Extreme Learning Machine for Multi-Categories Classification Applications

Hai-Jun Rong, Guang-Bin Huang, and Yew-Soon Ong

***Abstract*— In the paper, the multi-class pattern classification using extreme learning machine (ELM) is studied. The study is based on either a series of ELM binary classifiers or a single ELM classifier. When using binary ELM classifiers, the multi-class problem is decomposed into two-class problem using the one-against-all (OAA) and one-against-one (OAO) schemes, which are named as ELM-OAA and ELM-OAO respectively for brevity. In a single ELM classifier, the multi-class problem is implemented with an architecture of multi-output nodes which is equal to the number of pattern classes. Their performance is evaluated using some multi-class benchmark problems and simulation results show that ELM-OAA and ELM-OAO requires fewer hidden nodes than the single ELM classifier. In addition ELM-OAO usually has similar or less computation burden than the single ELM classifier when the pattern class labels is not larger than 10.**

I. Introduction

Recently, a new fast learning algorithm referred to as Extreme Learning Machine (ELM) with any hidden nodes has been developed for Single-Hidden Layer Feedforward Networks (SLFNs) in [1], [2], [3]. In ELM, one may randomly choose and fix the hidden node parameters. After the hidden nodes parameters are chosen randomly, SLFN becomes a linear system where the output weights of the network can be analytically determined using simple generalized inverse operation of the hidden layer output matrices. Compared with the traditional gradient-based learning algorithms, ELM not only learns much faster with higher generalization performance but also avoids many difficulties faced by gradient-based learning methods such as stopping criteria, learning rate, learning epochs, and local minima. ELM has been successfully applied to many real world applications [4], [5], [6], [7], [8], [9], [10], [11]. The universal approximation of ELM has been proved in Huang, et al [12], [13].

In the paper, the ELM algorithm is further studied for multi-class classification problem. The commonly used method for multi-class classification problem is to decompose the multi-class classification problem into multiple two-class problems according to the inherent class relations among the training data, two major class decomposition methods are one-against-all (OAA) and one-against-one (OAO) methods. Thus the multi-class classification problem studied in the paper using ELM is implemented by using the OAA and OAO decomposition schemes where each binary classifier is based on the ELM algorithm, for brevity, named as ELM-OAA and ELM-OAO. Meanwhile a single ELM classifier is studied in the paper to implement the multi-class classification problem with an architecture of multi-output nodes equal to the number of pattern classes.

H.-J. Rong and G.-B. Huang are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798. E-mail: {hjrong, egbhuang}@ntu.edu.sg

Y.-S. Ong is with the School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798. E-mail: asysong@ntu.edu.sg

The performances of the proposed ELM-OAA, ELM-OAO and ELM classifiers are evaluated based on some multi-class benchmark UCI problems [14]. The simulation results show the learning capabilities of the three classifiers are problem-dependent and determined by the number of training data and number of pattern classes. However ELM-OAA and ELM-OAO require smaller number of hidden nodes than the single ELM classifier.

This paper is organized as follows. Section II provides the details of multi-class classification problem using ELM. Section III presents a quantitative performance comparison of using a single ELM classifier and multiple binary ELM classifiers based on multi-class classification problems. Finally, Section IV summarizes the conclusions of the present study.

II. ELM for Multiple Class Recognition

In the section, single ELM classifier and multiple binary ELM classifiers based on OAA and OAO decomposition schemes are introduced.

A. ELM

A single ELM classifier implements multi-class classification problem using a network structure of multi-output nodes equal to the number of pattern classes m. The network output is $\mathbf{y} = (y_1, y_2, \cdots, y_m)^T$. For each training example $\mathbf{x}$, the target output $\mathbf{t}$ is coded into m bits: $(t_1, \cdots, t_m)^T$. For a pattern of class i, only the t_i is "1" and the rest is "-1".

