Neurocomputing 102 (2013) 135–143

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

EEG-based vigilance estimation using extreme learning machines

Li-Chen Shi [a], Bao-Liang Lu [a,b,c,*]

[a] Center for Brain-Like Computing and Machine Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai 200240, China
[b] MOE-Microsoft Key Laboratory for Intelligent Computing and Intelligent Systems, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai 200240, China
[c] Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai 200240, China

ARTICLE INFO

Available online 5 June 2012

Keywords:
Extreme learning machine
L_2 norm penalty
L_1 norm penalty
EEG
Vigilance estimation

ABSTRACT

For many human machine interaction systems, techniques for continuously estimating the vigilance of operators are highly desirable to ensure work safety. Up to now, various signals are studied for vigilance analysis. Among them, electroencephalogram (EEG) is the most commonly used signal. In this paper, extreme learning machine (ELM) and its modifications with L_1 norm and L_2 norm penalties are adopted for EEG-based vigilance estimation. A comparative study on system performance is conducted among ordinary ELM, its modifications, and support vector machines (SVMs). Experimental results show that, compared with SVMs, the ordinary ELM and its modifications can all dramatically speed up the training process while still achieving similar or better vigilance estimation accuracy. In addition, the following three observations have been made from the experiment results: (a) the ordinary ELM and the ELM with L_1 norm penalty (LARS-ELM) are sensitive on the number of hidden nodes; (b) the ELM with L_2 norm penalty (regularized-ELM) and the ELMs with both L_1 norm and L_2 norm penalties (LARS-EN-ELM, TROP-ELM) are stable and insensitive on the number of hidden nodes; and (c) regularized-ELM has a much faster training speed, while LARS-EN-ELM can achieve better vigilance estimation accuracy.

1. Introduction

Vigilance, or sustained attention, refers to the ability of observers to maintain their focus of attention and remain alert to stimuli for a prolonged period of time. During the past few decades, studies on vigilance have shown that vigilance estimation is very useful to our daily lives [1,2]. Especially, for many human machine interaction systems, techniques for realtime estimating the operator's vigilance are highly desirable to ensure work safety. Up to now, various signals are studied for vigilance analysis. Among them, EEG is the most commonly used signal, and has been proved to be very effective. In EEG-based vigilance analysis, many important observations have been pointed out, including the positive correlation of vigilance and some ERPs (P300, N200) amplitudes, the negative correlation of vigilance and theta rhythm activities, and the similarity between vigilance and the principal components of EEG spectrum [3–9]. Based on these correlations, and with the advancements of signal processing and machine learning techniques, many methods have been proposed for vigilance estimation [10–15]. In this paper, we focus on the machine learning aspect of EEG-based vigilance estimation.

Currently, EEG power spectra are the most used features, containing lots of vigilance information. Support vector machines are the most succeeded and commonly used algorithms to model the relationship between EEG features and vigilance states. However, the training process of SVMs is relatively long. To speed up the training process without dropping vigilance estimation accuracy, a recently developed machine learning algorithm, ELM, and its modifications are used as regression models for EEG-based vigilance estimation. ELM is proposed by Huang et al. [16–21], which is a kind of single-hidden-layer feedforward network (SLFN). The performance of ELM has been evaluated on a number of benchmark problems. Usually, ELM can achieve similar accuracy but requires one to two orders magnitude less training time than SVMs.

In this study, a visual recognition task is developed for vigilance indexing. Nine healthy subjects have participated in this vigilance experiment. The vigilance level is continuously indexed by the average vigilance-related performance within a 2 min moving time window in 2 s steps. The logarithms of EEG power spectra are used as the original features. Principal component analysis (PCA) and linear correlation coefficient between features and vigilance index are used for feature dimension reduction and feature selection. Compared with other kinds of data, the EEG data usually contain much more noises, which may

* Corresponding author at: Center for Brain-Like Computing and Machine Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai 200240, China.
E-mail addresses: bllu@sjtu.edu.cn, blu@cs.sjtu.edu.cn (B.-L. Lu).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.02.041

cause the ordinary ELM to be sensitive on the number of hidden nodes. To overcome this problem, some modified ELMs with L_1 norm penalty or L_2 norm penalty are adopted [22,23,20,24]. Then a comparative study on system performance among ELM, modified ELMs, and SVMs is conducted. The specifications are accuracy, stability, and training time. In [25], the ELM also was used to test in EEG application but on mental task classification. However, in this paper, the EEG-based vigilance estimation is on regression, not on classification.

Experiment results show that compared with SVMs, the ordinary ELM can dramatically speed up the training process while still achieving similar vigilance estimation accuracy, but it is sensitive on the number of hidden nodes. After adding an L_2 norm penalty, the modified ELM becomes stable and insensitive on the number of hidden nodes, while still maintaining the advantage of fast training and without dropping the vigilance estimation accuracy. After adding an L_1 norm penalty together with an L_2 norm penalty, this modified ELM also becomes stable and can achieve better vigilance estimation accuracy, but it needs more training time than ordinary ELM. However, its training time is still less than that of SVMs. Besides, only adding an L_1 norm penalty, the modified ELM is still sensitive on the number of hidden nodes.

This paper is organized as follows. In Section 2, the algorithms of ELM and its modifications and SVM are briefly described. In Section 3, the vigilance experiment setup is described. In Section 4, the data analysis process for EEG-based vigilance estimation is presented. In Section 5, performance comparisons of different algorithms are presented and discussed. Finally, some conclusions are given in Section 6.

2. Description of algorithms

2.1. ELM

The ELM, proposed by Huang et al. [16–21], is a kind of single-hidden-layer feedforward network (SLFN), and can be modeled as

$$f(x)=\sum_{i=1}^{\tilde{N}}\beta_i g(x;w_i,b_i), \tag{1}$$

where $\tilde{N}$ is the number of hidden nodes, β_i is the weight vector connecting the ith hidden node and output nodes, g is the activation function, w_i is the weight vector connecting the ith hidden node and input nodes, and b_i is the bias or impact factor of the ith hidden node.

