Neurocomputing 116 (2013) 87–93

Contents lists available at SciVerse ScienceDirect

Neurocomputing

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

An improved evolutionary extreme learning machine based on particle swarm optimization

Fei Han [a],*, Hai-Fen Yao [a], Qing-Hua Ling [b]

[a] School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang, Jiangsu 212013, China
[b] School of Computer Science and Engineering, Jiangsu University of Science and Technology, Zhenjiang, Jiangsu 212003, China

ARTICLE INFO

Available online 8 October 2012

Keywords:
Extreme learning machine
Particle swarm optimization
Generalization performance
Convergence rate

ABSTRACT

Recently Extreme Learning Machine (ELM) for single-hidden-layer feedforward neural networks (SLFN) has been attracting attentions for its faster learning speed and better generalization performance than those of traditional gradient-based learning algorithms. However, ELM may need high number of hidden neurons and lead to ill-condition problem due to the random determination of the input weights and hidden biases. In this paper, a hybrid learning algorithm is proposed to overcome the drawbacks of ELM, which uses an improved particle swarm optimization (PSO) algorithm to select the input weights and hidden biases and Moore–Penrose (MP) generalized inverse to analytically determine the output weights. In order to obtain optimal SLFN, the improved PSO optimizes the input weights and hidden biases according to not only the root mean squared error (RMSE) on validation set but also the norm of the output weights. The proposed algorithm has better generalization performance than traditional ELM and other evolutionary ELMs, and the conditioning of the SLFN trained by the proposed algorithm is also improved. Experiment results have verified the efficiency and effectiveness of the proposed method.

1. Introduction

Single-hidden-layer feedforward neural networks (SLFN) have been proven to be universal approximator and are widely used for regression and classification problem [1]. Gradient-based learning algorithms such as backpropagation (BP) and Levenberg–Marquardt (LM), have been extensively used in the training of SLFN due to their reasonable performance [2–4]. However, these gradient-based algorithms are apt to be trapped in local minima and very time-consuming due to improper learning steps in many applications [5–9]. Moreover, they only consider the desired input/output information without the structure properties of network, thus their generalization performance are limited [10–12].

In order to overcome the drawbacks of gradient-based methods, extreme learning machine (ELM) for SLFN was proposed in 2004 [13]. ELM randomly chooses the input weights and hidden biases and analytically determines the output weights of SLFN through simple generalized inverse operation of the hidden layer output matrix. ELM not only learns much faster with better generalization performance than traditional gradient-based learning algorithms but also avoids many difficulties faced by gradient-based learning methods such as stopping criteria, learning rate, learning epochs, and local minima [14]. However, it is also found that ELM tends to require more hidden neurons than traditional gradient-based learning algorithms as well as result in ill-condition problem due to randomly selecting input weights and hidden biases [14,15].

In the literature [14], a evolutionary ELM (E-ELM) was proposed which used the differential evolutionary algorithm to select the input weights and Moore–Penrose (MP) generalized inverse to analytically determine the output weights. The evolutionary ELM was able to achieve good generalization performance with much more compact networks. In the literature [15], in order to improve the conditioning of traditional ELM, an improved ELM was proposed by selecting input weights for an ELM with linear hidden neurons. This approach maintains testing accuracy with stable condition, but it was only limited to ELM with linear hidden neurons.

In recent years, particle swarm optimization (PSO) has been used increasingly as an effective technique for search global minima [16]. PSO has no complicated evolutionary operators and less parameter need to adjust [17]. Therefore, the hybrids of PSO and ELM should be promising for training feedforward neural networks. In the literature [18], particle swarm optimization (PSO) [19,20] was used to optimize the input weights and hidden biases of the SLFN to solve some prediction problems, which mainly encoded the boundary conditions into PSO to improve the performance of ELM.

* Corresponding author. Tel.: +86 511 88790321 533; fax: +86 511 88780371.
E-mail addresses: hanfei@ujs.edu.cn (F. Han), yao_hf0611@163.com (H.-F. Yao), lingee_2000@163.com (Q.-H. Ling).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2011.12.062

In this paper, a new method combining ELM with an improved PSO called as IPSO-ELM is proposed. In the proposed algorithm, the improved PSO is used to optimize the input weights and hidden biases, and the MP generalized inverse is used to analytically calculate the output weights. The improved PSO mainly focuses on decreasing the norm of the output weights of the SLFN and constraining the input weights and hidden biases within a reasonable range to improve the convergence performance of ELM.