The learning procedure of the single ELM classifier is given below. ELM was proposed in Huang, *et. al.* [3] and its key point is that the hidden node parameters $(\mathbf{c}_i, a_i)$ are randomly assigned instead of tuning. For N arbitrary distinct samples $(\mathbf{x}_k, \mathbf{t}_k) \in \mathbf{R}^n \times \mathbf{R}^m$, the single ELM classifier with $\tilde{N}$ hidden nodes becomes a linear system as,

$$\sum_{i=1}^{\tilde{N}} \boldsymbol{\beta}_i G(\mathbf{x}_k; \mathbf{c}_i, a_i) = \mathbf{t}_k, \ \ k = 1, \cdots, N. \qquad (1)$$

where $\mathbf{c}_i \in \mathbf{R}^n$ and $a_i \in R$ are the learning parameters of hidden nodes and randomly assigned, $\boldsymbol{\beta}_i$ is the weight connecting the ith hidden node to the output node, and $G(\mathbf{x}_k; \mathbf{c}_i, a_i)$ is the output of the ith hidden node with respect to the input $\mathbf{x}_k$. The matrix of output weights is defined as,

$$\boldsymbol{\beta} = \left[\begin{array}{ccc} \boldsymbol{\beta}_1^T & \cdots & \boldsymbol{\beta}_{\tilde{N}}^T \end{array} \right]_{m\times\tilde{N}}^T \quad (2)$$

Equation (1) can be written compactly as:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \quad (3)$$

where

$$\mathbf{H}(\mathbf{c}_1, \cdots, \mathbf{c}_{\tilde{N}}, a_1, \cdots, a_{\tilde{N}}, \mathbf{x}_1, \cdots, \mathbf{x}_N) = \left[\begin{array}{ccc} G(\mathbf{x}_1; \mathbf{c}_1, a_1) & \cdots & G(\mathbf{x}_1; \mathbf{c}_{\tilde{N}}, a_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ G(\mathbf{x}_N; \mathbf{c}_1, a_1) & \cdots & G(\mathbf{x}_N; \mathbf{c}_{\tilde{N}}, a_{\tilde{N}}) \end{array} \right]_{N\times\tilde{N}} \quad (4)$$

$$\mathbf{T} = \left[\begin{array}{ccc} \mathbf{t}_1^T & \cdots & \mathbf{t}_N^T \end{array} \right]_{m\times N}^T \quad (5)$$

The determination of the output weights $\boldsymbol{\beta}$ is as simple as finding the least-square solutions to the given linear system and given as,

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T} \quad (6)$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse [15] of the hidden layer output matrix $\mathbf{H}$.

The three-step simple learning algorithm named extreme learning machine (ELM) [1], [3], [12], [13] can be summarized as follows:

Algorithm ELM: Given a training set $\aleph = \{(\mathbf{x}_i, \mathbf{t}_i)|\mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^m, i = 1, \cdots, N\}$, the hidden node output function $G(\mathbf{x}; \mathbf{c}, a)$, and hidden node number $\tilde{N}$:

1) Randomly assign hidden node parameters $(\mathbf{c}_i, a_i)$, $i = 1, \cdots, \tilde{N}$.
2) Calculate the hidden layer output matrix $\mathbf{H}$.
3) Calculate the output weight $\boldsymbol{\beta}$: $\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$.

Thus for a pattern example $\mathbf{x}$, the ELM classifier produces m outputs ($\mathbf{y} = [y_1, y_2, \cdots, y_m]$) represented by,

$$\mathbf{y} = \sum_{i=1}^{\tilde{N}} \boldsymbol{\beta}_i G(\mathbf{x}; \mathbf{c}_i, a_i) \quad (7)$$

The class label which the pattern $\mathbf{x}$ belongs to is determined by the output node with the largest output value. Mathematically it is given as,

$$\hat{y}(\mathbf{x}) = \text{argmax}_{i=1,2,\cdots,m} y_i(\mathbf{x}) \quad (8)$$

The MATLAB codes of the single ELM classifier can be found in the ELM web portal: http://www.ntu.edu.sg/home/egbhuang/.