For an additive hidden node, the activation function can be modeled as

$$g(x;w_i,b_i)=g(x^Tw_i+b_i). \tag{2}$$

For a Radial Basis Function (RBF) hidden node, the activation function can be modeled as,

$$g(x;w_i,b_i)=g\left(\frac{\|x-w_i\|}{b_i}\right). \tag{3}$$

Compared with the ordinary SLFN, after fixing the hidden nodes and activation function in ELM, all the parameters except β_i can be randomly chosen. Then, for a training set $\{x_i,y_i\}$, the ELM criterion can be written compactly as a least square (LS) form:

$$L(X,Y;\beta)=\|Y-H\beta\|^2, \tag{4}$$

where

$$X=\begin{pmatrix}x_1^T\\ \vdots\\ x_N^T\end{pmatrix},\quad Y=\begin{pmatrix}y_1^T\\ \vdots\\ y_N^T\end{pmatrix},\quad \beta=\begin{pmatrix}\beta_1^T\\ \vdots\\ \beta_{\tilde{N}}^T\end{pmatrix},\quad \text{and}$$

$$H=\begin{pmatrix}g(x_1;w_1,b_1) & \cdots & g(x_1;w_{\tilde{N}},b_{\tilde{N}})\\ \vdots & \cdots & \vdots\\ g(x_N;w_1,b_1) & \cdots & g(x_N;w_{\tilde{N}},b_{\tilde{N}})\end{pmatrix}.$$

The parameter β can be determined directly by

$$\hat{\beta}=H^{\dagger}Y, \tag{5}$$

where $H^{\dagger}$ is the Moore–Penrose generalized inverse of H.

In this study, for vigilance estimation, the output of the ELM network is a scalar. The label set Y is a $N\times 1$ vector.

2.2. Modified ELMs

2.2.1. Regularized-ELM

The ELM is very efficient. But just like the LS, the ELM also may encounter the singularity problem. When the number of hidden nodes $\tilde{N}$ is greater than the number of training data N, or some hidden nodes are assigned with similar parameters (some columns of H are correlated), the ELM will be a singular LS problem because of $H\in\mathbb{R}^{N\times\tilde{N}}$, and its solution is unstable. To solve this problem, an L_2 norm penalty can be added for regularization [23,20,21]. Then the modified ELM, called regularized-ELM, can be expressed as

$$L(X,Y;\beta,\lambda)=\|Y-H\beta\|^2+\lambda_2\|\beta\|^2, \tag{6}$$

where λ_2 is the regularization parameter. The parameter β can be calculated by

$$\hat{\beta}=(H^TH+\lambda_2 I)^{-1}H^TY, \tag{7}$$

where I is the identity matrix.

According to the ridge regression theory [26], even without encountering the singularity problem, regularized-ELM can still have a better generalization ability than the ordinary ELM.

Besides, a similar method named extreme support vector machine classifier (ESVMC) is proposed by Liu et al. [27]. ESVMC, which is derived from ELM and proximal support vector machine classifiers, is originally used for classification problems [28]. Its solution is equivalent to Eq. (6). And, it can be directly used for regression problem. In this study, ESVMC is categorized as regularized-ELM.

2.2.2. LARS-ELM

In ELM, the hidden nodes and their parameters are randomly generated, and the EEG features usually contain lots of noises. As a result, some outputs of hidden nodes (columns of H) will be irrelevant to vigilance. On the one hand, if few hidden nodes are used, the ELM will be under-fitted. On the other hand, if too many hidden nodes are used, the ELM will be over-fitted. To solve this problem, first, more hidden nodes than necessary are generated. Then, an L_1 norm penalty is added to the ELM, which can prune the network of ELM, automatically select the relevant hidden nodes, and improve the generalization ability of ELM [22]. The modified ELM can be expressed as

$$L(X,Y;\beta,\lambda)=\|Y-H\beta\|^2+\lambda_1\|\beta\|^1, \tag{8}$$

where λ_1 is the penalty factor and $\|\cdot\|^1$ stands for l_1 norm. Eq. (8) is commonly known as the Lasso problem, and it can be efficiently solved by using the LARS algorithm [29]. In this paper, this modified ELM is called LARS-ELM, while it is called OP-ELM in [22].

2.2.3. LARS-EN-ELM

Although LARS-ELM can find the most relevant hidden nodes, it also will encounter the singularity problem when the outputs of selected hidden nods (columns of H) are correlated or the number of selected hidden nodes is greater than the number of training data. Because solving the LARS problem uses the inverse of the Gram matrix of the selected columns of H, if the Gram matrix is not full rank, the solution of LARS will be unstable [29]. To overcome the deficiency of LARS-ELM, an L_2 norm penalty is added to the LARS-ELM, and a new modified ELM can be expressed as

$$L(X,Y;\beta,\lambda_1,\lambda_2)=\|Y-H\beta\|^2+\lambda_1\|\beta\|^1+\lambda_2\|\beta\|^2, \tag{9}$$

which is known as the naive elastic net, and can be reformed as a Lasso problem:

$$L(X,Y^*;\beta^*,\gamma)=\|Y^*-H^*\beta^*\|^2+\lambda\|\beta^*\|^1, \tag{10}$$

where

$$Y^*=\begin{pmatrix} Y \\ 0_{\tilde{N}\times 1}\end{pmatrix},\quad H^*=(1+\lambda_2)^{-1/2}\begin{pmatrix} H \\ \sqrt{\lambda_2}I_{\tilde{N}\times\tilde{N}}\end{pmatrix},$$

$$\beta^*=\sqrt{1+\lambda_2}\beta,\quad \text{and}\quad \lambda=\lambda_1/\sqrt{1+\lambda_2}.$$

Eq. (10) can be efficiently solved by LARS algorithm. Let

$$\hat{\beta}^*=\arg\min_{\beta^*} L(X,Y^*;\beta^*,\lambda), \tag{11}$$

then the solution of Eq. (9) can be obtained as follows:

$$\hat{\beta}=\frac{1}{\sqrt{1+\lambda_2}}\hat{\beta}^*. \tag{12}$$

Therefore, Eq. (9) also can automatically select the relevant hidden nodes but without encountering the singularity problem. However, as mentioned in [30], compared with Eqs. (6) and (8), Eq. (9) causes too much coefficients shrinkage, and introduces more bias to the ELM, but only slightly reduces variance. To correct the bias, elastic net is introduced, which is a rescaled version of naive elastic net. Elastic net directly rescales the solution of naive elastic net as its solution:

$$\hat{\beta}(\text{elasticnet})=(1+\lambda_2)\hat{\beta}(\text{naive elastic net}). \tag{13}$$

The algorithm for solving elastic net is a modification of LARS, called LARS-EN [30]. In this paper, only the elastic net solution of Eq. (9) is used, and this modified ELM is called LARS-EN-ELM. However, there are two parameters, λ_1 and λ_2, which should be tuned. These two parameters lead to slowing down the training process of LARS-EN-ELM.