The rest of paper is organized as follows. Section 2 introduces some preliminaries of ELM and PSO. The proposed method is proposed in Section 3. In Section 4, experiment results and discussion for regression and classification problems are given to demonstrate the effectiveness of the proposed algorithm. Finally, the concluding remarks are offered in Section 5.

2. Preliminaries

2.1. Extreme learning machine

For N arbitrary distinct samples (x_i,t_i), where $x_i=[x_{i1},x_{i2},\ldots x_{in}]^T \in R^n$, $t_i=[t_{i1},t_{i2},\ldots,t_{im}]^T \in R^m$. A SLFN with H hidden neurons and activation function $g(\cdot)$ can approximate these N samples with zero error. This means that

$$Hwo=T \tag{1}$$

where

$$H(wh_1,\ldots,wh_H,b_1,\ldots,b_H,x_1,\ldots,x_N) = \begin{bmatrix} g(wh_1\bullet x_1+b_1) & \cdots & g(wh_H\bullet x_1+b_H) \\ \vdots & \cdots & \vdots \\ g(wh_1\bullet x_N+b_1) & \ldots & g(wh_H\bullet x_N+b_H) \end{bmatrix}_{N\times H}, \quad wo=\begin{bmatrix} wo_1^T \\ \vdots \\ wo_H^T \end{bmatrix}_{H\times m} \quad T=\begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N\times m} \tag{2}$$

where $wh_i=[wh_{i1},wh_{i2},\ldots,wh_{in}]^T$ is the weight vector connecting the i-th hidden neuron and the input neurons, $wo_i=[wo_{i1}, wo_{i2},\ldots,wo_{im}]^T$ is the weight vector connecting the i-th hidden neuron and the output neurons, and b_i is the bias of the i-th hidden neuron.

Thus, the determination of the output weights is to find the least-square (LS) solution to the given linear system. The minimum norm LS solution to the linear system (1) is

$$wo=H^+T \tag{3}$$

where H^+ is the MP generalized inverse of matrix H. The minimum norm LS solution is unique and has the smallest norm among all the LS solutions. ELM using such MP inverse method tends to obtain good generalization performance [14]. Since the solution is obtained by an analytical method and all the parameters of SLFN need not be adjusted, ELM converges much faster than gradient-based algorithms.

2.2. Particle swarm optimization

Particle swarm optimization (PSO) is a population-based stochastic optimization technique developed by Eberhart and Kennedy [19,20]. PSO simulates the social behavior of organisms, such as birds in a flock or fishes in a school, and can be described as an automatically evolving system.

PSO works by initializing a flock of birds randomly over the searching space, where every bird is called as a "particle". These particles fly with a certain velocity and find the global best position after some iterations. At every iteration, each particle adjusts its velocity vector according to its momentum and the influence of its best position (P_b) as well as the best position of its neighbors (P_g), and then a new position that the particle is to fly is obtained. Supposing the dimension of searching space is D, the total number of particles is n, the position of the i-th particle can be expressed as vector $X_i=(x_{i1},x_{i2},\ldots,x_{iD})$; the best position of the i-th particle searching until now is denoted as $P_{ib}=(p_{i1},p_{i2},\ldots,p_{iD})$, and the best position of all particles searching until now is denoted as vector $P_g=(p_{g1},p_{g2},\ldots,p_{gD})$; the velocity of the i-th particle is represented as vector $V_i=(v_{i1},v_{i2},\ldots,v_{iD})$. Then the original PSO [19,20] is described as

$$\begin{aligned} v_{id}(t+1) = v_{id}(t)+c_1^*rand()^*[p_{id}(t)-x_{id}(t)] \\ +c_2^*rand()^*[p_{gd}(t)-x_{id}(t)] \end{aligned} \tag{4}$$

$$x_{id}(t+1)=x_{id}(t)+v_{id}(t+1)\ 1\le i\le n,\ 1\le d\le D \tag{5}$$

where c_1, c_2 are the acceleration constants with positive values; $rand()$ is a random number between 0 and 1. In addition to the c_1 and c_2 parameters, the implementation of the original algorithm also requires to place a limit on the velocity (v_{max}).