B. ELM-OAA

In the ELM-OAA method, the m-class classification problem is implemented by m binary ELM classifiers, each of which is trained independently to classify one of the m pattern classes with the training data relevant to each class. The input data for each binary ELM classifier is the same but the target output data is different. For the qth binary ELM classifier, ELMq, the target output data need to be decomposed to two subsets: one labeled as "1" for all the qth class examples and the other labeled as "-1" for all the examples belonging to all other classes. Each binary ELM classifier ELMq has one output node with output value $y_q (q = 1, 2, \cdots, m)$. When a pattern example $\mathbf{x}$ is input, the ELM-OAA produces the corresponding m output values from the m binary ELM classifiers and thus a decision function is required to integrate the m output values to produce the final class label output $\hat{y}$ which the pattern $\mathbf{x}$ belongs to.

C. ELM-OAO

In the ELM-OAO, the m pattern classes are pairwise decomposed into $m(m-1)/2$ two different classes, each of which is trained by one ELM classifier (ELM$q(j, s)$, $q = 1, 2, \cdots, r (r = m(m-1)/2)$; $j = 1, 2, \cdots, m-1$; $s = j+1, \cdots, m$) to discriminate two distinct classes. The input and output data for each ELM$q(j, s)$ are only related with the data examples from class j and class s by ignoring the examples from other class examples. For the qth binary ELM classifier ELM$q(j, s)$ for discriminating the class j and class s, the target output data need to be decomposed to two subsets: one labelling as "1" for all the jth class examples and the other labelling as "-1" for all the sth class examples. When a pattern example $\mathbf{x}$ is input, the ELM-OAO produces $m(m-1)/2$ collective outputs for the m pattern classes and thus a decision function needs to be designed to decide the final class label $\hat{y}$ which the pattern $\mathbf{x}$ belongs to.

D. Learning Procedure of ELM-OAA and ELM-OAO

As described above, ELM-OAA and ELM-OAO decompose a multi-class classification problem into multiple two class problems, each of which is implemented by a binary classifier. It seems that it is troublesome to select multiple appropriate networks as the binary classifiers. However in real implementation of ELM-OAA and ELM-OAO, we can simplify the process. Since in ELM the parameters of hidden nodes are assigned randomly and independent of training data, all the binary classifiers in ELM-OAA or ELM-OAO share a common hidden node set. If for a m-class classification problem, the number of binary ELM classifiers required by ELM-OAA and ELM-OAO is r (in ELM-OAA, $r = m$ and in ELM-OAO, $r = m(m-1)/2$), the ELM-OAA and ELM-OAO can be considered as a combination of r SLFNs which share the same hidden nodes. Figure 1 illustrates the simplified structure of ELM-OAA and ELM-OAO. As illustrated in the figure, the same hidden nodes can be randomly generated for all the binary ELM classifiers in ELM-OAA and ELM-OAO, in other words, all these binary ELM classifiers can share the same hidden layer as done in Huang, et al [8] and thus a more compact network architecture is obtained. r groups of independent training data $(\mathbf{x}^{(q)}, \mathbf{t}^{(q)}, q = 1, 2, \cdots, r)$ are obtained from the training data according to the properties of m pattern classes to train the r independent binary classifiers. The r output neurons represent the outputs of the r binary classifiers and thus they are independently learned. When the qth group training data $(\mathbf{x}^{(q)}, \mathbf{t}^{(q)})$ is received, the weight $\beta^{(q)}$ connecting the qth binary classifier output neuron with the shared hidden nodes is learned without affecting other weights $(\beta^{(1)}, \cdots, \beta^{(q-1)}, \beta^{(q+1)}, \cdots, \beta^{(r)})$.

