Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 808292, 7 pages
http://dx.doi.org/10.1155/2014/808292

Research Article

Evolutionary Voting-Based Extreme Learning Machines

Nan Liu,[1] Jiuwen Cao,[2] Zhiping Lin,[3] Pin Pin Pek,[1] Zhi Xiong Koh,[1] and Marcus Eng Hock Ong[1,4]

[1] *Department of Emergency Medicine, Singapore General Hospital, Singapore 169608*
[2] *Institute of Information and Control, Hangzhou Dianzi University, Zhejiang 310018, China*
[3] *School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798*
[4] *Health Services and Systems Research, Duke-NUS Graduate Medical School, Singapore 169857*

Correspondence should be addressed to Nan Liu; liu.nan@sgh.com.sg

Received 13 June 2014; Accepted 29 July 2014; Published 14 August 2014

Academic Editor: Tao Chen

Voting-based extreme learning machine (V-ELM) was proposed to improve learning efficiency where majority voting was employed. V-ELM assumes that all individual classifiers contribute equally to the decision ensemble. However, in many real-world scenarios, this assumption does not work well. In this paper, we aim to enhance V-ELM by introducing weights to distinguish the importance of each individual ELM classifier in decision making. Genetic algorithm is used for optimizing these weights. This evolutionary V-ELM is named as EV-ELM. Results on several benchmark databases show that EV-ELM achieves the highest classification accuracy compared with V-ELM and ELM.

1. Introduction

The extreme learning machine (ELM) [1, 2] was proposed as a new type of learning algorithm for single-hidden layer feedforward neural network (SLFN). ELM adopts a training mechanism that does not require parameter tuning so that it learns faster than traditional gradient-based neural networks while achieving even better classification/regression performance [3]. ELM has received increasing attentions in recent years; many efforts have been dedicated to improve its performance [4–7] and apply it in various applications [8–14]. Among these ELM extensions, ensemble learning-based methods [15–23] have found their advantages such as better accuracy and less variance compared with the original ELM algorithm.

Ensemble learning has been popular for decades. Jain et al. [24] wrote a concise yet informative introduction to the classifier combination. Polikar [25] comprehensively reviewed the area of multiple classifier system (ensemble system) for decision making. Majority voting [26] is one of the most commonly used combining strategies. This rule seeks the class that receives the highest number of votes and assigns it to the predicted label for the testing pattern. Since each classifier in an ensemble does not necessarily contribute equally to the final decision, Littlestone and Warmuth [27] proposed weighing individual classifiers to make them discriminative. Ensemble learning-based ELM algorithms [16, 18, 19] were reported to successfully resolve the problems of predictive instability and overfitting.

Among ensemble-based extensions, voting-based ELM (V-ELM) [18] was proposed to perform multiple independent ELM training using a simple and effective learning architecture. A decision was made based on majority voting. V-ELM not only enhanced the classification performance, but also lowered the variance. In this paper, we aim to investigate the introduction of weights for all individual ELM classifiers to enhance the V-ELM algorithm. The hypothesis is that each individual ELM classifier presents various levels of confidence in decision making. In our proposed method, each weight represents the importance of an ELM classifier. Final decision is made with the weighted majority voting scheme.

The remainder of this paper is organized as follows. Section 2 briefly reviews ELM and V-ELM algorithms. Section 3 presents the proposed evolutionary V-ELM (EV-ELM) algorithm. Section 4 demonstrates the performance of

Inputs: Training samples $L = \{(\mathbf{x}_n, y_n)\}, n = 1, \ldots, N$ with labels $y_n \in \{1, \ldots, C\}$; $P(t)$: Population of candidates at generation t; p: Population size of each generation; p_c: Crossover probability; p_m: Mutation probability
Initialization: Set $t = 0$ and initialize the population $P(t)$ at random.
Evolutionary Process: Evaluate the fitness values of $P(t)$ using (5). Increase t with 1 in each iteration. To create a new generation $P(t+1)$, operations of selection, crossover and mutation on $P(t)$ are used. Repeat the following steps until the termination criteria of genetic algorithm (GA) is met.
(i) Firstly, $(1 - p_c)p$ members of $P(t)$ are probabilistically selected to $P(t+1)$ according to the fitness.
(ii) Secondly, the crossover operator is applied to half of not selected candidates in $P(t)$. The offsprings after crossover are added to $P(t+1)$.
(iii) Lastly, a number of chromosomes with a probability of p_m in $P(t+1)$ are subject to mutation.
Store weights $\{\omega_1^{\text{opt}}, \ldots, \omega_K^{\text{opt}}\}$ as the outputs where "opt" indicates "optimal".
Decision Making: Given a testing sample $\mathbf{x}$, use the following equation to predict its label

$$\varphi^{\omega}(\mathbf{x}, L) = \arg \max_{j=1}^{C} \sum_{k=1}^{K} \omega_k^{\text{opt}} D_{k,j}$$

