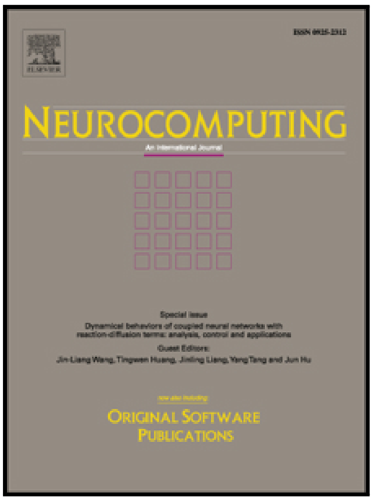


Online Sequential Prediction of Imbalance Data with Two-Stage Hybrid Strategy by Extreme Learning Machine

Wentao Mao, Jinwan Wang, Ling He, Yangyang Tian

PII: S0925-2312(17)30163-7
DOI: 10.1016/j.neucom.2016.05.111
Reference: NEUCOM 17975

To appear in: *Neurocomputing*

Received date: 26 September 2015
Revised date: 12 May 2016
Accepted date: 27 May 2016

Please cite this article as: Wentao Mao, Jinwan Wang, Ling He, Yangyang Tian, Online Sequential Prediction of Imbalance Data with Two-Stage Hybrid Strategy by Extreme Learning Machine, *Neurocomputing* (2017), doi: 10.1016/j.neucom.2016.05.111

Highlights

- We derive an efficient leave-one-out cross-validation error estimate for OS-ELM
- We propose a two-stage hybrid strategy for online sequential data imbalance problem
- We prove theoretically the rationality and validity of this strategy
- We proposed a new OS-ELM method for solving online sequential data imbalance problem.
- We conduct a number of computer experiments on UCI and real-life data sets

Online Sequential Prediction of Imbalance Data with Two-Stage Hybrid Strategy by Extreme Learning Machine

Wentao Mao[a,b,c,*], Jinwan Wang[a], Ling He[a], Yangyang Tian[a]

[a] *School of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China*

[b] *Engineering Technology Research Center for Computing Intelligence & Data Mining, Henan Province, Xinxiang 453007, China*

[c] *School of Mechanics and Civil & Architecture, Northwestern Polytechnical University, Xián, 710129, Shaanxi, P.R.China*

Abstract

In many practical engineering applications, data tend to be collected in online sequential way with imbalanced class. Many traditional machine learning methods such as support vector machine and so on generally get biased classifier which leads to lower classification precision for minor class than major class. To get fast and efficient classification, a new online sequential extreme learning machine method with two-stage hybrid strategy is proposed. In offline stage, data-based strategy is employed, and the principal curve is introduced to model the distribution of minority class data. In online stage, algorithm-based strategy is employed, and a new leave-one-out cross-validation method using Sherman-Morrison matrix inversion lemma is proposed to tackle online imbalance data, meanwhile, with add-delete mechanism for updating network weights. And the rationality of this strategy is proved theoretically. The proposed method is evaluated on four UCI

*Corresponding author
Email address: maowt.mail@gmail.com (Wentao Mao)

datasets and the real-world Macau air pollutant forecasting dataset. The experimental results show that, the proposed method outperforms the classical ELM, OS-ELM and meta-cognitive OS-ELM in terms of generalization performance and numerical stability.

Keywords:
Extreme learning machine, Imbalance problem, Principal curve, Leave-one-out cross-validation, Online sequential learning

1. Introduction

Data imbalance problem occurs commonly in real applications[1]. In this problem, the data are not uniformly distributed. Many batch learning algorithms suffer from this problem. Majority class data tend to cause the model learning repeatedly the same knowledge which can already be correctly classified. As a result, the prediction model will incur bias and over-fitting to the majority class. This is called ”undo” effect to minority class[2]. For example, suppose that there are 90% of training observations belonging to majority class while 10% to minority class. In this setting, even all of minority class samples are predicted wrongly, the total prediction precision is still 90% when all of majority class samples are predicted correctly. Obviously, the final numerical value cannot reflect the actual classification ability. To improve the generalization of imbalance data, two kinds of strategies are widely applied. Data-based strategy[3][4] only focus on sampling method that adjust the size of training data, with increasing data for minority class or decreasing data for majority class. Algorithm-based strategy[5][6] mainly involves introduction of the cost-sensitive information in a classification algorithm to handle data imbalance. These two strategies and their extensions have been proved effective for many data imbalance problems[7][8].

In this paper, we focus on one special imbalance problem. In many applications, especially in *Big Data*, data tend to be collected in online sequential way with imbalanced class. For example, in air pollutants forecasting application, the air quality data are usually collect sequentially(e.g., hourly or daily), and the number of observations with good air quality is generally far larger than the number with bad air quality[9]. In Big data application, data tend to reach chunk by chunk due to the limitation of RAM[10]. In the online sequential phenomenon we call it as *online sequential data imbalance problem*. The traditional imbalance strategy discussed above could suffer from this problem, for they generally don't possess the online and sequential characteristic. Therefore, it is of great significance for improving the generalization performance on online sequential imbalance data.

There are two main challenges to improve the generalization for online sequential data imbalance problem. One is how to choose a proper baseline algorithm for online sequential setting, and another is how to make the majority and minority classes balanced while obeying the original distribution of online sequential data. As a extension form of single-hidden layer feedforward neural network(SLFN), extreme learning machines(ELMs), introduced by Huang[11], have shown its very high learning speed and good generalization performance in solving many problems of regression estimate and pattern recognition[12][13]. As a sequential modification of ELM, online sequential ELM(OS-ELM) proposed by Liang[14] can learn data one-by-one or chunk-by-chunk. In many applications such as time-series forecasting, OS-ELMs also show good generalization at extremely fast learning speed[15]. Threfore, OS-ELM is a proper solution for the first challenge. ELM and OS-ELM-based methods have also shown their good performance in solving data imbalance problem[16][17][18]. To solve the second challenge, Vong[9] introduced prior duplication strategy to generate more minority class data, and utilized OS-ELM to train an online sequential prediction model. To some extent, this research serves as a meaningful probe to solve online sequential data imbalance problem. The experimental results on air pollutants forecasting data from Macau also show this method has higher generalization than many classical batch learning algorithms in online setting. To our best knowledge, there are very few other researches concerning this topic.

However, although OS-ELM in [9] works effectively on online sequential data, it still not yet applies better imbalance strategy in online stage. Specifically speaking, as the imbalance strategy in [9] is only duplicating minority class observations, it couldn't explore the real data distribution of minority class. In other words, it couldn't guarantee the new generated samples obeying the data distribution and to be valuable for prediction. On the other hand, in the level of algorithm, the method in [9] lacks an efficient mechanism to exclude the redundant and harmful samples including new generated virtual samples. Therefore, to solve this problem, this paper blends the data-based strategy and algorithm-based strategy, and proposes a new two-stage hybrid strategy. In offline stage, the principal curve is used to explore the data distribution and further establish an initial model on imbalance data. Principal curve has strict mathematical property to guarantee good ability to reflect data's skeleton. In online stage, some virtual samples are generated according to the principal curve, and a new leave-one-out(LOO) cross-validation method is proposed to tackle online unbalanced data with add-delete mechanism for updating network weights. This LOO cross-validation method uses Sherman-Morrison matrix inversion lemma, and can effectively reduce the redundant computation of matrix inversion. This article is an extended version of the paper in ELM2015[19]. Compared to the state-of-the-art literature for data imbalance problem, the contribution of this work includes three parts: 1)we propose a two-stage hybrid strategy for online sequential data imbalance problem which can improve the classification accuracy of minority class; 2)we derive an efficient leave-one-out cross-validation error estimate for OS-ELM to improve the generalization performance in the process of sample reconstruction, and 3)we provide a theoretical analysis of the rationality and validity for this strategy. According to our literature survey, there are no other researches so far about generalization evaluation of OS-ELM.

The paper is organized as follows. In section 2, a brief review to ELM and principal curve is provided. In section 3, we describe the online sequential learning paradigm on imbalance data, including offline stage and online stage. Section 4 is devoted to computer experiments on two different types of data sets, followed by a conclusion of the paper in the last section.

2. Background

2.1. Review of ELM and OS-ELM

As the theoretical foundations of ELM, [20] studied the learning performance of SLFN on small-size data set, and found that SLFN with at most N hidden neurons can learn N distinct samples with zero error by adopting any bounded nonlinear activation function. Then based on this concept, Huang[21][22] pointed out that ELM can analytically determine the output weights by a simple matrix inversion procedure as soon as the input weights and hidden layer biases are generated randomly, and then obtain good generalization performance with very high learning speed. Here a brief summary of ELM is provided[25].

