International Journal of Neural Systems, Vol. 16, No. 1 (2006) 29–38

World Scientific
www.worldscientific.com

CLASSIFICATION OF MENTAL TASKS FROM EEG SIGNALS USING EXTREME LEARNING MACHINE

NAN-YING LIANG*, PARAMASIVAN SARATCHANDRAN†, GUANG-BIN HUANG‡
and NARASIMHAN SUNDARARAJAN§
School of Electrical and Electronic Engineering, Nanyang Technological University,
Nanyang Avenue, Singapore 639798
**lny@pmail.ntu.edu.sg*
†*epsarat@ntu.edu.sg*
‡*egbhuang@ntu.edu.sg*
§*ensundara@ntu.edu.sg*

Received 2 February 2005
Revised 3 August 2005
Accepted 11 January 2006

In this paper, a recently developed machine learning algorithm referred to as Extreme Learning Machine (ELM) is used to classify five mental tasks from different subjects using electroencephalogram (EEG) signals available from a well-known database. Performance of ELM is compared in terms of training time and classification accuracy with a Backpropagation Neural Network (BPNN) classifier and also Support Vector Machines (SVMs). For SVMs, the comparisons have been made for both 1-against-1 and 1-against-all methods. Results show that ELM needs an order of magnitude less training time compared with SVMs and two orders of magnitude less compared with BPNN. The classification accuracy of ELM is similar to that of SVMs and BPNN. The study showed that smoothing of the classifiers' outputs can significantly improve their classification accuracies.

Keywords: Brain Computer Interfaces (BCIs); electroencephalogram (EEG); mental task classification; Support Vector Machines (SVMs); Backpropagation Neural Networks (BPNNs); Extreme Learning Machine (ELM).

1. Introduction

Recently, there has been a lot of interest in the area of Brain Computer Interface (BCI) research as it has the potential to provide communication and control capabilities to people with severe motor disabilities.[1,2] This is a multi-disciplinary research involving experts from neurobiology, psychology, engineering, mathematics, and computer science. Any practical implementation of BCI design requires an efficient signal processing scheme that includes signal pre-processing, feature extraction and classification. In this paper, we emphasize on the classification part. Ideally, a good classifier for BCI should produce high classification accuracy with minimal computational complexity.

Presently, a number of linear and nonlinear classifiers have been studied in the literature for this application.[3] They include statistical methods such as Linear Discriminant Analysis (LDA), Hidden Markov Classifier and z-scale base Discriminant Analysis (ZDA).[4] The main drawback of these methods is that they do not work well for nonlinear classification problems. Recently backpropagation neural networks (BPNNs) and support vector machines (SVMs) have been shown to improve the classification accuracy compared with linear techniques, such as LDA.[5] However, they need a lengthy and time consuming training procedure. To reduce the training time without compromising accuracy, a new classification algorithm referred to as Extreme Learning Machine (ELM) is used in this paper.[6,7]

ELM has been developed by Huang *et al.* and its performance on a number of benchmark problems has been documented.[8] Here, its performance for the classification of five mental tasks has been

evaluated and compared with BPNN and SVM classifiers. Raw EEG data consisting of five mental tasks were obtained from a well-known database.[9] First feature extraction was applied to the raw data. Then the resulted feature vectors were used to train the classifiers. At last the classifiers were tested with the data not seen during the training to evaluate their classification accuracy. The results indicate that the ELM classifier produces similar classification accuracy but requires one to two orders of magnitude less training time than SVMs and BPNN. For SVMs the results have been presented for both 1-against-1 SVMs and 1-against-all SVMs.

The paper is organized as follows. Section 2 gives a brief description of the mental task classification problem. Section 3 presents the details of the ELM algorithm and Sec. 4 presents a brief description of the BPNN and SVM classifiers. Performance comparison of ELM classifier with BPNN and SVM classifiers is reported in Sec. 5 along with a discussion of the results. Conclusions from this study are summarized in Sec. 6.

2. Mental Tasks Classification Problem

2.1. *Experimental setup*

To provide communication and control capabilities to people with severe or complete motor paralysis, Brain Computer Interface (BCI) is a potential choice. The measured brain electrical signals, such as electroencephalogram (EEG) and electrocorticogram (ECoG) signals, convey the information related to mental activities. Hence the essential problem of identifying mental tasks is to classify the recorded brain electrical signals into different classes, each of which corresponds to a mental task.

A well-known EEG database was experimented with in this paper, which was recorded by Zak Keirn,[10] and distributed by Anderson.[9] The EEG database consisted of signals recorded from seven subjects, each of which performed five 10-second trials of five pre-defined mental tasks: Relax, Letter Composition, Multiplication, Counting, and Rotation per session. The performance of each classifier is evaluated and compared session by session. Subject 4 is excluded in this experiment because of saturation or invalidation of the signals in several trials in this subject.

