problems, a novel online sequential WELM algorithm based on the optimization version is proposed in this paper. Its derivation process is similar to the OS-ELM algorithm [39].

Based on (8), we have

$$\beta = \left(\frac{I}{C} + HWH^T\right)^{-1} H^T WT$$
$$= \left(\frac{I}{C} + (\sqrt{WH})^T(\sqrt{WH})\right)^{-1} (\sqrt{WH})^T(\sqrt{WT}). \quad (14)$$

Then, (14) can be rewritten as

$$\beta = \left(\frac{I}{C} + U^T U\right)^{-1} U^T V \quad (15)$$

where

$$U = \sqrt{WH}$$
$$V = \sqrt{WT}. \quad (16)$$

Before the incremental learning starts, we only hold the initially labeled seed set, thus the first classification model can be represented as follows:

$$\beta^{(0)} = \left(\frac{I}{C} + U_0^T U_0\right)^{-1} U_0^T V_0 \quad (17)$$

where

$$U_0 = \sqrt{W_0 H_0}$$
$$V_0 = \sqrt{W_0 T_0} \quad (18)$$

where W_0, H_0, and T_0 are calculated with the initial seed set. Then, we consider the incremental procedure, for the $(K+1)$th iteration, the weight matrix of the output layer β could be represented as

$$\beta^{(K+1)} = \left(\frac{I}{C} + U_{K+\Delta K}^T U_{K+\Delta K}\right)^{-1} U_{K+\Delta K}^T V_{K+\Delta K} \quad (19)$$

where

$$U_{K+\Delta K} = \sqrt{W_{K+\Delta K} H_{K+\Delta K}} = \begin{bmatrix} U_K \\ U_{\Delta K} \end{bmatrix}$$
$$V_{K+\Delta K} = \sqrt{W_{K+\Delta K} T_{K+\Delta K}} = \begin{bmatrix} V_K \\ V_{\Delta K} \end{bmatrix} \quad (20)$$

and ΔK indicates the $(K+1)$th received the chunk of the incremental labeled instances. According to (20), (19) can be written as

$$\beta^{(K+1)} = \left(\frac{I}{C} + U_K^T U_K + U_{\Delta K}^T U_{\Delta K}\right)^{-1} (U_K^T V_K + U_{\Delta K}^T V_{\Delta K}). \quad (21)$$

Revisiting (17), let

$$P_0 = \frac{I}{C} + U_0^T U_0. \quad (22)$$

Combining (21) and (22), $\beta^{(1)}$ is given by

$$\beta^{(1)} = (P_0 + U_{\Delta 1}^T U_{\Delta 1})^{-1} U_0^T V_0 + U_{\Delta 1}^T V_{\Delta 1} \quad (23)$$

then P_1 can be represented as

$$P_1 = P_0 + U_{\Delta 1}^T U_{\Delta 1} = \frac{I}{C} + U_1^T U_1. \quad (24)$$

Thus, it is not difficult to deduce the expression of P_{K+1}

$$P_{K1} = P_K + U_{\Delta K}^T U_{\Delta K} = \frac{I}{C} + U_K^T U_K + U_{\Delta K}^T U_{\Delta K}. \quad (25)$$

Combining (23) and (27), we have

$$\beta^{(K+1)} = (P_K + U_{\Delta K}^T U_{\Delta K})^{-1} (U_K^T V_K + U_{\Delta K}^T V_{\Delta K}). \quad (26)$$

Then, we observe the second term in (28), it can be transferred to

$$U_K^T V_K + U_{\Delta K}^T V_{\Delta K} = P_K P_K^{-1} U_K^T V_K + U_{\Delta K}^T V_{\Delta K}$$
$$= P_K \left(\frac{I}{C} + U_K^T U_K\right)^{-1} U_K^T V_K + U_{\Delta K}^T V_{\Delta K}$$
$$= P_K \beta^{(K)} + U_{\Delta K}^T V_{\Delta K}. \quad (27)$$

Integrating the result into (28), we have

$$\beta^{(K+1)} = (P_K + U_{\Delta K}^T U_{\Delta K})^{-1} (P_K \beta^{(K)} + U_{\Delta K}^T V_{\Delta K}). \quad (28)$$

As P_K can be incrementally updated, $U_{\Delta K}$ and $V_{\Delta K}$ are merely correlated with the instances in the newly received chunk, thus $\beta^{(K+1)}$ can be deduced from $\beta^{(K)}$.

By observing (28), it is clear that the time-complexity of the learning procedure mentioned above is linear with respect to the number of the incrementally labeled instances because in each round, the new incremental training time is related to only the number of newly labeled training instances. Because the quality of ELM is determined only by the weight matrix of the output layer β, updating β means the implementation of online incremental learning.

B. Weight Update Rule

Next, we focus on the problem of weight setting for the newly received labeled instances during online learning. In the work of Zong *et al.* [37], the weight assignment is related to the reciprocal of the number of the instances belonging to each category. It is obviously improper to assign the weights by this strategy, as with an increase in the number of labeled examples, the weights used to punish the newly inserted instances would decrease sharply, resulting in the classification model focuses on the previous instances, but neglects the newly labeled ones. In other words, we need a more robust weight assignment strategy that should be irrelevant to the timeline. Here, we provide a stable weight update strategy for online learning on binary-class skewed data. For a newly labeled instance x_i, its weight can be assigned as

$$w_i = \begin{cases} \dfrac{|N^+|}{|N^+|+|N^-|}, & \text{if } x_i \text{ belong to the majority class} \\ \dfrac{|N^-|}{|N^+|+|N^-|}, & \text{if } x_i \text{ belong to the minority class} \end{cases} \tag{29}$$

where $|N^+|$ and $|N^-|$ denote the number of instances belonging to the positive class (minority class) and the negative class (majority class) in the currently labeled set, respectively. The weights rely only on the ratio between the number of instances belonging to two different classes, regardless of the absolute number of instances, so it would be more stable than (6) which might assign monotonically decreasing weights to the new instances and further gradually reduce the effect of the newly learned instances. Furthermore, the ratio of the weights between two different classes reflects the class imbalance ratio. However, during online learning, the ratio still fluctuates dynamically.

C. Early Stopping Criterion

As we know, the objective of active learning is to acquire a high-quality classification model in synchronization with reducing the number of labeled instances. Therefore, exhausting all unlabeled instances is unacceptable in practical active learning applications. In fact, an early stopping criterion should be previously designated to ask active learning when it should be stopped. The optimal stopping time should simultaneously guarantee two conditions: 1) the performance of the current classification model should be good enough and 2) the size of the total labeled instances should be small enough.