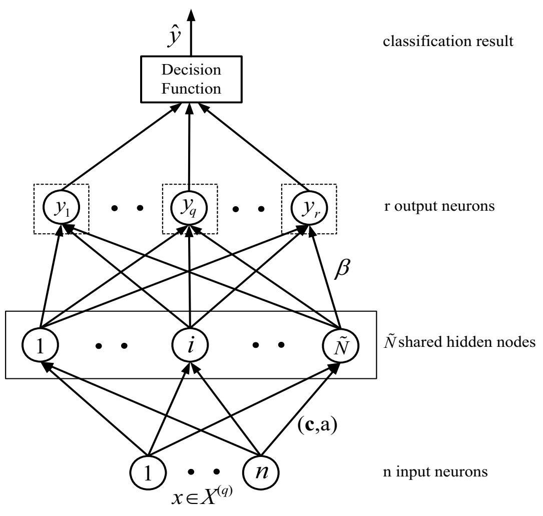

Fig. 1. Simplified Structure of ELM-OAA and ELM-OAO

Define weight vector $\beta^{(q)}$ as $[\beta_1^{(q)}, \cdots, \beta_{\tilde{N}}^{(q)}]^T$ where $\beta_i^{(q)} (i = 1, 2, \cdots, \tilde{N})$ denotes the weight connecting the ith shared hidden node to the qth binary classifier output neuron. $\beta^{(q)}$ can be determined as,

$$\beta^{(q)} = (\mathbf{H}^{(q)})^{\dagger} \mathbf{t}^{(q)} \tag{9}$$

where $\mathbf{t}^{(q)} (q = 1, 2, \cdots, r)$ is defined as,

$$\mathbf{t}^{(q)} = \left[\begin{array}{ccc} t_1^{(q)} & \cdots & t_{N^{(q)}}^{(q)} \end{array} \right]^T_{1 \times N^{(q)}} \tag{10}$$

Matrix $\mathbf{H}^{(q)} (q = 1, 2, \cdots, r)$ is defined as,

$$\begin{aligned} \mathbf{H}^{(q)} &= \mathbf{H}(\mathbf{c}_1, \cdots, \mathbf{c}_{\tilde{N}}, a_1, \cdots, a_{\tilde{N}}, \mathbf{x}_1^{(q)}, \cdots, \mathbf{x}_{N^{(q)}}^{(q)}) \\ &= \left[\begin{array}{ccc} G(\mathbf{x}_1^{(q)}; \mathbf{c}_1, a_1) & \cdots & G(\mathbf{x}_1^{(q)}; \mathbf{c}_{\tilde{N}}, a_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ G(\mathbf{x}_{N^{(q)}}^{(q)}; \mathbf{c}_1, a_1) & \cdots & G(\mathbf{x}_{N^{(q)}}^{(q)}; \mathbf{c}_{\tilde{N}}, a_{\tilde{N}}) \end{array} \right]_{N^{(q)} \times \tilde{N}} \end{aligned} \tag{11}$$

where $N^{(q)}$ is the training data size for the qth binary classifier.

Note that the training data $(\mathbf{x}^{(q)}, \mathbf{t}^{(q)}, q = 1, 2, \cdots, r)$ for r binary classifiers in ELM-OAA and ELM-OAO is different. We need to set the corresponding input-output training data for each binary classifier in ELM-OAA and ELM-OAO according to the OAA and OAO decomposition scheme described above. The outputs from all the binary classifiers represent collective outputs for the m pattern classes and thus a decision function to be described below is utilized to obtain the final classification result. Furthermore it should be noted that the number of the shared hidden nodes in Figure 1 is not fixed and is determined according to the OAA and OAO decomposition scheme and the problems considered. Comparing equations (4) and (5) with equations (10) and (11), one can observe that in ELM-OAA and ELM-OAO, the multi-class classification problem is decomposed into a series of two-class classification problems which are implemented by a series of binary classifiers trained by the subsets of the training data. Thus we may conclude that the network complexity required by the binary classifier is less than the single ELM classifier.