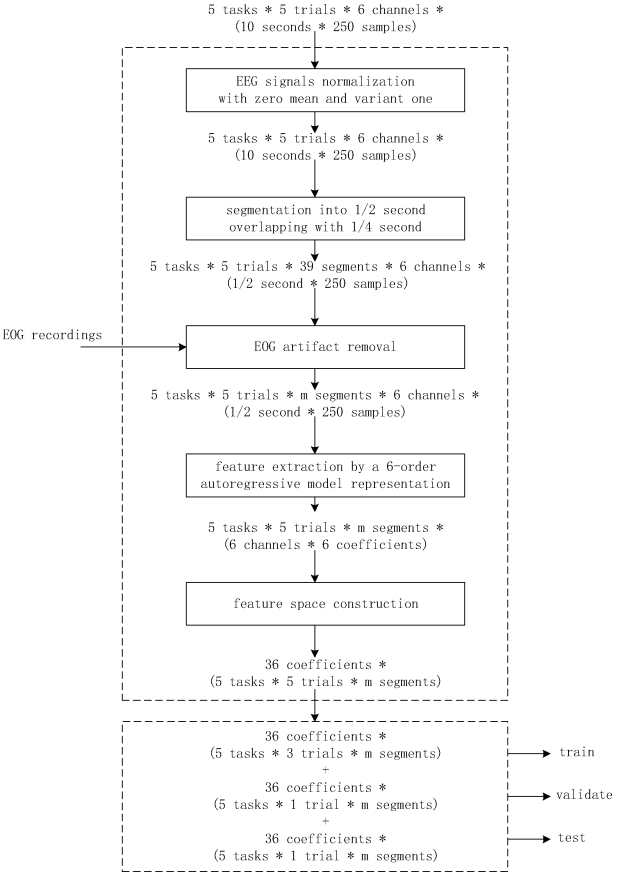


Fig. 1. Flow diagram shows the overall procedure for mental tasks classification using raw EEG signals.

Figure 1 shows the complete steps involved in processing the raw EEG signals recorded by 6 channels and sampled at 250 Hz. The EEG signals are first normalized with zero mean and variance one, and then divided into half-second segments with an overlap of one-quarter-second between adjacent segments. Thus 39 segments are produced. The segments with ocular artifacts are then detected and discarded using the detection scheme through a separated electrooculography (EOG) channel.[11] The detection scheme is based on a fixed-weight leakage normalized stochastic least mean fourth algorithm in a predictor mode. The scheme detects abrupt patterns in EOG channel caused by eye movements. The output of the detector is composed of 1s and 0s, with possible 0s between 1s. Therefore a morphological operation is followed, which results in two separated sets of EOG signals: eye-movement and non-eye-movement. EEG data collected during

the eye-movement set was discarded. Feature extraction is then performed on ocular artifact free EEG segments.

2.2. *Feature extraction*

The EEG signals undergo changes in the amplitude as well as frequency while different mental tasks are performed. So the discriminant features can be captured and extracted in the frequency domain before classification using modelling techniques, such as autoregressive (AR) model. AR model has been widely used for EEG analysis.[12] The model order that we used here is six, the same as that in Garret *et al.*[5] and Anderson *et al.*[13] for mental task classification using the same EEG database. So we deploy a sixth order AR model to fit into half-second EEG segments. The coefficients of the model for each segment are estimated by the Burg method, and then constructed into a 36-D feature vector (6 channels X 6 coefficients).[5,13]

3. A Brief Description of Extreme Learning Machine

In a seminal paper Baum implied that in a Single Layer Feedforward Network (SLFN) one may fix the connections on one level (i.e., weights between input neurons and hidden neurons) and only adjust the connections on the other level (i.e., weights between hidden neurons and output put neurons) and there is no gain achieved by an algorithm able to adjust the weights on both levels simultaneously.[14] Inspired by this work, Huang *et al.* have proposed a new learning algorithm referred to as Extreme Learning Machine (ELM).[6,7] ELM randomly chooses and fixes the weights between input neurons and hidden neurons based on some continuous probability density function, and then analytically determines the weights between hidden neurons and output neurons of the SLFN.

3.1. *Approximation problem of SLFNs*

For N samples $\{(\mathbf{x}_k, \mathbf{t}_k)\}_{k=1}^{N}$, where $\mathbf{x}_k = [x_{k1}, x_{k2}, \ldots, x_{kn}]^T$ and $\mathbf{t}_k = [t_{k1}, t_{k2}, \ldots, t_{km}]^T$, a standard SLFN with $\tilde{N}$ hidden neurons and activation function $g(x)$ is mathematically modeled as

$$\sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{w}_i \cdot \mathbf{x}_k + b_i) = \mathbf{o}_k, \quad k = 1, \ldots, N, \qquad (1)$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, \ldots, w_{in}]^T$ is the weight vector connecting the ith hidden neuron and the input neurons, $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^T$ is the weight vector connecting the ith hidden neuron and the output neurons, $\mathbf{o}_k = [o_{k1}, o_{k2}, \ldots, o_{km}]^T$ is the output vector of the SLFN, and b_i is the threshold of the ith hidden neuron. $\mathbf{w}_i \cdot \mathbf{x}_k$ denotes the inner product of $\mathbf{w}_i$ and $\mathbf{x}_k$. And these N equations can be written compactly as:

$$\mathbf{H}\beta = \mathbf{O} \qquad (2)$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}}, \qquad (3)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}{}^T \end{bmatrix}_{\tilde{N} \times m}, \qquad (4)$$

and

$$\mathbf{O} = \begin{bmatrix} \mathbf{o}_1^T \\ \vdots \\ \mathbf{o}_N^T \end{bmatrix}_{N \times m}. \qquad (5)$$

Here $\mathbf{H}$ is called the hidden layer output matrix.