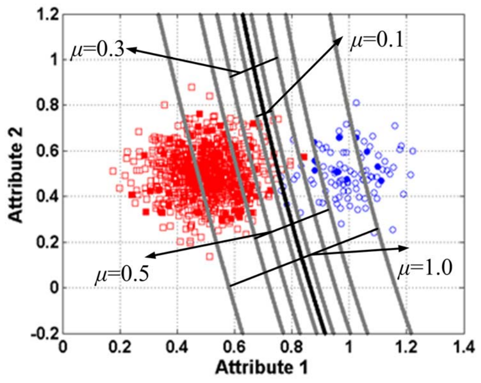

Fig. 5. Classification hyperplane (black line), contour lines (gray lines) directed by four different thresholds (0.1, 0.3, 0.5, and 1.0), and their margin ranges produced by WELM ($L = 100$, $C = 2^{10}$) on 10% randomly extracted labeled instances on D5 data set. □ and ■ represent the unlabeled and labeled instances of the majority class, while ○ and ● denote the unlabeled and labeled instances belonging to the minority class, respectively.

In this paper, we refer to the idea of the margin exhaustion criterion presented in the work of Ertekin *et al.* [33], and then design a more flexible early stopping criterion. Unlike SVM, there is no concept about margin in ELM, but in our previous work, we observed that the contour line of a designated actual output value in ELM could be used to describe the confidence distribution, while a pair of contour lines can direct a so-called margin. Fig. 5 presents the margin ranges beyond four different thresholds constructing on 10% randomly labeled instances of the $D5$ data set trained by WELM. Threshold μ denotes the actual output absolute value of the output node. Thus, a specific threshold μ corresponds to a pair of contour lines that indicate a margin. With an increase in the threshold μ, the margin will become larger. It is clear that the larger the threshold μ is designated, the lower the acceptable uncertainty level is, thus more instances must be labeled in the active learning procedure.

In this paper, margin exhaustion means that for a given threshold μ, finding two corresponding contour lines in the instance space, when and only when all instances laying between these two contour lines have been labeled, active learning can be stopped. Of course, in practical applications, we do not need to describe the contour lines, but only compare the actual output of an instance with the given threshold to determine whether the instance drops into the margin. In addition, the actual output of ELM can be considered a variant of posterior probability, so the proposed stopping criterion is also approximately equivalent to the maximum confidence stopping criterion [49]. Therefore, it is not difficult to observe that the proposed early stopping criterion is both effective and more flexible than the traditional margin exhaustion criterion. A discussion of the threshold μ is given in Section V.

D. AOW-ELM Algorithm Description

In this section, we expect to describe the detailed procedure of the AOW-ELM algorithm as follows.

TABLE II
DATA SETS USED IN THIS PAPER

Dataset	No. Features	No. Instances	Imbalance ratio	Per. Labeled (%)	Batch size	Minority class	Majority class	Source
cardiotocographyC1	21	2126	4.54	5	10	Class1	Class 2~10	UCI
cardiotocographyC5	21	2126	28.53	10	10	Class 5	Class 1~4, Class 6~10	UCI
cardiotocographyC0	21	2126	9.79	5	10	Class10	Class 1~9	UCI
cardiotocographyN2	21	2126	6.21	5	10	NSP2	NSP1, NSP3	UCI
cardiotocographyN3	21	2126	11.08	5	10	NSP3	NSP1, NSP2	UCI
cardiotocographyN1 vs. N3	21	2126	3.51	5	10	NSP3	NSP1	UCI
mfeat-mor0	6	2000	9.00	5	5	0	1~9	UCI
mfeat-mor01	6	2000	4.00	5	5	0,1	2~9	UCI
mfeat-mor012	6	2000	2.33	5	5	0~2	3~9	UCI
mfeat-mor0123	6	2000	1.50	5	5	0~3	4~9	UCI
vowel0	13	988	9.98	5	5	hid	remainder	UCI
seed2	7	210	2.00	5	1	2	1, 3	UCI
yeast-ME3	8	1484	9.00	5	5	ME3	remainder	UCI
yeast-ME2	8	1484	28.10	5	5	ME2	remainder	UCI
yeast-ME1	8	1484	32.73	5	5	ME1	remainder	UCI
banknote-authentication	4	1372	1.25	1	5	1	0	UCI
segmentation-grass	19	210	6.00	10	1	grass	Path, window, cement, poliage, sky, brickface	UCI
segment1	19	2310	6.00	5	10	1	2~7	UCI
segment12	19	2310	2.50	5	10	1, 2	3~7	UCI
segment123	19	2310	1.33	5	10	1~3	4~7	UCI
page-blocks5	10	5473	46.59	5	20	5	1~4	UCI
ILPD	10	583	2.49	5	1	2	1	UCI
abalone9	8	4177	5.06	5	10	9	remainder	UCI
abalone19	8	4177	129.53	10	10	19	remainder	UCI
credit card clients	23	10000	3.46	0.5	100	default payment next month1	default payment next month 0	UCI
wilt	5	4839	17.54	5	20	1	2	UCI
haberman	3	306	2.78	5	1	2	1	UCI
letter-A	16	20000	24.35	1	100	A	B~Z	UCI
magic	10	19020	1.84	1	100	1	2	UCI
SNP	25	3074	15.80	5	10	Real SNP	Remainder SNP Candidates	Ref.[50]
microRNA	32	8587	44.01	2	40	Real microRNA Precursors	Remainder microRNA candidates	Ref.[51]
Box H/ACA snoRNA	14	8510	129.92	2	40	Box H/ACA snoRNA	Remainder non-coding RNAs	Ref.[52]

No.Features denotes the number of attributes, No.Instances denotes the number of instances, and Per. Labeled (%) indicates the initial labeled percentage in the dataset.