Given a set of i.i.d. training samples $\{(\mathbf{x}_1, \mathbf{t}_1), \cdots, (\mathbf{x}_N, \mathbf{t}_N)\} \subset \mathbf{R}^d \times \mathbf{R}^n$, standard SLFNs with L hidden nodes are mathematically formulated as[18]:

$$\sum_{i=1}^{L} \beta_i g_i(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i g_i(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{o}_j, \; j = 1, ..., N \tag{1}$$

where $g(x)$ is activation function, $\mathbf{w}_i = [w_{i1}, w_{i2}, ..., w_{id}]^T$ is input weight vector connecting input nodes and the $i\text{th}$ hidden node, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{in}]^T$ is the output weight vector connecting output nodes and the $i\text{th}$ hidden node, b_i is bias of the $i\text{th}$ hidden node. Huang[17] has rigorously proved that for *N* arbitrary distinct samples and any $(\mathbf{w}_i, b_i)$ randomly chosen from $\mathbf{R}^d \times \mathbf{R}$ according to any continuous probability distribution, the hidden layer output matrix $\mathbf{H}$ of a standard SLFN with *N* hidden nodes is invertible and $\|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\| = 0$ with probability one if the activation function $g : \mathbf{R} \mapsto \mathbf{R}$ is infinitely differentiable in any interval. Then given $(\mathbf{w}_i, b_i)$, training a SLFN equals finding a least-squares solution of the following equation[21]:

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

where

$$\mathbf{H}(\mathbf{w}_1, ..., \mathbf{w}_L, b_1, ..., b_L, \mathbf{x}_1, ..., \mathbf{x}_L) = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ \vdots & \cdots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L}$$

$$\boldsymbol{\beta} = [\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_L]^T$$

$$\mathbf{T} = [\mathbf{t}_1, ..., \mathbf{t}_N]^T$$

Considering most cases that $L \ll N$, $\boldsymbol{\beta}$ cannot be computed through the direct matrix inversion. Therefore, the smallest norm least-squares solution of equation (2) is calculated as:

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T} \tag{3}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$. Based on the above analysis, Huang[21] proposed ELM whose framework can be stated as follows:

Step 1. Randomly generate input weight and bias $(\mathbf{w}_i, b_i), i = 1, \cdots, L$.
Step 2. Compute the hidden layer output matrix $\mathbf{H}$.
Step 3. Compute the output weight $\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T}$.

Therefore, the output of SLFN can be calculated by $(\mathbf{w}_i, b_i)$ and $\hat{\boldsymbol{\beta}}$:

$$f(\mathbf{x}_j) = \sum_{i=1}^{L} \hat{\beta}_i g_i(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \hat{\boldsymbol{\beta}} \cdot h(\mathbf{x}_j) \tag{4}$$

Like ELM, all the hidden node parameters in OS-ELM are randomly generated, and the output weights are analytically determined based on the sequentially arrived data. OS-ELM process is divided into two steps: initialization phase and sequential learning phase[14].

Step 1. Initialization phase: choose a small chunk $M_0 = \{(x_i, t_i), i = 1, 2, ..., N_0\}$ of initial training data, where $N_0 \geq L$.

1) Randomly generate the input weight $\mathbf{w}_i$ and bias b_i,$i = 1, 2, ..., L$. Calculate

the initial hidden layer output matrix $\mathbf{H_0}$:

$$\mathbf{H}_0 = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1) \\ \mathbf{h}(\mathbf{x}_2) \\ . \\ . \\ . \\ \mathbf{h}(\mathbf{x}_{N_0}) \end{bmatrix} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ g(\mathbf{w}_1 \cdot \mathbf{x}_2 + b_1) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_2 + b_L) \\ & ... & \\ & ... & \\ & ... & \\ g(\mathbf{w}_1 \cdot \mathbf{x}_{N_0} + b_1) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_{N_0} + b_L) \end{bmatrix}_{N_0 \times L} \tag{5}$$

2) Calculate the output weight vector:

$$\boldsymbol{\beta}^0 = \mathbf{D}_0 \mathbf{H}_0{}^T \mathbf{T}_0 \tag{6}$$

where $\mathbf{D}_0 = (\mathbf{H}_0{}^T \mathbf{H}_0)^{-1}, \quad \mathbf{T}_0 = [t_1, t_2, ...t_{N_0}]^T$.

3) Set $k = 0$

Step 2 Sequential learning phase

1) Learn the $(k+1)-$th training data: $d_{k+1} = (\mathbf{x}_{N_0+k+1}, t_{N_0+k+1})$

2) Calculate the partial hidden layer output matrix:

$$\mathbf{H}_{k+1} = [g(\mathbf{w}_1 \cdot \mathbf{x}_{N_0+k+1} + b_1) \quad ... \quad g(\mathbf{w}_L \cdot \mathbf{x}_{N_0+k+1} + b_L)]_{1 \times L} \tag{7}$$

Set $\mathbf{T}_{k+1} = [t_{N_0+k+1}]^T$.

3) Calculate the output weight vector

$$\mathbf{D}_{k+1} = \mathbf{D}_k - \mathbf{D}_k \mathbf{H}_{k+1}^T (\mathbf{I} + \mathbf{H}_{k+1} \mathbf{D}_k \mathbf{H}_{k+1}{}^T)^{-1} \mathbf{H}_{k+1} \mathbf{D}_k \tag{8}$$

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k + \mathbf{D}_{k+1} \mathbf{H}_{k+1}{}^T (\mathbf{T}_{k+1} - \mathbf{H}_{k+1} \beta^k) \tag{9}$$

4) Set $k = k + 1$. Go to step 2(1).

2.2. Review of principal curve

In 1983, Hastie[23] firstly introduced the theory of principal curve. Afterwards, this theory was successfully applied to solve practical problems, like data visualisation[24] and ecology analysis[26], etc. Principal curve is extension of principal component analysis and its basic idea is to find a continuous one-dimensional manifold that approximate the data in the sense of ”self-consistency”, i.e. the curve should coincide at each position with the expected value of the data projecting to that position[26]. Intuitively, this curve passes through the ”middle” of a high-dimensional data set. The difference between principal curve and regression is a non-parametric method is used to explore the trajectory in data set, without any assumption about causal relationships among instances[27]. Hastie argued that principal curve can truly reflect the shape of data set, i.e. the curve is skeleton and the data set is cloud[23].

In this paper, we choose $k-$ curve for its good practicability. [28] proved that for any data set with finite second moments there always exists a principal curve. The definition of $k-$ curve is listed as follows.

Definition 1 ($K-$ principal curve). *[28] For a data set $X = \{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}^d$, a curve f^* is called a K principal curve of length L for X if f^* minimizes $\triangle(f)$ over all curves of length less than or equal to L, where f is a continuous function $f : I \to \mathbb{R}^d$, $\triangle(f)$ is the expected squared distance between X and f and defined as:*

$\triangle(f) = E[\triangle(X, f)] = E[\inf_\lambda \|X - f(\lambda)\|^2] = E[\|X - f(\lambda_f(X))\|^2]$

where $\lambda_f(x) = \sup\{\lambda : \|X - f(\lambda)\| = \inf_\tau \|x - f(\tau)\|\}$ is called projection index.

The goal of K curve is to find a set of polygonal lines with $K-$ segments and with a given length to approximate the principal curve. The algorithm is based on a common model about complexity in statistical learning theory[28]. The framework of this algorithm can be summarized as follows. At the beginning, $f_{1,n}$ is initialized by the first principal component line and in each iteration step, a new vertex is added on $f_{i-1,n}$ which is obtained in $i-1$ step, to increase the number of segments. According to the principle of minimizing the projection distance, the position of vertexes are optimized to construct a new curve $f_{i,n}$. Kegl[28] gave a detailed description of this algorithm.

3. OS-ELM with two-stage hybrid strategy

Considering the data imbalance problem and online sequential learning problem simultaneously, this section presents a new OS-ELM method with two-stage hybrid strategy. This strategy uses data-based strategy in offline stage to establish better initial model for OS-ELM, and uses algorithm-based strategy in online stage to pick more valuable training samples including virtual minority class samples. This strategy works in two stages with the target of reducing large gap between minority and majority classes, so we call it as two-stage hybrid strategy. Here the offline and online stages are both in training process, absolutely same as OS-ELM.

3.1. Offline stage

In OS-ELM, the initial model in offline phase depends on the output weights in online phase heavily. Therefore, it is important to improve the initial model. As principal curve can truly reflect the shape of data set, we employ the principal curve in offline stage to balance the samples in minority and majority classes, i.e., generate virtual samples in minority class, and filter samples in majority class. Then the initial model is established using the obtained dataset.

