World Wide Web
DOI 10.1007/s11280-013-0230-8

Probability based voting extreme learning machine for multiclass XML documents classification

Xiangguo Zhao · Xin Bi · Baiyou Qiao

Received: 17 September 2012 / Revised: 18 April 2013 / Accepted: 3 June 2013

Abstract This paper presents a novel solution based on Extreme Learning Machine (ELM) for multiclass XML documents classification. ELM is a generalized Single-hidden Layer Feedforward Network (SLFN) with extremely fast learning capacity. An improved vector model DSVM (Distribution based Structured Vector Model) is proposed to represent XML documents with more structural information and more precise semantic information. The XML documents classifiers are conducted based on PV-ELM (Probablity based Voting ELM) with a postprocessing method ε-RCC (ε - Revoting of Confusing Classes) to refine the voting results. To evaluate the overall performance of this solution, a series of experiments are conducted on two real datasets of news feeds online. The experimental results show that DSVM represents the XML documents more effectively and PV-ELM with ε-RCC achieves a higher accuracy than original ELM algorithm for multiclass classification.

Keywords Extreme learning machine · XML representation model · Multiclass classification · Decomposition methods · Voting-ELM

1 Introduction

As a typical type of semi-structured data, XML has been widely accepted as the universal data format for information systems in various domains. Since classification

X. Zhao · X. Bi · B. Qiao
Key Laboratory of Medical Image Computing (Northeastern University), Ministry of Education, Shenyang, China

X. Zhao (✉) · X. Bi · B. Qiao
College of Information Science and Engineering, Northeastern University, Liaoning, Shenyang 110819, China
e-mail: zhaoxiangguo@mail.neu.edu.cn

is an important data mining task, the design and building of classifiers for XML documents classification solution has become one of the most important research topics.

The SLFNs based Extreme Learning Machine (ELM) is proposed in [6–8, 11, 12]. ELM, with its variants [2, 10, 16, 18, 23], achieves extremely fast learning capacity and good generalization capabilities. ELM for classification is less sensitive to user specified parameters and can be implemented easily [9]. In [21], a protein secondary structure prediction framework based on binary ELM classifiers is proposed to predict protein sequences, including a probability based combination method to combine binary prediction results and a helix postprocessing method to further improve the overall performance based on bioinformatic features. To make ELM more robust and generic, a systematic and automated pruned-ELM (P-ELM) ,which prunes irrelevant hidden layer nodes measured by probability measures, is proposed in [15] to avoid underfitting/overfitting issues caused by too few/many hidden layer nodes. Optimally Pruned ELM (OP-ELM) [17] is proposed on the basis of P-ELM for both classification and regression problems, using a leave-one-out (LOO) criterion to decide the appropriate number of neurons.

A multiclass classification problem is often decomposed into multiple binary classification problems [1, 4]. In [5], three methods, including One-against-all (OAA), One-against-one (OAO) and Directed Acyclic Graph (DAG), are compared. The experiments indicate that OAO and DAG methods are more suitable for practical classification problems. In [14], ELM is applied to multiclass classification based on both OAA and OAO. It is observed from experimental results that ELM based on OAO requires smaller number of hidden nodes than OAA and original ELM for multiclass classification. All these three methods have similar testing accuracy and training time while ELM based on OAO performs some sort of better than the others. In [20], a performance enhancement scheme for multiclass sparse data classification based on ELM is presented.Optimal number of hidden neurons can be selected by RCGA-ELM to result a better performance. An alternate and less computationally intensive approach S-ELM (sparse-ELM) is also proposed using k-fold validation. The enhancement scheme works effectively for sparse multiclass classification problems.

In this paper, a solution for multiclass classification of XML documents is proposed, which includes an XML representation model named Distribution based Structured Vector Model (DSVM), a improved ELM algorithm named Probability based Voting ELM (PV-ELM), and a postprocessing method ε-RCC. DSVM contains structural information to a great degree so that DSVM has a much better representation ability of both semantic and structural information of XML document. DSVM also improves the TFIDF calculation in VSM (Vector Space Model) [19] by taking the category distribution into account. PV-ELM improves voting-ELM (v-ELM)[22], which does not provide a flexible solution to OAA methods. PV-ELM introduces probability calculation into the training phase so that it is able to adopts both OAA and OAO methods with no limitations. The improved ε-RCC method based on Revoting of Confusing Classes (RCC) [22] handles the confusing classes discovered during the training phase more effectively.

This paper is organized as follows: In Section 2, we present the improved XML representation model DSVM. In Section 3, a brief introduction to ELM is given. PV-ELM based on OAO and OAA are proposed in Section 4. A series of experiments are conducted and the performance is evaluated in Section 5. Section 6 gives a summary of conclusion based on the theoretical analysis and experimental results.

2 XML representation model

Traditional VSM (Vector Space Model) [19], which is widely used to represent plain text documents, contains no structural information when it represents structured documents like XML. An extended model named SLVM (Structured Link Vector Model) is proposed in [13], containing both semantic and structural information of XML documents. SLVM, which extends a VSM (Vector Space Model) into a matrix, represents an XML document as an array of VSMs, each referring to an XML element.