Algorithm 1: Evolutionary voting based ELM.

EV-ELM and compares it with ELM and V-ELM. Section 5 concludes this study.

2. Background

2.1. Extreme Learning Machine. In the process of SLFN learning, ELM randomly selects weights and biases for hidden nodes. Then it analytically determines the output weights by finding the least squares solution. Given a training set consisting of N samples $L = \{(\mathbf{x}_n, \mathbf{t}_n) \mid \mathbf{x}_n \in \mathbf{R}^u, \mathbf{t}_n \in \mathbf{R}^v, n = 1, 2, ..., N\}$, where $\mathbf{x}_n$ is an $u \times 1$ input vector and $\mathbf{t}_n$ is an $v \times 1$ target vector, an SLFN with $\widetilde{N}$ hidden nodes is formulated as

$$f_{\widetilde{N}}(\mathbf{x}_n) = \sum_{i=1}^{\widetilde{N}} \boldsymbol{\beta}_i g(\mathbf{w}_i \cdot \mathbf{x}_n + b_i) = \mathbf{t}_n, \quad n = 1, 2, \ldots, N, \tag{1}$$

where the additive hidden node is employed. Weight vector $\mathbf{w}_i$ connects the ith hidden node and input neurons. In approximating N samples using $\widetilde{N}$ hidden nodes, $\boldsymbol{\beta}_i$, $\mathbf{w}_i$, and b_i are supposed to exist if zero error is obtained. Consequently, (1) can be written as

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

where $\mathbf{H}(\mathbf{w}_1, \ldots, \mathbf{w}_{\widetilde{N}}, b_1, \ldots, b_{\widetilde{N}}, \mathbf{x}_1, \ldots, \mathbf{x}_N)$ is hidden layer output matrix of the network; $h_{ni} = g(\mathbf{w}_i \cdot \mathbf{x}_n + b_i)$ is the output of ith hidden neuron with respect to $\mathbf{x}_n$, $i = 1, 2, \ldots, \widetilde{N}$ and $n = 1, 2, \ldots, N$; $\widehat{\boldsymbol{\beta}} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{\widetilde{N}}]^{\mathrm{T}}$ and $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]^{\mathrm{T}}$ are output weight matrix and target matrix, respectively.

The ELM algorithm can be summarized as three steps: (1) generate parameters $\mathbf{w}_i$ and b_i randomly for $i = 1, \ldots, \widetilde{N}$; (2) calculate the hidden layer output matrix $\mathbf{H}$; and (3) calculate the output weight using $\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$. It has been shown in [28] that any continuous target functions in $\mathbf{R}^n$ can be universally approximated using single SLFN with randomly chosen additive hidden nodes.

2.2. Voting-Based Extreme Learning Machine. In ELM, randomized hidden nodes are used and remain unchanged during the training. Some testing samples could be misclassified in certain situations, for example, when they are near the classification boundary. To tackle this issue, V-ELM incorporates multiple individual ELMs and makes decisions with majority voting. V-ELM uses a fixed number of hidden nodes for all individual ELMs. All these ELMs are trained with the same dataset and the learning parameters of each ELM are randomly initialized. The predicted class label is then determined by majority voting on all results obtained from ELMs.

3. Evolutionary Voting-Based ELM

Given a learning set L consisting of samples $\{(\mathbf{x}_n, y_n)\}$, $n = 1, 2, \ldots, N$, where y_n is the class label. We assume that $\mathbf{x}$ is the input and y is predicted by $\varphi(\mathbf{x}, L)$. In V-ELM, the aim is to better predict y using multiple ELMs than a single one. Suppose that $\varphi(\mathbf{x}, L)$ predicts a class label $j \in \{1, 2, \ldots, C\}$ and the prediction of kth classifier is $D_{k,j} \in \{0, 1\}$ where $k = 1, 2, \ldots, K$, the ensemble decision can be defined as

$$\varphi(\mathbf{x}, L) = \arg \max_{j=1}^{C} \sum_{k=1}^{K} D_{k,j}. \tag{3}$$

The voting is plurality version which means that the output is the value with highest number of votes whether or not the sum of votes exceeds half.