The concrete process can be described as follows:

Step 1. Plot the principal curve of minority or majority samples $D = \{(Dx_i, Dy_i)\}$ using k-curve[24] presented in section 2.2. Get the samples $S = \{(Sx_i, Sy_i)\}$ on principal curve using polynomial interpolation.
Step 2. Generate virtual samples for minority class by adding Gaussian white noise on original samples $Z = \{(Zx_i, Zy_i)\}$. Keep the majority class sample set unchanged.
Step 3. Filter samples for two classes using the following formulation:

$$\begin{cases} |Sx_i - Dx_j| \leq \delta_x \\ |Sy_i - Dy_j| \leq \delta_y \end{cases} \tag{10}$$

where δ_x, δ_y are pre-defined threshold. The samples not meeting equation (10) will be excluded from the minority or majority class sample set.

It is worthy noting that, to plot principal curve, we need to select the most important feature for multi-dimensional data. In experiment section, we employ RELIEF algorithm to determine the features for principal curve. Of course, other algorithms, like maximizing information entropy, also can be introduced to reach this goal.

Define the obtained sample set filtered by principal curve as $D = \{(\mathbf{x}_i, t_i) | i = 1, 2, ..., N\}$. Given activation function $g(x)$ and the number of hidden neurons *L*, choose input weight $\mathbf{w}_i$ and bias b_i, $i = 1, 2, ...L$ randomly and calculate the input matrix $\mathbf{H}_1$:

$$\mathbf{H}_1 = \begin{bmatrix} \mathbf{h}(\mathbf{x}_1) \\ \mathbf{h}(\mathbf{x}_2) \\ . \\ . \\ . \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ g(\mathbf{w}_1 \cdot \mathbf{x}_2 + b_1) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_2 + b_L) \\ & ... & \\ & ... & \\ & ... & \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & ... & g(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L} \quad (11)$$

Here the output vector is $\mathbf{T}_1 = [t_1, t_2, ..., t_N]^T$, and the output weight is:

$$\boldsymbol{\beta}_1 = \mathbf{H}_1{}^+\mathbf{T}_1 \quad (12)$$

where

$$\mathbf{H}_1{}^+ = (\mathbf{H}_1{}^T\mathbf{H}_1)^{-1}\mathbf{H}_1{}^T \quad (13)$$

Let $\mathbf{M}_1 = (\mathbf{H}_1{}^T\mathbf{H}_1)^{-1}$, equation (13) can be rewritten as $\mathbf{H}_1{}^+ = \mathbf{M}_1\mathbf{H}_1{}^T$.

3.2. Online stage

It is important to reduce the data imbalance phenomenon in online stage. Moreover, traditional OS-ELM treats old and new arrived training samples equally, and once new samples arrived, the output weights will immediately updated. This mechanism is so rigid that the learning machine cannot exclude redundant samples and unnecessary computational cost.

To solve the problems above, we employ leave-one-out(LOO) cross validation to choose more valuable samples. LOO cross validation is almost unbiased, and widely used to evaluate the generalization performance of prediction model. However, the traditional LOO cross validation is computationally expensive. In this section, we will derive a new fast LOO error estimation method. Meanwhile, we introduce a add-delete mechanism to update output weights in order to highlight the value of new arrived sample and keep the model simple.

3.2.1. Add new sample

Add the new arrived sample $(\mathbf{x}_{N+1}, t_{N+1})$ into training set. The output vector becomes $\mathbf{T}_2 = [t_1, t_2, ..., t_N, t_{N+1}]^T = [\mathbf{T}_1{}^T \ t_{N+1}]^T$, and the hidden layer matrix becomes $\mathbf{H}_2 = [\mathbf{h}_1{}^T, \mathbf{h}_2{}^T, ..., \mathbf{h}_N{}^T, \mathbf{h}_{N+1}{}^T]^T = [\mathbf{H}_1{}^T \ \mathbf{h}_{N+1}{}^T]^T$. Then we have:

$$\mathbf{H}_2{}^+ = (\mathbf{H}_2{}^T\mathbf{H}_2)^{-1}\mathbf{H}_2{}^T \tag{14}$$

Let $\mathbf{M}_2 = (\mathbf{H}_2{}^T\mathbf{H}_2)^{-1}$, then equation (14) becomes:

$$\mathbf{H}_2{}^+ = \mathbf{M}_2\mathbf{H}_2{}^T \tag{15}$$

Because

$$\mathbf{H}_2{}^T\mathbf{H}_2 = [\mathbf{H}_1{}^T \ \mathbf{h}_{N+1}{}^T][\mathbf{H}_1{}^T \ \mathbf{h}_{N+1}{}^T]^T = \mathbf{H}_1{}^T\mathbf{H}_1 + \mathbf{h}_{N+1}{}^T\mathbf{h}_{N+1} \tag{16}$$

we have

$$\mathbf{M}_2{}^{-1} = \mathbf{M}_1{}^{-1} + \mathbf{h}_{N+1}{}^T\mathbf{h}_{N+1} \tag{17}$$

Calculate the inversion of equation (17), and according to Sherman-Morrison matrix inversion lemma, we have:

$$\mathbf{M}_2 = (\mathbf{M}_1{}^{-1} + \mathbf{h}_{N+1}{}^T\mathbf{h}_{N+1})^{-1} = \mathbf{M}_1 - \frac{\mathbf{M}_1\mathbf{h}_{N+1}{}^T\mathbf{h}_{N+1}\mathbf{M}_1}{1 + \mathbf{h}_{N+1}\mathbf{M}_1\mathbf{h}_{N+1}{}^T} \tag{18}$$

As shown in equation (18), $\mathbf{M}_2$ can be calculated based on $\mathbf{M}_1$, which reduces computational cost largely. Then we have $\mathbf{H}_2{}^+$ by substituting equation (18) into equation (15).

3.2.2. *Delete old sample*

After adding new sample $(\mathbf{x}_{N+1}, t_{N+1})$, to reduce the negative effect of old sample and make the model simple, we need to exclude the oldest sample $(\mathbf{x}_1, t_1)$. After excluding $(\mathbf{x}_1, t_1)$, the output vector becomes $\mathbf{T}_3 = [t_2, t_3, ..., t_N, t_{N+1}]^T$, and the hidden layer matrix becomes $\mathbf{H}_3 = [\mathbf{h}_2{}^T, \mathbf{h}_3{}^T, ..., \mathbf{h}_N{}^T, \mathbf{h}_{N+1}{}^T]^T$. We have:

$$\mathbf{H}_3{}^+ = (\mathbf{H}_3{}^T\mathbf{H}_3)^{-1}\mathbf{H}_3{}^T \tag{19}$$

Let $\mathbf{M}_3 = (\mathbf{H}_3{}^T\mathbf{H}_3)^{-1}$, then we have:

$$\mathbf{H}_3{}^+ = \mathbf{M}_3\mathbf{H}_3{}^T \tag{20}$$

Because

$$\mathbf{H}_2{}^T\mathbf{H}_2 = [\mathbf{h}_1{}^T\ \mathbf{H}_3{}^T][\mathbf{h}_1{}^T\ \mathbf{H}_3{}^T]^T = \mathbf{h}_1{}^T\mathbf{h}_1 + \mathbf{H}_3{}^T\mathbf{H}_3 \tag{21}$$

we have:

$$\mathbf{M}_3{}^{-1} = \mathbf{M}_2{}^{-1} - \mathbf{h}_1{}^T\mathbf{h}_1 \tag{22}$$

We use Sherman-Morrison matrix inversion lemma again, and have:

$$\mathbf{M}_3 = (\mathbf{M}_2{}^{-1} - \mathbf{h}_1{}^T\mathbf{h}_1)^{-1} = \mathbf{M}_2 + \frac{\mathbf{M}_2\mathbf{h}_1{}^T\mathbf{h}_1\mathbf{M}_2}{1 - \mathbf{h}_1\mathbf{M}_2\mathbf{h}_1{}^T} \tag{23}$$

Similar to equation (18), $\mathbf{M}_3$ can be obtained directly from $\mathbf{M}_2$. Then we have $\mathbf{H}_3{}^+$ by substituting equation (23) into equation (20).

3.2.3. *Fast Online LOO error estimation*

In [29], Liu et al. derived a fast LOO error estimation of ELM. The generalization error in $i-\mathrm{th}$ LOO iteration can be expressed as:

$$r_i = t_i - f_i(\mathbf{x}_i) = \frac{t_i - \mathbf{H}_{\mathbf{x}_i}\mathbf{H}^+\mathbf{T}}{1 - (\mathbf{H}_{\mathbf{x}_i}\mathbf{H}^+)_i} \tag{24}$$

where $(\cdot)_i$ means the $i-\mathrm{th}$ element, **H** is hidden layer matrix, and $\mathbf{H}_{\mathbf{x}_i}$ means the row about the sample $\mathbf{x}_i$ in **H**.