The adaptive particle swarm optimization (APSO) algorithm is based on the original PSO algorithm, proposed by Shi & Eberhart [21]. The APSO can be described as follows:

$$\begin{aligned} v_{id}(t+1) = w^*v_{id}(t)+c_1^*rand()^*[p_{id}(t)-x_{id}(t)] \\ +c_2^*rand()^*[p_{gd}(t)-x_{id}(t)] \end{aligned} \tag{6}$$

where w is a new inertial weight. The parameter can reduce gradually as the generation increases according to $w(t)=w_{max}-t(w_{max}-w_{min})/itermax$ where w_{max}, w_{min} and $itermax$ are the initial inertial weight, the final inertial weight and the maximum searching generations, respectively. The APSO is more effective than original PSO, because the searching space reduces step by step.

3. The improved extreme learning machine (IPSO-ELM)

ELM need not spend much time to tune the input weights and hidden biases of the SLFN by randomly choosing these parameters. Since the output weights are computed based on the input weights and hidden biases, there inevitably exists a set of nonoptimal or unnecessary input weights and hidden biases. Thus, two problems may be resulted from randomly choosing parameters in ELM. One is that ELM may require more hidden neurons than conventional gradient-based learning algorithms in some applications, which may make ELM respond slowly to unknown testing data [14]. The other is that an ill-conditioned hidden output matrix H may arise, especially when a large number of hidden neurons are used, which may cause worse generalization performance [15].

The conditioning of a matrix can be qualitatively characterized by condition number. According to the literatures [15,22], the 2-norm condition number of the matrix H can be computed as follows:

$$K_2(H)=\sqrt{\frac{\lambda_{\max}\left(H^TH\right)}{\lambda_{\min}\left(H^TH\right)}} \tag{7}$$

where $\lambda_{\max}\left(H^TH\right)$ and $\lambda_{\min}\left(H^TH\right)$ are the largest and smallest eigenvalues of the matrix H^TH. Condition number is a good indicator of a system conditioning which shows how close a system is to be ill-conditioned.

In this paper, an improved approach named IPSO-ELM combining PSO with ELM is proposed. This improved ELM uses PSO to select the input weights to improve the generalization performance and the conditioning of the SLFN. The detailed steps of the proposed method are as follows:

Firstly, the swarm is randomly generated. Each particle in the swarm is composed of a set of input weights and hidden biases:

$P_i=[wh_{11},wh_{12},\ldots,wh_{1n},\ \ wh_{21},wh_{22},\ldots,wh_{2n},\ \ wh_{H1},wh_{H2},\ldots,wh_{Hn},$ $b_1,b_2,\ldots,b_H]$. All components in the particle are randomly initialized within the range of $[-1,1]$.

Secondly, for each particle, the corresponding output weights are computed according to Eq. (3). Then the fitness of each particle is evaluated. In order to avoid overfitting of the SLFN, the fitness of each particle is adopted as the root mean squared error (RMSE) on the validation set only instead of the whole training set as used in [14,23]:

$$f() = \sqrt{\frac{\sum_{j=1}^{n_v} \left|\left| \sum_{i=1}^{H} wo_i g(wh_i \bullet x_j + b_j) - t_j \right|\right|_2^2}{n_v}} \tag{8}$$

where n_v is the number of the validation samples.

Thirdly, with the fitness of all particles, the P_bs for all particles and the P_g for the swarm are computed. As analyzed by Zhu [14] and Bartlett [24], neural networks tend to have better generalization performance with the weights of smaller norm. Therefore, in order to further improve generalization performance, the norm of output weights along with the RMSE on the validation set are considered to determine the P_bs for all particles and the P_g for the swarm. The corresponding details are described as follows:

$$P_{ib} = \begin{cases} P_i & (f(P_{ib}) - f(P_i) > \eta f(P_{ib}))\ or\ (|f(P_{ib}) - f(P_i)| < \eta f(P_{ib}) \\ & and\ ||wo_{P_i}|| < ||wo_{P_{ib}}||) \\ P_{ib} & else \end{cases} \tag{9}$$

$$P_g = \begin{cases} P_i & (f(P_g) - f(P_i) > \eta f(P_g))\ or\ (|f(P_g) - f(P_i)|) < \eta f(P_g) \\ & and\ ||wo_{P_i}|| < ||wo_{P_g}|| \\ P_g & else \end{cases} \tag{10}$$

where $f(P_i)$, $f(P_{ib})$ and $f(P_g)$ are the corresponding fitness values for the i-th particle, the best position of the i-th particle and global best position of all particles, respectively. wo_{P_i}, $wo_{P_{ib}}$ and wo_{P_g} are the corresponding output weights obtained by MP generalized inverse when the input weights are set as the i-th particle, the best position of the i-th particle and global best position of all particles, respectively. $\eta > 0$ is a tolerance rate.