1) *Algorithm:* AOW-ELM.
2) *Input:* A collected unlabeled instance set Θ, the number of initially labeled instances κ, the initially labeled set $\psi = \phi$, the number of labeled instances on each round υ, the number of the hidden nodes L, the penalty factor C, and the stopping threshold μ.
3) *Output:* The final weight matrix $\beta^{(F)}$.
4) *Procedure.*
    a) Clustering all instances in Θ into κ different groups by hierarchical clustering technique.
    b) Find the instance that is closest to the centroid in each cluster extract and label them, and then shift them from Θ to ψ.
    c) Count and record the number of minority instances $|N^+|$ and majority instances $|N^-|$ in ψ.
    d) Adopt (29) to calculate and acquire the initial weighting matrix W_0.
    e) Generate the hidden-layer parameters randomly for L hidden nodes.
    f) Calculate $\beta^{(0)}$ using (8) and P_0 using (22).
    g) Calculate the actual output of each instance in Θ, if there exist outputs smaller than the threshold μ, continue to conduct the next step, else, stop, and return the current β as $\beta^{(F)}$.
    h) Rank all instances in Θ according to their actual output absolute values in ascending order, and then extract υ first instances to submit to human experts for labeling.
    i) Transfer υ new labeled instances from Θ to ψ.
    j) Update $|N^+|$ and $|N^-|$, and assign the weights for these υ new labeled instances using (29).
    k) Compute $U_{\Delta K}$ and $V_{\Delta K}$ for υ new labeled instances taking advantage of the random parameters generated for the hidden layer in step. e.
    l) Update P using (25).
    m) Update β using (28), and then return to step. g.

V. EXPERIMENTS

A. Data Sets

In this paper, we focus on conducting active learning on binary-class imbalanced data. Overall, 32 binary-class imbalanced data sets that are used, most of which come from the University of California-Irvine (UCI) machine learning data repository [50], and the others are from several publications about bioinformatics [51]–[53]. These data sets have different number of instances, number of features, and class imbalance ratios. In addition, for each data set, half of the instances are reserved as the test set, and in the other half, a specific percentage of instances is extracted as an initially labeled seed set. The percentage correlates with the size of the data set and class imbalance ratio to avoid cold starts as much as possible. Furthermore, the batch size of active learning, i.e., the number of instances labeled on each round, is also associated with the size of data set. The detailed description and settings about these data sets are provided in Table II.

B. Parameter Settings

To show the effectiveness of the proposed AOW-ELM algorithm, we compare it with six other algorithms.

1) *AO-ELM:* It combines the AL-ELM algorithm [35] with the OS-ELM algorithm [39], but does not adopt the balance control strategy during active learning. In fact, it can be regarded as a baseline algorithm that is used to indicate the necessity of a balance control strategy.
2) *Random Online-sequential Weighting (ROW)-ELM:* It adopts online sequential WELM, but on each round, the new incremental samples are selected randomly. It can also be seen as a baseline algorithm to make clear the necessity of active learning.
3) *RUS-ELM [25]:* It adopts RUS as the balance control strategy in the procedure of active learning. Specifically, it is required to reserve the current undersampling set to

TABLE III
PERFORMANCE (ALC METRIC) COMPARISON OF VARIOUS ALGORITHMS

Dataset	AO-ELM	ROW-ELM	RUS-ELM	ROS-ELM	BootOS-ELM	ACS-SVM	AOW-ELM
cardiotocographyC1	0.7526	0.8445	0.8378	0.8114	0.8310	**0.8684**	0.8583
cardiotocographyC5	0.3637	0.8528	0.8179	0.7360	0.7216	0.8195	**0.8835**
cardiotocographyC0	0.7173	0.8955	0.8729	0.8417	0.8627	0.8990	**0.9030**
cardiotocographyN2	0.7573	0.8630	0.8582	0.8324	0.8530	0.8709	**0.8735**
cardiotocographyN3	0.8460	0.9022	0.8947	0.8837	0.9006	**0.9238**	0.9079
cardiotocographyN1 vs. N3	0.8507	0.8848	0.8894	0.8765	0.8878	**0.9052**	0.8896
mfeat-mor0	0.7339	0.9899	0.9590	0.9898	**0.9907**	0.9901	0.9903
mfeat-mor01	0.5413	0.9738	0.9030	0.9660	0.9708	0.9692	**0.9755**
mfeat-mor012	0.4920	0.8981	0.8517	0.8684	0.8812	0.8939	**0.9059**
mfeat-mor0123	0.4831	0.8247	0.6990	0.7767	0.7994	**0.8368**	0.8285
vowel0	0.9671	0.9651	0.9358	0.9513	**0.9860**	0.9788	0.9857
seed2	0.9766	0.9750	0.9044	**0.9833**	0.9829	0.9621	0.9810
yeast-ME3	0.8202	0.9183	0.8711	0.8934	0.8985	0.9105	**0.9188**
yeast-ME2	0.6991	0.9313	0.9178	0.8940	0.9336	0.9146	**0.9421**
yeast-ME1	0.6981	0.9326	0.9201	0.8732	0.9069	0.9255	**0.9403**
banknote-authentication	0.9943	0.9885	0.8331	0.9854	0.9970	0.9873	**0.9973**
segmentation-grass	0.9965	0.9977	0.9746	0.9974	0.9998	**1.0000**	0.9962
segment1	0.8554	0.9892	0.9565	0.9897	0.9873	**0.9929**	0.9904
segment12	0.8301	0.9921	0.9731	0.9899	0.9904	**0.9958**	0.9932
segment123	0.7970	0.9320	0.9137	0.9197	0.9243	**0.9629**	0.9403
page-blocks5	0.6744	**0.8853**	0.8363	0.8328	0.8294	0.8402	0.8739
ILPD	0.4839	0.6266	0.5607	0.5482	0.5957	0.6206	**0.6303**
abalone9	0.1448	0.6405	0.6209	0.5560	0.5863	0.5388	**0.6445**
abalone19	0.0000	0.6115	**0.6782**	0.5207	0.5607	0.0732	0.5985
credit card clients	0.5369	**0.6654**	0.6487	0.6157	0.6362	0.6616	0.6539
wilt	0.6556	0.9224	0.8258	0.8342	0.9005	**0.9512**	0.9316
haberman	0.4912	**0.5892**	0.5045	0.5173	0.5457	0.5552	0.5714
letter-A	0.9271	0.9613	0.8920	0.9397	0.9506	**0.9738**	0.9707
magic	0.8174	0.8178	0.0000	0.7927	0.8123	**0.8344**	0.7703
SNP	0.1934	**0.6510**	0.6259	0.4944	0.4592	0.5737	0.6177
microRNA	0.4393	0.8483	0.8061	0.6708	0.7775	**0.8628**	0.8491
Box H/ACA snoRNA	0.4078	**0.9259**	0.8099	0.7167	0.8630	0.8861	0.9240

The best result in each dataset has been highlighted in boldface

conduct undersampling on next round, thus we did not adopt online learning in this algorithm.