In this section, we propose an improved DSVM (Distribution based Structured Vector Model) based on SLVM. DSVM gives consideration to both distributive and structural information by implementing a revised TFIDF calculation and distribution modifying factors.

We first introduce the revised IDF, which is defined as

$$\mathrm{IDF}_{\mathrm{ex}}(w_i, c_j) = \log\left(\frac{|D|}{\mathrm{DF}(w_i) - \mathrm{DF}_{\mathrm{ex}}(w_i, c_j)}\right) \tag{1}$$

where $\mathrm{IDF}_{\mathrm{ex}}(w_i, c_j)$ is the revised IDF value of term w_i that belongs to class c_j, $|D|$ is still the number of all the documents. $\mathrm{DF}(w_i)$ is the traditional DF value, which denotes the number of documents containing term w_i. $\mathrm{DF}(w_i) - \mathrm{DF}_{\mathrm{ex}}(w_i, c_j)$ therefore is the number of documents belonging to the classes excluding c_j and containing term w_i. So the revised IDF can express the importance of term w_i in the documents belonging to other classes more precisely, ignoring the misleading situation that a term appears not many times in other classes having a high IDF value or many times but only in its own class having a low IDF.

Traditional document representation models take few account of the interact factors within or between classes. In [3] , a new term weighting system is proposed to take Intra-class Factors, Inter-class Factors and Root Mean Square Normalization into consideration except for traditional TF and IDF value.

Two improved interact factors are applied to the XML documents representation model in this paper. Among Classes Discrimination (ACD) measures the distribution balance of a term among all the classes of the data set. Within Class Discrimination (WCD) measures the distribution balance of a term within a specific class.

We consider a data set D with m classes, D $= [c_1, c_2, \ldots, c_m]$, each class c_j contains $|c_j|$ documents, and class $c_j = [d_1, d_2, \ldots, d_k]$ and d_k is a document belonging to c_j. WCD is defined as

$$\mathrm{WCD}(w_i, c_j) = \sqrt{\frac{\sum_{k=1}^{|c_j|} [\mathrm{TF}(w_i, d_k) - \mathrm{TFC}(w_i, c_j)]^2}{|c_j|}} \tag{2}$$

where

$$\mathrm{TFC}(w_i, c_j) = \sqrt{\frac{\sum_{k=1}^{|c_j|} \mathrm{TF}(w_i, d_k)^2}{|c_j|}} \tag{3}$$

$\mathrm{TFC}(\mathrm{w_i}, \mathrm{c_j})$ is the root mean square frequency of term w_i in class c_j. ACD is defined as

$$\mathrm{ACD}(w_i, c_j) = \sqrt{\frac{\sum_{j=1}^{m} [\mathrm{TFC}(w_i, c_\mathrm{j}) - \mathrm{TFA}]^2}{m}} \tag{4}$$

where

$$\mathrm{TFA} = \sqrt{\frac{\sum_{j=1}^{m} \mathrm{TFC}(w_i, c_j)^2}{m}} \tag{5}$$

and TFA is the root mean square frequency of all the terms in all the classes.

We argue that the greater the WCD of a term is, which also means the more unbalanced it is within a class, the less distinguish and express ability it has. In contrast, the greater the ACD of a term is, the more distinguish ability of different classes it has. The interact factors are introduced into SLVM as a united weight, calculating the reciprocal of arithmetic product of WCD and ACD.

Therefore we have a model description of DSVM as follows

$$\mathbf{d\_dsvm} = \left[d\_dsvm_1, d\_dsvm_2, \ldots, d\_dsvm_n\right]^\mathrm{T} \tag{6}$$

where **d_dsvm** is the n-dimensional feature vector of an XML document, d_dsvm_i is the ith represent feature which is described as

$$d\_dsvm_i = \sum_{j=1}^{m} (\mathrm{TF}(w_i, doc.e_j) \cdot \varepsilon_j) \cdot \mathrm{IDF}_{\mathrm{ex}}(w_i, c) \cdot \rho_{\mathrm{CD}} \tag{7}$$

where m is the number of elements in document doc, $doc.e_j$ is the jth element e_j of doc, ε_j, which is the unit vector of $doc.e_j$ in SLVM, is the dot product of s-dimensional unit vector $\mathbf{u}_j$ and s-dimensional weight vector $\mathbf{v}$. $\mathrm{IDF}_{\mathrm{ex}}(w_i, c)$ is the revised IDF presented above. The factor ρ_{CD} is the distribution modifying factor, which equals the reciprocal of arithmetic product of WCD and ACD.

3 Brief of extreme learning machine

In this section, we present a brief overview of Extreme Learning Machine (ELM). ELM is a generalized Single Hidden-layer Feedforward Network. In ELM, the hidden-layer node parameters is mathematically calculated instead of being iteratively tuned, providing good generalization performance at thousands of times faster speed than traditional popular learning algorithms for feedforward neural networks.