2.2.4. TROP-ELM

Besides, to solve the singularity problem in LARS-ELM, Miche et al. proposed an algorithm, called TROP-ELM [24]. They first use LARS to select the best hidden nodes, then use the selected hidden nodes to form an ELM network together with an L_2 norm penalty to solve a regularized LS problem. They aim to speed up the process of tuning parameters, λ_1 and λ_2, in Eq. (9) by tuning them separately. According to their approach, they first tune λ_1 with $\lambda_2=0$, then they tune λ_2 with the well tuned λ_1. However, their method cannot guarantee to find out the optimal parameters unless the two parameters are checked in the dimension parameter space together. And TROP-ELM still faces the singularity problem in calculating the LARS solution and the bias problem of the naive elastic net. For performance comparison, TROP-ELM is also adopted, but the parameters, λ_1 and λ_2, are checked through the two dimension parameter space together.

2.3. SVM for regression

SVM for regression is proposed by Vapnik et al. [31]. For regression, SVM is formed as

$$f(x)=\sum_{i=1}^{l}\beta_i k(x,x_i)+b, \tag{14}$$

where β_i is the weight factor, x_i is the support vector, b is the bias, and $k(\cdot,\cdot)$ is the kernel function. The final optimization problem is usually formed as

$$\text{maximize}\quad \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)k(x_i,x_j), \\ -\varepsilon\sum_{i=1}^{l}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{l}y_i(\alpha_i-\alpha_i^*), \end{cases}$$

$$\text{subject to}\quad \sum_{i=1}^{l}(\alpha_i-\alpha_i^*)=0\ \text{ and }\ \alpha_i,\alpha_i^*\in[0,C], \tag{15}$$

where y_i is the real output of sample x_i, α_i and α_i^* are the Lagrange multipliers, ε is the tolerance factor, and C is the constrained factor. Eq. (15) can be efficiently solved by using quadratic programming. Then, in Eq. (14), $\beta_i=\alpha_i-\alpha_i^*$ and sample data x_i corresponding to nonzero β_i is the support vector. If some $\alpha_i^{(*)}\in(0,C)$, the bias b can be calculated by

$$b=\begin{cases} y_i-\varepsilon-\sum_{i=1}^{l}\beta_i k(x,x_i), & \alpha_i\in(0,C), \\ -y_i-\varepsilon+\sum_{i=1}^{l}\beta_i k(x,x_i), & \alpha_i^*\in(0,C), \end{cases} \tag{16}$$

where, $\alpha_i\alpha_i^*=0$ is always satisfied. They can never be a set of α_i,α_i^*, which are both nonzero. For more detail on SVM for regression, refer [31,32].

3. Experimental setup

3.1. Subjects and procedure

This is a monotonous visual task, which is shown in Fig. 1 [33]. Subjects sit in a comfortable chair, two feet away from the LCD. There are four colors of traffic signs being randomly presented in the LCD by the NeuroScan *Stim*^{2} software, and there are more than 40 different traffic signs for each color. Each trial is 5.5–7.5 s long, including 5–7 s black screen and 0.5 s traffic signs presented. Each section consists of more than 600 trials. The stimulus sequence of the vigilance task is shown in Fig. 2, and some of these four different traffic signs are shown in Fig. 3. In each trial, the subjects are asked to recognize the sign color and press the

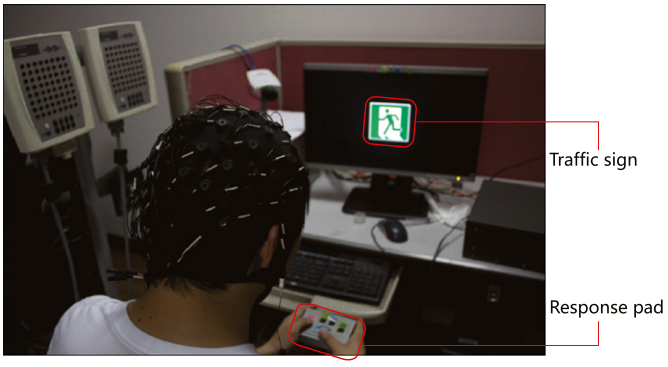


Fig. 1. The vigilance experimental scene.

correct button on the response pad immediately after seeing the traffic sign. There are four buttons on the response pad corresponding to the four different colors of traffic signs. The response sequence of the vigilance task is shown in Fig. 4. A total of nine healthy subjects of 19–28 years old have participated in this experiment. After training, each subject has finished at least two sessions on different days. Sessions are carried out in a small sound-proof room with normal illumination during 13:00–15:00 after lunch.

3.2. Data collection

For each session, the visual stimulus sequence and the response sequence are recorded by the NeuroScan system sampled at 500 Hz. Simultaneously, a total of 62 EEG channels are also recorded by the NeuroScan system sampled at 500 Hz and filtered between 0.1 and 100 Hz. The electrodes are arranged based on the extended 10/20 system with a reference on the top of the scalp [33].

3.3. Vigilance measurement

To evaluate the performances of different models, a reference vigilance index is necessary. In our experiments, the local error rate of the subject's performance is used as the reference vigilance level, which is defined as the current probability that the subject failed to respond to a presented target within a time window of constant width [4]. Because the fluctuations of vigilance level with cycle lengths are usually longer than 4 min [4], in our experiments, to eliminate the variance at cycle lengths shorter than 2 min, the local error rate series $e(t)$ are derived by computing the target false recognition rate within a 2-min time window at 2-s step as

$$e(t)=\frac{NumF_{(ST+2t-L/2,ST+2t-1+L/2)}}{NumT_{(ST+2t-L/2,ST+2t-1+L/2)}}, \tag{17}$$

where, ST is the start time for vigilance measurement, L is the 120 s (2 min) window length, $NumF_{(i,j)}$ is the number of false responses within the time window (i,j), and $NumT_{(i,j)}$ is the number of total stimuli within the time window (i,j). The original performance data and the local error rate series are shown in Fig. 5.

Fig. 2. The stimulus sequence of the vigilance task.