4) *ROS-ELM [25]*: It adopts ROS as the balance control strategy in the procedure of active learning. In particular, during active learning, all currently labeled instances must be reserved for generating the oversampling instances on the next round. In each round, a new increased oversampling set will be learned incrementally.
5) *BootOS-ELM [25]:* It adopts the BootOS algorithm as the balance control strategy in the procedure of active learning. The procedure of BootOS is similar to ROS. Parameter K in BootOS was designated a default value 5. When the number of the labeled instances belonging to the minority class is smaller than K, it adopts ROS to copy minority instances.
6) *Active Cost Sensitive (ACS)-SVM [26]:* CS-SVM is adopted as the balance control strategy during active learning, and SVM is used as the baseline classifier. All parameters inherit from [26].

All algorithms were implemented in MATLAB 2013a environment, and experiments were conducted on an Intel(R) Core i7 6700HQ 8 cores CPU (main frequency: 2.60 GHz for each core) and 16-GB RAM.

To evaluate the performance of each algorithm at a specific time point to construct the learning curve, we adopted the G-mean metric that can be calculated as

$$G\text{-mean} = \sqrt{\text{Acc}_+ \times \text{Acc}_-} \tag{30}$$

where Acc_+ and Acc_- denote the accuracy of the minority class and the majority class, respectively. The G-mean metric evaluates the tradeoff between the accuracy of the minority class and that of the majority class. Then, to evaluate the quality of the overall learning procedure, we adopt a popular active learning evaluation metric: area under the learning curve (ALC) [54]. The ALC metric is calculated based on the G-mean values of all time points on the learning curve. Therefore, we can say that the G-mean metric here is only used to serve for the ALC metric. To compare various algorithms fairly, the ALC metric was calculated upon exhausting all unlabeled instances.

For each algorithm related to ELM, a sigmoid function is used to calculate the hidden-layer output matrix, and two main parameters L and C are determined by grid search, where $L \in \{10, 20, \ldots, 200\}$ and $C \in \{2^{-4}, 2^{-2}, \ldots, 2^{20}\}$. For ACS-SVM, the Gaussian radial basis kernel function is adopted, and the penalty factor C and the width parameter σ are also tuned by grid search, where $C \in [2^{-8}, 2^{-7}, \ldots, 2^{8}]$ and $\sigma \in [2^{-8}, 2^{-7}, \ldots, 2^{8}]$.

Considering the randomness of the experiments, the experimental results may be unstable. Therefore, we adopt 50 times random fivefold cross-validation to calculate the average result for each algorithm.

C. Comparison of Classification Performance

We first test the ALC metric of various compared algorithms on 32 given data sets. Table III presents the average results of 50 times random fivefold cross-validation.

We can see from Table III that on most data sets, the algorithms with the balance control strategy improve the performance to some extent. That means that inserting balance control strategy into the procedure of active learning is effective and necessary. However, on several other data sets, including vowel0, banknote-authentication, segmentation grass and magic, the promotion is not clear. Specifically, on the data sets mentioned earlier, it is possible to reduce the performance for those algorithms with the balance control strategy. As discussed in Section III, the performance of active learning upon the skewed data distribution has a relationship with multiple factors. Therefore, we consider that there might be a low imbalance ratio, large class margin, and no existence of small disjunctions in those four data sets. In fact, we observe that the imbalance ratio of those data sets is indeed very low. As expected, the results testify our hypothesis.

For sampling-based active learning algorithms, BootOS-ELM significantly outperforms ROS-ELM on a majority of the data sets, as it effectively alleviates the overfitting of the classification model, further help to extract more useful unlabeled instances during active learning. It is also surprising that RUS-ELM can produce a quite high performance on many data sets, which is not in accordance with the conclusion of Zhu and Hovy [25]. Zhu and Hovy [25] thought that the RUS strategy had two obvious drawbacks as follows: 1) RUS wasted many labeled instances and 2) RUS had a distinct reduction in performance. In fact, when we seriously examined the results in Table III, it was not difficult to observe that RUS-ELM generally performed better on those highly imbalanced data sets, such as abalone19, page-blocks5, microRNA, and Box H/ACA snoRNA. Of course, the performance of the RUS-ELM algorithm was quite unstable, it produced significantly lower ALC metrics on some data sets, e.g., 0.0000 on the magic data set. Therefore, we suggest using it with caution in practical applications.

When comparing ROW-ELM with the proposed AOW-ELM algorithm, we see that on a majority of the data sets, AOW-ELM significantly outperforms ROW-ELM, indicating that uncertainty-based instance selection can be faster than random instance selection to improve the quality of the classification model. In other words, the results also demonstrate the effectiveness of our proposed uncertainty estimation and selection strategy in the work of Yu *et al.* [21]. In addition, we also note that on several highly imbalanced data sets, ROW-ELM often performed better than AOW-ELM. We think that on these data sets, there may have existed much more noise that could influence the accuracy of uncertainty estimation, thereby uncertainty sampling was even worse than random sampling.

In contrast to the ACS-SVM algorithm [26], our proposed AOW-ELM algorithm seems not to be obviously superior. In particular, ACS-SVM achieves the best results on 12 data sets, while the best results of AOW-ELM only cover 11 data sets. Considering that they both adopt the idea of cost-sensitive learning as a balance control strategy, it is not difficult to understand why they had similar performance. Here, the distinction between ACS-SVM and AOW-ELM is merely correlated with different classification models adopted by them. On these data sets, SVM outperforms ELM, while on the other data sets, ELM outperforms SVM.

To present a thorough understanding of the comparison of various algorithms, we also provide their statistical results. Specifically, the Friedman test is used to detect a significant difference among a group of results, and the Holm post hoc test is adopted to examine whether the proposed algorithm is distinctive among a $1 \times N$ comparison [55], [56]. The post hoc procedure allows us to know whether a hypothesis of comparison of means could be rejected at a specified level of significance α. The adjusted p-value (APV) is also calculated to denote the lowest level of significance of a hypothesis that results in a rejection. Furthermore, we consider the average rankings of the algorithms to measure how good an algorithm is with respect to its partners. The ranking could be calculated by assigning a position to each algorithm depending on its performance on each data set. The algorithm that achieves the best performance on a specific data set will be given rank 1 (value 1), then the algorithm with the second-best result is assigned rank 2, and so forth. This task is conducted over all data sets, and finally an average ranking is calculated. The statistical results are presented in Table IV.

TABLE IV
STATISTICAL RESULTS OF VARIOUS ALGORITHMS

Algorithm	Average Ranking	APV
AOW-ELM	2.03	-
AO-ELM	6.47	**1.26×10^{-15}**
ROW-ELM	2.69	0.45
RUS-ELM	5.06	**7.96×10^{-8}**
ROS-ELM	5.19	**2.54×10^{-8}**
BootOS-ELM	3.94	**1.24×10^{-3}**
ACS-SVM	2.63	0.45

For a standard level of significance of α=0.05, the results which have significant difference compared with the proposed algorithm have been highlighted in boldface.