E. Decision Function

In ELM-OAA and ELM-OAO the multiple outputs from each binary ELM classifier represent collective outputs for the m pattern classes and thus they need to be decoded to obtain the final class label for the pattern example $\mathbf{x}$. Here the loss-based decoding approach [16] is utilized to decode the outputs from each binary ELM classifiers. In the approach, the chosen class label is most consistent with the output y_i in the sense that the total loss for example $(\mathbf{x}, i)$ is the minimum for all the class labels $(i = 1, 2, \cdots, m)$ if example $\mathbf{x}$ is labeled as class i. The total loss on an example $(\mathbf{x}, i)$ is given as,

$$d_L(M(i), y(\mathbf{x})) = \sum_{q=1}^{r} L(M(i, q) y_q(\mathbf{x})) \tag{12}$$

where M is a coding matrix: $\{-1, 0, +1\}^{m \times r}$. For ELM-OAA, M is a $m \times m$ matrix where the diagonal elements are $+1$ and the other elements are -1 and for ELM-OAO, M is a $m \times m(m-1)/2$ matrix where a distinct class pair (j, s) is included in each column which has $+1$ in row j, -1 in row s and the others are zero. $y_q(\mathbf{x})$ (in ELM-OAA, $q = 1, 2, \cdots, m$ and in ELM-OAO, $q = 1, 2, \cdots, m(m-1)/2$) is output value for the qth binary classifier on an example $\mathbf{x}$ and L is the exponential loss function.

Based on the above loss function (equation (12)), the final class output for the training example $\mathbf{x}$ is given by

$$\hat{y} = \mathrm{argmin}_{i=1,2,\cdots,m} d_L(M(i), y(\mathbf{x})) \tag{13}$$

III. Performance Evaluation

In this section, the performances of the ELM, ELM-OAO and ELM-OAA are evaluated in details based on multiple class pattern classification problems. Twelve real-world multiple class classification problems from UCI ML repository [14] are considered for verifying their performance, which are summarized in Table I with separate training and testing datasets. Generally there are two types of datasets: one has maximum 10 classes, and one with more than 10 classes.

All simulations have been conducted within the MATLAB 7.1 environment on a Pentium IV 1.7 GHZ CPU and 512M RAM personal computer. In each problem except letter recognition problem, Optical Recognition of Handwritten Digits and Pen-Based Recognition of Handwritten Digits problems, the optimal number of hidden nodes used in the ELM, ELM-OAO and ELM-OAA is chosen in the range of [10, 500] with a step size of 10 nodes. In the letter recognition problem, Optical Recognition of Handwritten

TABLE I

SPECIFICATION OF REAL-WORLD CLASSIFICATION BENCHMARK PROBLEMS

Type	Datasets	# Attributes	# Classes	# Observations	
				Training	Testing
Type I	Glass	9	6	110	104
	Vehicle	18	4	420	426
	Page Blocks	10	5	2,700	2,773
	Image Segmentation	19	7	1,100	1,210
	Satellite Image	36	6	4,400	2,035
	Shuttle	9	7	43,500	14,500
	DNA	180	3	1,000	1,186
	Optical Recognition of Handwritten Digits	64	10	3,823	1,797
	Pen-Based Recognition of Handwritten Digits	16	10	7,494	3,498
Type II	Cancer	98	14	144	46
	Arrhythmia	279	16	300	152
	Letter Recognition	16	26	10,000	10,000

Digits and Pen-Based Recognition of Handwritten Digits problems, the optimal number of hidden nodes for ELM-OAO is selected from the range [100,1000] with the interval of 10 hidden nodes and the optimal of hidden nodes for ELM and ELM-OAA is selected from the range [300,2000] with the interval of 10 hidden nodes. For each trial of simulation, 75% data and 25% data of the training dataset are randomly chosen for the training and validation purpose. The average cross-validation results over 25 trials of simulations have been obtained for ELM, ELM-OAA and ELM-OAO with fixed number of nodes and the number of nodes increases 10 by 10 in order to obtain the best cross-validation results. The ELM, ELM-OAA and ELM-OAO with best cross-validation results are finally chosen and the average testing accuracy of 50 trials on this finally chosen ELM, ELM-OAA and ELM-OAO is obtained and reported in this paper.