4. Data analysis

4.1. Vigilance estimation

Although 62 EEG channels are recorded, there is no need to use all of them, because only the EEG activities measured around the posterior regions of the scalp are highly correlated with the vigilance changes. As a result, in this study, six EEG channels (P1, Pz, P2, Po3, Poz, Po4), which are measured from the posterior regions of the scalp, are used for vigilance estimation. The goal of this work is to compare the performance of different machine learning algorithms. Therefore, a commonly used signal processing strategy for EEG-based vigilance estimation is adopted. The whole vigilance estimation framework is shown in Fig. 6. This framework consists of the following five main components: (1) a bandpass filter (1–50 Hz) is used to remove the low-frequency noise and the high frequency noise for each EEG channel; (2) the logarithm of each EEG channel's power spectral density (PSD) from 2 Hz to 18 Hz is calculated for every 2 s interval and averaged on the frequency domain with 2 Hz frequency resolution; for six EEG channels, there are totally 54 PSD features; (3) each PSD feature is smoothed with a 2 min moving-average filter to make the features more stable; (4) the top 10 principal components (PCs) of the 54 PSD features are calculated as the candidate features, which account for more than 90% of training set variance, maintain most of the original information and are more stable than the original PSD features. Because not all of the PC features are related with vigilance changes, to reduce the influences of unrelated PC features, linear correlation coefficients between the PC features and vigilance index are calculated, and only the PC features with correlation coefficient greater than 0.15 are reserved as the final features; and (5) ordinary ELM, regularized-ELM, LARS-ELM, LARS-EN-ELM, TROP-ELM, and SVM are adopted as the regression models for vigilance estimation with a 2 s time resolution.

For LARS-ELM, LARS-EN-ELM, and TROP-ELM, the initial number of hidden nodes is set to 100. All of these six kinds of regression models use the following Gaussian RBF kernels:

$$g(x;w_i,b_i)=\exp\left(-\frac{\|x-w_i\|^2}{b_i}\right), \quad k(x_i,x_j)=\exp(-\gamma\|x_i-x_j\|^2), \tag{18}$$

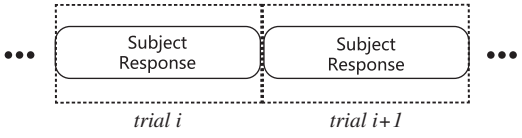


Fig. 4. The response sequence of the vigilance task.

Fig. 3. The 40 different traffic signs from a stimulus sequence slice.

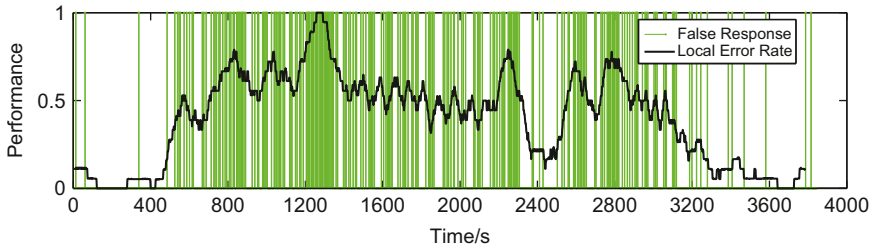


Fig. 5. Original performance data and local error rate series from session 1_1. Each green stem denotes a false response occurred on that moment. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

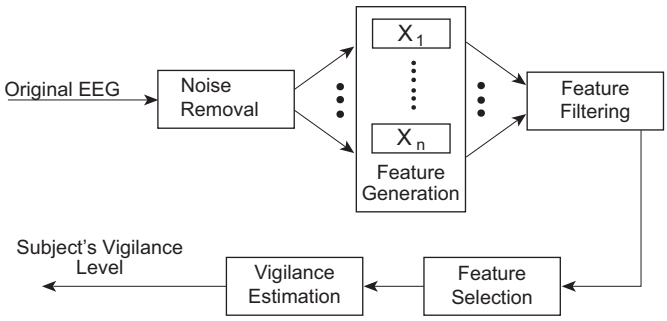


Fig. 6. The framework of vigilance estimation.

Table 1
Parameters need to be tuned in the regression algorithms. Here, 'R-ELM', 'L-ELM', 'LE-ELM', and 'T-ELM' stand for regularized-ELM, LARS-ELM, LARS-EN-ELM, and TROP-ELM, respectively, and 'Nodes' stands for the number of hidden nodes in ELM or the number of support vectors in SVMs.

Parameters	ELM	R-ELM	L-ELM	LE-ELM	T-ELM	SVM
Nodes	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	No
λ_1	No	No	No	No	No	No
λ_2	No	**Yes**	No	**Yes**	**Yes**	No
C	No	No	No	No	No	**Yes**
ε	No	No	No	No	No	**Yes**
γ	No	No	No	No	No	**Yes**

where $g(x; w_i, b_i)$ is used for ELMs, $k(x_i, x_j)$ is used for SVM, and γ is the impactor.

4.2. Parameters selection

The regression models are trained for each subject individually, because the EEG-vigilance relationships are usually different for different subjects [4]. As the EEG features are time dependent (processed by a 2 min moving-average filter), the training set and testing set are generated from different sessions of each subject. During training, for determining the number of ELM hidden nodes or searching parameters, such as λ_2 in Eq. (6), cross-validation is adopted [33].

It should be noted that for cross-validation (CV), as the EEG features are time dependent, and not independent, the traditional random partition method of CV for independent identical distributed data sets is not suitable, which will lead to large over-fitting when the adjacent (dependent) EEG features on the time domain are divided into both the training set and validation set. Besides, as the EEG features close to the boundary of the adjacent partitions are time dependent, if too many partitions are generated, the proportion of dependent EEG features in each partition will increase, which also will lead to large over-fitting. As a result, during CV, to reduce the time dependent influence of the EEG features and to control over-fitting, each subject's training set is equally divided into three partitions in the chronological order, but not randomly.

The parameter λ_1 in Lasso is used to control the number of selected hidden nodes. Instead of tuning λ_1, we directly tune the number of selected hidden nodes. An early stopping strategy is used in the LARS algorithm. That is, when the number of nonzero coefficients of β meets the predefined number of selected hidden nodes, the LARS algorithm is stopped. In the same way, only the number of selected hidden nodes and λ_2 need to be tuned in LARS-EN-ELM and TROP-ELM. Table 1 lists the parameters to be tuned in the six kinds of algorithms used in this paper.