Given N arbitrary samples $(\mathbf{x}_i, \mathbf{t}_i) \in \mathbf{R}^{n\times m}$ and activation function $\mathrm{g}(x)$, standard SLFNs are modeled mathematically as

$$\sum_{i=1}^{L} \beta_i \mathrm{g}_i(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i \, \mathrm{g}(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = \mathbf{o}_j,\ j = 1, ..., N \tag{8}$$

where L is the number of hidden layer nodes, $\mathbf{w}_i = [w_{i1}, w_{i2}, ..., w_{in}]^T$ is the input weight vector, $\beta_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{im}]^T$ is the output weight vector, b_i is the bias of ith hidden node, and $\mathbf{o}_j$ is the output of the jth node.

To approximate these samples with zero errors means that $\sum_{j=1}^{L} ||\mathbf{o}_j - \mathbf{t}_j|| = 0$ [12], where exist β_i, $\mathbf{w}_i$ and b_i satisfying that

$$\sum_{i=1}^{L} \beta_i \, \mathrm{g}(\mathbf{w}_i \mathbf{x}_j + b_i) = \mathbf{t}_j,\ j = 1, ..., n \tag{9}$$

which can be rewritten in terms as

$$\mathbf{H}\beta = \mathbf{T} \tag{10}$$

where $\mathbf{T} = \left[\mathbf{t}_1^T, ..., \mathbf{t}_L^T\right]_{m\times L}^T$, $\beta = [\beta_1^T, ..., \beta_L^T]_{m\times L}^T$ and

$$\mathbf{H} = \begin{bmatrix} \mathrm{h}(\mathbf{x}_1) \\ \vdots \\ \mathrm{h}(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \mathrm{g}(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & \mathrm{g}(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ \vdots & \cdots & \vdots \\ \mathrm{g}(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & \mathrm{g}(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}_{n\times L} \tag{11}$$

$\mathbf{H}$ is the ELM feature space to map the n-dimensional input data space into l-dimensional hidden nodes space. The decision function is described as

$$\mathrm{f}(x) = \mathrm{sign}\left(\sum_{i=1}^{L} \beta_i \, \mathrm{g}(\mathbf{w}_i, b_i, \mathbf{x})\right) = \mathrm{sign}(\beta \cdot \mathrm{H}(\mathbf{x})) \tag{12}$$

and therefore ELM Algorithm [12] is described as Algorithm 1.

Algorithm 1 ELM

```
1: for $i = 1$ to $n$ do
2:     randomly assign input weight $\mathbf{w}_i$
3:     randomly assign bias $b_i$
4: end for
5: calculate $\mathbf{H}$
6: calculate $\beta = \mathbf{H}^{\dagger}\mathbf{T}$
```

In ELM, the parameters of hidden layer nodes, namely $\mathbf{w}_i$ and b_i, is chosen randomly without acknowledging the training data sets. The output weight β is then calculated with matrix computation formula $\beta = \mathbf{H}^{\dagger}\mathbf{T}$, where $\mathbf{H}^{\dagger}$ is the Moore-Penrose Inverse of $\mathbf{H}$.

4 Probability based classifiers for multiclass classification

In [22], we proposed v-ELM (voting-ELM) to improve the performance of ELM for multiclass classification problems. Based on voting theory, v-ELM gains a higher accuracy of classification than the original ELM for multiclass classification. However, v-ELM has two major defects in implementation. OAA can not be applied to v-ELM easily, which is restricted by voting results combination. Additionally, the performance of v-ELM heavily rely on the postprocessing methods REV (Revoting of Equal Votes). In REV method, the revote result of the single binary ELM classifier is fully trusted and assigned as the final result. Therefore, improved ELM methods based on v-ELM for multiclass classification are proposed in this section, which achieve higher accuracy and better generalization performance.

4.1 Probability based voting extreme learning machine

In this section, probability based Voting ELM (PV-ELM) is presented ,which further improves v-ELM by introducing the probability calculation into the training phase instead of the postprocessing phase. The probabilities of correct outcomes, type I errors and type II errors will not be taken into consideration after the original votes but directly during the decision procedure. TP (True Positive) and TN (True Negative) are correct outcomes. Type I error is FP (False Positive) and Type II error is FN (False Negative).

In [22], an improved ELM algorithm for multiclass classification named v-ELM (voting-ELM) is proposed. PV-ELM applies the probabilistic votes into training phase of v-ELM. PV-ELM intends to map the 1 or 0 integral votes into the decimal numbers between 0 and 1 interval. The class with the vote closed to 1 most is the final result of the classification. Therefore, the focus of the decision function shifts from the revoting postprocessing to the every single original binary votes. The inflexible postprocessing methods are effectively avoided while maintaining similar or even better classification accuracy compared with postprocessing methods like REV.

ELM classifiers based on OAA in [21] is practical for protein secondary structure prediction dues to the small number of protein patterns. It is only a tree-class classification problem, which results to the possibility of listing all the eight vote results combinations. The defects of voting theory will be reduced during the postprocessing phase. However, the possible results combination table will expand exponentially when the the number of pattern class increases. We conclude that, for v-ELM based on OAA, postprocessing method does not have a high generalization applicability.