3.2. *ELM learning algorithm*

In the case of learning an arbitrary function with zero training error, Baum had presented several constructions of SLFNs with sufficient hidden neurons.[14] However, in practice, the number of hidden neurons required to achieve a proper generalization performance on novel patterns is much less. And the resulting training error might not approach to zero but can be minimized by solving the following problem:

$$\min_{\mathbf{w}_i, b_i, \beta} \|\mathbf{H}(\mathbf{w}_1, \ldots, \mathbf{w}_{\tilde{N}}, b_1, \ldots, b_{\tilde{N}})\beta - \mathbf{T}\|^2, \qquad (6)$$

where

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m}. \qquad (7)$$

ELM randomly assigns and fixes the input weights $\mathbf{w}_i$ and biases b_i based on some continuous probability distribution function in the case of

learning a structured function, only leaving output weights β_i to be adjusted according to:

$$\min_{\beta} \|\mathbf{H}\beta - \mathbf{T}\|^2. \tag{8}$$

The above problem is well established and known as a linear system optimization problem. Its unique least-squares solution with minimum norm is given by[15]:

$$\hat{\beta} = \mathbf{H}^{\dagger}\mathbf{T}, \tag{9}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of the matrix $\mathbf{H}$. As analyzed by Bartlett[16] and Huang,[6,7] the generalization performance of a SLFN tends to be better with smaller magnitude of output weights. From this sense, the solution produced by ELM in Eq. (9) not only achieves the minimum square training error but also the best generalization performance on novel patterns. We now summarize ELM as the follows:

ELM Algorithm: Given a training set $\aleph = \{(\mathbf{x}_k, \mathbf{t}_k) \,|\, \mathbf{x}_k \in \mathbf{R}^n, \mathbf{t}_k \in \mathbf{R}^m, k = 1, \ldots, N\}$, an activation function $g(x)$, and the number of hidden neurons $\tilde{N}$,

(i) Randomly assign input weights $\mathbf{w}_i$ and biases b_i according to some continuous probability density function.
(ii) Calculate the hidden layer output matrix $\mathbf{H}$.
(iii) Calculate the output weights β_i: $\hat{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$.

In our experiments with ELM in this paper, the activation function is a sigmoidal function: $g(x) = 1/(1 + e^{-x})$. And the probability density function is a uniform distribution function in the range from -1 to 1.

4. Other Classifiers — BPNN and SVMs

4.1. *Backpropagation neural network (BPNN)*

Levenberg-Marquardt backpropagation (LMBP) learning algorithm appears to be the fastest method for training moderate-sized feedforward neural networks (up to several hundred weights).[17] According to the algorithm, backpropagation is used to calculate the Jacobian $\mathbf{J}$ of the performance index $F(\mathbf{W}) = \sum_{i=1}^{N} \mathbf{e}_i^2(\mathbf{W})$ with respect to the weight and bias variables $\mathbf{W}$. Each variable is then adjusted according to:

$$\mathbf{W}_{k+1} = \mathbf{W}_k - [\mathbf{J}^T\mathbf{J} + \mu\mathbf{I}]^{-1}\mathbf{J}^T\mathbf{E}, \tag{10}$$

where $\mathbf{E}$ is the error vector, and $\mathbf{I}$ is the identity matrix. The learning parameter μ is adaptively changed. It is increased by μ_{inc} as long as the change in Eq. (10) results in a reduced performance index. If the performance index no longer reduces, then μ is decreased by μ_{dec}.

In our experiment, LMBP provided in MATLAB is used to train the standard SLFNs with the same activation function as ELM. We initialize $\mu = 0.001$ and set $\mu_{\text{inc}} = 10$ and $\mu_{\text{dec}} = 0.1$.