5. Performance evaluation

The LIBSVM package is used for SVMs and run in the C environment [34]. Other algorithms are implemented and run in the Matlab environment. All the experiments are run on an HP server with 64 bit Windows Sever 2003, Intel Xeon E5440 CPU, and 16 GB memory.

As the hidden nodes of ELM are randomly generated, when the number of hidden nodes is too small, the testing results of well trained ELM in different training trials will change. Therefore, the ordinary ELM and the modified ELMs are run for 10 times with random initializations, and their mean results of 10 trials are used for performance comparison. Because the result of SVM is stable after fixing the parameters, SVM is only run once.

To evaluate the performance of different algorithms in the testing set, root-mean-square error (RMSE) between the real vigilance index and the estimated vigilance levels is used to represent the accuracy. The smaller the RMSE is, the higher the accuracy is. In the following figures or tables, if not specified, the specifications, such as the RMSE and the training time, usually indicate the 10 trials mean specifications of ELMs; the averaged specifications, such averaged RMSE and averaged training time, usually indicate the specifications averaged on nine different subjects.

The performance of the six kinds of regression algorithms used in this paper are shown in Table 2. From this table, we can obtain the following observations: (1) the accuracy of LARS-EN-ELM is the best, while the accuracies of other algorithms are very close; (2) ordinary ELM and regularized-ELM require the least training time, other modified ELMs require a little more training time, but still less than SVMs; (3) LARS-EN-ELM can find the most compact ELM network; and (4) the five kinds of ELMs all require many fewer nodes than SVMs, therefore their test processes should be much faster than SVMs.

The relationship between averaged testing accuracy and the number of hidden nodes is shown in Fig. 7, which reflects the stability of different kinds of ELMs. From this figure, we can

Table 2
Performance of different algorithms on EEG-based vigilance estimation. Here, 'Spec.' stands for specification, 'R-ELM', 'L-ELM', 'LE-ELM', and 'T-ELM' stand for regularized-ELM, LARS-ELM, LARS-EN-ELM, and TROP-ELM, respectively, 'Tr.T' stands for single trial training time (in sec), and 'Nodes' stands for the number of hidden nodes in ELM or the number of support vectors in SVMs.

Subject	Spec.	Algorithm					
		ELM	R-ELM	L-ELM	LE-ELM	T-ELM	SVM
No. 1	RMSE	0.172	0.154	0.159	**0.146**	0.159	0.192
	Tr.T	$<$ **0.001**	$<$ **0.001**	0.119	0.145	0.114	1.000
	Nodes	8	8	7	**5**	9	724
No. 2	RMSE	0.178	0.184	0.185	**0.173**	0.181	0.177
	Tr.T	$<$ **0.001**	$<$ **0.001**	0.156	0.150	0.136	0.813
	Nodes	10	18	15	**6**	12	601
No. 3	RMSE	0.189	0.187	**0.186**	0.201	0.189	0.204
	Tr.T	0.031	$<$ **0.001**	0.134	0.142	0.177	0.328
	Nodes	26	11	**6**	8	13	385
No. 4	RMSE	0.152	0.154	0.157	0.155	0.159	**0.135**
	Tr.T	$<$ **0.001**	$<$ **0.001**	0.113	0.155	0.125	0.078
	Nodes	**8**	12	11	11	**8**	61
No. 5	RMSE	0.195	0.174	0.177	**0.171**	0.176	0.176
	Tr.T	$<$ **0.001**	0.094	0.058	0.089	0.048	0.734
	Nodes	13	23	13	**9**	17	769
No. 6	RMSE	**0.150**	0.190	0.191	0.166	0.189	0.186
	Tr.T	$<$ **0.001**	$<$ **0.001**	0.038	0.064	0.058	0.500
	Nodes	6	8	**5**	**5**	9	626
No. 7	RMSE	0.148	0.139	0.138	0.144	0.137	**0.125**
	Tr.T	$<$ **0.001**	0.016	0.098	0.130	0.134	0.641
	Nodes	8	13	9	**7**	15	578
No. 8	RMSE	0.215	0.203	0.170	**0.163**	0.183	0.165
	Tr.T	$<$ **0.001**	$<$ **0.001**	0.134	0.114	0.139	0.844
	Nodes	10	16	13	11	**10**	476
No. 9	RMSE	**0.079**	0.095	0.096	0.089	0.098	0.103
	Tr.T	0.109	$<$ **0.001**	0.095	0.127	0.206	0.344
	Nodes	12	21	8	**7**	21	311
Average $\pm$ sd	RMSE	0.164 $\pm$ 0.039	0.164 $\pm$ 0.033	0.162 $\pm$ **0.030**	**0.156** $\pm$ 0.031	0.163 $\pm$ **0.030**	0.163 $\pm$ 0.034
	Tr.T	0.016 $\pm$ 0.036	**0.012** $\pm$ 0.031	0.105 $\pm$ 0.038	0.124 $\pm$ **0.030**	0.126 $\pm$ 0.050	0.587 $\pm$ 0.297
	Nodes	11.2 $\pm$ 6.0	14.4 $\pm$ 5.4	9.7 $\pm$ 3.5	**7.7** $\pm$ **2.3**	12.7 $\pm$ 4.3	503.4 $\pm$ 222.7

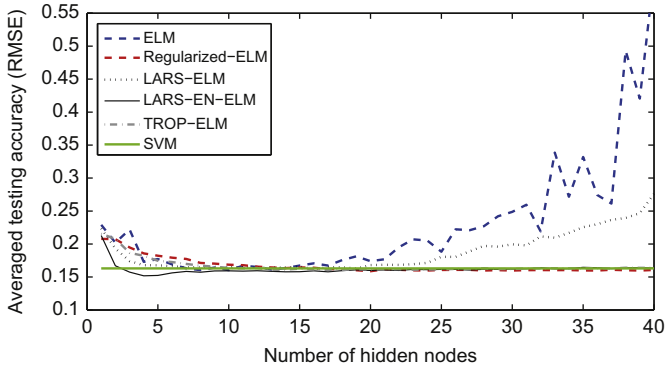


Fig. 7. The relationship between averaged testing accuracy of nine subjects and the number of hidden nodes. The accuracy of SVM is used as the baseline. (The RMSE of ELM or LARS-ELM increased dramatically when with more than 40 hidden nodes, while the RMSEs of other algorithms were still stable and close to the RMSE of SVM. In order to depict the RMSE divergences of different algorithms with small scale resolution clearly, only the accuracies corresponding to the number of hidden nodes between 1 and 40 are depicted.)