On the basis of PV-ELM, OAA can be easily applied to v-ELM with no concern that the combination of voting results will be extremely complicated. The training phase of PV-ELM based on OAA is presented as Algorithm 2.

The v-ELM algorithm decomposes a m-class classification problem into $m(m-1)/2$ binary classification problems using traditional OAO method. Each binary classifier $\mathrm{elm}_{(j,k)}$ is trained by samples belonging to class j and class k. During the test phase, $\mathrm{elm}_{(j,k)}$ votes for class j if the output is $+1$ and for class k if -1. The class with the highest votes is the final classification result.

Applied to PV-ELM, OAO performs more effectively. In PV-ELM based on OAO, when a document is to be classified, the binary classifiers cast a vote using the probabilities instead of integral number 1 to reduce the misleading vote from

Algorithm 2 PV-ELM based on OAA

```
1: for $j = 1$ to $m$ do
2:     for $i = 1$ to $m$ and $i != j$ do
3:         Assign class $k$ with the union of class $k$ and class $i$
4:     end for
5:     Train elm$_{(j,k)}$ with training samples of class $j$ and class $k$
6:     Generate probability values of elm$_{(j,k)}$
7: end for
```

the binary classifiers not related to the class of the document. The training phase of PV-ELM based on OAO is presented as Algorithm 3.

Algorithm 3 PV-ELM based on OAO

```
1: for $j = 1$ to $m$ do
2:     for $k = 1$ to $m$ do
3:         Train elm$_{(j,k)}$ with training samples of class $j$ and class $k$
4:         Generate probability values of elm$_{(j,k)}$
5:     end for
6: end for
```

After the training phase of PV-ELM based on OAA/OAO, the integral votes from the multiple binary classifiers are multiplied by the probability of the binary classifier for the specific class. When a testing sample is tested by a elm$_{(j,k)}$, the vote is multiplied by a weight $\mathrm{ACC}^{j}_{(j,k)}$ if the output of the classifier is for j or by $\mathrm{ACC}^{k}_{(j,k)}$ for k, where

$$\mathrm{ACC}^{j}_{(j,k)} = \left(\mathrm{TP}^{j}_{(j,k)} + \mathrm{TN}^{k}_{(j,k)}\right) / (\mathrm{T+N}) \tag{13}$$

and

$$\mathrm{ACC}^{k}_{(j,k)} = \left(\mathrm{TP}_{(}j, k)^{k} + \mathrm{TN}^{j}_{(j,k)}\right) / (\mathrm{T+N}) \tag{14}$$

$\mathrm{TP}^{k}_{(j,k)}$, for instance, denotes the True Positive of the binary classifier elm$_{(j,k)}$ for class k. The subscript of $\mathrm{TP}^{k}_{(j,k)}$ identifies the binary ELM classifier and the superscript is referred to the class of the probability being calculated. ACC is the accuracy criterion which evaluats a binary ELM classifier.

4.2 Postprocessing method ε-RCC

In [22], a postprocessing method named RCC (Revoting of Confusing Classes) invokes the revote action to avoid the effects caused by confusing classes. RCC takes the top 2 classes, which have the highest wrongly classified probability between each other, as confusing classes. The matrix **confClass** is a diagonal matrix, indicating that if class k is a confusing class to class j, class j is also a confusing class to class k. However, confusing relation is not always reciprocal and the selection of top 2 to define confusing class is not rational enough.

In this section, an improved ε-RCC (ε-Revoting of Confusing Classes) is proposed to refine the voting result correction based on RCC. In ε-RCC method, non-diagonal matrix and determine threshold is applied instead. If the probability of wrongly classifying a document from class j to class k is greater than the predefined threshold ε, class k is a confusing class to class j. After the voting of training phase, ε-RCC decides whether to revote or not. We believe that the binary classifier $\text{elm}_{(j,k)}$ has the most accurate classification result for the documents of class j and class k theoretically.

Algorithm 4 ε-RCC

```
 1: generate voting result matrix of training data
 2: assign FP $\times$ FN of each class to matrix **errorPos**
 3: **for** each element in **errorPos** **do**
 4:     **if** **errorPos**$_{(}j, k)$ $> \varepsilon$ **then**
 5:         **confClass**$_{(j,k)} = 1$
 6:     **else**
 7:         **confClass**$_{(j,k)}$=0
 8:     **end if**
 9: **end for**
10: generate voting result vector **scores** of a test sample
11: assign top 2 of **scores** to class *top* and class *scnd* respectively
12: **if** **confClass**$_{(}top, scnd) = 1$ or **confClass**$_{(}top, scnd)$=1 **then**
13:     assign the result of $\text{elm}_{(j,k)}$ to $revoteResult$
14: **else**
15:     assign the original result to $revoteResult$
16: **end if**
```

Wrongly classified probability matrix **errorPos** is a $m \times m$ dimensional matrix. Each element **errorPos**$_{(}j, k)$ is the probability of the case that a document of class j is wrongly classified to class k. Confusion classes matrix **confClass** is a Boolean matrix calculated based on matrix **errorPos**. Element **confClass**$_{(}j, k)$ equal to 1 indicates that class j is a confusion class to class k and 0 indicates not.

