

# Dynamic ensemble extreme learning machine based on sample entropy

Jun-hai Zhai · Hong-yu Xu · Xi-zhao Wang

Published online: 7 February 2012  
© Springer-Verlag 2012

**Abstract** Extreme learning machine (ELM) as a new learning algorithm has been proposed for single-hidden layer feed-forward neural networks, ELM can overcome many drawbacks in the traditional gradient-based learning algorithm such as local minimal, improper learning rate, and low learning speed by randomly selecting input weights and hidden layer bias. However, ELM suffers from instability and over-fitting, especially on large datasets. In this paper, a dynamic ensemble extreme learning machine based on sample entropy is proposed, which can alleviate to some extent the problems of instability and over-fitting, and increase the prediction accuracy. The experimental results show that the proposed approach is robust and efficient.

**Keywords** Extreme learning machine · Dynamic ensemble · AdaBoost · Bagging · Sample entropy

## 1 Introduction

Extreme learning machine (ELM) was recently proposed by Huang et al. (2006, 2011) for single-hidden layer

feed-forward neural networks (SLFNNs). In ELM the input weights and the hidden layer biases can be chosen randomly, the output weights can be analytically determined with Moore–Penrose generalized inverse  $H^*$  of the hidden layer output matrix  $H$ . Unlike other gradient descent-based learning algorithms [such as back-propagation algorithm (BP)] for feed-forward networks, the ELM does not require iterative techniques to adjust input weights and hidden layer biases during training process, so it becomes a simple learning method with extremely fast learning speed (Feng et al. 2009; Liang et al. 2006; Huang et al. 2010; Wu et al. 2011; Wang et al. 2011). Although the ELM has simplified the learning approach for SLFNNs avoiding iterative and descent steps, the following issues still remain in ELM, especially dealing with large datasets.

1. Predictive instability caused by randomly selecting the input weights and the hidden layer biases;
2. Over-fitting problem caused by the complexity of distribution of input instances and much more hidden nodes on large datasets;
3. The order of matrix  $H$  is  $N \times M$ , where  $N$  is the number of samples, and  $M$  is the number of hidden layer nodes. For large datasets, the order of  $H$  is very high. Large memory is required to calculate the Moore–Penrose generalized inverse  $H^*$ .

Ensemble learning (EL) or combining classifiers (CC) could solve the problems mentioned above (Hansen and Salamon 1990; Zhou et al. 2002; Kittler et al. 1998; Rogova 1994; Pal and Pal 2001; Zhang et al. 2011; Biggio et al. 2010). EL is a learning paradigm where a collection of a finite number of base classifiers such as neural network or decision tree is trained for the same task (Kittler et al. 1998; Pal and Pal 2001), and can significantly improve the

---

J. Zhai (✉) · H. Xu · X. Wang  
Key Laboratory of Machine Learning and Computational Intelligence, College of Mathematics and Computer Science, Hebei University, Baoding 071002, Hebei, China  
e-mail: mczjh@hbu.edu.cn

H. Xu  
e-mail: xuhongyufat@126.com

X. Wang  
e-mail: xizhaowang@ieee.org

generalization ability of classification system (Hansen and Salamon 1990; Kittler et al. 1998).

In general, an ensemble of classifiers is generated in two steps:

1. Training a number of the base classifiers;
2. Combining the predictions of classifiers.

The most prevailing approaches of training base classifiers are Bagging and AdaBoost (Breiman 1996; Freund and Schapire 1997). Bagging generates diverse classifiers by randomly selecting subsets of samples to train classifiers (Kuncheva and Whitaker 2003; Brown et al. 2005; Mao et al. 2011). Intuitively, we would expect classifiers trained by different sample subsets to exhibit different behaviors. AdaBoost (Freund and Schapire 1997; Wu et al. 2008) also uses parts of samples to train classifiers, but not randomly, it maintains a set of weights over the original training set and adjusts these weights after each classifier is learned. The adjustments increase the weight of examples that are misclassified and decrease the weight of examples that are correctly classified. Liu and Wang (2010) proposed an approach of ensemble-based extreme learning machine (EN-ELM) to enhance the generalization ability (Wang and Dong 2009; Wang et al. 2008, 2011). EN-ELM uses the cross validation scheme to create an ensemble of ELM classifiers for classification. EN-ELM uses static ensemble strategy (Woods et al. 1997) to classify a test sample, all base classifiers are considered equally important. Actually, for different test samples the base classifiers have different degrees of confidence due to the diversity, and then their importance is different. Wang and Li (2010) proposed the dynamic AdaBoost ensemble extreme learning machine, which regards the extreme learning machine as weak learning machine, dynamic AdaBoost ensemble algorithm is used to integrate the outputs of weak learning machines, and makes use of fuzzy activation function as activation function of extreme learning machine. In this paper, different from the works in Liu and Wang (2010) and Wang and Li (2010), we propose a method of dynamic ensemble (Ko et al. 2008) ELM classifier based on sample entropy, which can alleviate the problems of instability and over-fitting, and increase the prediction accuracy.

In our method, we use AdaBoost to generate  $N$  training subsets from training set, and then train one ELM classifier for each of training subsets, hence  $N$  classifiers can be obtained in all; finally, based on the strategy of dynamic ensemble with sample entropy, an unseen instance can be classified.

The remaining of this paper is organized as follows. Section 2 will briefly present the ELM algorithm and the AdaBoost algorithm. In Sect. 3, we described the proposed method of dynamic ensemble ELM classifiers based on

sample entropy. Performance evaluation is presented in Sect. 4. Section 5 gives conclusion.

## 2 Brief reviews of extreme learning machine and AdaBoost algorithm

In this section, we briefly review the extreme learning machine and the AdaBoost algorithm.