In many applications, not all the classifiers contribute equally to decision making. The overall performance of the ensemble system is able to be improved by weighing the decisions prior to combination [27]. In this section, an evolutionary voting-based ELM (EV-ELM) using weighted majority voting is proposed. The general algorithm is elaborated in Algorithm 1.

Table 1: Databases used in the experiments.

	Database	Number of training data	Number of testing data	Number of attributes	Number of classes
UCI	Balance	400	225	4	3
	Diabetes	576	192	8	2
	Digit	7494	3498	16	10
	Hayes	132	28	4	3
	Heart	100	170	13	2
	Iris	100	50	4	3
	Monk1	124	432	6	2
	Monk2	169	432	6	2
	Monk3	122	432	6	2
	Sonar	100	108	60	2
	Waveform	3000	2000	21	3
	Wine	100	78	13	3
Face	Combo	555	575	81	75
	FERET	1280	1433	81	320
	GTFD	400	350	81	50

We denote ω_k as the weights for kth individual ELM. The mathematical representation of weighted voting algorithm is shown as

$$\varphi^{\omega}(\mathbf{x}, L) = \arg\max_{j=1}^{C} \sum_{k=1}^{K} \omega_k D_{k,j}. \tag{4}$$

In the framework of weighted majority voting algorithm, the weight ω_k needs to be optimized to improve the generalization performance. If we know certain classifiers working better, we are able to assign larger weights to the corresponding ones. However, such knowledge is usually absent.

Conventional parameter updating methods find the optimal weights to provide better generalization performance. But the optimization process has the risk of getting the local minima and maxima. Methods that discover global optimum can be implemented to further improve classification accuracy. Genetic algorithm (GA) [29] is a class of optimization procedures inspired by the biological mechanisms of reproduction. Many applications utilize the advantages of GA to find optimal solutions, for example, face recognition [30] and clustering techniques [31]. GA is implemented in this paper for demonstration purpose. In practice, many new emerging techniques are potential alternatives such as differential evolution [32] and particle swarm optimization [33].

In order to use GA to select proper weights, the chromosomes are formed by $\omega_1, \omega_2, \ldots, \omega_K$. At the beginning, a population of N_c chromosomes is generated randomly. Then, the fitness function f is calculated for each chromosome as

$$f = \frac{\sum_{k=1}^{K} \omega_k A_k}{\sum_{k=1}^{K} \omega_k}, \tag{5}$$

where ω_k and A_k are the weight and training accuracy for the kth ELM. Maximizing the fitness implies that by choosing appropriate weights we are able to achieve the best normalized training accuracy across all K ELM classifiers in the ensemble. Such a set of optimal weights provide the decision ensemble good generalization performance on unseen testing samples. The fitness function is the most important measurement to determine the composition of the next generation and to guide the entire evolutionary process.

After chromosome selection, parts of the current population are inherited into the next generation. The remaining strings are reproduced and some of the parent chromosomes have undergone crossover operation. The crossover probability is defined as p_c. To extend the search space to find global optimum, mutation is applied to some of offsprings randomly with a very small mutation probability p_m. The mutation will introduce a degree of diversity to the population and prevent a premature convergence. The evolutionary process will be terminated after N_g generations, which are considered enough for convergence.

4. Performance Evaluation

Experiments were carried out in MATLAB 7 environment under a desktop equipped with Intel 3.2 GHz CPU and 4 G RAM. The learning and testing processes were repeated 50 times and the mean and standard deviations were reported in results. In the experiments, the range of weight was $[\exp(0), \exp(1)]$ where the exponential function was used to enhance the difference between lower-bound and upper-bound values of the weight. Moreover, the crossover probability p_c and mutation probability p_m were chosen as 0.65 and 0.004, respectively. The population size N_c was 100 and the number of generations in GA was 150 for all experiments.