Algorithm 4 represents the main procedure of ε-RCC. During the training phase, original votes are wrongly classified probabilities are calculated. Confusing classes matrix is then calculated and checked whether the result vote of ELM output is a confusion class after the original votes are casted in the test phase. If the class with original highest vote is a confusion class to or of the class with second highest vote, the document is revoted by the binary classifier trained by these two classes. Otherwise, the original output of ELM is trusted completely.

5 Experimental results

In this section, a series of experiments are conducted to evaluate the multi-class classification performance of the solution proposed in this paper. The experiments environment includes Visual Studio 2008 and MATLAB R2009a. We built the codes of algorithms and ran the programs on a PC with 3.0 GHz Intel duo CPU and 2 GB RAM.

Table 1 Datasets of XML documents from RSS feeds

Dataset I	Class	Industry	Cloud	Linux	XML	OpenSource	Java
	# of docs	751	1039	510	677	994	483
Dataset II	Class	Business	Politics	Health	Technology	Sports	Money
	# of docs	1,000	950	655	871	1,000	702

5.1 Datasets and evaluation criteria

The experiments use RSS feeds as original datasets. RSS (Really Simple Syndication) is a family of standard web feed formats. An RSS feed is a standardized XML file containing web content titles, abstract, author information, full text body, etc. Many portal websites provide RSS feeds of syndicated multimedia and information. In our experiments, two RSS feeds from IBM Developer Works (Dataset I) and ABC News (Dataset II) are fetched to provide XML documents of several classes as datasets. The XML documents clips we captured from these two RSS feeds are presents in Table 1.

We use the traditional evaluation criteria to evaluate the performance of classifiers built in our experiments. Table 2 presents a brief summary of the traditional criteria measures with computing formula.

The traditional F-measure is the harmonic mean of precision and recall. The experiments in this section, excluding the ones comparing precision and recall criteria, are evaluated by F-measure unless otherwise specified.

5.2 Experiments setup

Our experiments are mainly conducted to the following three aspects: (1) representation capability comparison between different XML representation models including DSVM proposed in this paper and the traditional ones; (2) classification results evaluation of PV-ELM and postprocessing method ε-RCC; (3) performance comparison among the strategies proposed in this paper, which includes DSVM, PV-ELM based on OAA/OAO and ε-RCC, and traditional classification algorithms SVM and BP Neural Network.

All the datasets are divided in the ratio of 3:1, that is, 75 % of the datasets are used as training samples and 25 % as testing samples. The number of hidden layer nodes should be chosen before the experiments. We randomly pickup half of the XML documents and test the precision criterion with different numbers of hidden layer nodes. After adequate times of test, we set 100 as the number of the hidden layer nodes.

Table 2 Brief summary of classification evaluation criteria

Criterion	Computing formula
Precision	TP/(TP + FP)
Recall	TP/(TP + FN)
F-measure (F1 score)	2 × Precision × Recall/ (precision + Recall)

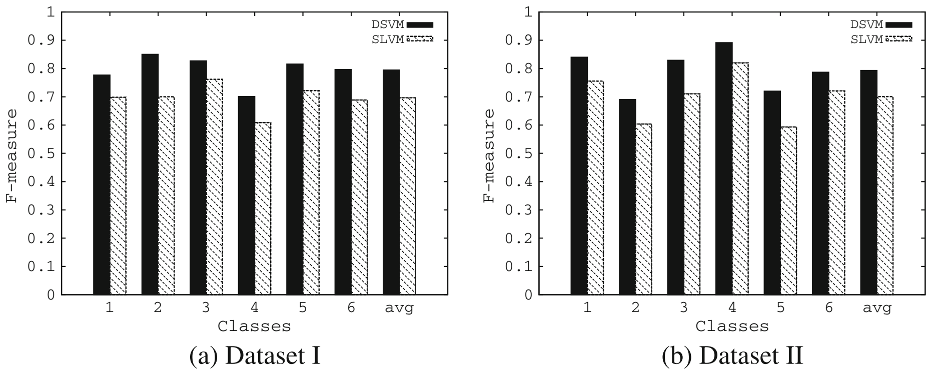


Figure 1 Performance comparison between SLVM and DSVM on ELM

5.3 Performance evaluation of XML representation model

The XML representation model performance is compared between SLVM (Structured Link Vector Model) and DSVM (Distributed Structured Vector Model) proposed in this paper. The experimental result is shown in Figure 1.

Figure 1a presents the performance comparison between SLVM and DSVM on the XML documents of Dataset I and Figure 1b of Dataset II. Due to the fully consideration of distribution information's effects on TFIDF and the more rational weight settings of structural information, DSVM performs much better on both two datasets measured by F-measure. All the following experiments are conducted by using DSVM to represent XML documents.