However, this LOO estimation cannot be directly applied to online sequential scenario. We observe in equation (24), when adding a sample and delete another sample, the only affected element is **H**. So, we simply set $\mathbf{H}^+ = \mathbf{H}_3{}^+$ which is calculated from equation (20), and the generalization error in $i-\mathrm{th}$ LOO iteration can be expressed as:

$$r_i = t_i - f_i(\mathbf{x}_i) = \frac{t_i - \mathbf{H}\mathbf{x}_i\mathbf{H}^+\mathbf{T}}{1-(\mathbf{H}\mathbf{x}_i\mathbf{H}^+)_i} \tag{25}$$

After introducing PRESS statistic, we have the LOO error estimation:

$$LOO = \frac{1}{N}\sum_{i=1}^{N} {r_i}^2 \tag{26}$$

Equation (26) can be applied effectively to solve online sequential data imbalance problem. We can calculate LOO values from equation (26) for each new arrived sample $(\mathbf{x}_{N+1}, t_{N+1})$, whatever majority class or virtual minority class samples, and make a decision of retaining this sample or not. If LOO value decreases, it is obvious that this sample can improve the generalization ability of current model, then we use equation (27) from add-delete mechanism to update output weights:

$$\boldsymbol{\beta} = \mathbf{H}_3{}^+\mathbf{T}_3 \tag{27}$$

Otherwise, the new arrived sample is proved as invalid sample and should be rejected while output weights remain unchanged.

3.3. Algorithm

Following the above idea, we propose a new OS-ELM algorithm based on principal curve and fast LOO cross-validation. We call it as PL-OSELM. Based on the series of analysis, this algorithm can be summarized as follows:

Step 1. Initial offline stage

(1) Use principal curve to get initial training set $N_0 = \{(\mathbf{x}_i, t_i)|i = 1, 2, ..., N_0\}$, as stated in section 3.1. Use equation (28)to calculate the network weights based on total N_0 training samples:

$$\begin{cases} \boldsymbol{\beta}_0 = \mathbf{H}_0{}^+\mathbf{T}_0 \\ \mathbf{H}_0{}^+ = \mathbf{M}_0\mathbf{H}_0{}^T \\ \mathbf{M}_0 = (\mathbf{H}_0{}^T\mathbf{H}_0)^{-1} \end{cases} \tag{28}$$

where $\mathbf{H}_0$ is hidden layer matrix based on new training set, $\mathbf{T}_0 = [t_1, t_2, ..., t_{N_0}]$

(2) Set $k = 0$.

Step 2. Online learning stage

(1) Set the $(k+1)$ $-$th arrived sample as $(\mathbf{x}_{N_0+k+1}, t_{N_0+k+1})$. Now the output vector becomes $\mathbf{T}_Z = [\mathbf{T}_k{}^T \; t_{N_0+k+1}]^T$, and the hidden layer matrix becomes $\mathbf{H}_Z = [\mathbf{H}_k{}^T \; \mathbf{h}_{N_0+k+1}{}^T]^T$. Calculate $\mathbf{H}_Z{}^+$ by equation (29):

$$\mathbf{H}_Z{}^+ = \mathbf{M}_Z\mathbf{H}_Z{}^T \tag{29}$$

where

$$\mathbf{M}_Z = (\mathbf{M}_k{}^{-1} + \mathbf{h}_{N_0+k+1}{}^T\mathbf{h}_{N_0+k+1})^{-1} = \mathbf{M}_k - \frac{\mathbf{M}_k\mathbf{h}_{N_0+k+1}{}^T\mathbf{h}_{N_0+k+1}\mathbf{M}_k}{1 + \mathbf{h}_{N_0+k+1}\mathbf{M}_k\mathbf{h}_{N_0+k+1}{}^T}$$

. (2) Delete the oldest sample $(\mathbf{x}_{k+1}, t_{k+1})$. Now the output vector becomes: $\mathbf{T}_S = [t_{k+2}, t_{k+3}, \cdots, t_{N_0+k+1}]^T$, and the hidden layer matrix becomes $\mathbf{H}_S =$

$[\mathbf{h}_{k+2}, \mathbf{h}_{k+3}, ..., \mathbf{h}_{N_0+k+1}]^T$. Calculate $\mathbf{H}_S{}^+$ by equation (30):

$$\mathbf{H}_S{}^+ = \mathbf{M}_S \mathbf{H}_S{}^T \tag{30}$$

where

$$\mathbf{M}_S = (\mathbf{M}_S{}^{-1} - \mathbf{h}_{k+1}{}^T \mathbf{h}_{k+1})^{-1} = \mathbf{M}_Z + \frac{\mathbf{M}_Z \mathbf{h}_{k+1}{}^T \mathbf{h}_{k+1} \mathbf{M}_Z}{1 - \mathbf{h}_{k+1} \mathbf{M}_Z \mathbf{h}_{k+1}{}^T}$$

(3) Let $\mathbf{H}_{k+1} = \mathbf{H}_S$, $\mathbf{T}_{k+1} = \mathbf{T}_S$, $\mathbf{M}_{k+1} = \mathbf{M}_S$. If $(\mathbf{x}_{N_0+k+1}, t_{N_0+k+1})$ belongs to minority class, generate virtual minority class sample using the principal curve-based method in section 3.1. If $(\mathbf{x}_{N_0+k+1}, t_{N_0+k+1})$ belongs to majority class, the training set remain unchanged.

(4) Calculate LOO error using equation (31):

$$\begin{cases} r_i = t_i - f_i(\mathbf{x}_i) = \frac{t_i - \mathbf{H}\mathbf{x}_i \mathbf{H}_{k+1}{}^+ \mathbf{T}}{1 - (\mathbf{H}\mathbf{x}_i \mathbf{H}_{k+1}{}^+)_i} \\ LOO = \frac{1}{N} \sum\limits_{i=1}^{N} r_i{}^2 \end{cases} \tag{31}$$

where $i = 1, 2, ...N_0$. If the LOO value decreases, update output weights using equation (32):

$$\boldsymbol{\beta}_{k+1} = \mathbf{H}_{k+1}{}^+ \mathbf{T}_{k+1} \tag{32}$$

Otherwise, exclude sample $(\mathbf{x}_{N_0+k+1}, t_{N_0+k+1})$ with no updating weights.

(5) Set $k = k + 1$, and go to Step 2.

3.4. Theoretical analysis

According to the discussion in Section 3.2, this paper introduces LOO error estimate to select valuable majority samples in online stage. For the majority samples which reach sequentially, we can find that these samples are more valuable for model amendment if their LOO error estimates reduce. Conversely, the samples with greater value of LOO error estimate will make little sense to update online model, even produce redundant or negative information. For the latter case, the samples should be removed. In order to testify theoretically the rationality of using LOO error estimate to select valuable samples, we give the upper bound of information loss for majority class under-sampling in online stage from the perspective of information entropy.

Denote by r_i the LOO error of $i-\mathrm{th}$ sequential learning stage. Then total set of LOO error for $(i{+}1)\ -\mathrm{th}$ sequential learning stage is $\Psi = \{r_i | i = 1, 2, ..., k+1\}$. For convenience, we introduce the following definition:

Definition 2 (Sample Weight). *For* $\Psi = \{r_i | i = 1, 2, ..., k+1\}$*, the sample weight of* $i-\mathrm{th}$ *sequential learning stage is:*

$$w_i = 1 - \frac{r_i}{\sum\limits_{j=1}^{k+1} r_j}$$

According to Definition 1, the lower the LOO error estimate r_i is, the greater the sample weight of $i-\mathrm{th}$ sequential learning stage as well as the effect on model amendment is.

Suppose that the set of LOO error estimate for the reserved *m* majority samples is Ψ_1. Then in online stage, the set of LOO error estimate for the abandoned majority samples is $\Psi_2 = \Psi - \Psi_1 = \{r_{q_i} | i = 1, 2, ..., k+1-m\}$, where q_i is the index of r_{q_i} in Ψ. Obviously, the whole sample weight of Ψ_2 is $\sum\limits_{i=1}^{k+1-m} w_{q_i} = \sum\limits_{i=1}^{k+1-m} (1 - \frac{r_{q_i}}{\sum\limits_{j=1}^{k+1} r_j})$. Because $\sum\limits_{j=1}^{k+1} r_j$ is constant, we set $\Delta = \sum\limits_{j=1}^{k+1} r_j$, then the whole sample weight of Ψ_2 is $\sum\limits_{i=1}^{k+1-m} (1 - \frac{r_{q_i}}{\Delta}) = (k+1-m) - \frac{1}{\Delta} \sum\limits_{i=1}^{k+1-m} r_{q_i}$. Here we have the following theorem.