Fourthly, each particle updates its position according to Eqs. (5) and (6), and the new population is generated. According to the literatures [5,13,18], all components in the particle should be limited within the range of $[-1,1]$. This *a priori* information is encoded in the improved PSO by the following equation:

$$x_{id}(t+1) = \begin{cases} 1 & x_{id}(t+1) > 1 \\ -1 & x_{id}(t+1) < -1 \end{cases}, 1 \le i \le n, 1 \le d \le D \tag{11}$$

Finally, the above optimization process is repeated until the goal is met or the maximum optimization epochs are completed. Thus the ELM with the optimal input weights and hidden biases are obtained, and then the optimal ELM is applied to the testing data.

From Eqs. (9) and (10), the proposed method tends to select those input weights and hidden biases which results in smaller norm of the corresponding output weights of the SLFN. The smaller the norm value of the output weights of the SLFN is, the smaller is the condition value of the corresponding hidden output matrix.

4. Experiment results and discussion

In this section, the IPSO-ELM are compared with PSO-ELM [18], E-ELM [14], LM (one of the fastest implementation of BP algorithms and is provided in the neural networks tools box of MATLAB.) and ELM. The parameters in all algorithms in all experiments are determined by trial and error. For IPSO-ELM, PSO-ELM and E-ELM, the maximum optimization epochs are 20, and the population size is 200. For IPSO-ELM and PSO-ELM, the initial inertial weight, w_{max}, and the final inertial weight, w_{min}, are selected as 1.2 and 0.4 in all experiments. All the results shown in this paper are the mean values of 50 trails. We conduct simulations on function approximation and benchmark classification problems. All the programs are run in MATLAB 7.0 environment.

4.1. Function approximation

In this section, all the five algorithms (IPSO-ELM, PSO-ELM, E-ELM, LM and ELM) are used to approximate the 'SinC' function:

$$y = \begin{cases} \sin(x)/x & x \neq 0 \\ 1 & x = 0 \end{cases} \tag{12}$$

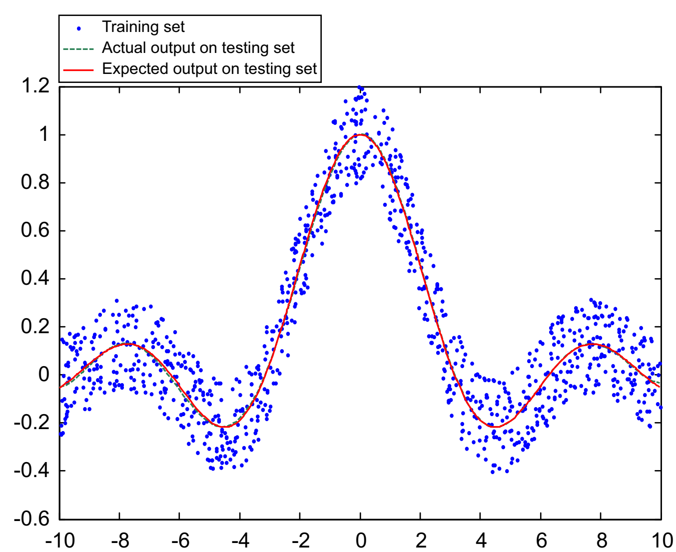


Fig. 1. Outputs of the IPSO-ELM learning algorithm.

Table 1
The performance of approximating Sin C function with five learning algorithms.