5.4 Performance evaluation of training time

Training time is one of the most important performance indicators in classification problems. In this section, two training time comparisons are made. One is between PV-ELM based on OAA/OAO and original ELM, while the other one is between

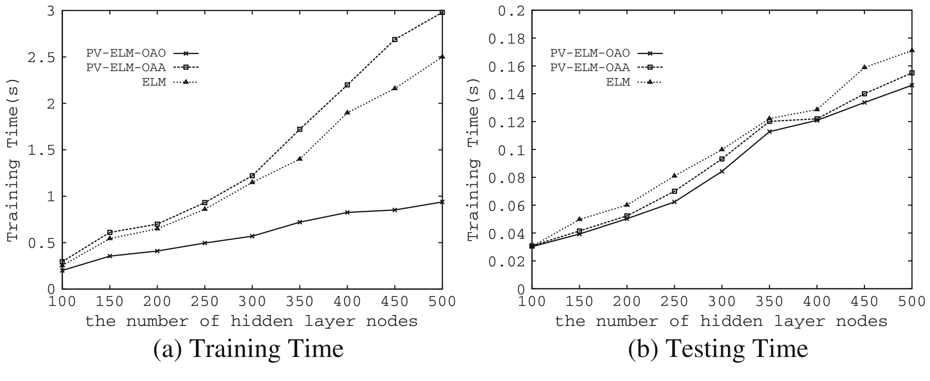


Figure 2 Training/testing time comparison between ELM and each single PV-ELM

Table 3 Training/testing time comparison among ELM, PV-ELM, SVM and BP

Algorithm	Training time (s)	Testing time (s)
ELM	0.3950	0.1222
PV-ELM-OAO	7.2355	0.3950
PV-ELM-OAA	2.3812	0.2793
SVM	10.0182	1.5005
BP	382.1565	0.1345

ELM based algorithm and traditional classification algorithm including SVM and BP Neural Network. Figure 2 shows the training time of PV-ELM based on OAA/OAO and original ELM for multiclass classification.

Figure 2a shows the comparison on training time among PV-ELM based on OAA/OAO and original ELM for multiclass classification. The training time of original ELM denotes the training time of ELM multiclass classifier, while the training time of PV-ELM is the average training time of all the binary classifiers. With a fixed number of hidden layer nodes, each single binary classifier of PV-ELM based on OAO process the least samples, which belongs to only two classes, and therefore costs the lowest training time. However, PV-ELM-OAO has the most number, namely $m(m-1)/2$, of binary ELM classifiers, the total training time of PV-ELM-OAO is the highest of all. Likewise, PV-ELM-OAA, which is composed of m binary ELM classifiers, has the second highest total training time and original ELM the lowest.

Figure 2b reflects the testing time of single ELM classifiers. All these three algorithms have similar testing time. However, due to the different number of single ELM classifiers, PV-ELM-OAO, similar to training time, costs the highest testing time, followed by PV-ELM-OAA and then original ELM.

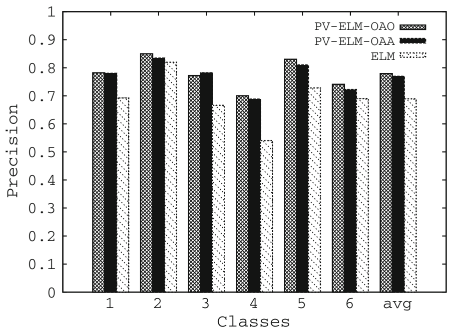
Comparisons of training time and testing time among PV-ELM based on OAA/OAO, original ELM, SVM and BP Neural Network are also made in this section. From Table 3 we can see that v-ELM takes more training time and testing time than original ELM but less than SVM and BP neural networks. Compared with BP neural networks, v-ELM and ELM cost two orders of magnitudes less time. This is a huge advantage of ELM based algorithm. ELM is a BP neural network essentially, so the testing time of ELM is close to BP neural network. V-ELM costs little more time than ELM and BP because of its larger number of binary ELM classification units. Both the training time and testing time of SVM is not far different with ELM based algorithm. The testing time of SVM, whose structure differs from neural networks, is not significantly larger than the other algorithms.

Based on the experimental results mentioned above, we thus conclude that PV-ELM costs more training and testing time than original ELM, especially PV-ELM-OAO. Despite of this, both of PV-ELM and ELM are extremely faster than traditional BP neural networks. SVM has a close training time to ELM based algorithm but longest testing time among all these algorithms.

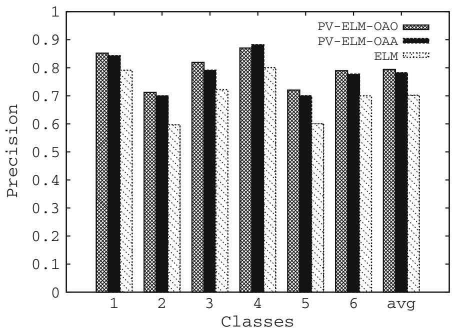
5.5 Performance evaluation of PV-ELM with ε-RCC

In this section, we place great emphasis on the performance comparison between original ELM for multiclass classification and PV-ELM with the postprocessing method ε-RCC. We first compare the performance evaluation between original ELM and PV-ELM. The experimental results are shown in Figure 3.