Theorem 1. *Let $H(\Psi_2)$ denote the information loss for majority class undersampling in online stage, then $H(\Psi_2)$ can be bounded by:*

$$H(\Psi_2) \leq ((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})\log((k+1-m)/((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i}))$$

And this bound is only relevant to the sum of LOO error estimate of Ψ_2.

PROOF. According to the definition of entropy, we have $H(\Psi_2) = -\sum_{j=1}^{k+1-m} w_{q_j}\log w_{q_j}$. According to the principle of maximum entropy, $H(\Psi)$ reaches the maximum if and only if each w_{q_j} holds the same value $((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})/(k+1-m)$. We have:

$$\begin{aligned} H(\Psi_2) &= -\sum_{q_j=1}^{k+1-m} w_{q_j}\log w_{q_j} \\ &\leq -\sum_{q_j=1}^{k+1-m} \frac{((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})}{(k+1-m)}\log(\frac{((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})}{(k+1-m)}) \\ &= \sum_{q_j=1}^{k+1-m} \frac{((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})}{(k+1-m)}\log(\frac{(k+1-m)}{((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})}) \\ &= ((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})\log((k+1-m)/((k+1-m)-\frac{1}{\Delta}\sum_{i=1}^{k+1-m} r_{q_i})) \end{aligned} \tag{33}$$

From equation (33), the upper bound of $H(\Psi_2)$ is only relevant to $\sum_{i=1}^{k+1-m} r_{q_i}$, i.e., the sum of LOO error estimate of Ψ_2. The greater $\sum_{i=1}^{k+1-m} r_{q_i}$ is, the lower the upper bound is.

Theorem 1 proves theoretically the rationality of the method using LOO error estimate to select valuable samples. We have a high confidence that the information loss tend to be infinitesimal if the sum of LOO error estimate of Ψ_2 tends to be infinite, i.e., $\sum_{i=1}^{k+1-m} r_{q_i} \to \infty$, which demonstrates that abandoning these samples has negligible effect on the whole online model.

4. Experimental Results

In this section, we run experiments to test the proposed algorithm. Our goal is to demonstrate that the proposed algorithm can efficiently improve the generalization performance of OS-ELM in data imbalance problem. For comparison, we choose four baselines. The first is classical ELM[21]. The second is OS-ELM[14]. The third is meta-cognitive OS-ELM(MC-OSELM) proposed in [9][30]. In this approach, minority class samples are duplicated directly in online phase. And the forth is the improved OS-ELM based on principal curve and SMOTE(PCI-OSELM), which utilizes pricipal curve in offline stage to reduce the gap between majority and minority classes and chooses the suitable samples in online stage according to the degree of membership[31]. Moreover, the proposed OS-ELM algorithm based on principal curve and fast LOO cross-validation is named PL-OSELM. It is worth to say that the algorithms of PCI-OSELM and PL-OSELM both employ the principal curve to extract the data distribution characteristics and then choose more valuable samples in the process of sample reconstruction, which is effective to solve the online sequential imbalance classification. Strictly speaking, this paper is the improved version of [31] and gives theoretical analysis which is not found in [31].

For completeness, we examine two types of data sets. We start with four UCI classification data. To simulate the imbalance data, we randomly choose a small number of one class samples as minority class, keeping the ratio between majority class and minority class more than 5:1. And then we use OS-ELM to establish sequential model via importing data chunk by chunk. We further present results on a real-world data set, i.e., air pollutants forecasting in Macau[9]. The goal is to test the generalization performance on online sequential data imbalance problem as well as the running speed. In each experiment, all results are the mean of 100 trials. RBF activation function is used in each algorithm. Each variable is linearly rescaled. A method with higher classification accuracy is better.

4.1. UCI dataset

In this section, we introduce four UCI classification datasets[32]. Some statistics of these four data sets are listed in Table 1. These data sets are divided into offline training, online training and test set. Note that only ELM does not need to set online set.

It is worthy noting that, in Pima/Banknote/Blood, minority class is all set 1 class while majority class is 0 class. As Abalone has three classes, we choose F

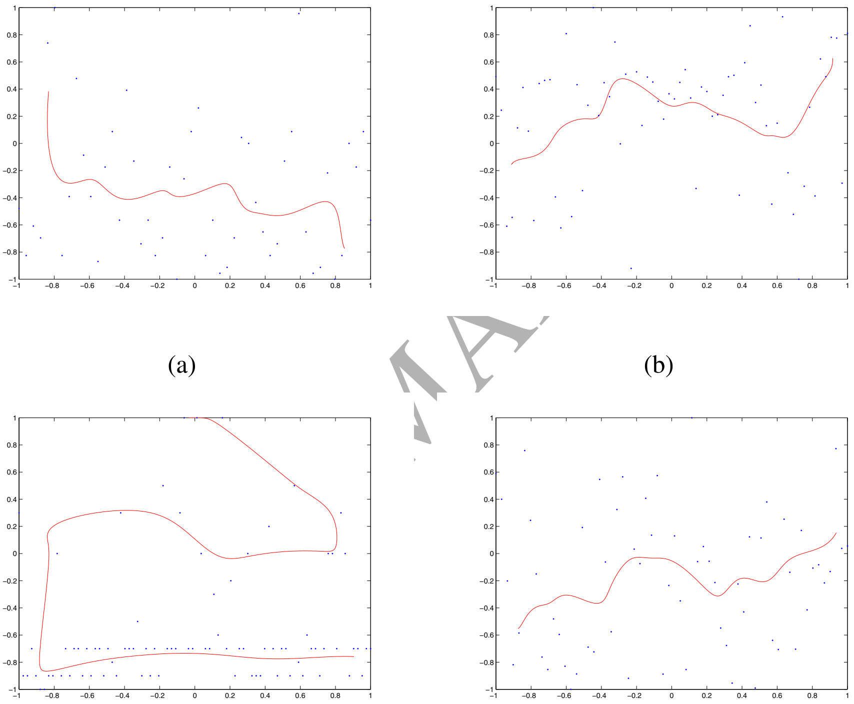
Table 1: Specifications of four UCI classification data sets

Name	Feature	Training Data				Test Data	
		Offline		Online			
		Majority	Minority	Majority	Minority	Majority	Minority
Pima	8	250	50	167	33	83	17
Banknote	4	334	66	247	53	181	31
Blood	4	266	84	187	63	117	31
Abalone	8	638	62	361	39	308	29

and I class as majority and minority class, respectively.

First, we examine whether the principal curve can reflect the shape of data set. We use K-principal curve[23] to plot the middle shape in minority class of four data sets, as shown in Figure 1.

Obviously, even in the small sample size, principal curve also could pass



(a)

(b)

(c)

(d)

Figure 1: Principal curve in minority class of data sets of (a) Pima, (b) Banknote, (c) Blood and (d) Abalone

through the middle part of data set, reflecting the trend or distribution of data. Especially in Fig.1(c), principal curve truly reflects the data distribution. Note that in minority class, due to the small quantity, each sample should be considered equally and should not be regarded as outlier or noisy sample. The tendency in Fig.1(d) is not very clear. It may be caused by small sample size.

In the offline stage, we need to re-generate samples to reduce the gap between majority and minority classes. We use principal curve-based method described in section 3.1 to produce some virtual minority class samples and delete some redundant majority class samples. The values of threshold in equation (10) are listed in Table 2.

After applying principal curve-based method, we have new training set whose sample size is listed in Table 3. Obviously, at each data set, we have balanced the sample size of two classes.

Table 2: Values of threshold used in section 3.1

Name	Minority		Majority	
	δ_x	δ_y	δ_x	δ_y
Pima	0.02	0.05	0.02	0.05
Banknote	0.02	0.05	0.002	0.005
Blood	0.001	0.004	0.01	0.01
Abalone	0.05	0.05	0.002	0.001

Second, we report the numerical results on four data sets. We check the running time, classification accuracy on majority/minority/both classes. And we most care about the accuracy of minority class. The mean results of 100 trials on four data sets are listed in Table 4-7. Here the numbers of neurons are 30, 17, 25, 45, respectively.