4.2. *Support vector machines (SVMs)*

Support vector machines (SVMs) were originally designed for binary classification problems.[18] Currently different types of extensions of SVMs to multiclass classification have been studied and some of them include 1-against-all, 1-against-1, DAGSVM, and all-together.[19] The following gives the mathematical details of 1-against-1 and 1-against-all SVMs, which are used in our experiments:

4.2.1. *1-against-1 SVM*

Given a set of training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T$ and $y_i \in \{1, \ldots, k\}$ is the class index of $\mathbf{x}_i$ and k is the number of classes, 1-against-1 method constructs $k(k-1)/2$ SVM models. Each model is trained on the data from two classes. For training data from the ith and the jth class, the following problem is solved:

$$\text{minimize } \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} y_m y_n K(x_m, x_n)\alpha_m^{ij}\alpha_n^{ij} - \sum_{m=1}^{N}\alpha_m^{ij}$$

$$\text{subject to } \sum_{m=1}^{N} y_m \alpha_m^{ij} = 0 \quad 0 \le \alpha_m^{ij} \le C. \tag{11}$$

where $C > 0$ is the penalty parameter, and $K(x_m, x_n)$ is the kernel function. After solving Eq. (11), there are $k(k-1)/2$ decision functions. The classification is made by using the so called "Max Wins" strategy[19]: if $sign(\sum_{m=1}^{N} \alpha_m^{ij} y_m K(x_m, x) + b^{ij})$ says x is in the ith class, then the vote for the ith class is added by one; otherwise, the vote of the jth class is increased by one. Then the testing data x will be predicted in the class with the largest vote.

4.2.2. *1-against-all SVM*

1-against-all method constructs k SVM models. The ith SVM is trained with all of the samples in the ith class with positive labels, and all other samples with negative labels. Thus the ith SVM solves the following problem:

$$\text{minimize } \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} y_m y_n K(x_m, x_n)\alpha_m^i \alpha_n^i - \sum_{m=1}^{N}\alpha_m^i$$
$$\text{subject to } \sum_{m=1}^{N} y_m \alpha_m^i = 0 \quad 0 \le \alpha_m^i \le C. \tag{12}$$

After solving Eq. (12), there are k decision functions. The classification is made by taking the maximum of the real-valued output among the decision functions:

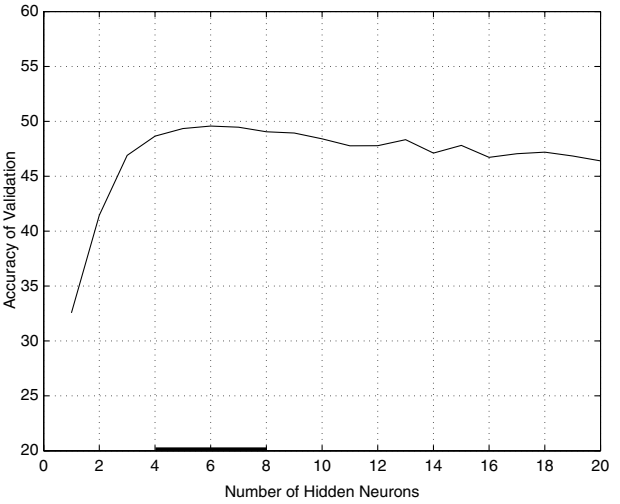
$$\text{class of } x \equiv \arg\max_{i=1,\ldots,k} \sum_{m=1}^{N} \alpha_m^i y_m K(x_m, x) + b^i. \tag{13}$$

The basic types of kernel functions which can be found in the implementation of SVMs include linear, polynomial, sigmoidal, and Gaussian kernel. Gaussian kernel, $\mathbf{K}(\mathbf{x}, \mathbf{x}_i) = exp(-\gamma\|\mathbf{x} - \mathbf{x}_i\|^2)$, is widely used and usually the first choice compared with other kernels in the literature.[20] Also, in the area of cognitive mental task classification using SVMs, it is the popular choice.[5] We have hence chosen Gaussian kernel in the implementation of 1-against-1 and 1-against-all SVMs in our comparisons.