Algorithms	Training RMSE	Testing RMSE	Testing Dev	Cpu time Training	Hidden nodes	Condition	Norm
LM	0.1137	0.0223	0.0039	0.3079 s	15	/	/
ELM	0.1138	0.0207	0.0038	0.0078 s	30	2.9462e ± 14	3.2015e ± 9
E-ELM	0.1156	0.0168	0.0034	9.4652 s	10	2.1416e ± 6	3.9983e ± 4
PSO-ELM	0.1154	0.0174	0.0033	10.2662 s	10	1.2710e ± 7	2.3159e ± 5
IPSO-ELM	0.1155	0.0163	0.0028	9.5808 s	10	1.7569e ± 6	2.3355e ± 4

A training set (x_i,y_i) and testing set (x_j,y_j) with 1000 data, respectively, are created where x_is and x_js are uniformly randomly distributed on the interval $(-10,10)$. Moreover, large uniform noise distributed in $[-0.2,0.2]$ has been added to all the training samples while testing data remain noise-free. The corresponding results are shown in Table 1 and Fig. 1.

From Table 1, some conclusions can be drawn as follows:

Firstly, all ELMs have smaller testing RMSE than LM. Only ten hidden nodes are assigned for E-ELM, PSO-ELM and IPSO-ELM, while 15 and 30 hidden nodes are used in LM and ELM, respectively.

Secondly, compared with ELM, the E-ELM, PSO-ELM and IPSO-ELM obtained smaller testing RMSE with less hidden nodes. This indicates that the E-ELM, PSO-ELM and IPSO-ELM can achieve good generalization performance with much more compact networks. Training Cpu time for the E-ELM, PSO-ELM and IPSO-ELM is more than that of the other algorithms, which is mainly spent on the selection of input weights.

Thirdly, the condition value of the hidden matrix H in the E-ELM, PSO-ELM and IPSO-ELM is much less than those of ELM. This shows that the networks trained by the E-ELM, PSO-ELM and IPSO-ELM are more well-conditioned than that of ELM. From the norm value of the output weights, it can be concluded that the E-ELM, PSO-ELM and IPSO-ELM have better generalization than ELM. It is also found that the smaller the norm of the output weights is, the smaller the condition value of the corresponding hidden output matrix is.

Finally, compared with E-ELM and PSO-ELM, IPSO-ELM has smaller RMSE, Dev., condition and norm value, which shows that the IPSO-ELM is superior to the E-ELM and PSO-ELM on approximating the SinC function.

4.2. Benchmark classification

The performance of IPSO-ELM is also tested on three real benchmark classification problems. The specification of these problems is listed in Table 2. The training, testing and validation datasets are randomly regenerated at each trial of simulations according to Table 2 for all the algorithms. The corresponding performance of five algorithms on three classification problems is listed in Tables 3–5.

From Tables 3–5, we can find that the E-ELM, PSO-ELM and IPSO-ELM need much more time to train SLFN than the LM and ELM. There is no much difference in training time among the IPSO-ELM, PSO-ELM and the E-ELM. Training time of the IPSO-ELM is slightly less than that of the E-ELM and PSO-ELM on the diabetes and satellite image classification, while the IPSO-ELM

Table 2
Specification of three classification problems.

Names	Attributes	Classes	Number of observations		
			Training	Testing	Validation
Diabetes	8	2	252	258	258
Satellite image	36	7	4435	1000	1000
Image segmentation	19	7	1500	405	405

Table 3
Performance of five algorithms on diabetes classification problem.

Algorithms	Cpu time	Accuracy(%)		Hidden neurons	Norm	Condition
	Training	Training	Testing+Dev.			
LM	1.7386 s	95.25	68.49 ± 0.0312	20	/	/
ELM	0.0061 s	81.45	75.19 ± 0.0282	30	382.5409	7.6265e+3
E-ELM	12.2287 s	76.98	75.51 ± 0.0273	10	62.5096	1.2576e+3
PSO-ELM	12.4323 s	77.15	76.38 ± 0.0251	10	232.9444	3.1850e+3
IPSO-ELM	12.1810 s	77.70	76.40 ± 0.0241	10	72.4941	1.0556e+3

Table 4
Performance of five algorithms on Satellite image classification problem.

Algorithms	Cpu time	Accuracy(%)		Hidden neurons	Norm	Condition
	Training	Training	Testing+Dev.			
LM	Out of memory					
ELM	2.0034 s	89.50	88.12 ± 0.0103	300	28.6327	3235.5
E-ELM	3.111e+3 s	88.39	87.25 ± 0.0098	90	2.4371	129.0319
PSO-ELM	3.213e+3 s	88.21	87.18 ± 0.0106	90	15.3416	808.9574
IPSO-ELM	3.105e+3 s	88.03	87.30 ± 0.0115	90	14.7202	630.1026

Table 5
Performance of five algorithms on Image segmentation classification problem.