obtain the following observations: (1) the ordinary ELM and LARS-ELM are sensitive on the number of hidden nodes and the performances of the ordinary ELM and LARS-ELM become worse when too few or too many hidden nodes are used; (2) regularized-ELM, LARS-EN-ELM, and TROP-ELM are insensitive on the number of hidden nodes; (3) the performances of regularized-ELM and TROP-ELM become stable and are comparable with that of SVMs after using enough hidden nodes, and the performance of LARS-EN-ELM is stable and a little better than that of SVM. However, the performance of LARS-EN-ELM will drop slightly after using more number of hidden nodes, but is still comparable with that of SVMs.

The relationship between averaged training time and the number of hidden nodes is shown in Fig. 8. From this figure, it can be seen that the ordinary ELM and regularized-ELM are much faster than SVMs even when using 100 hidden nodes for single trial training. When the number of hidden nodes is less than 40, LARS-ELM, LARS-EN-ELM, and TROP-ELM are faster than SVMs. Otherwise, they are slower than SVMs. For vigilance estimation task, however, 20 or fewer hidden nodes are enough. Therefore, SVMs are the slowest.

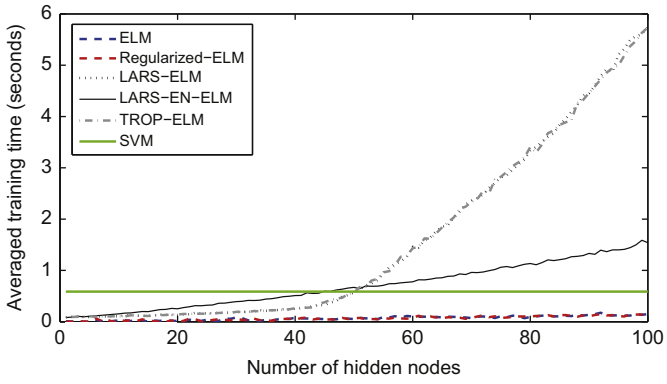


Fig. 8. The relationship between averaged training time of nine subjects and the number of hidden nodes. The training time of SVM is used as the baseline. (The averaged training time curves of LARS-ELM and TROP-ELM are very close.)

Table 3
Averaged performance comparisons of different algorithms by considering the cross-validation process. Here, 'Avg.' stands for averaged, 'R-ELM', 'L-ELM', 'LE-ELM' and 'T-ELM' stand for regularized-ELM, LARS-ELM, LARS-EN-ELM, and TROP-ELM, respectively, 'Tr.T' stands for single trial training time (in s), 'Nodes' stands for the number of hidden nodes in ELM or the number of support vectors in SVMs, 'CV. (No. Para.)' stands for number of parameters to be tuned by cross-validation, and 'Sup. L.' stands for superlinear.

Specification	Algorithm					
	ELM	R-ELM	L-ELM	LE-ELM	T-ELM	SVM
Avg. RMSE	0.164	0.164	0.162	**0.156**	0.163	0.163
Avg. Tr.T	0.016	**0.012**	0.105	0.124	0.126	0.587
Avg. Nodes	11.2	14.4	9.7	**7.7**	12.7	503.4
CV. (No. Para.)	**1**	2	**1**	2	2	3
Stability	No	**Yes**	No	**Yes**	**Yes**	**Yes**
Tr.T on Nodes	**linear**	**linear**	Sup. L.	**linear**	Sup. L.	Sup. L.

From Fig. 8, we can also see that the training times of the ordinary ELM, regularized-ELM, and LARS-EN-ELM are almost linear with the number of hidden nodes, but the training time of LARS-ELM or TROP-ELM is superlinear with the number of hidden nodes. The main reason is that, both LARS-ELM and TROP-ELM will encounter the singularity problem when many hidden nodes are used, and they both require more time to finish the LARS algorithm in LARS-ELM and TROP-ELM.

Now, the averaged performance comparisons will be discussed by considering the cross-validation (parameter selection) process. The results are shown in Table 3. From this table, we can see that the total training time should be calculated by multiplying two factors: single trial training time and the number of round for validation. Usually, there is an exponential relationship between the number of round for validation and the number of parameters to be tuned. For the total training time with no more than 100 hidden nodes, the training speed of the ordinary ELM and its modifications are all much faster than that of SVMs, and their ranks are shown in Table 4. After considering stability, accuracy and the total training speed, a comprehensive ranking of the ordinary ELM and its modifications can be obtained, which also is shown in Table 4. For comprehensive ranks, the algorithm stability is first considered, followed by the training speed and accuracy.

The ordinary ELM and LARS-ELM are sensitive on the number of hidden nodes and unstable. They are ranked into the worst level. Regularized-ELM has the second fastest training speed and can achieve comparable accuracy. Compared with Regularized-ELM, LARS-EN-ELM has slower training speed but can achieve better accuracy. Therefore, regularized-ELM and LARS-EN-ELM are ranked into the top level. TROP-ELM is ranked into the second level, because its accuracy is worse than that of LARS-EN-ELM and its training speed will decrease dramatically when many hidden nodes are used.

Finally, to illustrate the vigilance estimation results, the estimated local error rate curves from the test session of subject No. 2 are shown in Figs. 9–11, which are estimated by using LARS-EN-ELM, SVM and ordinary ELM, respectively.

Table 4
Performance rank of different algorithms. ("R-ELM", "L-ELM", "LE-ELM" and "T-ELM" stand for regularized-ELM, LARS-ELM, LARS-EN-ELM, TROP-ELM, respectively.)

Specification	Algorithm					
	ELM	R-ELM	L-ELM	LE-ELM	T-ELM	SVM
Total training speed Rank	**1**	2	2[a]	3	3[b]	4
Comprehensive Rank	4	**1**	4	**1**	2	3

[a] Because the training time of L-ELM is superlinear with the number of hidden nodes, the total training speed of L-ELM is comparable with that of R-ELM only with less hidden nodes.

[b] Because the training time of T-ELM is superlinear with the number of hidden nodes, the total training speed of T-ELM is comparable with that of LE-ELM only with less hidden nodes.