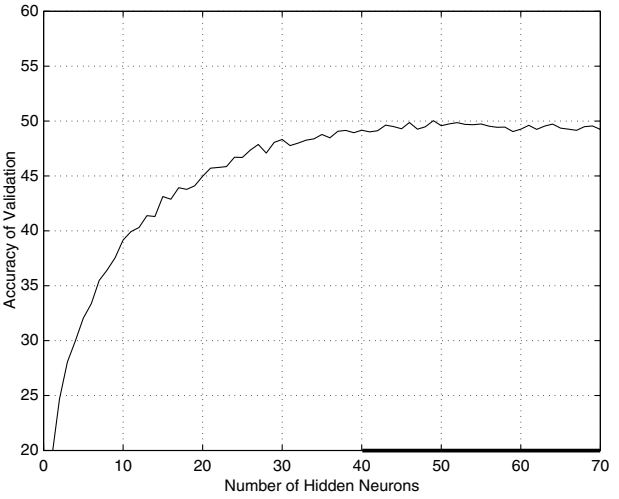
5. Performance Evaluation

5.1. *Classifier parameter selection*

Before performance comparison, the parameters of a classifier achieved the best generalization performance are pre-estimated. For Levenberg-Marquardt BPNN and ELM, the parameter only involves the number of hidden neurons used in a standard SLFN to be determined. For 1-against-1 and 1-against-all SVMs using Gaussian kernel, they involve the penalty parameter C as well as kernel parameter γ to be optimized for each binary SVM simultaneously. However, in order to reduce the search complexity, we consider that C and γ of all binary SVMs in 1-against-1 and 1-against-all SVMs are the same, which has generally been followed by Hsu and Lin as well.[19] Then the generalization performance of each classifier is evaluated as the classification accuracy on validation data set and averaged over sessions. In this context, Fig. 2(a) shows the validation accuracy against the number of hidden neurons for BPNN, in which early stopping was used. The increase of validation accuracy is significant until the number of hidden neurons in the SLFN is equal to 4 and an obvious decrease can be observed after 8. Therefore, the optimal number of hidden neurons is found from 4 to 8 for BPNN. Figure 2(b) shows similar results for ELM and the optimal number of hidden neurons is above 40. For the sake of simplicity, 6 hidden neurons and 50 hidden neurons are chosen for BPNN and ELM in our final implementations, respectively.



(a) BPNN



(b) ELM

Fig. 2. Validation accuracy vs. number of hidden neurons for BPNN and ELM.

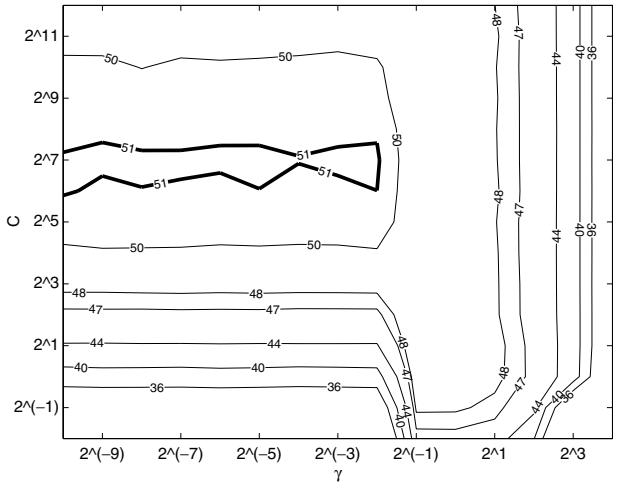


Fig. 3. Contours of validation accuracy for 1-against-1 SVMs' parameters (C, γ).

For SVMs with Gaussian kernel, it involves optimizing two parameters, C and γ, simultaneously, therefore a "grid-searching" procedure on the plane of $C - \gamma$ was recommended by Hsu and Lin and is used here.[19] Figure 3 shows the validation accuracy contours for various values of C and γ for 1-against-1 SVMs. The contour lines are labelled by the magnitude of validation accuracy matrix which is interpreted as heights with respect to the $\gamma - C$ plane. Specified values of validation accuracy: Refs. $36, 40, 44, 47, 48, 50$ and 51 are shown. The optimal combination of $(C, \gamma) = (2^7, 2^{-4})$ is obtained, for which produces the best validation accuracy of 51.04%. Similarly, the optimal combination of $(C, \gamma) = (2^8, 2^{-9})$ for 1-against-all SVM is found and produces the best validation accuracy of 51.83%. In our simulations, we found that ELM needs only around 21 minutes to find the optimal classifier parameter whereas 1-against-1 SVMs, 1-against-all SVMs and BPNN, require about 13, 23 and 28 hours, respectively.

As far as the simulations are concerned, all the results presented are based on an IBM T40 platform with Window XP, 1.5 GHz processor, and 512 MB RAM. The popular compiled C-coded SVM package, **LIBSVM**, is used in our simulations.[a] For BPNN, Levenberg-Marquardt backpropagation (LMBP) learning algorithm, one of the fastest algorithms of BP's variants, is used. LMBP in MATLAB has been well optimized at quite low level, such as the binary code level. ELM is implemented in MATLAB version.[b]

5.2. *Performance evaluation of BPNN, SVM and ELM classifiers*

Out of five trials one is selected as the testing data. Of the remaining four trials, one was chosen to be the validation data. The remaining three were then randomly shuffled to form the training data. The performance of each classifier is then evaluated session by session. Performance evaluation and comparison are carried out in terms of training time and testing accuracy. The pre-estimated optimal classifier parameters were applied for each classifier. And then each was trained using the training data set obtained from each session. The training time taken by the classifiers for each session is shown in Table 1. It can be seen from the table that ELM requires an order of magnitude less training time than SVMs and two orders of magnitude less than BPNN. It is to be noted that ELM is run in the MATLAB environment which is about 10 to 50 times slower than the C executable envirnoment used for SVMs. In spite of this ELM achieves a reduced training time.

The results for testing accuracy for each mental task and for each session have been shown in Table 2. The definitions for testing accuracy are given below the table. The best testing accuracies for each mental task for various sessions are highlighted. Average testing accuracy over all sessions is given in the last four rows together with standard deviation. It can be seen from the table that ELM produces similar accuracy compared with BPNN and SVM classifiers.