4.1. Databases. We evaluated the proposed EV-ELM with two types of data: UCI machine learning data and face recognition data. The UCI data was used to test the methods in general-purpose classification problems; the face data was able to examine how good the methods were at handling data with the problem of small sample size (i.e., each pattern class had

Table 2: Comparison results among ELM, V-ELM, and EV-ELM algorithms using benchmark UCI and face databases.

	Database	Algorithms	Number of nodes	Number of ensembles	Training time (s)	Testing accuracy (%)	Standard deviation (%)
UCI	Balance	ELM	100	NA	0.06	89.56	1.35
		V-ELM	100	20	1.22	90.24	0.68
		EV-ELM	100	20	3.16	**90.84**	**0.67**
	Diabetes	ELM	30	NA	0.06	76.65	1.73
		V-ELM	30	20	0.35	78.67	0.69
		EV-ELM	30	20	2.27	**79.18**	**0.66**
	Digit	ELM	200	NA	1.15	96.98	0.23
		V-ELM	200	20	28.51	97.18	0.08
		EV-ELM	200	20	30.77	**97.30**	**0.07**
	Hayes	ELM	60	NA	<0.01	74.64	5.63
		V-ELM	60	20	0.41	78.42	3.58
		EV-ELM	60	20	2.51	**79.93**	**3.21**
	Heart	ELM	20	NA	<0.01	79.01	2.82
		V-ELM	20	20	0.19	82.14	1.25
		EV-ELM	20	20	1.99	**82.78**	**1.12**
	Iris	ELM	20	NA	<0.01	97.20	1.39
		V-ELM	20	20	0.12	98.00	0.70
		EV-ELM	20	20	2.16	**98.00**	**0.39**
	Monk1	ELM	100	NA	0.05	78.15	2.63
		V-ELM	100	20	0.79	85.79	1.37
		EV-ELM	100	20	2.82	**87.23**	**1.17**
	Monk2	ELM	100	NA	0.06	78.84	2.07
		V-ELM	100	20	0.91	83.51	0.82
		EV-ELM	100	20	3.21	**83.81**	**0.75**
	Monk3	ELM	100	NA	0.06	80.62	3.13
		V-ELM	100	20	0.68	89.00	1.03
		EV-ELM	100	20	2.68	**89.48**	**0.97**
	Sonar	ELM	60	NA	<0.01	77.25	3.44
		V-ELM	60	20	0.29	86.79	1.95
		EV-ELM	60	20	2.23	**87.17**	**1.77**
	Waveform	ELM	200	NA	0.44	84.87	0.52
		V-ELM	200	20	12.71	86.44	0.21
		EV-ELM	200	20	14.77	**86.60**	**0.19**
	Wine	ELM	20	NA	<0.01	97.31	1.50
		V-ELM	20	20	0.12	98.89	0.85
		EV-ELM	20	20	2.21	**99.36**	**0.69**
Face	Combo	ELM	200	NA	0.21	84.88	1.06
		V-ELM	200	20	5.16	87.80	0.60
		EV-ELM	200	20	7.58	**88.42**	**0.59**
	FERET	ELM	100	NA	0.12	41.85	0.80
		V-ELM	100	20	7.19	49.06	0.49
		EV-ELM	100	20	13.95	**49.20**	**0.47**
	GTFD	ELM	200	NA	0.19	54.86	2.01
		V-ELM	200	20	4.60	66.91	1.39
		EV-ELM	200	20	6.77	**67.14**	**1.26**

only a few samples). Details of these databases are presented in Table 1. A total of 12 real world datasets were downloaded from the UCI database [34]. Five benchmark face databases were used, namely, ORL [35], UMIST [36], Yale [37], FERET [38], and Georgia Tech face database (GTFD) [39]. ORL, UMIST, and Yale formed a combo database for testing. The combo set consisted of training samples and 575 testing samples in total, and all images belonged to 75 different classes with large variations of illumination, poses, and facial expressions. The FERET database used in this paper was