Table IV shows that our proposed AOW-ELM algorithm achieved the lowest average ranking, indicating that it is the best among all the algorithms. In addition, we observe that the APVs of the AO-ELM, RUS-ELM, ROS-ELM, and BootOS-ELM are lower than a standard level of significance of $\alpha = 0.05$. That means that these four algorithms are significantly different from the proposed AOW-ELM algorithm. However, we cannot say that AOW-ELM is significantly different from ROW-ELM and ACS-SVM, although it has a lower APV value than those two algorithms.

Fig. 6 presents the learning curves of seven compared algorithms on three representative data sets. In Fig. 6, the first subgraph is representative of most data sets that have a medium imbalance ratio and class overlapping. On this type of data set, all algorithms could rapidly promote the quality of the classification model in the initial stage of active learning, and then the learning curves would be steady or even declining. As expected, AOW-ELM has a faster learning speed than ROW-ELM, which demonstrates the effectiveness of the uncertainty-based instance selection strategy again. The second subgraph presents the learning curves generated on highly imbalanced data sets. As we know, on the data sets with a high imbalance ratio, there generally exists noise, and most minority instances are even embedded into the majority class; therefore, RUS may be more helpful for eliminating noise and promoting the quality of the classification model. For AO-ELM, which does not adopt any balance control strategy, all minority instances can be even misclassified. Of course, on this type of data set, our proposed algorithm is also effective to some extent, but generally, it is not the best choice.

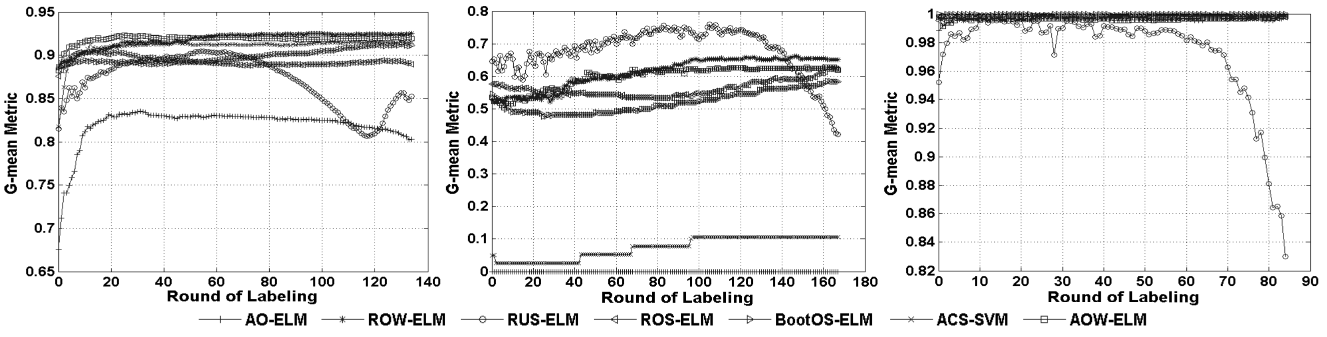

Fig. 6. Learning curves of seven compared active learning algorithms on three representative data sets: yeast-ME3, abalone19, and segmentation-grass, respectively (from left to right).

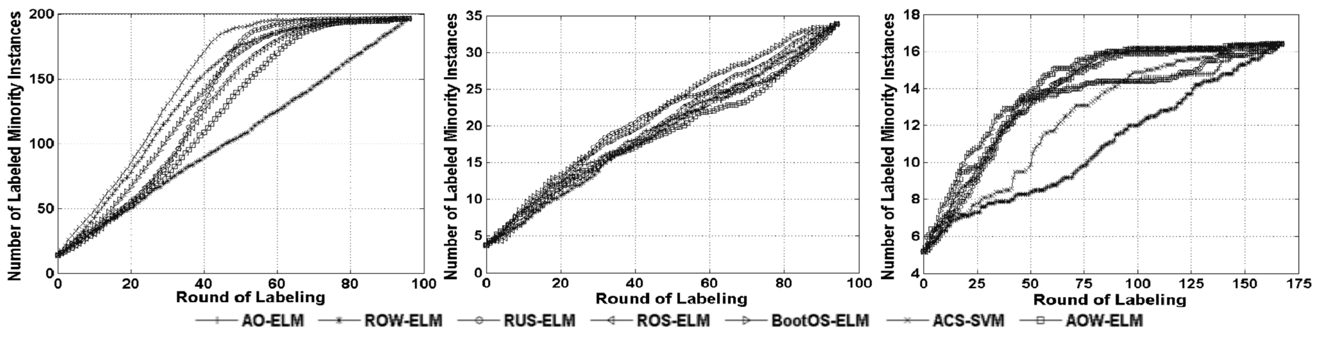

Fig. 7. Comparison of seven compared algorithms on the number of labeled instances belonging to the minority class in synchronization with the increase of labeling rounds on three representative data sets: cardiotocographyc1, seed2, and abalone19, respectively (from left to right).

The third subgraph in Fig. 6 generally represents the easy data sets that have a low imbalance ratio, large class margin, and low noise. Therefore, we can observe that there is no clear distinction among different learning algorithms except RUS-ELM. Obviously, for this type of data sets, RUS is not an excellent balance control strategy, and even the learning algorithm without a balance control strategy can generate good performance.

In addition to the observations discussed earlier, we also see an interesting phenomenon that although AOW-ELM generally performs better than ROW-ELM in the initial stage of active learning, it could be surpassed gradually. We think that this result is related to the proposed weight update rule. Actually, uncertainty-based sampling strategy often tends to collect more minority instances than a random sampling strategy in the initial stage of active learning. For the AOW-ELM algorithm, the total weights assigned to the minority instances is likely higher than that assigned for the majority instances, causing the later classification models to focus on the minority class, and then reduce the G-mean metric. Fortunately, in practical applications, we generally do not need to exhaust all unlabeled instances, so AOW-ELM is still a better choice than ROW-ELM.

Tomanek and Hahn [31] thought that an excellent active learning algorithm constructing upon the skewed data distribution should collect the unlabeled minority instances rapidly in the initial stage of learning. Here, we examined the rising trend of the minority labeled instances during active learning for each compared algorithm on three representative data sets, named cardiotocographyC1, seed2, and abalone19 (see Fig. 7).