### 2.1 Extreme learning machine

Extreme learning machine proposed by Huang et al. (2006) is an efficient and practical learning mechanism for the SLFNNs, see Fig. 1. According to Theorem 2.1 of reference (Huang et al. 2006), the input weights and biases do not need to be adjusted. It is possible to analytically determine the output weights by finding the least-square solution. The neural network is obtained after very few steps with very low computational cost. Since Huang's seminal work (Huang et al. 2006), many researchers have paid their attention to ELM recently, such as Wang et al. (2011) studied the effectiveness of extreme learning machine. José et al. (2011) studied the regularized extreme learning machine for regression problems. Mohammed et al. (2011) applied ELM to face recognition. Chacko et al. (2011) successfully apply ELM to handwritten character recognition field. Emilio et al. proposed a Bayesian approach to ELM, which allows the introduction of a priori knowledge, and presents high generalization capabilities (Emilio et al. 2011). As pointed out in Emilio et al. (2011), ELM represents a suitable approach to obtain models from databases, especially from huge databases within a reasonable time. An excellent survey papers on ELM can be found in Huang et al. (2011).

Given a training dataset,  $L = \{(x_i, t_i) | x_i \in R^n, t_i \in R^m, i = 1, 2, \dots, N\}$ , where  $x_i = (x_{i1}, x_{i2}, \dots, x_{in})^T$  and  $t_i = (t_{i1}, t_{i2}, \dots, t_{im})^T$ . A SLFNN with  $M$  hidden nodes is formulated as

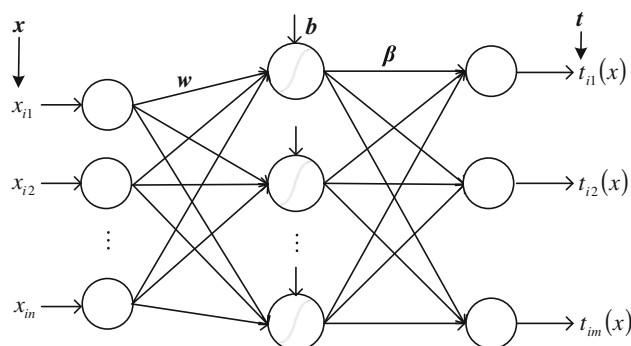


Fig. 1 The SLFNN trained with ELM algorithm

$$\sum_{j=1}^M \beta_j g(w_j \cdot x_i + b_j) = t_i \quad (i = 1, 2, \dots, N) \tag{1}$$

where  $w_j = [w_{j1}, w_{j2}, \dots, w_{jm}]^T$  is the weight vector connecting the  $j$ th hidden node with the input nodes.  $\beta_j = [\beta_{j1}, \beta_{j2}, \dots, \beta_{jm}]^T$  is the weight vector connecting the  $j$ th hidden node with the output nodes, and  $b_j$  is the threshold of the  $j$ th hidden node. Equation (1) can be written in a more compact format as

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

where

$$H = \begin{bmatrix} g(w_1 \cdot x_1 + b_1) & \cdots & g(w_M \cdot x_1 + b_M) \\ \vdots & \cdots & \vdots \\ g(w_1 \cdot x_N + b_1) & \cdots & g(w_M \cdot x_N + b_M) \end{bmatrix}_{N \times M} \tag{3}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_M^T \end{bmatrix}_{M \times m} \quad \text{and} \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \tag{4}$$

$H$  is the hidden layer output matrix of the network, where the  $j$ th column of  $H$  is the  $j$ th hidden node's output vector with respect to inputs  $x_i$ , and the  $i$ th row of  $H$  is the output vector of the hidden layer with respect to input  $x_i$ . If the number of hidden nodes is equal to the number of distinct training samples, the matrix  $H$  is square, and SLFNNs can approximate these training samples with zero error. But generally, the number of hidden nodes is much less than the number of training samples. Therefore,  $H$  is a non-square matrix and we can not expect an exact solution of the system (2). Fortunately, it has been proved in Huang et al. (2006) and Huang and Chen (2007) that SLFNNs with random hidden nodes have the universal approximation capability; the hidden nodes could be randomly generated. According to the definition of the Moore–penrose generalized inverse, the smallest norm least-squares solution of (2) is given in Huang et al. (2006):

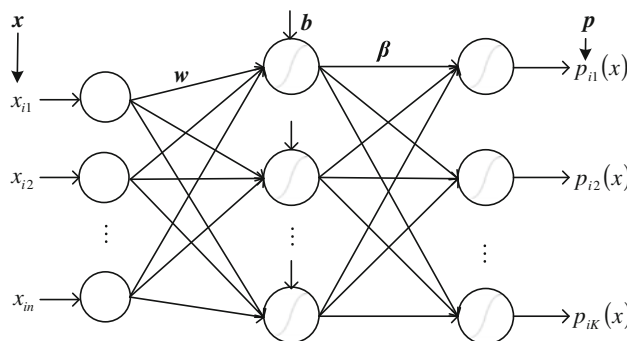
$$\hat{\beta} = H^*T \tag{5}$$

where  $H^*$  is the Moore–penrose generalized inverse of matrix  $H$  (Serre 2002). In the following, the ELM algorithm (Huang et al. 2006) is introduced.

**ELM algorithm**

**Input:** A training dataset  $\{(x_i, t_i) | x_i \in R^n, t_i \in R^m, i = 1, 2, \dots, N\}$  an activation function  $g$ , and the number of hidden nodes  $M$

**Output:** A weights matrix



**Fig. 2** The SLFNN trained with ELM with little change in output layer

1. Randomly assign input weights  $w_j$  and biases  $b_j, j = 1, \dots, M$ .
2. Calculate the hidden layer output matrix  $H$ ;
3. Calculate output weights matrix  $\hat{\beta} = H^*T$ .