Table 3: Sample size of new training set

Name	Sample	
	Majority	Minority
Pima	169	168
Banknote	224	213
Blood	178	167
Abalone	366	352

Table 4: Classification accuracy of four algorithms on Pima data set

	PL-OSELM	OSELM	ELM	MC-OSELM	PCI-OSELM
Training time(s)	9.5535	0.4384	0.0437	0.0860	0.4682
Test time(s)	0.0187	0.0125	0.0012	0.0908	0.0195
Minority training accuracy(%)	69.18	31.93	30.96	68.21	82.31
Majority training accuracy(%)	88.99	97.58	97.82	84.31	89.74
Minority test accuracy(%)	61.76	27.65	25.29	47.06	62.11
Majority test accuracy(%)	91.93	96.27	96.87	87.11	88.46
Whole training accuracy(%)	81.21	86.68	86.72	78.88	88.36
Whole test accuracy(%)	84.30	84.70	84.60	82.80	84.31

Table 5: Classification accuracy of four algorithms on Banknote data set

	PL-OSELM	OSELM	ELM	MC-OSELM	PCI-OSELM
Training time(s)	14.5393	0.2746	0.0125	0.0523	0.3124
Test time(s)	0.0003	0.0006	0.0062	0.0544	0.0043
Minority training accuracy(%)	98.95	97.73	96.05	96.88	98.41
Majority training accuracy(%)	97.69	99.28	98.49	95.69	98.52
Minority test accuracy(%)	98.71	95.81	95.16	93.87	99.13
Majority test accuracy(%)	98.84	99.67	99.28	97.57	98.94
Whole training accuracy(%)	97.39	99.01	98.07	96.87	99.46
Whole test accuracy(%)	98.11	99.10	98.68	97.74	99.11

Table 6: Classification accuracy of four algorithms on Blood data set

	PL-OSELM	OSELM	ELM	MC-OSELM	PCI-OSELM
Training time(s)	10.1822	0.4649	0.0187	0.0927	0.4162
Test time(s)	0.0187	0.0062	0.0062	0.0967	0.0248
Minority training accuracy(%)	58.10	31.36	32.45	51.26	58.26
Majority training accuracy(%)	93.81	94.99	95.01	79.40	82.31
Minority test accuracy(%)	49.03	27.42	27.10	32.58	45.07
Majority test accuracy(%)	89.32	95.04	94.53	83.08	86.37
Whole training accuracy(%)	77.71	79.75	79.68	71.38	69.32
Whole test accuracy(%)	75.95	80.88	80.41	77.43	73.59

Table 7: Classification accuracy of four algorithms on Abalone data set

	PL-OSELM	OSELM	ELM	MC-OSELM	PCI-OSELM
Training time(s)	64.0727	1.1450	0.0655	0.2350	1.5983
Test time(s)	0.0187	0.0031	0.0047	0.2487	0.0163
Minority training accuracy(%)	94.68	69.11	67.82	95.39	93.36
Majority training accuracy(%)	97.52	98.03	98.01	94.21	97.28
Minority test accuracy(%)	88.82	87.06	86.47	88.24	86.47
Majority test accuracy(%)	92.51	98.69	98.52	93.11	92.31
Whole training accuracy(%)	96.53	95.37	95.24	94.58	95.48
Whole test accuracy(%)	92.15	97.70	97.50	92.75	92.31

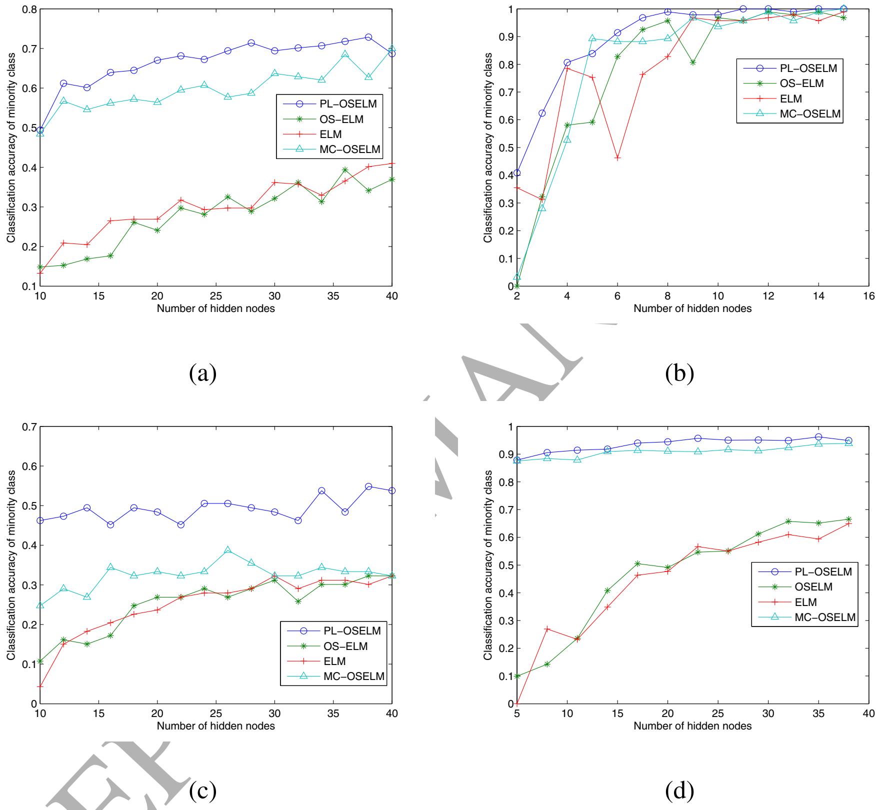
Obviously, although PL-OSELM cannot get the highest whole accuracy, it gets relatively higher test accuracy in minority class on all four data sets compared with the other algorithms on four datasets, which shows the good performance of our proposed method. Morevoer, PL-OSELM has higher accuracy on minority class than PCI-OSELM on the datasets Banknote and Abalone. It's worthy noting that PCI-OSELM has less training time than PL-OSELM. The reason is that PCI-OSELM chooses more valuable samples according to the degree of membership based on the principal curve from two stage of offline stage and online stage. But there is little difference between PCI-OSELM and PL-OSELM on the accuracy of minority class, that is to say the two algorithms both can solve the problem of online sequential imbalance data classification effectively. So we no longer consider PCI-OSELM in the following analysis and focus on the comparison of PL-OSELM, OS-ELM, ELM and MC-OSELM.

It is not hard to find that the proposed method gets the highest accuracy on minority class than OS-ELM, ELM and MC-OSELM. Specially speaking, PL-

OSELM also gets highest training accuracy on Abalone data set than the other three methods. The results can demonstrate the effectiveness of PL-OSELM in solving online sequential data imbalance problem. Note that, although MC-OSELM gets success in forecasting time-series data of air pollutants, it tends to get lower test accuracy in this experiment. The key reason is that direct prior duplication may be effective for time-series data, but it cannot explore the real data distribution accurately, and tends to cause over-fitting. However, the corresponding cost of high accuracy is long running time. The computational time in PL-OSELM is much larger than other three algorithms. The part of most time consuming is principal curve. However, it is lucky that the long running time is consumed in training phase, rather than test phase. On the contrary, the difference of test time of four algorithms is not very significant, diversed in the same order of magnitude. Hence we think PL-OSELM is enough practical.

The number of hidden neurons is the only adjustable parameter. We also check the effect of it. Just as the above description, here we only compare the performance of PL-OSELM, OS-ELM, ELM and MC-OSELM. Figure 2 illustrates the variation tendency of classification accuracy on minority class with different numbers of hidden neurons. To reduce the randomness of ELM, we also report the mean results of 100 experimental trials. Even so, some results in Fig.2 are still fluctuate slightly.

From Fig.2, PL-OSELM generally get most stable prediction on four data sets. And PL-OSELM tends to get highest accuracy, which keeps pace with the numerical results listed above. The stable performance is likely to be produced by LOO cross validation. On the contrary, we observe that MC-OSELM is more unstable, which is also reflected in numerical results. The results remind us that



(a)

(b)

(c)

(d)

Figure 2: Classification accuracy on minority class with different number of hidden neurons for data set (a) Pima, (b) Banknote, (c) Blood and (d) Abalone

improper imbalance strategy maybe deteriorate the generalization performance on online sequential imbalance data. OS-ELM plays most similar performance with PL-OSELM, which demonstrates the prominent property of it.

4.2. Air pollutants forecasting data

As shown in [9], air pollutants data is a typical time-series imbalance data. These data were collected from three land zones, and contain several features such as suspended particulate matters($\mathtt{PM}_{10}$), nitrogen dioxide($\mathtt{NO}_2$), sulfur dioxide($\mathtt{SO}_2$), etc. We choose the values of $\mathtt{PM}_{10}$, $\mathtt{NO}_2$, $\mathtt{SO}_2$ and $\mathtt{O}_3$ to construct input sample, as

$$\mathbf{x} = (\mathrm{d}\,(\mathrm{PM}_{10})\,, \mathrm{d}\,(\mathrm{SO}_2)\,, \mathrm{d}\,(\mathrm{NO}_2)\,, \mathrm{d}\,(\mathrm{O}_3))$$

The output sample is the value of next day's $\mathtt{PM}_{10}$, i.e., $t = \mathrm{d} + 1\,(\mathrm{PM}_{10})$.

We use the data collected from 2010 to 2013 year to conduct experiment. Specifically, the data in 2010 are used for initial offline training, the data in 2011 are used for online training, the data in 2012 are used for validation, and the data in 2013 are used for test.