Algorithms	Cpu time	Accuracy(%)		Hidden neurons	Norm	Condition
	Training	Training	Testing+Dev.			
LM	599.7319 s	99.64	97.00 ± 0.0092	120	/	/
ELM	0.4116 s	96.46	92.16 ± 0.0053	150	163.8552	1.2811e+4
E-ELM	1.1111e+3 s	95.61	94.51 ± 0.0125	100	52.6075	1.0254e+4
PSO-ELM	1.3231e+3 s	95.94	94.12 ± 0.0130	100	126.0788	7.6494e+3
IPSO-ELM	1.2731e+3 s	95.33	94.62 ± 0.0123	100	94.0883	5.113e+3

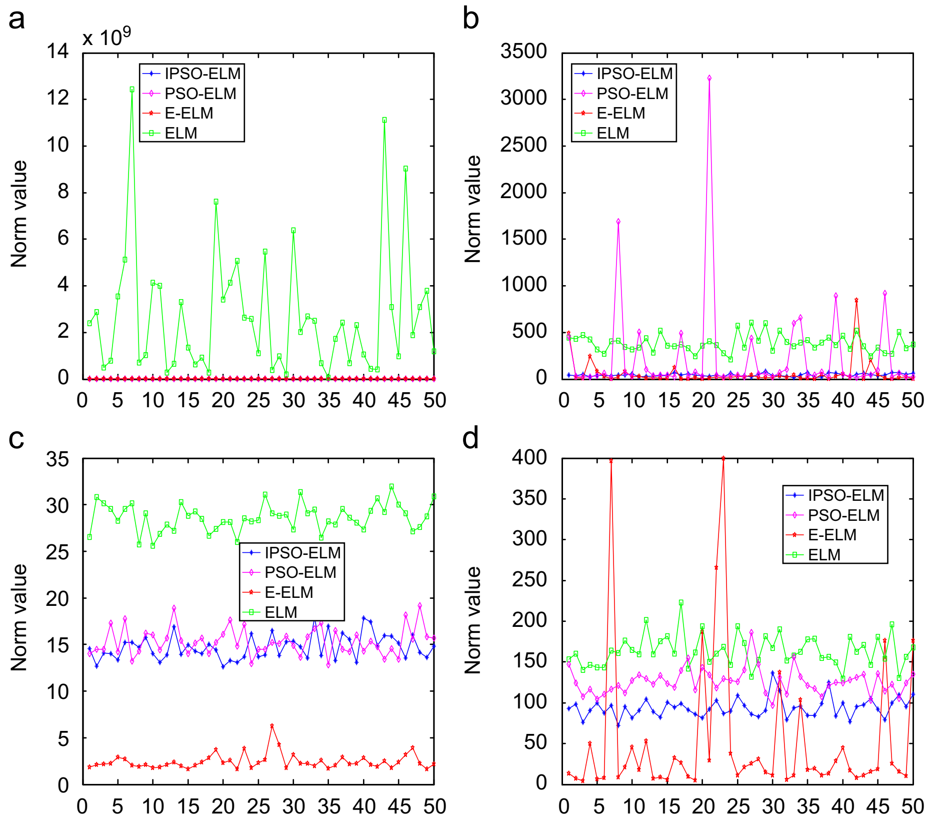


Fig. 2. The norm value of the output weights with four ELMs (a) function approximation, (b) diabetes, (c) satellite image and (d) segmentation image.