In this paper, we will focus on the classification problem. Let  $Y = \{\omega_1, \omega_2, \dots, \omega_K\}$  be a set of class labels of samples, and let  $L = \{(x_i, y_i) | x_i \in R^n, y_i \in Y, i = 1, 2, \dots, N\}$ , we will use the ELM algorithm with little change as the base classifiers, see Fig. 2, i.e. the output nodes with sigmoid active function. The output of the SLFNN becomes

$$p_i(x) = g\left(\sum_{j=1}^M \beta_j g(w_j \cdot x_i + b_j)\right) \tag{6}$$

where  $p_i(x) = (p_{i1}(x), p_{i2}(x), \dots, p_{iK}(x))$ ,  $p_{ik}(x)$  denotes the probability (or membership degree) of sample  $x$  belongs to class  $k, 1 \leq p_{ik}(x) \leq 1, k = 1, 2, \dots, K$ .

**2.2 The AdaBoost algorithm**

The boosting algorithm was originally proposed by Schapire (1990), AdaBoost is an improved version of boosting algorithm (Freund and Schapire 1997). In this paper, we use AdaBoost algorithm to generate the subsets of sample set used for training the base ELM classifiers. For convenience, we list this algorithm in the following.

**AdaBoost algorithm**

**Input:**  $L = \{(x_i, y_i) | x_i \in R^n, y_i \in Y, i = 1, 2, \dots, N\}$ ,  $J$ , the number of iterations, BaseLearner, a base learning algorithm

**Output:** The final hypothesis

**Steps of the algorithm**

**Step 1:** Initialize distribution of weights on all samples  $x_i, (1 \leq i \leq N), D_1(x_i) = \frac{1}{N}$

- Step 2:** For  $j = 1$  to  $J$
- Step 3:** Train *BaseLearner* with  $D_j$   $C_j = \text{BaseLearner}(L, D_j)$
- Step 4:** Calculate the error of  $C_j$ ,  $e_j = \sum_{C_j(x_i) \neq y_i} D_j(x_i)$
- Step 5:** If  $e_j > \frac{1}{2}$ , then set  $J = j - 1$  and abort loop
- Step 6:** Set  $\beta_j = \frac{e_j}{1-e_j}$
- Step 7:** Update weights,  

$$D_{j+1}(x_i) = \begin{cases} D_j(x_i) \times \beta_j & \text{if } (C_j(x_i) \neq y_i) \\ D_j(x_i) & \text{otherwise} \end{cases}$$
- Step 8:** Normalize weights,  $D_{j+1}(x_i) = \frac{D_{j+1}(x_i)}{\sum_{i=1}^N D_{j+1}(x_i)}$
- Step 9:** Output  $C'(x) = \arg \max_{y \in Y} \sum_{j: C_j(x)=y} \log \frac{1}{\beta_j}$

### 3 Dynamic ensembles extreme learning machine based on sample entropy

In this section, we will present our method of dynamic ensemble ELM classifier based on sample entropy. We first give the basic concepts used in this paper.

**Definition 1** Given  $L = \{(x_i, y_i) | x_i \in R^n, y_i \in Y, i = 1, 2, \dots, N\}$ ,  $Y = \{\omega_1, \omega_2, \dots, \omega_K\}$  be a set of class labels of samples, the entropy of  $L$  is defined as follows.

$$E(L) = - \sum_{i=1}^K p_i \log_2 p_i \tag{7}$$

where  $p_i$  is the proportion of examples in  $\omega_i$  ( $i = 1, 2, \dots, K$ ).

**Definition 2** Given  $L = \{(x_i, y_i) | x_i \in R^n, y_i \in Y, i = 1, 2, \dots, N\}$ ,  $Y = \{\omega_1, \omega_2, \dots, \omega_K\}$  be a set of class labels of samples, the entropy of sample  $x_i$  is defined as follows.

$$\text{Entropy}(x_i) = - \sum_{k=1}^K p_k(x_i) \log_2 p_k(x_i) \tag{8}$$

where  $p_k(x_i)$  is the probability (or the membership degree) of instance  $x$  belongs to class  $k$  ( $k = 1, 2, \dots, K$ ).

**Definition 3** Given  $L = \{(x_i, y_i) | x_i \in R^n, y_i \in Y, i = 1, 2, \dots, N\}$ ,  $Y = \{\omega_1, \omega_2, \dots, \omega_K\}$  be a set of class labels of samples,  $C = \{C_1, C_2, \dots, C_J\}$  be  $J$  classifiers. The entropy of sample  $x_i$  with respect to classifier  $C_j$  ( $j = 1, 2, \dots, J$ ) is defined as follows.

$$\text{Entropy}_j(x_i) = - \sum_{k=1}^K p_{jk}(x_i) \log_2 p_{jk}(x_i) \tag{9}$$

**Definition 4** Given  $L = \{(x_i, y_i) | x_i \in R^n, y_i \in Y, i = 1, 2, \dots, N\}$ ,  $Y = \{\omega_1, \omega_2, \dots, \omega_K\}$  be a set of class labels of samples,  $C = \{C_1, C_2, \dots, C_J\}$  be  $J$  classifiers. The normalized entropy of sample  $x_i$  with respect to classifier  $C_j$  ( $j = 1, 2, \dots, J$ ) is defined as follows.

$$N\text{Entropy}_j(x_i) = \frac{\text{Entropy}_j(x_i)}{\sum_{j=1}^J \text{Entropy}_j(x_i)} \tag{10}$$

**Definition 5** Given a test sample  $x$ ,  $C = \{C_1, C_2, \dots, C_J\}$  be  $J$  basic classifiers. The threshold of entropy of sample  $x$  is defined as follows.

$$\gamma = \frac{1}{2} \left( \arg \max_{1 \leq j \leq J} (\text{Entropy}_j(x)) - \arg \min_{1 \leq j \leq J} (\text{Entropy}_j(x)) \right) \tag{11}$$

Let  $C = \{C_1, C_2, \dots, C_J\}$  be the trained-ELM classifiers,  $D = (D_1, D_2, \dots, D_J)$  be the distribution of weights on samples of  $L$ . The proposed algorithm is described as follows.