The first subgraph in Fig. 7 presents a common trend that existed on most of the data sets. Here, the curve of ROW-ELM could be regarded as a benchmark, as it is approximately linear with respect to the imbalance ratio of the initially unlabeled set. As we can see in the first subgraph, all six other algorithms discover many more minority unlabeled instances than ROW-ELM does in the early stage of active learning. In particular, the AO-ELM algorithm could even exhaust all unlabeled minority instances approximately halfway through the learning. It is not difficult to understand the phenomenon by reviewing the discussions in Section III. Due to AO-ELM not adopting any balance control strategy, the classification hyperplane tends to appear in the region of the minority class, causing more minority instances to be extracted in the early stage of active learning. In contrast with several other algorithms, our proposed algorithm presents a more moderate trend, indicating that discovering more unlabeled minority instances in the early stage of learning should be helpful for promoting the quality of the classification model rapidly, but excessive extraction may be unwanted. The second subgraph in Fig. 7 is based on an easy data set. That means, in this type of data set, there existed a large margin between the instances belonging to the two different classes. As we know, for those easy data sets, nearly all learning algorithms could acquire satisfying results, thus they all approximately present a linear relation with the imbalance ratio of the

TABLE V
RUNNING TIME (SECONDS) COMPARISON OF VARIOUS ALGORITHMS

Dataset	AO-ELM	ROW-ELM	RUS-ELM	ROS-ELM	BootOS-ELM	ACS-SVM	AOW-ELM
cardiotocographyC1	2.24	2.37	3.43	4.03	6.53	7.53	2.38
cardiotocographyC5	2.04	2.16	2.29	4.61	7.94	5.17	2.17
cardiotocographyC0	2.25	2.40	2.96	5.08	7.91	5.57	2.39
cardiotocographyN2	2.24	2.34	3.28	5.42	7.21	6.64	2.39
cardiotocographyN3	2.29	2.42	2.92	5.36	7.23	4.40	2.48
cardiotocographyN1 vs. N3	2.22	2.33	3.73	4.58	6.84	6.55	2.37
mfeat-mor0	3.96	4.12	4.44	5.64	7.37	2.06	4.16
mfeat-mor01	4.01	4.16	4.85	6.53	8.70	3.39	4.22
mfeat-mor012	3.98	4.09	5.57	6.45	7.43	7.16	4.19
mfeat-mor0123	3.78	3.95	5.94	5.72	6.78	6.45	3.96
vowel0	0.97	1.03	1.15	2.53	3.03	2.22	1.08
seed2	0.28	0.44	0.34	0.65	0.68	0.74	0.47
yeast-ME3	2.18	2.32	2.51	3.69	5.59	2.86	2.35
yeast-ME2	2.17	2.37	2.14	4.27	5.45	4.77	2.41
yeast-ME1	2.20	2.35	2.36	4.18	6.28	4.71	2.41
banknote-authentication	1.94	2.10	2.15	4.25	4.42	5.70	2.16
segmentation-grass	0.24	0.39	0.28	0.63	0.65	0.64	0.38
segment1	2.48	2.55	3.20	5.50	8.42	4.80	2.66
segment12	2.46	2.57	3.85	5.91	8.19	5.66	2.66
segment123	2.44	2.50	4.39	5.93	7.65	6.26	2.59
page-blocks5	7.31	7.35	7.90	13.68	16.59	11.16	7.48
ILPD	1.74	2.08	1.91	4.51	4.83	7.94	2.16
abalone9	8.35	8.43	11.60	27.15	33.03	55.92	8.63
abalone19	7.33	7.53	7.62	31.58	39.05	48.25	7.60
credit card clients	5.35	5.21	10.54	18.60	22.99	98.66	5.41
wilt	5.59	5.46	5.85	11.47	15.33	17.19	5.51
haberman	0.52	0.71	0.61	1.25	1.33	2.99	0.82
letter-A	20.64	20.49	22.60	71.21	110.51	90.46	20.52
magic	16.96	16.87	19.35	70.26	82.25	269.89	17.19
SNP	5.45	5.38	6.35	19.28	23.90	26.30	5.49
microRNA	9.82	9.75	10.61	43.08	62.78	78.68	9.98
Box H/ACA snoRNA	9.60	9.29	9.77	31.92	41.82	63.37	9.22

initially unlabeled set. For those difficult data sets that have a high imbalance ratio, large class overlapping, and high noise, all algorithms with uncertainty sampling can help discover more unlabeled minority instances than the benchmark, i.e., ROW-ELM, in the early stage of learning.

In our experiments, we have not run the proposed algorithm on multiclass data sets. In fact, on multiclass imbalance scenarios, two main challenges are presented: 1) cold start frequently happens as the instances belonging to some classes are extremely scarce and 2) the weight updating rule will lose efficacy as most multiclass imbalanced data sets are highly skewed. We believe that the cold start problem has to be solved by some participation by human experts, and the second problem could be addressed by improving the weight updating rule to equifrequently extract unlabeled instances from different classes during active learning

D. Comparison of Running Time

Nevertheless, we should also consider the computation time. From Table V, we can see that AOW-ELM generally has a similar running time as AO-ELM and ROW-ELM do, but needs less running time than several other algorithms do. This is because both RUS-ELM and ACS-SVM do not adopt incremental learning, ROS-ELM wastes some time to copy the minority instances, while BootOS-ELM consumes much time on neighborhood computation and synthetic instance generation. Therefore, we can say that AOW-ELM is a time-saving algorithm. Its superiority would be increasingly obvious in synchronization with increases in the instances and attributes existing in a data set. Of course, the comparison between ACS-SVM and AOW-ELM may be unfair as ACS-SVM does not adopt incremental learning. However, considering that training an individual SVM classification model is generally much more time-consuming than training an ELM model, we believe AOW-ELM would still be more time-saving than ACS-SVM with incremental learning.

E. Discussion About Stopping Condition

Finally, we wish to discuss the stopping condition to make clear the optimal setting for the threshold μ. We randomly select three representative data sets that reflect the concepts of easy, medium, and difficult. We let μ increase from 0.1 to 1.0 with an increment of 0.1, and then the distinction between the performance of each stopping point and the overall best performance was presented in Fig. 8.