The three methods are compared concerning the network complexity, the testing accuracy, the training and testing time required.

A. Network Complexity

Table II shows that comparison of network complexity of ELM-OAA, ELM and ELM-OAO. From the table one can clearly observe that since ELM-OAA and ELM-OAO decompose the multi-class classification problem into multiple binary classifiers, each binary classifier requires smaller number of hidden nodes than ELM. Besides ELM-OAO requires smaller number of hidden nodes than ELM-OAA since each binary classifier in ELM-OAO utilizes smaller number of training data than that in ELM-OAA.

B. Testing Accuracy

The comparison of testing accuracy of ELM-OAA, ELM and ELM-OAO is given in Table III. As observed in the table, the three methods achieve similar testing accuracy and thus it is hard to give a clear conclusion about their advantage on the testing accuracy.

C. Computation Complexity

Table IV gives the comparison of training and testing time of ELM-OAA, ELM and ELM-OAO. It can be observed from the table that the ELM-OAO requires similar or less training time than ELM and ELM-OAA when the number of pattern classes is small, that is not larger than 10. Furthermore one can find that for the large pattern class size that is larger than 10, the training time required by ELM is the least. ELM generally requires less testing time than ELM-OAA and ELM-OAO. The reason is that in ELM-OAA and ELM-OAO multiple binary classifiers need to be decoded into the final output by the decision function.

TABLE II

COMPARISON OF NETWORK COMPLEXITY OF ELM-OAA, ELM AND ELM-OAO

Datasets	ELM-OAA	ELM	ELM-OAO
Glass	30	30	***20***
Vehicle	90	110	***50***
Page Blocks	130	160	***50***
Image Segmentation	200	210	***30***
Satellite Image	450	470	***150***
Shuttle	260	340	***80***
DNA	350	450	***290***
Handwritten Optical Recognition	420	470	***110***
Handwritten Pen-Based Recognition	650	690	***190***
Cancer	40	50	***10***
Arrhythmia	50	70	***20***
Letter Recognition	1100	1500	***510***

IV. CONCLUSIONS

In this paper, the single ELM classifier and its one-against-one and one-against-all variants (ELM-OAO and ELM-OAA) on multi-class classification problems have been invesitated. From the simulation results, we observe that the three methods produce similar testing accuracy but ELM-OAO requires smaller number of hidden nodes than the single ELM classifier and ELM-OAA. The training time required

TABLE IV

COMPARISON OF TRAINING AND TESTING TIME (SECONDS) OF ELM-OAA, ELM AND ELM-OAO

Datasets	ELM-OAA (s)		ELM (s)		ELM-OAO (s)	
	Training	Testing	Training	Testing	Training	Testing
Glass	0.0250	0.0617	***0.0117***	0.0773	0.0133	***0.0219***
Vehicle	0.1508	0.0617	0.0680	0.0531	***0.0547***	***0.0313***
Page Blocks	1.7125	0.5023	0.6641	***0.2070***	***0.3422***	0.2836
Image Segmentation	3.3547	0.2953	0.6000	***0.0906***	***0.1453***	0.1102
Satellite Image	43.838	1.6500	8.4320	***0.2758***	***5.1039***	1.0383
Shuttle	151.88	7.4594	36.614	***1.6516***	***30.395***	4.8422
DNA	7.0797	0.6227	5.0328	***0.2359***	***4.7070***	0.4578
Handwritten Optical Recognition	39.002	2.0734	6.7414	***0.3461***	***4.2766***	2.5727
Handwritten Pen-Based Recognition	140.17	5.3305	18.127	***0.7742***	***17.923***	7.3648
Cancer	0.0491	***0.0191***	***0.0072***	0.0194	0.0469	0.0422
Arrhythmia	0.1250	0.0586	***0.0266***	***0.0273***	0.1414	0.2281
Letter Recognition	2295.6	73.006	***76.066***	***1.9700***	1429.5	279.57