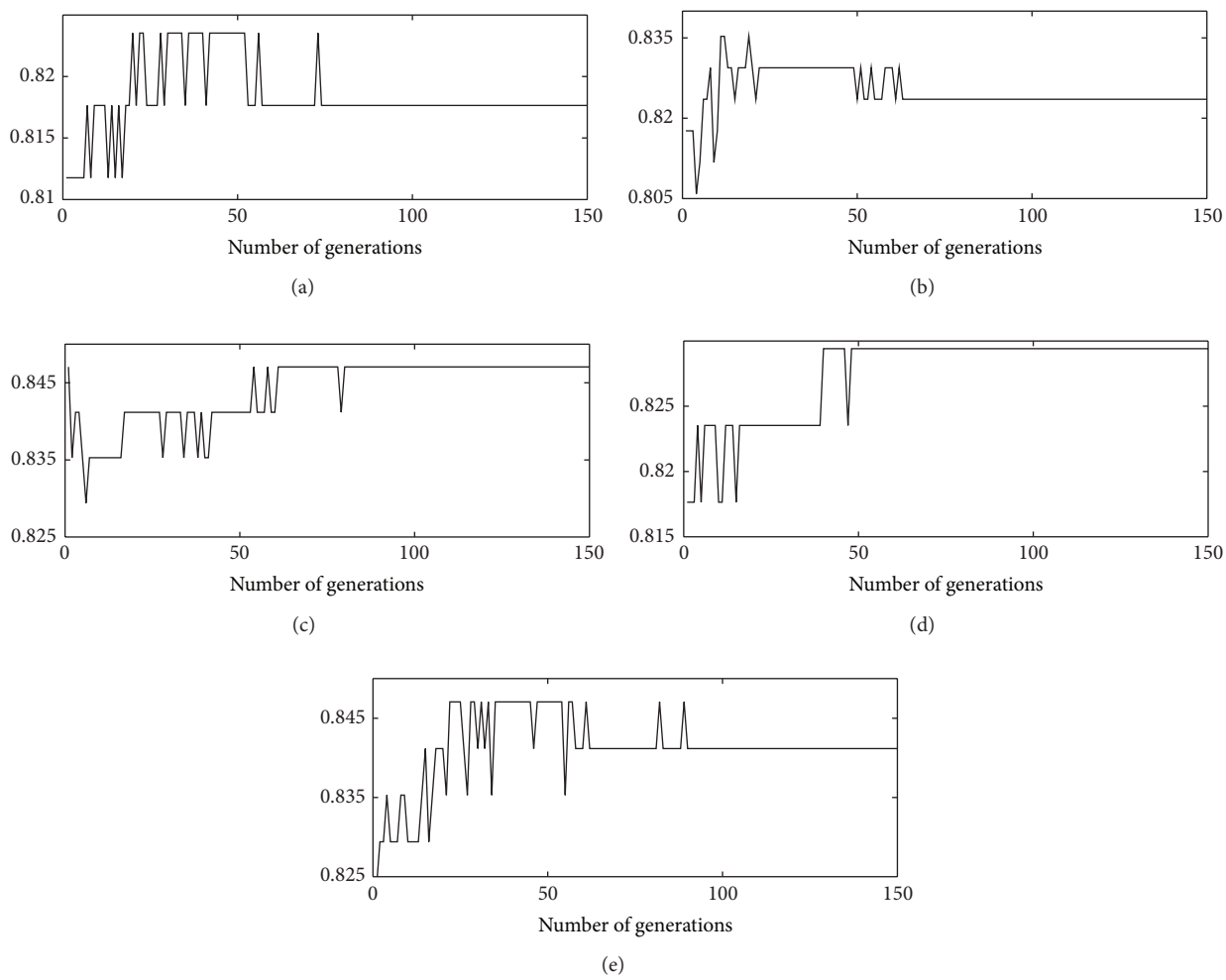

Figure 1: Five examples show the changes in classification performance during the evolutionary process. The x-axis is the number of generations and the y-axis is classification accuracy during each generation. The results are based on UCI Heart dataset.

a preprocessed subset [40] composed of 2713 images from 320 subjects. In GTFD, each of 50 subjects had 15 images. Before the experimental evaluation, we cropped and resized images in Yale and GTFD databases to 112×92 to make their dimensions identical to those of samples in ORL and UMIST. Furthermore, we applied the discrete cosine transform (DCT) [41] to convert 2D face images to low-dimensional vectors of DCT coefficients.