[9] classified the air quality into three level. In this section we simply set two classes as pollution and without pollution, using the following formulation:

$$\lambda = \begin{cases} -1 & \mathrm{d} + 1\,(\mathrm{PM}_{10}) > 100 \\ 1 & \mathrm{d} + 1\,(\mathrm{PM}_{10}) \leq 100 \end{cases} \tag{34}$$

Using equation (34), we provide the description of data set, as in Table 8.

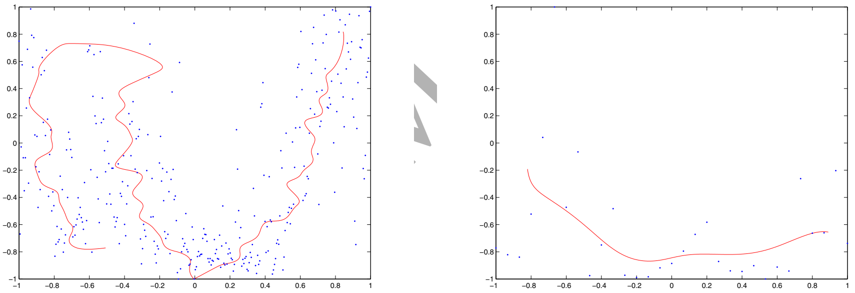
It is clear that the data of every year is severely imbalanced. Meanwhile, it is still time series. Hence we use these data to examine the performance of PL-OSELM. First, we use principal curve-based method described in section 3.1 to expand minority sample and cut down redundant majority sample in initial offline

Table 8: Description of original data set from 2010 to 2013

	2010	2011	2012	2013
Minority sample	31	30	29	51
Majority sample	334	335	337	313
Ratio of minority	8.49%	8.22%	7.92%	14.01%
Ratio of majority	91.51%	91.78%	92.08%	85.99%

stage. We use k-principal curve[24] to plot the principal curve of majority and minority data in 2010, respectively, as shown in Fig.3.

Obviously, the principal curves have revealed the variation tendency. Based on the obtained principal curve, we exclude the redundant majority samples, and generate virtual minority samples by adding white noise whose intensity is set 10dB. With the threshold $\delta_x = 0.002$, $\delta_y = 0.005$, we get the new training set described in Table 9.



(a)

(b)

Figure 3: Principal curve of $\texttt{PM}_{10}$ data in 2010 with (a) majority class and (b) minority class

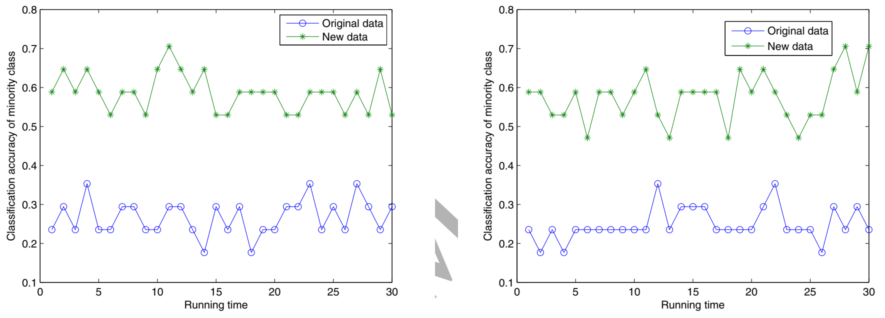
After preprocessing using principal curve, the ratio between majority and minority classes becomes almost 1:1. The data imbalance problem has been released

Table 9: Description of new data set based on principal curve in 2010

	Majority sample	Minority sample	Ratio of majority	Ratio of minority
Orignal data set	334	31	91.51%	8.49%
New data set	154	131	**54.03%**	**45.97%**

to some extents. To examine the quality of new data set, we run ELM and OS-ELM 100 times, and the accuracy results are illustrated in Fig. 4.

From Fig.4, even ELM and OS-ELM both improve the generalization performance largely, which demonstrates the rationality of the new data set based on principal curve. Table 10 provides the comparative results of five algorithms. Same to the experiment above, we mainly focus on the accuracy on minority class. Here the hidden neurons are set 50.



(a)

(b)

Figure 4: Comparative accuracy on minority class of original and new data sets using (a) ELM and (b) OS-ELM

As the results on UCI datasets, PCI-OSELM obtains more classification accuracy on minority samples than PL-OSELM but there is not much difference. To highlight the proposed method, here we only compared PL-OSELM with the other

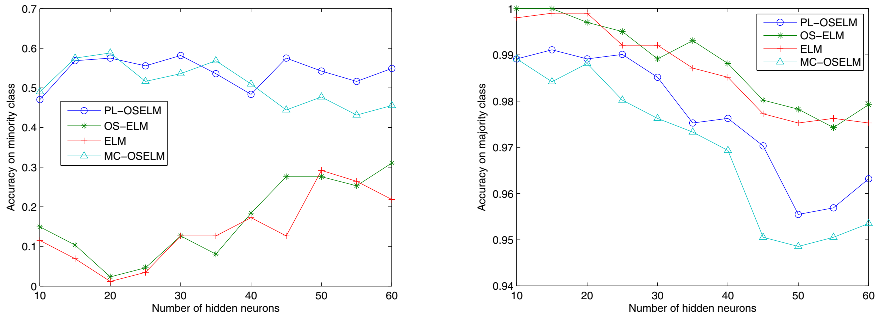
Table 10: Classification accuracy on minority class of four algorithms on air pollutant data set

	PLOSELM	OSELM	ELM	MCOSELM	PCI-OSELM
Training time(s)	14.7473	0.4940	0.0468	0.2979	0.5873
Test time(s)	0.0052	0.0052	0.0260	0.3093	0.0062
Minority training accuracy(%)	80.31	20.22	21.31	79.22	91.46
Majority training accuracy(%)	93.52	99.30	99.05	95.60	88.34
Minority test accuracy(%)	53.44	21.84	18.49	50.34	62.47
Majority test accuracy(%)	96.34	99.80	99.31	97.73	86.73
Whole training accuracy(%)	88.04	92.69	92.56	91.65	91.64
Whole test accuracy(%)	89.71	92.26	92.08	90.80	84.57

three algorithms. From Table 10, although PL-OSELM get relative low accuracy on majority class and whole data, but it still obtains the highest training and test accuracy on minority class, which is precisely our algorithm's value. Similar to the results in [9], MC-OSELM also gets much higher accuracy on minority class than OS-ELM and ELM, which also demonstrates the necessity of solving online sequential data imbalance problem. OS-ELM gets highest accuracy on whole data, but as stated in section Introduction, this value is meaningless because the correct prediction gathers nearly in majority class.

We also examine the effect of hidden neurons. Figure 5 shows the accuracy of four algorithms with different number of hidden neurons.

From Fig.5, PL-OSELM and MC-OSELM both get satisfying results on minority class, with aggravated performance on majority class, which demonstrate their good ability to handle online sequential imbalance data. In comparison, PL-OSELM behaves in more stable way. Although OS-ELM and ELM get much better accuracy in Fig.5(b), the results have no reference value because almost all



(a)

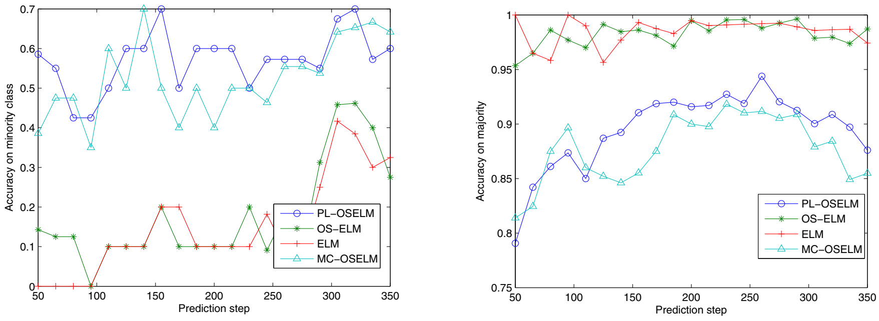
(b)

Figure 5: Classification accuracy of Macau air pollutant data with different number of hidden neurons on (a) minority class and (b) majority class

minority class samples are incorrectly predicted.

Prediction step determines the number of test samples substituted into the prediction model for forecasting at one time. This index reflects the generalization ability of algorithm especially for time series data. We report the accuracy of four algorithms with different prediction step, as in Figure 6. As mentioned above, the slight fluctuation of results are caused by the randomness of ELM.