#### DE-ELM algorithm

**Input:**  $L, J$ , the number of iterations,  $n$ , the number of instances of subset  $L_j \subset L, j = 1, 2, \dots, J$

**Output:**  $C = \{C_1, C_2, \dots, C_J\}$

**Steps of the algorithm**

//Generate initial distribution

**STEP1:** For  $i=1$  to  $N$

$$D_1(x_i) = \frac{1}{N};$$

// Generate  $J$  base classifiers

**STEP2:** For  $j=1$  to  $J$

//Generate subset  $L_j \subset L$ , which satisfy  $|L_j| = \lambda|L| = \lambda N, \lambda \in (0,1)$

Let  $L_j = \phi$

While  $|L_j| < \lambda N$

Generate a random number  $r \in (0,1)$ ;

For  $i=1$  to  $N$

If  $D_1(x_i) \geq r$  then

$$L_j = L_j \cup \{x_i\};$$

$$T = T - \{x_i\};$$

Train a base learner  $ELM_j$  using  $L_j$ , the output of the  $ELM_j$  is a probability distribution  $(p_{j1}(x), p_{j2}(x), \dots, p_{jK}(x))$  where  $p_{jk}(x)$  ( $1 \leq j \leq J; 1 \leq k \leq K$ ) denotes the membership degree of instance  $x$  belong to class  $k$  based on classifier  $ELM_j$ ;

```

// Calculate error of ELMj, ej
For i=1 to N
    ej = ∑ELMj(xi)≠yi Dj(xi)
If ej > 1/2 break
Let βj = ej / (1 - ej)
//Update weights Dj+1(xi)
For i=1 to N
    If ELMj(xi) ≠ yi
        Dj+1(xi) = Dj(xi) × βj;
    Else
        Dj+1(xi) = Dj(xi);
// Normalize weights Dj+1(xi)
For i=1 to N
    Dj+1(xi) =  $\frac{D_{j+1}(x_i)}{\sum_{x_i \in L} D_{j+1}(x_i)}$ ;
// Calculate the entropy of test sample x with respect to classifier ELMj
STEP3: For j=1 to J
    For each test instance x
        Calculate sample entropy with (9)
        Calculate normalized sample entropy with (10)
//Dynamic ensemble classifiers for classifying test instance x
STEP4: For each test instance x
    Calculate the threshold of entropy of sample x with (11)

    Suppose that there exists T basic classifiers Ct (1 ≤ t ≤ T)
    satisfied condition NEntropyj(x) ≤ γ

    The label C' of x is determined by
    C' = arg max1 ≤ t ≤ T  $\left\{ \frac{1}{T} \sum_{j=1}^T \frac{NEntropy_j^{-1}(x)}{\sum_{j=1}^T NEntropy_j^{-1}(x)} p_{ji} \right\}$ 

```

### 4 Experimental results and analysis

The effectiveness of our proposed method is demonstrated through numerical experiments in the environment of Matlab 7.0 on a Pentium 4 PC. Totally our experiments select 8 UCI datasets which are Statlog (Shuttle) (DB1), letter recognition (DB2), pen-based recognition of handwritten digits (DB3), mushroom (DB4), landsat satellite image (DB5), optical recognition of handwritten digits (DB6), waveform (DB7), and car evaluation (DB8). The basic information of the eight datasets is listed in Table 1.

**Table 1** The basic information of the eight datasets used in our experiments

Name	Number of instances	Number of attributes	Number of classes
DB1	58,000	9	7
DB2	20,000	16	26
DB3	10,992	16	10
DB4	8,124	22	2
DB5	6,435	36	7
DB6	5,620	64	10
DB7	5,000	21	3
DB8	1,728	6	4

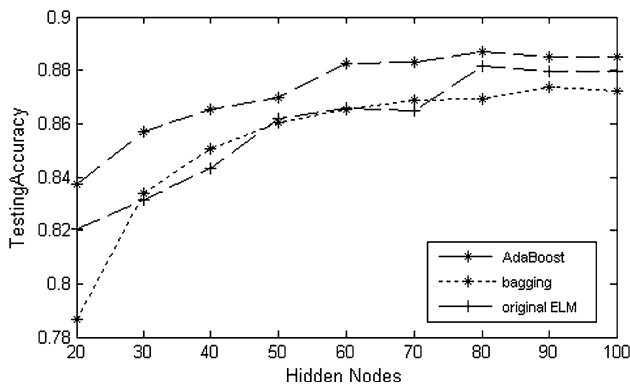
From Table 1, we can see that among the eight UCI datasets there are three large datasets, four medium datasets, and one small dataset. In our experiments, 50% samples chosen randomly are used as training set, and the other 50% samples are used for testing set. We set λ = 0.60, and J = 10, i.e. there are ten sub-classifiers (SLFNNs trained with ELM) are generated with AdaBoost algorithm. For each sub-classifier, the number of hidden nodes is determined using the method proposed in Feng et al. (2009). The performance of proposed method DE-ELM is compared with original ELM in three aspects, which are the influence of construction of sub-classifiers on ensemble system, average testing accuracy, stability.

**Experiment 1** The influence of construction of ELM base classifiers (i.e. SLFNNs trained with ELM) on ensemble system.

There are two commonly used methods (i.e. Bagging and AdaBoost) to generate the base classifiers with diversity. We compare the two ensemble methods of constructing ELM-base classifier with dataset landsat satellite image (DB5). In the experiment, we increase the number of hidden nodes of SLFNNs from 20 to 100, the experimental results shown in Fig. 3 illustrate the relationship between the testing accuracy and the number of hidden nodes. In addition, in this experiment, we also experiment on dataset landsat satellite image (DB5) with ELM. In the view of testing accuracy, the performance of ELM is superior to Bagging when the number of hidden nodes is greater than 73. For other datasets, the experimental results are similar. Based on the experimental results, we conclude that in the framework of ensemble SLFNNs trained with ELM AdaBoost is superior to Bagging, so in our method we prefer to select AdaBoost rather than Bagging.