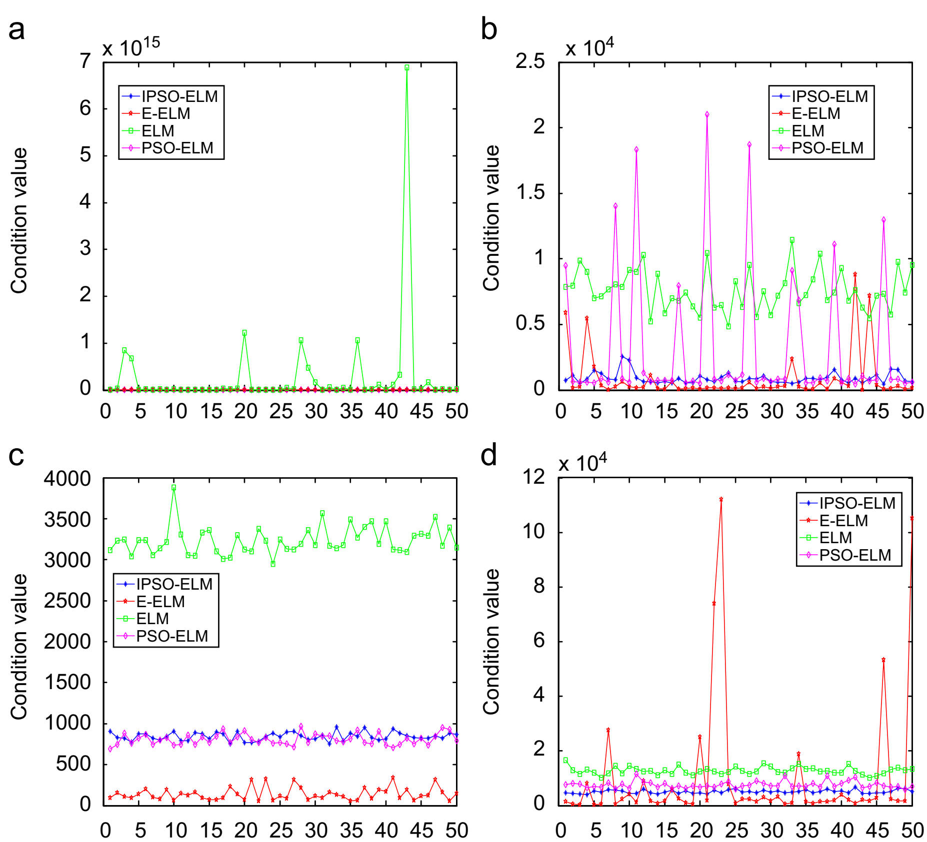


Fig. 3. The condition value of the SLFN obtained by four ELMs (a) function approximation, (b) diabetes, (c) satellite image and (d) segmentation image.

spends most of the time in training the SLFN on image segmentation classification than other algorithms except PSO-ELM.

From Table 3, the proposed algorithm achieves best testing accuracy with least hidden neurons. From Table 4, ELM obtains the highest testing accuracy, but it requires 300 hidden nodes. Unfortunately, the training of LM on satellite image case could not be completed as it ran out of memory. From Table 5, the IPSO-ELM obtains higher testing accuracy with 100 hidden neurons than other learning algorithms except LM.

From Tables 3–5, the IPSO-ELM, PSO-ELM and E-ELM are able to obtain the smaller norm of the output weights than the ELM, and they also train the SLFN in a better condition with smaller condition number than the ELM. Moreover, the condition value of the corresponding hidden output matrix has a downward trend as the norm value of the output weights of the SLFN decreases.

From Table 1, the difference between the training RMSE and testing RMSE of the IPSO-ELM on function approximation is the most in those of all algorithms. From Tables 3–5, the difference between the training accuracy and testing accuracy of the IPSO-ELM on three benchmark classification problems is almost the least in those of all algorithms. This also verifies that the proposed algorithm has the best generalization performance in all algorithms.

Moreover, in order to test the performance of the IPSO-ELM on improving the SLFN's conditioning and reducing the norm of the output weights in detail, the corresponding norm and condition values for four ELMs on the function approximation and benchmark classification problems are shown in Figs. 2 and 3, respectively.

From Fig. 2, the norm value of the output weights obtained by the IPSO-ELM, PSO-ELM and E-ELM is almost less than that obtained by the ELM on all four cases in each trial except the E-ELM and PSO-ELM on segmentation image classification and diabetes classification. The norm value obtained by the IPSO-ELM is very steady in all cases, while the norm value curves for the ELM, PSO-ELM and E-ELM contain several sharp spiking in some cases.

Similarly, from Fig. 3, the condition value of hidden output matrix in the IPSO-ELM, PSO-ELM and E-ELM is smaller than that in the ELM, and the condition curve for the IPSO-ELM is more stable than the other three ELMs.