TABLE III

COMPARISON OF TESTING ACCURACY OF ELM-OAA, ELM AND ELM-OAO

Datasets	ELM-OAA	ELM	ELM-OAO
Glass	***66.346***	64.423	65.385
Vehicle	79.513	79.489	***81.378***
Page Blocks	***95.873***	95.761	95.397
Image Segmentation	***95.216***	94.667	94.493
Satellite Image	89.670	89.663	***89.955***
Shuttle	99.667	***99.715***	99.581
DNA	***94.866***	94.828	94.496
Handwritten Optical Recognition	96.928	***96.948***	96.343
Handwritten Pen-Based Recognition	98.294	***98.302***	97.819
Cancer	***78.652***	77.522	73.000
Arrhythmia	***65.441***	65.395	61.770
Letter Recognition	***93.417***	93.029	93.055

by ELM-OAO is similar or less than ELM and ELM-OAA when the number of pattern classes is not larger than 10. However when the number of pattern classes is larger than 10, the training time cost by ELM-OAO is most likely higher than the single ELM classifier but still smaller than the ELM-OAA.

REFERENCES

[1] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *Proceedings of International Joint Conference on Neural Networks (IJCNN2004)*, vol. 2, (Budapest, Hungary), pp. 985–990, 25-29 July, 2004.

[2] G.-B. Huang, Q.-Y. Zhu, K. Z. Mao, C.-K. Siew, P. Saratchandran, and N. Sundararajan, "Can threshold networks be trained directly?," *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 3, pp. 187–191, 2006.

[3] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, pp. 489–501, 2006.

[4] N.-Y. Liang, P. Saratchandran, G.-B. Huang, and N. Sundararajan, "Classification of mental tasks from EEG signals using extreme learning machine," *International Journal of Neural Systems*, vol. 16, no. 1, pp. 29–38, 2006.

[5] S. D. Handoko, K. C. Keong, Y.-S. Ong, G. L. Zhang, and V. Brusic, "Extreme learning machine for predicting HLA-peptide binding," *Lecture Notes in Computer Science*, vol. 3973, pp. 716–721, 2006.

[6] J.-X. Xu, W. Wang, J. C. H. Goh, and G. Lee, "Internal model approach for gait modeling and classification," in *the 27th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, (Shanghai, China), September 1-4, 2005.

[7] C.-W. T. Yeu, M.-H. Lim, G.-B. Huang, A. Agarwal, and Y.-S. Ong, "A new machine learning paradigm for terrain reconstruction," *IEEE Geoscience And Remote Sensing Letters*, vol. 3, no. 3, pp. 382–386, 2006.

[8] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Real-time learning capability of neural networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 863–878, 2006.

[9] J. Kim, H. Shin, Y. Lee, and M. Lee, "Algorithm for classifying arrhythmia using extreme learning machine and principal component analysis," in *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, pp. 3257–3260, Lyon, France, August 23-26, 2007.

[10] H. S. Shin and M. Lee, "Activity of daily living classification with acceleration signal using extreme learning machine," in *The International Conference on Electrical Engineering 2007 (ICEE2007)*, Hong Kong, July 8-12, 2007.

[11] G. Wang, Y. Zhao, and D. Wang, "A protein secondary structure prediction framework based on the extreme learning machine," *Neurocomputing (in press)*, 2008.

[12] G.-B. Huang, L. Chen, and C.-K. 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.

[13] G.-B. Huang and L. Chen, "Convex incremental extreme learning machine," *Neurocomputing*, vol. 70, pp. 3056–3062, 2007.

[14] C. Blake and C. Merz, "UCI repository of machine learning databases [online]," in *http://www.ics.uci.edu/mlearn/MLRepository.html*, (Department of Information and Computer Sciences, University of California, Irvine, USA), 1998.

[15] C. Rao and S. K. Mitra, *Generalized Inverse of Matrices and Its Applications*. John Wiley & Sons, Inc, New York, 1971.

[16] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," *The Journal of Machine Learning Research*, vol. 1, pp. 113–141, 2001.