**Experiment 2** Comparison with original ELM in average testing accuracy

In this experiment, we compare DE-ELM with the original ELM in average testing accuracy on eight datasets.



**Fig. 3** The influence of construction of ELM sub-classifiers on ensemble system

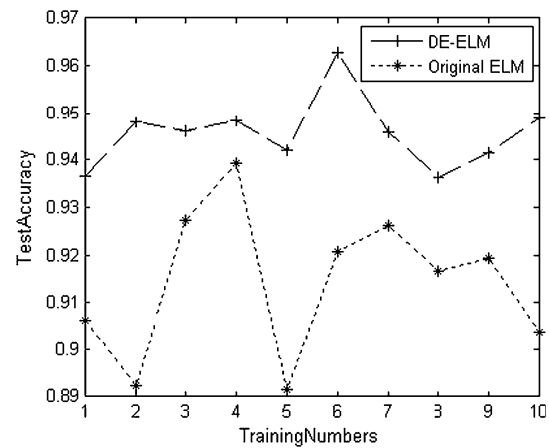
**Table 2** Comparison between DE-ELM and ELM

DBs	Average testing accuracy		Number of hidden nodes
	ELM	DE-ELM	
DB1	0.9298	0.9561	50
DB2	0.4880	0.5528	100
DB3	0.4303	0.4432	20
DB4	0.9139	0.9809	10
DB5	0.8260	0.8813	100
DB6	0.8320	0.8920	20
DB7	0.8452	0.8760	100
DB8	0.8391	0.8808	30

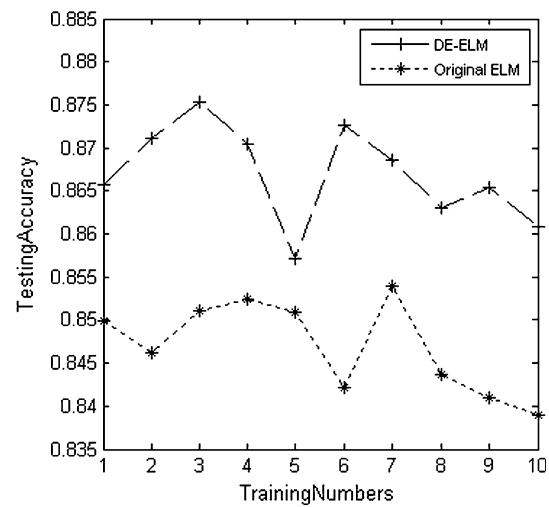
Table 2 shows the comparison results. For each dataset, we run the experiment 10 times. The experimental results are the average of the ten outputs. It can be seen from Table 2 that the average testing accuracies of DE-ELM are consistently higher than the ones of ELM no matter how small or big is the dataset. For example, for large dataset Statlog (shuttle) (DB1), we run the experiments ten times using original ELM and DE-ELM, the ten experimental results are shown in Fig. 4. Similarly, for medium dataset landsat satellite image (DB5), and optical recognition of handwritten digits (DB6), the ten experimental results are shown in Figs. 5 and 6, respectively. For small dataset car evaluation (DB8), the ten experimental results are shown in Fig. 7.

**Experiment 3** Comparison with original ELM in stability

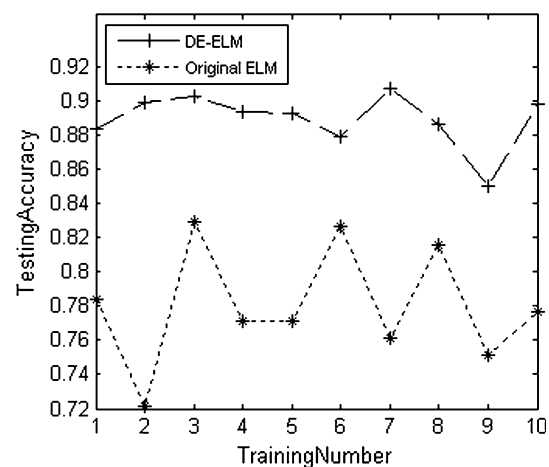
In this experiment, we compare DE-ELM with the original ELM in stability on three datasets, which are large dataset letter recognition (DB2), medium dataset landsat satellite image (DB5), and small dataset car evaluation (DB8), respectively. For each dataset, we run the experiments ten times using original ELM and DE-ELM, the



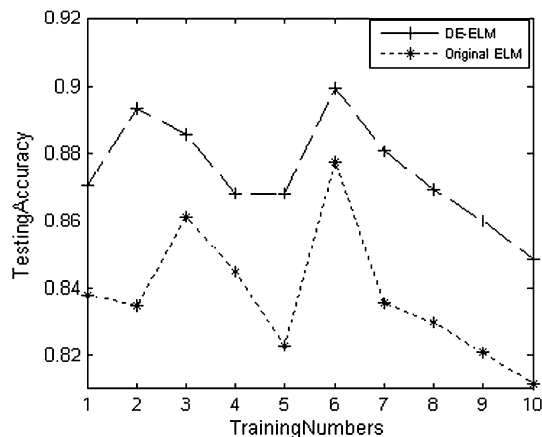
**Fig. 4** The experimental results in average testing accuracy on large dataset Statlog (Shuttle) (DB1)



**Fig. 5** The experimental results in average testing accuracy on medium dataset landsat satellite image (DB5)



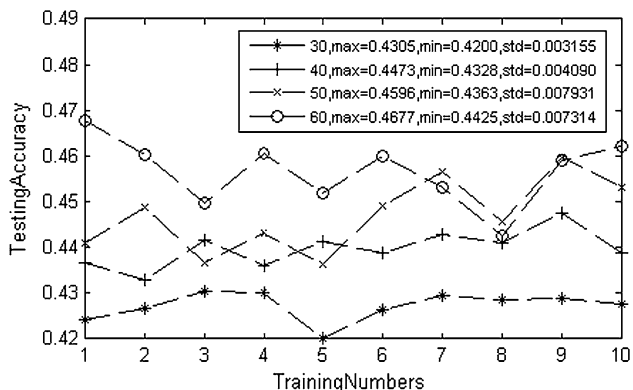
**Fig. 6** The experimental results in average testing accuracy on medium dataset optical recognition of handwritten digits (DB6)