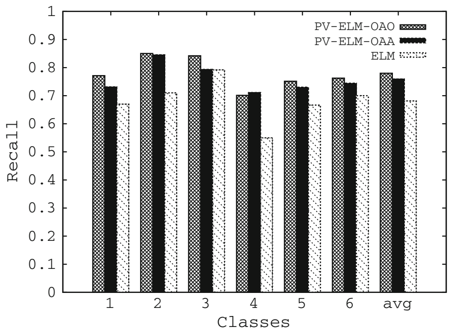
It is clear from Figure 3 that the performance of PV-ELM based on OAA/OAO, especially OAO, is higher than original ELM. In PV-ELM based on OAO, each binary classifier is trained by only two classes of samples using ELM algorithm and therefore gains higher classification accuracy than ELM. The shortages of voting theory, e.g. decision making of equal votes, is effectively compensated by



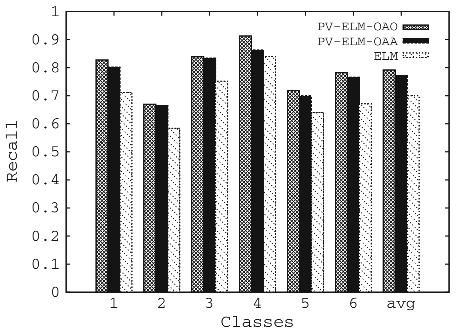
(a) Precision comparison on Dataset I



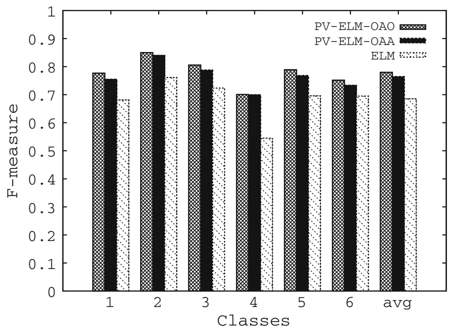
(b) Precision comparison on Dataset II



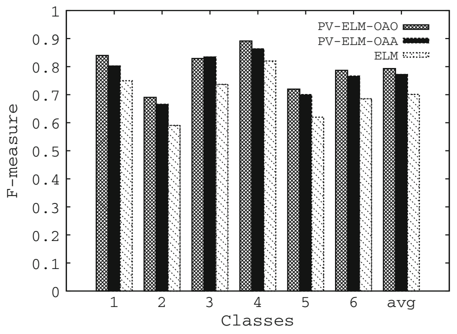
(c) Recall comparison on Dataset I



(d) Recall comparison on Dataset II



(e) F1 comparison on Dataset I



(f) F1 comparison on Dataset II

Figure 3 Performance comparison between PV-ELM and ELM

Table 4 Performance evaluation of postprocessing methods

	Dataset I			Dataset II		
	Precision	Recall	F1	Precision	Recall	F1
v-ELM & RCC	0.7854	0.7796	0.7825	0.7973	0.7846	0.7909
PV-ELM-OAA & RCC	0.8039	0.7892	0.7964	0.8109	0.7993	0.8051
PV-ELM-OAO & RCC	0.8143	0.8004	0.8072	0.8273	0.8286	0.8279
PV-ELM-OAA & ε-RCC	0.8307	0.8271	0.8289	0.8395	0.8300	0.8347
PV-ELM-OAO & ε-RCC	0.8466	0.8557	0.8511	0.8572	0.8661	0.8616

introducing probabilities into PV-ELM during training phase. Combining all the binary classifiers, PV-ELM based on OAA/OAO performs better than original ELM on precision, recall and therefore F-measure, which is the harmonic mean of precision and recall.

Performance of postprocessing method ε-RCC is also evaluated in the experiments. Table 4 shows that ε-RCC introduced into v-ELM improves the performance to a certain degree. Samples belong to class i may frequently misclassified to class k, but it does not mean that samples from class k may also frequently misclassified to class j. Both these two cases of confusing classes are taken into consideration in ε-RCC to achieve higher performance. Besides, compared with RCC, ε-RCC has a more rational confusing classes decision strategy and a high probability of correcting misclassified confusing classes. For this reason, the improvement achieved by ε-RCC is obvious, though the proportion of confusing classes in the whole data sets is not big.

5.6 Performance comparison with traditional classifiers

The experimental results presented in this section focus on the comparison among the performances of different classification algorithms. Traditional SVM and BP Neural Network are selected to compare with ELM and the improved PV-ELM algorithm (Table 5).

Experimental results show that PV-ELM based on OAA/OAO with ε-RCC is the most performing method on both datasets. PV-ELM based on OAO has a bit higher performance than PV-ELM based on OAA. Original ELM and SVM have similar performance to each other. Traditional BP neural network performs the worst in all these classification algorithms.