In the present scheme, all classifier's accuracies are only around 50%. This is because of the oscillatory nature of the classifier's outputs. For a typical case, we plotted the actual output vectors of ELM for session 2 of subject 1 in Fig. 4(a). In order to improve the accuracy, a post-processing scheme has been studied and deployed here. Its details are given in the following.

[a]http://www.csie.ntu.edu.tw/∼cjlin/libsvm/.
[b]http://www.ntu.edu.sg/home/egbhuang/.

Table 1. Training time (in sec) for BPNN, SVM and ELM classifiers.

Subject		BP[a]	ELM[a]	I-against-all SVM[b]	I-against-1 SVM[b]
1	Session 1	5.2100	0.0210	0.5431	0.2055
	Session 2	5.9389	0.0221	0.6719	0.2570
2	Session 1	4.1880	0.0181	0.3830	0.1460
3	Session 1	5.4207	0.0225	1.2027	0.4585
	Session 2	5.3543	0.0236	0.8350	0.2650
5	Session 1	4.9592	0.0210	0.6115	0.2220
	Session 2	5.4236	0.0235	0.8065	0.2995
	Session 3	5.6322	0.0240	0.8250	0.3015
6	Session 1	4.8865	0.0185	0.5070	0.2195
	Session 2	5.5849	0.0236	0.8890	0.3290
7	Session 1	5.7938	0.0236	0.7240	0.2400
	Average	5.3084	0.0219	0.7271	0.2676
	±sd	±0.4906	±0.0021	±0.2230	±0.0814

[a]Run in MATLAB environment.
[b]Run in C executable environment which is faster than MATLAB by 10–50 times.

Table 2. Testing accuracy (in %) for each mental task and for each session for all classifiers.

Subject		Classifier	Base[a]	Math[a]	Letter[a]	Rotate[a]	Count[a]	Session[b]
1	Session 1	BP	64.44	62.78	52.39	35.74	52.72	53.26
		ELM	**76.79**	**63.38**	62.77	48.08	51.94	60.29
		1-against-all SVM	71.63	60.65	62.83	**54.21**	**56.76**	60.86
		1-against-1 SVM	74.74	62.04	**63.90**	53.64	53.78	**61.30**
	Session 2	BP	54.65	73.71	**52.30**	36.70	61.05	55.76
		ELM	56.55	**85.54**	51.71	53.76	56.51	60.86
		1-against-all SVM	**60.23**	83.84	51.85	**59.51**	**64.27**	**63.96**
		1-against-1 SVM	60.23	78.89	47.87	55.56	63.61	61.36
2	Session 1	BP	51.90	68.48	46.62	60.90	38.76	53.11
		ELM	58.15	74.38	38.07	65.54	**48.60**	56.96
		1-against-all SVM	**65.04**	82.88	**45.95**	64.92	48.33	**61.37**
		1-against-1 SVM	64.29	**83.47**	43.81	**65.85**	45.61	60.52
3	Session 1	BP	45.26	**19.24**	42.83	18.66	**31.83**	31.81
		ELM	41.98	14.80	45.97	34.15	29.60	33.64
		1-against-all SVM	45.46	13.70	**51.33**	31.19	27.26	**34.08**
		1-against-1 SVM	**46.55**	17.00	45.55	**35.74**	21.78	33.79
	Session 2	BP	40.12	37.96	58.25	27.65	52.11	43.59
		ELM	44.49	33.55	65.92	20.13	57.26	44.39
		1-against-all SVM	44.86	**41.51**	**69.19**	39.96	**68.67**	**51.29**
		1-against-1 SVM	**45.84**	41.47	66.10	**30.33**	66.55	50.35
5	Session 1	BP	26.11	35.63	45.56	59.07	15.37	36.40
		ELM	34.44	**53.14**	59.44	69.44	22.04	47.65
		1-against-all SVM	33.70	52.77	**59.63**	**71.30**	25.19	**48.50**
		1-against-1 SVM	**41.85**	50.91	55.00	63.15	**26.30**	47.45
	Session 2	BP	12.63	39.24	**58.50**	55.35	28.82	38.76
		ELM	15.44	43.14	51.66	**68.41**	**32.29**	42.10
		1-against-all SVM	15.50	**49.78**	56.02	64.93	29.71	**43.23**
		1-against-1 SVM	**23.11**	48.24	50.84	53.86	27.04	40.52

Table 2. (*Continued*)