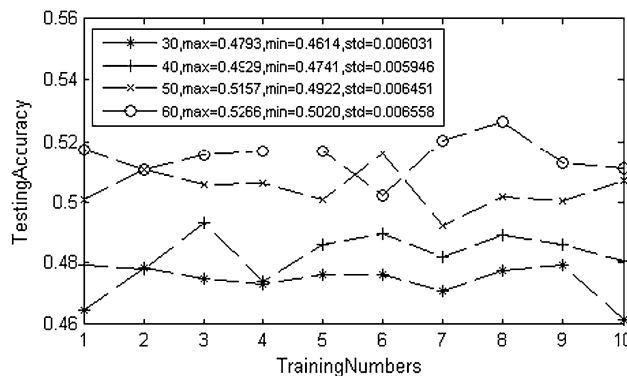
**Fig. 7** The experimental results in average testing accuracy on small dataset car evaluation dataset

number of hidden nodes are set to be 30, 40, 50, and 60. Figures 8 and 9 are the experimental results on large dataset letter recognition (DB2) with original ELM and DE-ELM. The curves that the testing accuracies change with different number nodes are given in six figures (from Figs. 8 to 13). From the fluctuating curves, we have observed following

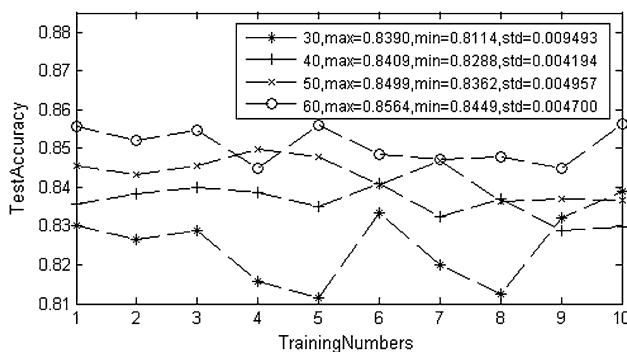
1. In all datasets, the testing accuracies of the two methods become higher and higher with the hidden nodes increasing from 30 to 60, and the accuracies of our method are always higher than that of original ELM.
2. In large dataset, our proposed method (Fig. 9) has more stable than original ELM (Fig. 8) when the number of hidden nodes is 50 or 60, which indicates that our proposed method has more stability than the original ELM with improvement of the testing accuracy.
3. In medium size of dataset, our proposed method has more stable than the original ELM in all four conditions (see Figs. 10, 11). For small dataset, it is same as medium size of dataset (see Figs. 12, 13).



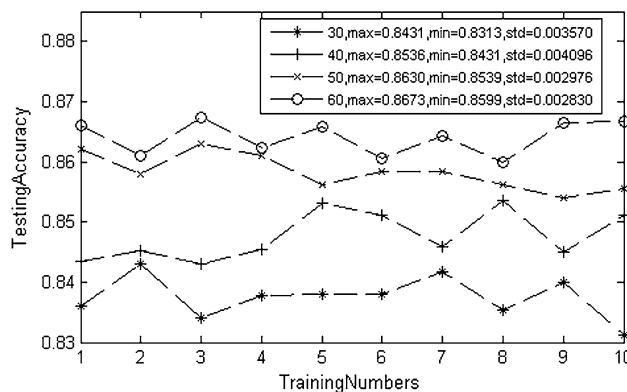
**Fig. 8** The experimental results in stability on large dataset letter recognition (DB2) with original ELM



**Fig. 9** The experimental results in stability on large dataset letter recognition (DB2) with DE-ELM



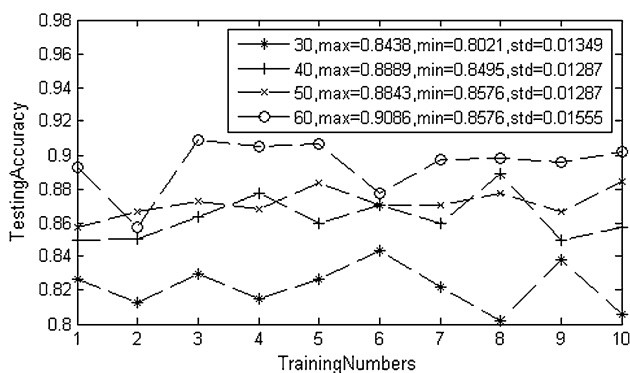
**Fig. 10** The experimental results in stability on medium dataset landsat satellite image (DB5) with original ELM



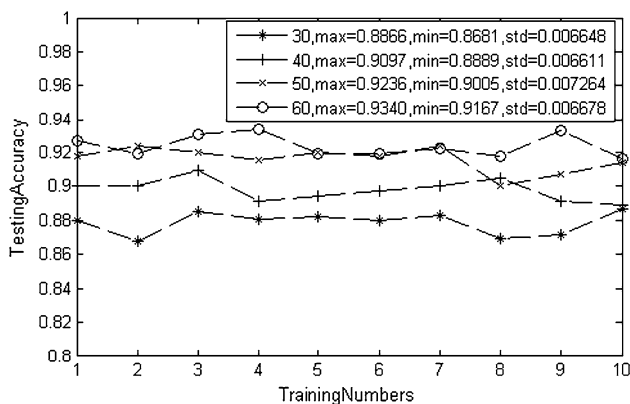
**Fig. 11** The experimental results in stability on medium dataset landsat satellite image (DB5) with DE-ELM