In Fig. 8, we observe an approximately uniform trend on the three different data sets. That is, at the beginning, the distinction always decreased sharply in synchronization with the increase of the threshold μ, and it then gradually became stable. It is easy to understand this phenomenon: the threshold denotes the maximum confidence level. As the threshold increases, more uncertain instances have been labeled, so the classification model must be closer to the optimal performance. Although the trends are similar on the three different data sets, their best thresholds corresponding to the inflection points are totally different. On the easy data set seed2, $\mu = 0.3$ could guarantee to acquire a good enough performance, while on the medium data set cardiotocographyC1, the best threshold is 0.4. For the difficult data set Box H/ACA snoRNA, when the threshold μ equals 0.6, it provides the best tradeoff between the classification performance and labeling expenses. Revisiting Fig. 5, it is not difficult to understand the phenomenon earlier. In Fig. 5, a small threshold means exhausting the instances in a narrow margin surrounding the classification boundary, while a large threshold means enlarging the margin and exhausting more instances. Considering the easy data

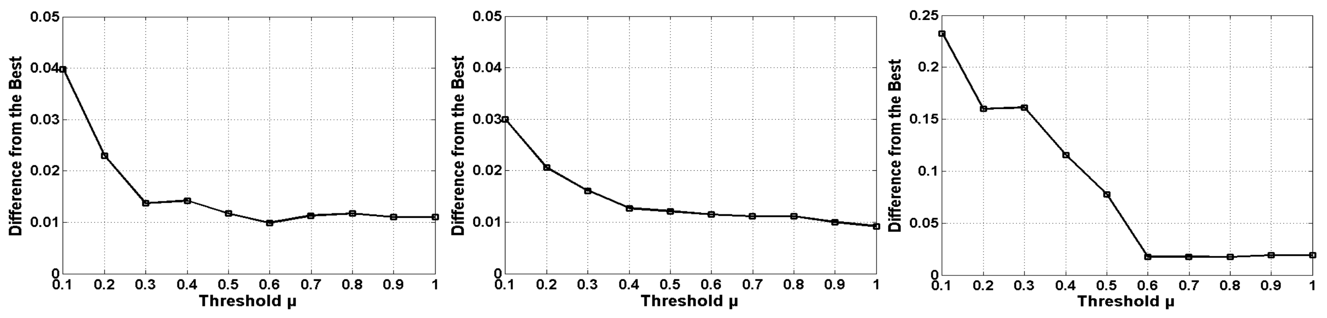

Fig. 8. Difference between the performance of each stopping point (denoting by a specific threshold μ) and the best performance on three representative data sets: seed2, cardiotocographyC1, and Box H/ACA snoRNA, respectively (from left to right).

sets, which have low imbalance ratio and small classification overlapping, a small threshold is enough to guarantee precisely describing the classification boundary. However, for those difficult data sets with a high imbalance ratio, large class overlapping, high noise, and even small disjunctions, we have to assign a large threshold for them to construct excellent enough classification models. In practical applications, the readers are suggested to examine the instance distribution on the initially labeled set, and then allocate an appropriate threshold.

VI. Conclusion

In this paper, we explore the problem of active learning in class imbalance scenario, and present a solution of online WELM named the AOW-ELM algorithm. We find that the harmfulness of skewed data distribution is related to multiple factors, and can be seen as a combination of these factors. Hierarchical clustering can be effectively used to previously extract initial representative instances into a seed set to address the potential missed cluster effect and cold start phenomenon. The comparison between the proposed AOW-ELM algorithm and some other benchmark algorithms indicates that AOW-ELM is an effective strategy to address the problem of active learning in a class imbalance scenario. The merits of the AOW-ELM algorithm can be summarized as follows.

1) It has a robust weight update rule.
2) Its running time is fast and linear with the training instances.
3) It has a flexible early stopping criterion.
4) It is appropriate for various types of data sets.

In the future work, we will focus more on the problem of active learning on multiclass imbalanced data sets. In addition, the active learning strategies addressing imbalanced and unlabeled data streams with handling concept drifts will also be investigated.

References

[1] B. Settles, "Active learning literature survey," Dept. Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 1648, Jan. 2010.

[2] M. Sugiyama and S. Nakajima, "Pool-based active learning in approximate linear regression," *Mach. Learn.*, vol. 75, no. 3, pp. 249–274, 2009.

[3] A. K. McCallumzy and K. Nigam, "Employing EM and pool-based active learning for text classification," in *Proc. ICML*, 1998, pp. 359–367.

[4] J. Smailović, M. Grčar, N. Lavrač, and M. Žnidaršič, "Stream-based active learning for sentiment analysis in the financial domain," *Inf. Sci.*, vol. 285, pp. 181–203, Nov. 2014.

[5] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu, "Batch mode active learning and its application to medical image classification," in *Proc. ICML*, 2006, pp. 417–424.

[6] D. Lewis and J. Catlett, "Heterogeneous uncertainty sampling for supervised learning," in *Proc. ICML*, 1994, pp. 148–156.

[7] M. Lindenbaum, S. Markovitch, and D. Rusakov, "Selective sampling for nearest neighbor classifiers," *Mach. Learn.*, vol. 54, no. 2, pp. 125–152, 2004.

[8] S. Dasgupta and D. Hsu, "Hierarchical sampling for active learning," in *Proc. ICML*, 2008, pp. 208–215.

[9] R. Wang, S. Kwong, and D. Chen, "Inconsistency-based active learning for support vector machines," *Pattern Recognit.*, vol. 45, no. 10, pp. 3751–3767, 2012.

[10] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Large-scale text categorization by batch mode active learning," in *Proc. WWW*, 2006, pp. 633–642.

[11] N. Roy and A. McCallum, "Toward optimal active learning through Monte Carlo estimation of error reduction," in *Proc. ICML*, 2001, pp. 441–448.

[12] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proc. ACM SIGIR*, 1994, pp. 3–12.

[13] J. He and J. G. Carbonell, "Nearest-neighbor-based active learning for rare category detection," in *Proc. NIPS*, 2008, p. 633–640.

[14] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Proc. NIPS*, 1995, pp. 231–238.

[15] K. Fukumizu, "Statistical active learning in multilayer perceptrons," *IEEE Trans. Neural Netw.*, vol. 11, no. 1, pp. 17–26, Jan. 2000.

[16] A. I. Schein and L. H. Ungar, "Active learning for logistic regression: An evaluation," *Mach. Learn.*, vol. 68, no. 3, pp. 235–265, 2007.

[17] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Nov. 2001.

[18] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *Proc. 9th ACM Int. Conf. Multimedia*, 2001, pp. 107–118.

[19] Y. Qi and G. Zhang, "Strategy of active learning support vector machine for image retrieval," *IET Comput. Vis.*, vol. 10, no. 1, pp. 87–94, 2016.