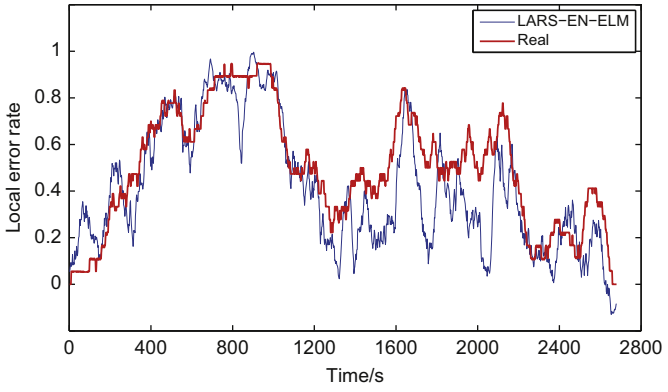


Fig. 9. The LARS-EN-ELM estimated and real local error rate curves of session 2_2. The RMSE between the two curves is 0.173.

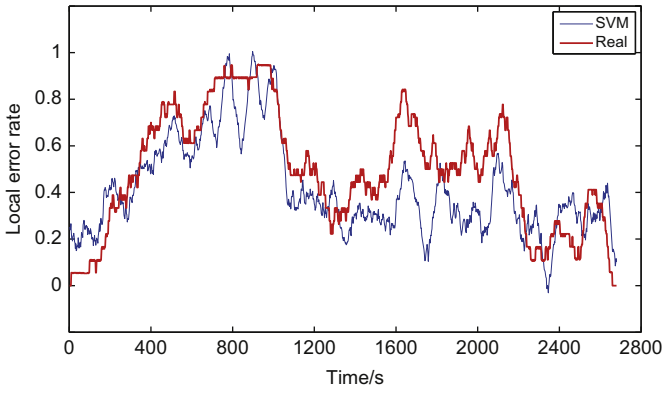


Fig. 10. The SVM estimated and real local error rate curves of session 2_2. The RMSE between the two curves is 0.177.

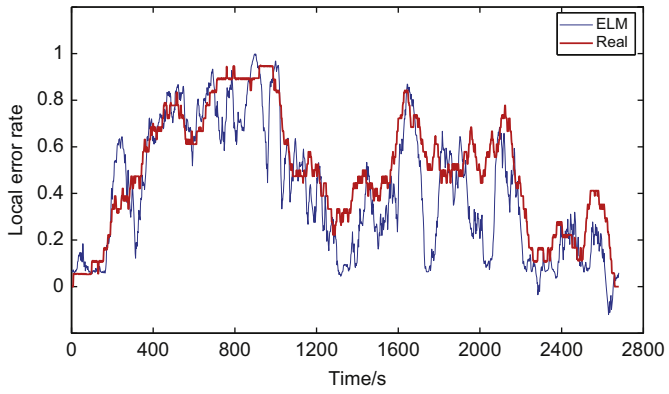


Fig. 11. The ordinary ELM estimated and real local error rate curves of session 2_2. The RMSE between the two curves is 0.178.

6. Conclusions

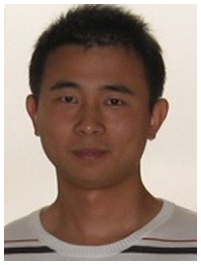
In this paper, ELM and its modifications with L_1 norm penalty and L_2 norm penalty are adopted for EEG-based vigilance estimation. LARS-EN-ELM for solving ELM with an L_1 norm penalty together with an L_2 norm penalty is introduced. A comparative study on system performance is conducted between ELMs and SVMs. Experiment results indicate that, in comparison with SVMs, the ordinary ELM can achieve similar vigilance estimation accuracy with much less training time, but is very sensitive on the number of hidden nodes. After adding an L_1 norm penalty to the ELM, the modified ELM, called LARS-ELM, is still very sensitive on the number of hidden nodes. After adding an L_2 norm penalty to ELM, the modified ELM, called regularized-ELM, becomes stable and can achieve similar vigilance estimation accuracy with much less training time. Finally, L_1 norm penalty together with L_2 norm penalty are added to the ELM, and LARS-EN-ELM and TROP-ELM are implemented. Experiment results show that both LARS-EN-ELM and TROP-ELM require more training time than regularized-ELM but use less training time than SVMs. The accuracy of TROP-ELM is similar with that of SVMs, while the accuracy of LARS-EN-ELM is better than that of SVMs. Considering the stability, accuracy and total training time, regularized-ELM and LARS-EN-ELM are ranked into the top level among the six algorithms. Both of them have outperformed SVMs. Regularized-ELM has a much faster training speed, while LARS-EN-ELM has better vigilance estimation accuracy.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grant no. 90820018), the National Basic Research Program of China (Grant no. 2009CB320901), and the European Union Seventh Framework Programme (Grant no. 247619). The authors would like to thank Prof. Guang-Bin Huang, Nanyang Technological University, Singapore, for providing the ELM source code and the helpful discussions on regularized-ELM.