So, we conclude that our proposed method has more stability than original ELM

In order to further verify the effectiveness of our proposed method, we statistically analyze the experimental results using Wilcoxon test and paired *t* test (Demars 2006). First, for each dataset, we run original ELM and our method 10, 30, and 50 times and then obtain six statistics, which are denoted with  $X_i(1 \leq i \leq 3)$  and  $X'_i(1 \leq i \leq 3)$ ,



**Fig. 12** The experimental results in stability on small dataset car evaluation (DB8) with original ELM



**Fig. 13** The experimental results in stability on small dataset car evaluation (DB8) with original ELM

respectively, where  $X_i(1 \leq i \leq 3)$  is corresponding to original ELM,  $X'_i(1 \leq i \leq 3)$  is corresponding to our proposed method.  $X_1$  and  $X'_1$  are both 10-dimensional vectors,  $X_2$  and  $X'_2$  are both 30-dimensional vectors,  $X_3$  and  $X'_3$  are both 50-dimensional vectors. Next, we apply Wilcoxon test to the experimental results by computing the values of MATLAB function  $\text{ranksum}(X_1, X'_1)$ ,  $\text{ranksum}(X_2, X'_2)$ , and  $\text{ranksum}(X_3, X'_3)$ . Similarly, applying paired  $t$  test to

the experimental results by computing the values of MATLAB function  $ttest2(X_1, X'_1)$ ,  $ttest2(X_2, X'_2)$  and  $ttest2(X_3, X'_3)$ . The  $p$  values and  $h$  values of Wilcoxon test are listed in Table 3. The  $p$  values of paired  $t$  test are listed in Table 4.

The small  $p$  values cast doubt on the validity of the null hypothesis. Statistically, the  $p$  values of Wilcoxon test and paired  $t$  test (shown in Tables 3, 4) verified the effectiveness of our proposed method, and the  $h$  values of Wilcoxon test further certify that our method outperform original ELM.

Moreover, we statistically analyze the stability of our method with coefficient of variation (CV) of test accuracy. In our experiments, the numbers of hidden nodes are 30, 40, 50, and 60, respectively. The coefficient of variation is calculated with the following formula.

$$CV = \frac{\sigma}{\mu}$$

where  $\sigma = (\frac{1}{n-1} \sum_{i=1}^N (x_i - \mu)^2)^{\frac{1}{2}}$  and  $\mu = \frac{1}{n} \sum_{i=1}^n x_i$ . The results are listed in Table 5, where CV value 1 and CV value 2 are corresponding to our method and original ELM. The smaller CV value is the better stability the method has. The results listed in Table 5 certify that our method is superior to original ELM in stability to some extent.

### 5 Conclusions

In this paper, based on sample entropy the DE-ELM is proposed, which use AdaBoost to train  $N$  SLFNs trained with ELM as  $N$  base classifiers and classify a new sample by strategy of dynamic ensemble. The DE-ELM not only can overcome many shortages in the traditional gradient-based learning algorithm such as local minimal, improper learning rate, and low learning speed but also can alleviate the problems of instability and over-fitting in original ELM, and increase the prediction accuracy. The experimental results show that the proposed approach is robust and efficient.

**Table 3**  $p$  values and  $h$  values of Wilcoxon test

DB#	10 times		30 times		50 times	
	$p$ value 1	$h$ values 1	$p$ value 2	$h$ values 2	$p$ value 3	$h$ values 3
DB1	3.30E-04	1	1.10E-03	1	3.67E-06	1
DB2	8.00E-05	1	3.01E-11	1	2.11E-06	1
DB3	7.20E-03	1	1.12E-01	0	3.55E-02	1
DB4	1.30E-03	1	1.25E-07	1	5.83E-13	1
DB5	1.83E-04	1	2.97E-11	1	6.92E-18	1
DB6	1.82E-04	1	1.01E-10	1	7.01E-18	1
DB7	1.79E-04	1	2.96E-11	1	3.11E-17	1
DB8	1.70E-03	1	3.70E-10	1	1.23E-15	1



**Table 4**  $p$  values of paired  $t$  test

DB#	10 times $p$ value 1	30 times $p$ value 2	50 times $p$ value 3
DB1	2.04E-05	1.36E-04	8.83E-07
DB2	5.50E-15	1.22E-31	3.64E-11
DB3	4.87E-03	4.07E-02	7.32E-03
DB4	7.42E-03	1.24E-07	1.53E-13
DB5	1.95E-07	7.40E-29	2.19E-47
DB6	4.19E-08	2.11E-25	2.64E-38
DB7	1.85E-13	2.17E-27	2.37E-12
DB8	2.00E-04	3.70E-15	8.29E-24

**Table 5** Comparison of stability between our method and original ELM

DB#	30 nodes		40 nodes		50 nodes		60 nodes	
	CV value 1	CV value 2	CV value 1	CV value 2	CV value 1	CV value 2	CV value 1	CV value 2
DB1	0.0043	0.0115	0.0048	0.0050	0.0035	0.0058	0.0033	0.0055
DB2	0.0074	0.0127	0.0093	0.0123	0.0177	0.0128	0.0160	0.0127
DB3	0.0038	0.0043	0.0029	0.0029	0.0013	0.0034	0.0029	0.0031
DB4	0.0033	0.0064	0.0015	0.0045	0.0011	0.0027	0.0018	0.0049
DB5	0.0043	0.0115	0.0048	0.0050	0.0030	0.0049	0.0033	0.0055
DB6	0.0059	0.0129	0.0069	0.0126	0.0043	0.0136	0.0044	0.0079
DB7	0.0066	0.0089	0.0035	0.0097	0.0039	0.0078	0.0027	0.0091
DB8	0.0076	0.0164	0.0074	0.0149	0.0079	0.0092	0.0072	0.0174

**Acknowledgments** This research is supported by the national natural science foundation of China (61170040), by the natural science foundation of Hebei Province (F2010000323, F2011201063), by the Key Scientific Research Foundation of Education Department of Hebei Province (ZD2010139), and by the natural science foundation of Hebei University (2011-228).