[20] C. Pan, D. S. Park, H. Lu, and X. Wu, "Color image segmentation by fixation-based active learning with ELM," *Soft Comput.*, vol. 16, no. 9, pp. 1569–1584, 2012.

[21] H. Yu, C. Sun, W. Yang, X. Yang, and X. Zuo, "AL-ELM: One uncertainty-based active learning algorithm using extreme learning machine," *Neurocomputing*, vol. 166, pp. 140–150, Oct. 2015.

[22] H. Liao, L. Chen, Y. Song, and H. Ming, "Visualization-based active learning for video annotation," *IEEE Trans. Multimedia*, vol. 18, no. 11, pp. 2196–2205, Nov. 2016.

[23] M. Wang and X. S. Hua, "Active learning in multimedia annotation and retrieval: A survey," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 2, p. 10, 2011.

[24] P. H. Gosselin and M. Cord, "Active learning methods for interactive image retrieval," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1200–1211, Jul. 2008.

[25] J. Zhu and E. H. Hovy, "Active learning for word sense disambiguation with methods for addressing the class imbalance problem," in *Proc. EMNLP-CoNLL*, 2007, pp. 783–790.

[26] M. Bloodgood and K. Vijay-Shanker, "Taking into account the differences between actively and passively acquired data: The case of active learning with support vector machines for imbalanced datasets," in *Proc. Hum. Lang. Technol.*, 2009, pp. 137–140.

[27] S. Rajan, J. Ghosh, and M. M. Crawford, "An active learning approach to hyperspectral data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 4, pp. 1231–1242, Apr. 2008.

[28] G. Riccardi and D. Hakkani-Tur, "Active learning: Theory and applications to automatic speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 13, no. 4, pp. 504–511, Jul. 2005.

[29] Y. Li and L. Guo, "An active learning based TCM-KNN algorithm for supervised network intrusion detection," *Comput. Secur.*, vol. 26, no. 7, pp. 459–467, 2007.

[30] W. Xiong, L. Xie, S. Zhou, and J. Guan, "Active learning for protein function prediction in protein–protein interaction networks," *Neurocomputing*, vol. 145, pp. 44–52, Dec. 2014.

[31] K. Tomanek and U. Hahn, "Reducing class imbalance during active learning for named entity annotation," in *Proc. 5th Int. Conf. Knowl. Capture*, 2009, pp. 105–112.

[32] S. Ertekin, J. Huang, and C. L. Giles, "Active learning for class imbalance problem," in *Proc. SIGIR*, 2007, pp. 823–824.

[33] S. Ertekin, J. Huang, L. Bottou, and C. L. Giles, "Learning on the border: Active learning in imbalanced data classification," in *Proc. CIKM*, 2007, pp. 127–136.

[34] 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.

[35] 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.

[36] 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.

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

[38] R. Fletcher, "Constrained optimization," in *Practical Methods of Optimization*, vol. 2. Hoboken, NJ, USA: Wiley, 1981.

[39] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

[40] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–450, 2002.

[41] 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.

[42] H. Yu, C. Mu, C. Sun, W. Yang, X. Yang, and X. Zuo, "Support vector machine-based optimized decision threshold adjustment strategy for classifying imbalanced data," *Knowl.-Based Syst.*, vol. 76, pp. 67–78, Mar. 2015.

[43] 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.

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

[45] K. Tomanek, F. Laws, U. Hahn, and H. Schütze, "On proper unit selection in active learning: co-selection effects for named entity recognition," in *Proc. NAACL HLT Workshop Act. Learn. Natural Lang. Process.*, 2009, pp. 9–17.

[46] H. He and Y. Ma, *Imbalanced Learning: Foundations, Algorithms, and Applications*. Hoboken, NJ, USA: Wiley, 2013.

[47] B. Mirza, Z. Lin, and K.-A. Toh, "Weighted online sequential extreme learning machine for class imbalance learning," *Neural Process. Lett.*, vol. 38, no. 3, pp. 465–486, 2013.

[48] S. Ding *et al.*, "Kernel based online learning for imbalance multiclass classification," *Neurocomputing*, vol. 277, pp. 139–148, Feb. 2018.

[49] J. Zhu, H. Wang, E. Hovy, and M. Ma, "Confidence-based stopping criteria for active learning for data annotation," *ACM Trans. Speech Lang. Process.*, vol. 6, no. 3, p. 3, 2010.

[50] 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

[51] Q. Zou, M. Guo, Y. Liu, and J. Wang, "A classification method for class imbalanced data and its application on bioinformatics," *J. Comput. Res. Develop.*, vol. 47, no. 8, pp. 1407–1414, 2010.

[52] C. Xue, F. Li, T. He, G.-P. Liu, Y. Li, and X. Zhang, "Classification of real and pseudo microRNA precursors using local structure-sequence features and support vector machine," *BMC Bioinf.*, vol. 6, p. 310, Dec. 2005.

[53] J. Hertel, I. L. Hofacker, and P. F. Stadler, "SnoReport: Computational identification of snoRNAs with unknown targets," *Bioinformatics*, vol. 24, no. 2, pp. 158–164, 2008.

[54] I. Guyon, G. Gawley, G. Dror, and V. Lemaire, "Results of the active learning challenge," in *Proc. JMLR, Workshop Conf.*, vol. 16, 2011, pp. 19–45.

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

[56] S. Garcia, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, pp. 2044–2064, May 2010.

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

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. He has authored or co-authored more than 60 journal and conference papers, and four books. His current research interests include machine learning, data mining, and bioinformatics.

Dr. Yu is the member in the Organizing Committee of several international conferences. He is also the member of ACM, China Computer Federation, and the Youth Committee of the Chinese Association of Automation. He is the reviewer for more than 20 high-quality international journals.

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, Jiangsu University of Science and Technology. He has authored or co-authored over 100 research articles on the professional journals and conferences. His current research interests include machine learning, knowledge discovery, granular computing, and rough set theory.

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, 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, Jiangsu University of Science and Technology, Zhenjiang, China. He has authored or co-authored more than 10 top journal and conference papers. His current research interests include software repository mining, software maintenance, software evolution, and intelligent computation.

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.

In 2004, he joined the Department of Computer Science, Chinese University of Hong Kong, Hong Kong, as a Post-Doctoral Fellow. 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, Nanjing. 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 the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS since 2010, *Neural Processing Letters* since 2008, the *International Journal of Swarm Intelligence Research* since 2010, and the Recent Parents of Computer Science since 2008. He is involved in organizing several international conferences.