	Subject	Classifier	Base[a]	Math[a]	Letter[a]	Rotate[a]	Count[a]	Session[b]
	Session 3	BP	45.84	43.93	18.41	54.36	49.42	42.25
		ELM	50.90	**45.46**	19.16	56.81	**54.55**	45.15
		1-against-all SVM	55.61	42.23	16.82	**62.11**	54.13	45.96
		1-against-1 SVM	**62.48**	41.28	**19.47**	58.45	53.85	**46.89**
6	Session 1	BP	54.77	55.20	39.47	31.16	**61.63**	48.57
		ELM	55.55	63.56	45.70	39.57	58.44	50.56
		1-against-all SVM	**59.47**	**65.80**	38.68	38.37	58.61	52.30
		1-against-1 SVM	57.38	63.41	**43.72**	**41.72**	55.42	**52.46**
	Session 2	BP	53.21	64.96	20.97	42.92	**43.39**	44.95
		ELM	62.47	**71.05**	29.84	43.99	42.10	49.70
		1-against-all SVM	65.52	70.22	28.71	**48.34**	42.26	**50.78**
		1-against-1 SVM	**67.60**	64.57	**35.16**	44.60	37.26	49.58
7	Session 1	BP	48.96	50.36	69.20	49.07	51.43	53.81
		ELM	45.91	45.73	65.00	61.55	56.85	55.13
		1-against-all SVM	58.67	**53.33**	**70.13**	**61.66**	**57.28**	**60.47**
		1-against-1 SVM	**60.49**	51.19	66.53	59.51	49.49	57.64
	Average ±sd	BP	45.26 ±14.58	50.14 ±16.64	45.86 ±15.38	42.87 ±13.99	44.23 ±14.33	45.66 ±7.95
		ELM	49.33 ±15.99	53.98 ±20.24	47.75 ±15.50	51.04 ±15.60	46.38 ±12.92	49.68 ±8.30
		1-against-all SVM	52.33 ±16.37	**56.07 ±20.16**	**50.10 ±16.65**	**53.32 ±14.36**	**48.41 ±15.24**	**52.07 ±9.11**
		1-against-1 SVM	**54.96 ±14.49**	54.77 ±18.67	48.90 ±14.11	51.13 ±11.48	45.52 ±15.37	51.08 ±8.88

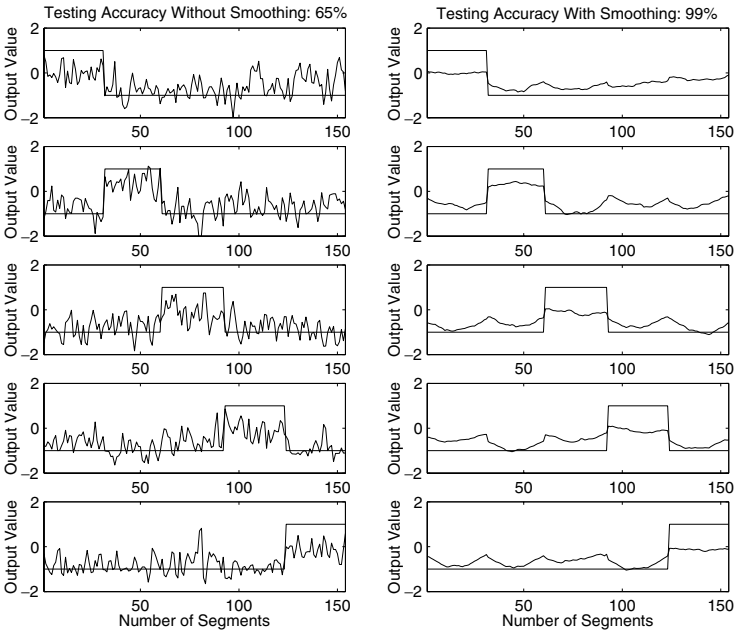
[a]The testing accuracy per mental task is defined as: the number of successfully classified segments of a mental task in a session over the total number of tested segments of that mental task in that session and expressed in %.
[b]The testing accuracy per session is defined as: the number of successfully classified segments of a session over the total number of tested segments of that session and expressed in %.

5.3. *Enhancing the classifier performance by post-processing*

A cursory inspection of the trend of output vectors in Fig. 4(a) indicates that better classification accuracy can be achieved by smoothing the classification algorithm's raw outputs over several segments and by basing the classification decision on the smoothed outputs. Therefore we smoothed the output vectors over 20 consecutive segments.[13] Figure 4(b) shows the resulting smoothed output vectors. The testing accuracies obtained by all the classifiers for each session with and without smoothing are given in Table 3. The results indicate that the accuracies have been improved to around 65%, which is a significant improvement compared to the case when smoothing was not employed. The accuracy of ELM in this case is also comparable to that of SVMs. Based on detailed studies on the smoothing of the classifier outputs it is found that smoothing can work well only if the classification accuracy without smoothing is above 20% in the five-class recognition case.

6. Conclusions

In this paper, we have evaluated the performance of three classifiers, namely BPNN, SVMs and ELM on the classification of mental tasks based on EEG signals. The study indicates that ELM needs about 1 to 2 orders of magnitude less training time compared with SVMs and BPNN. The classification accuracy



(a) (b)

Fig. 4. A typical output of ELM classifier from session 2 of subject 1 without and with smoothing: (a) without smoothing; (b) with smoothing.

Table 3. Testing accuracy (in %) comparisons with and without smoothing the classifier outputs.