Table 5 Performance (F-measure) comparison among different classifiers

	Dataset I	Dataset II	# of SVs/ Neurons
ELM	0.7689	0.7714	150
PV-ELM-OAA & ε-RCC	0.8289	0.8347	100
PV-ELM-OAO & ε-RCC	0.8511	0.8616	100
SVM	0.7621	0.7720	407
BP	0.7215	0.7206	200

6 Conclusions

Taking into account the theoretical analysis and experimental results presented in the previous sections, we can see that ELM, which can be implemented easily with less sensitive parameters, has a extreme learning speed and a higher performance comparing with traditional classification algorithms, especially traditional BP neural network. Introducing probability calculation into the training phase of classification efficiently avoids the drawbacks of voting theory for multiclass classification. XML document representation model DSVM enhances the capability of representing structural information of XML documents to a great extend. Postprocessing method ε-RCC is also proposed to refine classification accuracy. The PV-ELM based on OAA/OAO with DSVM and ε-RCC proposed in this paper satisfactorily increases the classification performance.

Acknowledgements This research was partially supported by the National Natural Science Foundation of China under Grant Nos. 60933001, 61272181, and 61073063; the National Basic Research Program of China under Grant No. 2011CB302200-G; the 863 Program under Grant No. 2012AA011004, and the Public Science and Technology Research Funds Projects of Ocean Grant No. 201105033.

References

1. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers. J. Mach. Learn. Res. **1**, 9–16 (2000)
2. Feng, G., Huang, G.-B., Lin, Q., Gay, R.K.L.: Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans. Neural Netw. **20**, 1352–1357 (2009)
3. Han, E.-H., Karypis, G.: Centroid-based document classification: analysis and experimental results. In: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 424–431. Springer-Verlag, New York (2000)
4. Ho, T.K., Hull, J.J., Srihari, S.N.: Decision combination in multiple classifier systems. IEEE Trans. Pattern Anal. **16**, 66–75 (1994)
5. Hsu, C.-W., Lin, C.-J.: A comparison of methods for multiclass support vector machines. IEEE Trans. Neural Netw. **13**, 415–425 (2002)
6. Huang, G.-B., Chen, L.: Convex incremental extreme learning machine. Neurocomputing **70**, 3056–3062 (2007)
7. Huang, G.-B., Chen, L.: Enhanced random search based incremental extreme learning machine. Neurocomputing **71**, 3460–3468 (2008)
8. Huang, G.-B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural Netw. **17**, 879–892 (2006)
9. Huang, G.-B., Ding, X., Zhou, H.: Optimization method based extreme learning machine for classification. Neurocomputing **74**, 155–163 (2010)
10. Huang, G.-B., Zhu, Q.-Y., Mao, K.Z., Siew, C.-K., Saratchandran, P., Sundararajan, N.: Can threshold networks be trained directly?. IEEE T. Circuits-II **53**, 187–191 (2006)
11. Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: Extreme learning machine: a new learning scheme of feedforward neural networks. In: Proceedings IEEE International Joint Conference on Neural Networks, vol. 2, pp. 985–990 (2004)
12. Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: Extreme learning machine: theory and applications. Neurocomputing **70**, 489–501 (2006)
13. Jianwu, Y., Xiaoou, C.: A semi-structured document model for text mining. J. Comput. Sci. Technol. **17**, 603–610 (2002)
14. Jun Rong, H., Huang, G.-B., Soon Ong, Y.: Extreme learning machine for multi-categories classification applications. In: IJCNN, pp. 1709–1713 (2008)
15. Jun Rong, H., Soon Ong, Y., Hwee Tan, A., Zhu, Z.: A fast pruned-extreme learning machine for classification. Neurocomputing **72**, 359–366 (2008)

16. Li, M.-B., Huang, G.-B., Saratchandran, P., Sundararajan, N.: Fully complex extreme learning machine. Neurocomputing **68**, 306–314 (2005)
17. Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A.: Op-elm: optimally pruned extreme learning machine. IEEE Trans. Neural Netw. **21**, 158–162 (2010)
18. Rong, H.-J., Huang, G.-B., Sundararajan, N., Saratchandran, P.: Online sequential fuzzy extreme learning machine for function approximation and classification problems. IEEE Trans. Syst. Man Cybern. **39**, 1067–1072 (2009)
19. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York, NY, USA (1986)
20. Suresh, S., Saraswathi, S., Sundararajan, N.: Performance enhancement of extreme learning machine for multi-category sparse data classification problems. Eng. Appl. Artif. Intell. **23**, 1149–1157 (2010)
21. Wang, G., Zhao, Y., Wang, D.: A protein secondary structure prediction framework based on the extreme learning machine. Neurocomputing **72**, 262–268 (2008)
22. Zhao, X., Wang, G., Bi, X., Gong, P., Zhao, Y.: Xml document classification based on elm. Neurocomputing **74**(16), 2444–2451 (2011)
23. Zhu, Q.-Y., Qin, A.K., Suganthan, P.N., Huang, G.-B.: Evolutionary extreme learning machine. Pattern Recogn. **38**, 1759–1763 (2005)