4.2. Results and Discussion. Table 2 presents the comparison results where both training time and testing accuracy are averaged across 50 repeats of the evaluation process. It is shown that ELM is the fastest learner but performs poorly in classification. V-ELM and EV-ELM achieve much better performance. Since both V-ELM and EV-ELM create an ensemble of individual ELM classifiers, they run slower compared with ELM unless a parallel computing structure is implemented. In all databases, EV-ELM outperforms V-ELM in terms of both accuracy and variance. However, this improvement is not as much as that between voting-based methods and the original ELM algorithm. The results also show that the evolutionary weighing method is able to increase classification accuracy while bringing down the variance. In general, EV-ELM needs more time than V-ELM to train a model. When in an application that online training is not required, EV-ELM is a good alternative to V-ELM.

Figure 1 illustrates five examples on the changes in classification performance during the evolutionary process. Within 150 generations, the GA process is usually able to converge. Reflected in the figure, the classification accuracy tends to become stable toward the end of the evolutionary process. The selected weights after 150 generations cannot guarantee achieving the best classification accuracy, but they are able to provide a consistent output. This characteristic is important as testing accuracy is not available during classifier training, and it cannot be used to guide the evolutionary process. Figure 2 depicts the changes of 3 weights as examples during the GA evolution. Initially, these weights are randomly generated. With the increasing of generations, they tend to converge to either upper- or lower-bound values so that the fitness is maximized. The variety among the weights creates a dynamic ELM ensemble for decision making.

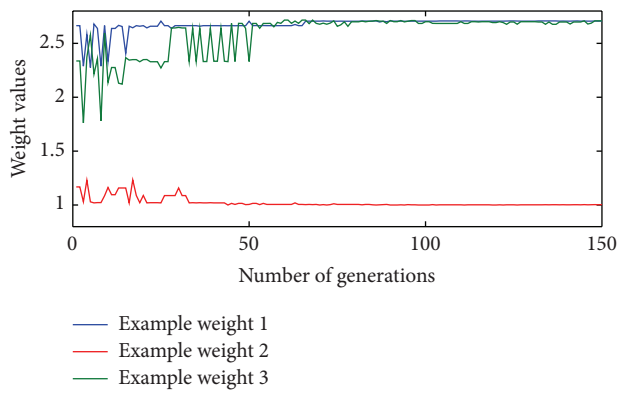

Figure 2: There example weights and their value changes in value changes during the evolutionary process. The results are based on UCI Heart dataset.

5. Conclusions

In this paper, we proposed an enhanced V-ELM method. Weights were introduced to distinguish the difference among various individual ELM classifiers and the genetic algorithm was used for optimization. Experimental results demonstrated the effectiveness of EV-ELM in terms of classification accuracy. However, slow training speed prohibits the use of EV-ELM in applications that require online training. This study is a preliminary research on optimizing ELM ensembles; many evolutionary algorithms are potentially useful in optimizing weights. Furthermore, reducing training time is of great interest in future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors would like to thank the editor and reviewers for the constructive comments and suggestions.

References

[1] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.

[2] G. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.

[3] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[4] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, "OP-ELM: optimally pruned extreme learning machine," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 158–162, 2010.

[5] Q.-Y. Zhu, A. K. Qin, P. N. Suganthan, and G.-B. Huang, "Evolutionary extreme learning machine," *Pattern Recognition*, vol. 38, no. 10, pp. 1759–1763, 2005.

[6] Z. Sun, K.-F. Au, and T.-M. Choi, "A neuro-fuzzy inference system through integration of fuzzy logic and extreme learning machines," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 37, no. 5, pp. 1321–1331, 2007.

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

[8] R. Zhang, G. Huang, N. Sundararajan, and P. Saratchandran, "Multicategory classification using an extreme learning machine for microarray gene expression cancer diagnosis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, no. 3, pp. 485–495, 2007.

[9] Z.-L. Sun, T.-M. Choi, K.-F. Au, and Y. Yu, "Sales forecasting using extreme learning machine with applications in fashion retailing," *Decision Support Systems*, vol. 46, no. 1, pp. 411–419, 2008.

[10] N. Liu and H. Wang, "Evolutionary extreme learning machine and its application to image analysis," *Journal of Signal Processing Systems*, vol. 73, pp. 1–9, 2013.

[11] S. Suresh, R. V. Babu, and H. J. Kim, "No-reference image quality assessment using modified extreme learning machine classifier," *Applied Soft Computing*, vol. 9, no. 2, pp. 541–552, 2009.