Subjects		BP		ELM		1-against-all SVM		1-against-1 SVM	
and sessions		Without	With	Without	With	Without	With	Without	With
1	Session 1	53.26	72.74	60.29	78.02	60.86	77.87	**61.30**	**80.13**
	Session 2	55.76	78.24	60.86	**86.70**	**63.96**	85.74	61.36	85.62
2	Session 1	53.11	71.28	56.96	75.99	**61.37**	**78.76**	60.52	77.44
3	Session 1	31.81	36.49	33.64	38.57	**34.08**	38.55	33.79	**40.04**
	Session 2	43.59	51.89	44.39	57.37	**51.29**	**64.60**	50.35	63.11
5	Session 1	36.40	43.79	47.65	**62.79**	**48.50**	61.17	47.45	61.15
	Session 2	38.76	48.02	42.10	50.86	**43.23**	**53.51**	40.52	49.22
	Session 3	42.25	58.33	45.15	61.41	45.96	62.09	**46.89**	**63.43**
6	Session 1	48.57	69.19	50.56	72.62	52.30	73.19	**52.46**	**76.42**
	Session 2	44.95	61.06	49.70	**69.47**	**50.78**	67.99	49.58	67.89
7	Session 1	53.81	73.67	55.13	74.81	**60.47**	**79.77**	57.64	75.40
Average		45.66	60.43	49.68	66.23	**52.07**	**67.57**	51.08	67.26
±sd		±7.95	±13.86	±8.30	±13.76	**±9.11**	**±13.64**	±8.88	±13.76

of ELM is similar to SVM and BPNN. Furthermore, the time taken to search the optimal classifier parameters for ELM is significantly lower than the other two classifiers. Also, significant improvement in the testing accuracy can be achieved for all the three classifiers by smoothing their raw outputs.

References

1. J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson and T. M. Vaughan, Brain-computer interface technology: A review of the first international meeting, *IEEE Transactions on Rehabilitation Engineering* **8** (2000) 164–173.

2. T. M. Vaughan, Guest editorial brain-computer interface technology: A review of the second international meeting, *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **11** (2003) 94–109.
3. K.-R. Müller, C. W. Anderson and G. E. Birch, Linear and nonlinear methods for brain-computer interfaces, *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **11** (2003) 165–169.
4. T. Hinterberger, A. Kübler, J. Kaiser, N. Neumann and N. Birbaumer, A brain computer interface (BCI) for the locked-in: Comparison of different EEG classifications for the thought translation device, *Clinical Neurophysiology* **114** (2003) 416–425.
5. D. Garrett, D. A. Peterson, C. W. Anderson and M. H. Thaut, Comparison of linear, nonlinear, and feature selection methods for EEG signal classification, *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **11** (2003) 141–144.
6. G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, Extreme learning machine: A new learning scheme of feedforward neural networks, in *Proc. IEEE Int. Joint Conf. Neural Networks* **2** (2004) (Budapest, Hungary) 985–990.
7. G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, Extreme learning machine: Theory and applications, accepted in *Neurocomputing*, (2006).
8. G.-B. Huang, Benchmarking ELM, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, (2004). Available at http://www.ntu. edu.sg/home/egbhuang.
9. C. W. Anderson, Classification of electroencephalogram (EEG) signals for brainmachine interfaces, Computer Science Department, Colorado State University, USA, (2003). Available at http://www.cs.colostate.edu/∼anderson/res/eeg.
10. Z. A. Keirn, Alternative modes of communication between man and machine, Masters thesis, Electrical Engineering Department, Purdue University, USA, (1998).
11. P. Celka, B. Boashash and P. Colditz, Preprocessing and time-frequency analysis of newborn EEG seizures, *IEEE Engineering in Medicine and Biology Magazine* **20** (2001) 30–39.
12. J. Pardey, S. Roberts and L. Tarassenko, A review of parametric modelling techniques for EEG analysis, *Medical Engineering & Physics* **18** (1996) 2–11.
13. C. W. Anderson and Z. Sijercic, Classification of EEG signals from four subjects during five mental tasks, *Solving Engineering Problems with Neural Networks: Proc. Conf. Engineering Applications in Neural Networks*, (1996) (Turku, Finland) pp. 407–414.
14. E. B. Baum, On the capabilities of multilayer perceptrons, *J. Complexity* **4** (1988) 193–215.
15. D. Serre, Matrices: Theory and Applications, (New York, Springer-Verlag, 2002).
16. P. L. Bartlett, The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network, *IEEE Transactions on Information Theory* **44** (1998) 525–536.
17. M. T. Hagan and M. B. Menhaj, Training feedforward networks with the marquardt algorithm, *IEEE Transactions on Neural Networks* **5** (1994) 989–993.
18. V. N. Vapnik, Statistical Learning Theory, (Wiley, New York, 1998).
19. C.-W. Hsu and C.-J. Lin, A comparison of methods for multiclass support vector machines, *IEEE Transactions on Neural Networks* **13** (2002) 415–425.
20. C.-W. Hsu, C.-C. Chang and C.-J. Lin, A practical guide to support vector classification, (2003). Available at http://www.csie.ntu.edu.tw/∼cjlin/papers/guide/guide.pdf.