References

[1] R. Molloy, R. Parasuraman, Monitoring an automated system for a single failure: vigilance and task complexity effects, Hum. Factors 8 (1996) 311–322.
[2] M.B. Weinger, Vigilance, boredom, and sleepiness, J. Clin. Monit. Comput. 15 (1999) 549–552.
[3] T. Jung, S. Makeig, Estimating level of alertness from EEG, Eng. Med. Biol. Soc. 2 (1994) 1103–1104.
[4] S. Makeig, M. Inlow, Lapses in alertness: coherence of fluctuations in performance and EEG spectrum, Electroencephalogr. Clin. Neurophysiol. 86 (1) (1993) 23–35.
[5] S. Makeig, T. Jung, Changes in alertness are a principal component of variance in the EEG spectrum, Neuroreport 7 (1995) 213–216.
[6] T. Jung, S. Makeig, M. Stensmo, Estimating alertness from the EEG power spectrum, IEEE Trans. Biomed. Eng. 44 (1997) 60–69.
[7] D. Loewy, K. Campbell, D. de Lugt, M. Elton, The mismatch negativity during natural sleep: intensity deviants, Clin. Neurophysiol. 111 (2000) 863–872.
[8] R. Huang, T. Jung, A. Delorme, S. Makeig, Tonic and phasic electroencephalographic dynamics during continuous compensatory tracking, Neuroimage 39 (4) (2008) 1896–1909.
[9] C. Lin, K. Huang, C. Chao, J. Chen, T. Chiu, L. Ko, T. Jung, Tonic and phasic EEG and behavioral changes induced by arousing feedback, Neuroimage 52 (2) (2010) 633–642.
[10] C. Lin, R. Wu, S. Liang, W. Chao, Y. Chen, T. Jung, EEG-based drowsiness estimation for safety driving using independent component analysis, IEEE Trans. Circuits. Syst. I 52 (12) (2005) 2726–2738.
[11] M. Johns, A. Tucker, R. Chapman, A new method for monitoring the drowsiness of drivers, in: Proceedings of the International Conference on Fatigue Management in Transportation Operations, 2005, pp. 11–15.
[12] C. Lin, L. Ko, I. Chung, T. Huang, Y. Chen, T. Jun, S. Liang, Adaptive EEG-based alertness estimation system by using ICA-based fuzzy neural networks, IEEE Trans. Circuits Syst I 53 (11) (2006) 2469–2476.
[13] P.R. Davidson, R.D. Jones, M.T.R. Peiris, EEG-based lapse detection with high temporal resolution, IEEE Trans. Biomed. Eng. 54 (5) (2007) 832–839.
[14] L.C. Shi, B.L. Lu, Dynamic clustering for vigilance analysis based on EEG, in: Proceedings of the IEEE International Conference of Engineering in Medicine and Biology Society, Vancouver, Canada, 2008, pp. 54–57.
[15] L.C. Shi, B.L. Lu, Off-line and on-line vigilance estimation based on linear dynamical system and manifold learning, in: Proceedings of the IEEE International Conference of Engineering in Medicine and Biology Society, Buenos Aires, Argentina, 2010, pp. 6587–6590.
[16] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (2006) 489–501.
[17] G.-B. Huang, L. Chen, C.-K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, IEEE Trans. Neural Networks 17 (4) (2006) 879–892.
[18] G.-B. Huang, L. Chen, Convex incremental extreme learning machine, Neurocomputing 70 (2007) 3056–3062.
[19] G.-B. Huang, L. Chen, Enhanced random search based incremental extreme learning machine, Neurocomputing 71 (2008) 3460–3468.
[20] G.-B. Huang, D.-H. Wang, Y. Lan, Extreme learning machines: a survey, Int. J. Mach. Learn. Cyber. 2 (2011) 107–122.
[21] G-B Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multi-class classification, IEEE Trans. Syst. Man Cyber. Part B 42 (2012) 513–529.
[22] Y. Miche, P. Bas, C. Jutten, O. Simula, A. Lendasse, A methodology for building regression models using extreme learning machine: Op-elm, in: Proceedings of the 16th European Symposium on Artificial Neural Networks, Bruges, Belgium, 2008, pp. 1–6.
[23] W. Deng, Q. Zheng, L. Chen, Regularized extreme learning machine, in: Proceedings of IEEE Symposium on Computational Intelligence and Data Mining, 2009, pp. 389–395.
[24] Y. Miche, M. Heeswijk, P. Bas, O. Simula, A. Lendasse, TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization, Neurocomputing (2011)http://dx.doi.org/10.1016/j.neucom.2010.12.042.
[25] N.-Y. Liang, P. Saratchandran, G.-B. Huang, N. Sundararajan, Classification of mental tasks from eeg signals using extreme learning machines, Int. J. Neural Syst. 16 (2006) 29–38.
[26] A.E. Hoerl, R.W. Kennard, Ridge regression: biased estimation for nonorthogonal problems, Technometrics 12 (1) (1970) 55–67.
[27] Q. Liu, Q. He, Z. Shi, Extreme support vector machine classifier, in: Proceedings of the 12th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, 2008, pp. 222–233.
[28] O. Mangasarian, E. Wild, Proximal support vector machine classifiers, in: Proceedings of Knowledge Discovery and Data Mining, 2001, pp. 77–86.
[29] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, Least angle regression, Ann. Statist. 32 (2) (2004) 407–499.
[30] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc. Ser. B 67 (2) (2005) 301–320.
[31] V. Vapnik, S.E. Golowich, A.J. Smola, Support vector method for function approximation, regression estimation, and signal processing, Adv. Neural Inf. Process. Syst. 9 (1996) 281–287.
[32] A.J. Smola, B. Scholkopf, A tutorial on support vector regression, Statist. Comput. 14 (3) (2004) 199–222.
[33] L.C. Shi, X.W. Wang, H. Sun, B.L. Lu, BCMI-EEG-Database: an EEG database for continuous vigilance estimation, 〈http://bcmi.sjtu.edu.cn/~Vigilance_EEG_database〉, 2011.
[34] C.C. Chang, C.J. Lin, LIBSVM: a library for support vector machines, 〈http://www.csie.ntu.edu.tw/~cjlin/libsvm〉, 2001.



Li-Chen Shi received the B.S. degree in Computer Science from Fudan University in 2004. He is currently a Ph.D. candidate in the Center for Brain-like Computing and Machine Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University. His research interests include pattern recognition, signal processing, and EEG-based Brain-Computer Interface.

Bao-Liang Lu is a professor of Computer Science and Engineering at Shanghai Jiao Tong University (SJTU). He received his B.S. degree in instrument and control engineering from Qingdao University of Science and Technology, China, in 1982, the M.S. degree in computer science and engineering from Northwestern Polytechnical University, China, in 1989, and the Dr. Eng. degree in electrical engineering from Kyoto University, Japan, in 1994. From 1982 to 1986, he was with the Qingdao University of Science and Technology. From April 1994 to March 1999, he was a Frontier Researcher at the Bio-Mimetic Control Research Center, the Institute of Physical and Chemical Research (RIKEN), Japan. From April 1999 to August 2002, he was a Research Scientist at the RIKEN Brain Science Institute. Since August 2002, he has been a full Professor at the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His research interests include brain-like computing, neural network, machine learning, pattern recognition, brain–computer interface, computational linguistics, and computational biology and bioinformatics. He was past President of the Asia Pacific Neural Network Assembly (APNNA) and the general Chair of the 18th International Conference on Neural Information Processing (ICONIP2011). He is an Associate Editor of the Neural Networks and a senior member of the IEEE.