## References

- Biggio B, Fumera G, Roli F (2010) Multiple classifier systems for robust classifier design in adversarial environments. *Int J Mach Learn Cybern* 1(1-4):27-41
- Breiman L (1996) Bagging predictors. *Mach Learn* 6(2):123-140
- Brown G, Wyatt J, Harris R, Yao X (2005) Diversity creation methods: a survey and categorisation. *Inf Fusion* 6(1):5-20
- Chacko BP, Vimal Krishnan VR, Raju G et al (2011) Handwritten character recognition using wavelet energy and extreme learning machine. *Int J Mach Learn Cybern*. doi:10.1007/s13042-011-0049-5
- Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7(1):1-30
- Emilio SO, Juan GS, Martín JD et al (2011) BELM: Bayesian extreme learning machine. *IEEE Trans Neural Netw* 22(3):505-509
- Feng GR, Huang GB, Lin QP, Gay R (2009) Error minimized extreme learning machine with growth of hidden nodes and incremental learning. *IEEE Trans Neural Netw* 20(8):1352-1357
- Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci* 55(1):119-139
- Hansen LK, Salamon P (1990) Neural network ensembles. *IEEE Trans Pattern Anal Mach Intell* 12(10):993-1001
- Huang GB, Chen L (2007) Convex incremental extreme learning machine. *Neurocomputing* 70(16-18):3056-3062
- Huang GB, Zhu QY, Siew CK (2006a) Extreme learning machine: theory and applications. *Neurocomputing* 70(1-3):489-501
- Huang GB, Chen L, Siew CK (2006b) Universal approximation using incremental constructive feed forward networks with random hidden nodes. *IEEE Trans Neural Netw* 17(4):879-892
- Huang GB, Ding XJ, Zhou HM (2010) Optimization method based extreme learning machine for classification. *Neurocomputing* 74(1-3):155-163
- Huang GB, Wang DH, Lan Y (2011) Extreme learning machines: a survey. *Int J Mach Learn Cybern* 2(2):107-122
- José MM, Pablo EM, Emilio SO et al (2011) Regularized extreme learning machine for regression problems. *Neurocomputing* 74(17):3716-3721
- Kittler J, Hatef M, Duin RPW et al (1998) On combining classifiers. *IEEE Trans Pattern Anal Mach Intell* 20(3):226-239
- Ko AHR, Sabourin R, Britto AS (2008) From dynamic classifier selection to dynamic ensemble selection [J]. *Pattern Recogn* 41(5):1718-1731
- Kuncheva LI (2001) Combining classifiers: soft computing solutions. In: Pal SK, Pal A (eds) *Pattern recognition: from classical to modern approaches*, World Scientific, pp 427-451
- Kuncheva LI, Whitaker CJ (2003) Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Mach Learn* 51(2):181-207
- Liang NY, Huang GB, Saratchandran P, Sundararajan N (2006) A fast and accurate online sequential learning algorithm for feed forward networks. *IEEE Trans Neural Netw* 17(6):1411-1423

- Liu N, Wang H (2010) Ensemble based extreme learning machine. *IEEE Signal Process Lett* 17(8):754–757
- Mao S, Jiao LC, Xiong L, Gou SP (2011) Greedy optimization classifiers ensemble based on diversity. *Pattern Recogn* 44(6): 1245–1261
- Mohammed AA, Minhas R, Jonathan QM et al (2011) Human face recognition based on multidimensional PCA and extreme learning machine. *Pattern Recogn* 44(10–11):2588–2597
- Rogova G (1994) Combining the results of several neural network classifiers. *Neural Netw* 7(5):777–781
- Schapire RE (1990) The strength of weak learnability. *Mach Learn* 5(2):197–227
- Serre D (2002) *Matrices: theory and applications*. Springer, New York
- Wang XZ, Dong CR (2009) Improving generalization of fuzzy if-then rules by maximizing fuzzy entropy. *IEEE Trans Fuzzy Syst* 17(3):556–567
- Wang GT, Li P (2010) Dynamic Adaboost ensemble extreme learning machine. In: 3rd international conference on advanced computer theory and engineering (ICACTE), vol 3, pp 54–58
- Wang XZ, Zhai JH, Lu SX (2008) Induction of multiple fuzzy decision trees based on rough set technique. *Inf Sci* 178(16): 3188–3202
- Wang X, Chen A, Feng H (2011a) Upper integral network with extreme learning mechanism. *Neurocomputing* 74(16):2520–2525
- Wang YG, Cao FL, Yuan YB (2011b) A study on effectiveness of extreme learning machine. *Neurocomputing* 74(16):2483–2490
- Wang XZ, Dong LC, Yan JH (2011c) Maximum ambiguity based sample selection in fuzzy decision tree induction. *IEEE Trans Knowl Data Eng*. doi:[10.1109/TKDE.2011.67](https://doi.org/10.1109/TKDE.2011.67)
- Woods K, Kegelmeyer WP, Bowyer K (1997) Combination of multiple classifiers using local accuracy estimates. *IEEE Trans Pattern Anal Mach Intell* 19(4):405–410
- Wu J, Wang S, Fu-lai C (2011) Positive and negative fuzzy rule system, extreme learning machine and image classification. *Int J Mach Learn Cybern* 2(4):261–271
- Wu X, Kumar V, Quinlan JR et al (2008) Top 10 algorithms in data mining. *Knowl Inf Syst* 14(1):1–37
- Zhang S, McCullagh P, Nugent C, Zheng H, Baumgarten M (2011) Optimal model selection for posture recognition in home-based healthcare. *Int J Mach Learn Cybern* 1(2):1–14
- Zhou ZH, Wu J, Tang W (2002) Ensembling neural networks: many could be better than all. *Artif Intell* 137(1–2):239–263