[12] Y. Jin, J. Cao, Q. Ruan, and X. Wang, "Cross-modality 2D-3D face recognition via multiview smooth discriminant analysis based on ELM," *Journal of Electrical and Computer Engineering*, vol. 2014, Article ID 584241, 9 pages, 2014.

[13] L. Mao, L. Zhang, X. Liu, C. Li, and H. Yang, "Improved extreme learning machine and its application in image quality assessment," *Mathematical Problems in Engineering*, vol. 2014, Article ID 426152, 7 pages, 2014.

[14] J. Cao and L. Xiong, "Protein sequence classification with improved extreme learning machine algorithms," *BioMed Research International*, vol. 2014, Article ID 103054, 12 pages, 2014.

[15] Y. Lan, Y. C. Soh, and G.-B. Huang, "Ensemble of online sequential extreme learning machine," *Neurocomputing*, vol. 72, no. 13–15, pp. 3391–3395, 2009.

[16] N. Liu and H. Wang, "Ensemble based extreme learning machine," *IEEE Signal Processing Letters*, vol. 17, no. 8, pp. 754–757, 2010.

[17] M. Van Heeswijk, Y. Miche, E. Oja, and A. Lendasse, "GPU-accelerated and parallelized ELM ensembles for large-scale regression," *Neurocomputing*, vol. 74, no. 16, pp. 2430–2437, 2011.

[18] J. Cao, Z. Lin, G. Huang, and N. Liu, "Voting based extreme learning machine," *Information Sciences*, vol. 185, pp. 66–77, 2012.

[19] J.-H. Zhai, H.-Y. Xu, and X.-Z. Wang, "Dynamic ensemble extreme learning machine based on sample entropy," *Soft Computing*, vol. 16, no. 9, pp. 1493–1502, 2012.

[20] D. Wang and M. Alhamdoosh, "Evolutionary extreme learning machine ensembles with size control," *Neurocomputing*, vol. 102, pp. 98–110, 2013.

[21] Q. Yu, M. van Heeswijk, Y. Miche et al., "Ensemble delta test-extreme learning machine (DT-ELM) for regression," *Neurocomputing*, vol. 129, pp. 153–158, 2014.

[22] H.-J. Lu, C.-L. An, E.-H. Zheng, and Y. Lu, "Dissimilarity based ensemble of extreme learning machine for gene expression data classification," *Neurocomputing*, vol. 128, pp. 22–30, 2014.

[23] X. Xue, M. Yao, Z. Wu, and J. Yang, "Genetic ensemble of extreme learning machine," *Neurocomputing*, vol. 129, pp. 175–184, 2014.

[24] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[25] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–45, 2006.

[26] L. I. Kuncheva, *Combining Pattern Classifiers, Methods and Algorithms*, Wiley-Interscience, New York, NY, USA, 2005.

[27] N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," *Information and Computation*, vol. 108, no. 2, pp. 212–261, 1994.

[28] G. Huang, L. Chen, and C. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.

[29] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[30] C. Liu and H. Wechsler, "Evolutionary pursuit and its application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 570–582, 2000.

[31] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern Recognition*, vol. 33, no. 9, pp. 1455–1465, 2000.

[32] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, December 1995.

[34] K. Bache and M. Lichman, "UCI machine learning repository," 2013, http://archive.ics.uci.edu/ml.

[35] F. S. Samaria, *Face recognition using hidden markov models [Ph.D. thesis]*, University of Cambridge, Cambridge, UK, 1994.

[36] D. B. Graham and N. M. Allinson, "Characterizing virtual eigensignatures for general purpose face recognition," in *Face Recognition: From Theory to Applications*, H. Wechsler, P. J. Phillips, V. Bruce, F. Fogelman-Soulie, and T. S. Huang, Eds., vol. 163 of *NATO ASI Series F, Computer and Systems Sciences*, pp. 446–456, 1998.

[37] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[38] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1090–1104, 2000.

[39] L. Chen, H. Man, and A. V. Nefian, "Face recognition based on multi-class mapping of fisher scores," *Pattern Recognition*, vol. 38, no. 6, pp. 799–811, 2005.

[40] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Uncorrelated multilinear discriminant analysis with regularization and aggregation for tensor object recognition," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 103–123, 2009.

[41] W. Chen, M. J. Er, and S. Wu, "Illumination compensation and normalization for robust face recognition using discrete cosine transform in logarithm domain," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 36, no. 2, pp. 458–466, 2006.