From Fig.6, with large prediction step, PL-OSELM can still get stable and high accuracy on minority class. The results demonstrate PL-OSELM can utilize information of online samples effectively, and establish a valid prediction model. MC-OSELM obtains similar results, even though a relative lower accuracy. ELM and OS-ELM can not learn enough information of minority class, then the accuracy in Fig.5(a) continuous to deteriorate. Contrary results are illustrated in Fig.5(b), where the accuracies on majority class of PL-OSELM and MC-OSELM both fall down with increasing step. But ELM and OS-ELM get stable prediction, in which the prediction on minority class is always almost incorrect.



(a) (b)

Figure 6: Accuracy of four algorithms with different prediction step on (a) minority class and (b) majority class

5. Conclusion and Future work

In this paper, a kind of data imbalance problem, online sequential data imbalance problem, is addressed. The key idea is to reduce the data imbalance phenomenon in online sequential learning process. Data-based strategy and algorithm-based strategy are integrated into a two-stage hybrid strategy. To realize this strategy, this paper utilizes principal curve in offline stage to reduce the gap between majority and minority classes, and proposes a fast leave-one-out cross-validation error estimation to reduce data imbalance in online stage. Following this strategy, a new OS-ELM algorithm based on principal curve and fast leave-one-out error estimation is proposed. This algorithm can adjust training samples in more efficient way, and update output weights automatically with ”add-delete” mechanism. The experimental results on UCI data sets and a real-world data set demonstrate the effectiveness of the proposed approach.

It is a matter of choosing more efficient method for calculating leave-one-out error. Many current methods including the one in this paper are still time consuming. A more simple computational method can increase efficiency. Another problem is how to extend the proposed algorithm to multi-class classification, which should be achieved by introducing the corresponding background knowledge. We also observe the data sets used in this paper are not concerned with sparsity. Therefore, how to tackle sparsity in very high-dimensional data and evaluate the effect of ultra sparsity on imbalance is a new problem, and will be studied in our future research.

Acknowledgement

We wish to thank the author C.M.Vong of [9] for useful discussion and instruction. This work was supported by the National Natural Science Foundation of China(No. U1204609), Postdoctoral Science Foundation of China(No. 2014M550508), the funding scheme of University Science & Technology Innovation in Henan Province(15HASTIT022), the funding scheme of University Young Core Instructor in Henan Province(2014GGJS-046) and the foundation of Henan Normal University for Excellent Young Teachers(No.14YQ007).

References

[1] Murphey, Y.L., Guo, H., & Feldkamp, L.A. (2004). Neural learning from unbalanced data. *Appl Intell*, *21*, 117-128.

[2] Lu, W.-Z., & Wang, D. (2008). Ground-level ozone prediction by support vector machine approach with a cost-sensitive classification scheme. *Sci.TotalEnviron*, *395*, 109-116.

[3] Blagus, R., & Lusa, L. (2010). Class prediction for high-dimensional class-imbalanced data. *BMC Bioinf*, *11*, 523.

[4] Batista, G.E., Prati, R.C., & Monard, M.C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl*, *6*, 20-29.

[5] Ducange, P., Lazzerini, B., & Marcelloni, F. (2010). Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets. *SoftComput-A Fusion of Foundations*, *Methodol Appl*, *14*, 713-728.

[6] Pang, S., Zhu, L., & Chen, G. (2013). Dynamic class imbalance learning for incremental LPSVM. *Neural Networks*, *44*, 87-100.

[7] Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced datasets. *Comput.Intell*, *20*, 18-36.

[8] Tang, Y., Zhang, Y.-Q., Chawla, N.V., & Krasser, S. (2009). SVMs modeling for highly imbalanced classification. *IEEE Transactions on Systems*, *Man*, *and Cybernetics*, *Part B:Cybernetics*, *39*, 281-288.

[9] Vong, C.M., IP, W.F., Wong, P.K., & Chiu, C.C. (2014). Predicting minority class for suspended particulate matters level by extreme learning machine. *Neurocomputing*, *128*, 136-144.

[10] Krawczyk, B., Stefanowski, J., & Wozniak, M. (2014). Data stream classification and big data analytics. *Neurocomputing*, *150*, 238-239.

[11] Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2004). Extreme learning machine: A new learning scheme of feedforward neural networks. In *Proceedings of International Joint Conference on Neural Networks*: *vol. 2*.(Budapest, Hungary), 985-990.

[12] Liu, X., Gao, C., & Li, P. (2012). A comparative analysis of support vector machines and extreme learning machines. *Neural Networks*, *33*, 58-66.

[13] Deng, W., Zheng, Q., & Wang, Z. (2014). Cross-person activity recognition using reduced kernel extreme learning machine. *Neural Networks*, *53*, 1-7.

[14] Liang, N.Y., & Huang, G.-B. (2006). A fast accurate online sequential learning algorithm for feedforword networks. *IEEE Transaction on Neural Networks*, *17*, 1411-1423.

[15] Wang, X., Han, M. (2014). Online sequential extreme learning machine with kernels for nonstationary time series prediction. *Neurocomputing*, *(*145), 90-97.

[16] Zong, W., Huang, G.-B., & Chen, Y. (2013). Weighted extreme learning machine for imbalance learning. *Neurocomputing*, *101*, 229-242.

[17] Li, K., Kong, X., & Lu, Z. (2014). Boosting weighted ELM for imbalanced learning. *Neurocomputing*, *128*, 15-21.

[18] Zong, W., Huang, G.-B., & Chen, Y. (2013). Weighted extreme learning machine for imbalance learning. *Neurocomputing*, *101(3)*, 229-242.

[19] Mao, W., Wang, J., He, L., & Tian, Y. (2015). Two-stage hybrid Extreme Learning Machine for Sequential Imbalanced Data. *In Proceedings of ELM-2015, Hangzhou, China, December, 2015*.

[20] Huang, G.-B., & Babri, H.A. (1998). Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions. *IEEE Transactions on Neural Networks*, *9(1)*, 224-229.

[21] Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing*, *70*, 489-501.

[22] Huang, G.-B., Zhou, H., Ding, X., & Zhang, R. (2012). Extreme Learning Machine for Regression and Multiclass Classification. *IEEE Transactions on Systems*, *Man*, *and Cybernetics - Part B: Cybernetics*, *42(2)*, 513-529.

[23] Hastie, T. (1984). Principal curves and surfaces. *Stanford University , Department of Statistics :Technical Report 11*.

[24] Hermann, T., Meinicke, P., & Ritter, H. (2000). Principal curve sonification. In *Proceedings of International Conference on Auditory Display, Atlanda , USA*, 81-86.

[25] Deng, W., Zheng, Q., & Lian, S. (2010). Ordinal extreme learning machine. *Neurocomputing*, *74(1-3)*, 447-456.

[26] Déath, G. (1999). Principal curves: A new technique for indirect and direct gradient analysis. *Ecology*, *80(7)*, 2237-2253.

[27] Zhang, J., & Wang, J. (2003). An overview of principal curves. *Chinese Journal of Computers*, *26(2)*, 129-146.

[28] Kégl, B., Krzyzak, A., Linder, T., & Zeger, K. (2000). Learning and design of principal curves. *IEEE Trans. on Pattern Recognition and Machine Intelligence*, *22(3)*, 281-297.

[29] Liu, X., Li, P., & Gao, C. (2011). Fast leave-one-out cross-validation algorithm for extreme learning machine. *Journal of Shanghai Jiaotong University*, *45(8)*, 6-11.

[30] Chong, C.C. (2013). Online Sequential Prediction of Minority Class of Suspended Particulate Matters by Meta-Cognitive OS-ELM. *University of Macau*.

[31] Mao, W., Wang, J., & Wang, L. (2015). Online sequential classification of imbalanced data by combining extreme learning machine and improved SMOTE algorithm. *International Joint Conference on Neural Networks, IEEE*.

[32] Newman, D.J., Hettich, S., Blake, C.L., & Merz, C.J. UCI Repository of machine learning databases[http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Biography

Wentao Mao received his M.S. degree in Computer Science from Chongqing University of Posts and Telecommunications in 2006, and the Ph.D degree in engineering mechanics from Xi'an Jiaotong University, China, in 2011. He is currently working at Henan Normal University, China. His current research interests include machine learning, kernel methods and evolutionary computation. Now he have conducted and is conducting about 10 research projects such as National Natural Science Foundation of China as project principal or main researcher.

Jinwan Wang received her Bachelor degree in School of Computer and Information Engineering, Henan Normal University, China, in 2014. She is currently studying in same university for her M.S. degree. Her research interests include extreme learning machine and deep learning.

Ling He received her Bachelor degree in 2013. Now she is working for her M.S. degree in Henan Normal University, China. Her research interests include support vector machine and learning theory.

Yangyang Tian received her Bachelor degree in Henan Normal University, China, in 2015. Now she is working for her M.S. degree in Donghua University, China. Now her research interests include extreme learning machine and deep learning.