Below, we discuss the effects of the parameter, the tolerance rate, η, with the IPSO-ELM for the diabetes and image segmentation classification problems. The training, validation and testing set are kept unchanged, and the tolerance rate is selected from [0.01, 0.2] at identically spaced intervals. Fig. 4 shows the corresponding simulation results. From Fig. 4(a), the testing accuracy has a decreasing trend as the tolerance rate increases on the diabetes classification and the testing accuracy is highest as the tolerance rate is 0.04, while the testing accuracy has not been changed significantly as the tolerance rate increases on image segmentation classification. From Fig. 4(b), the norm value of the output weights have a downward trend as the tolerance rate increases on both of the two classification problems. From Fig. 4(c), the condition value is not sensitive to the tolerance rate on diabetes classification, while the condition value reduces slowly and the corresponding curves are zigzag on image segmentation classification.

In summary, the IPSO-ELM achieves better performance by using PSO to select the input weights of the SLFN than the ELM, E-ELM, PSO-ELM and LM algorithms. Moreover, the proposed algorithm is most stable in the five learning algorithms.

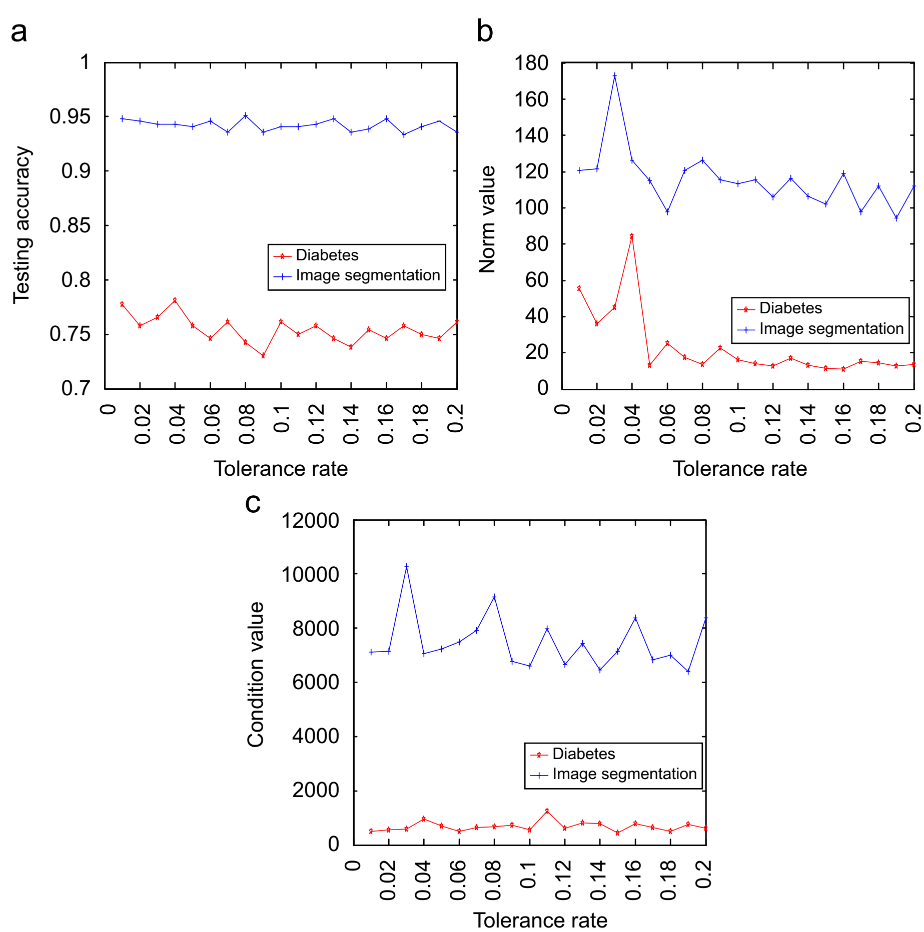


Fig. 4. The effects of the tolerance rate on two datasets with IPSO-ELM (a) testing accuracy, (b) norm value and (c) condition value.

5. Conclusions

In this paper, an improved evolutionary extreme learning machine based on particle swarm optimization (IPSO-ELM) was proposed. In the new algorithm, an improved PSO was used to optimize the input weights and hidden biases, and minimum norm least-square scheme to analytically determine the output weights. In the process of selecting the input weights and hidden biases, the improved PSO considered not only the RMSE on validation set but also the norm of the output weights as well as constrained the input weights and hidden biases within the reasonable range. The proposed algorithm has better generalization performance than the E-ELM, PSO-ELM, ELM and LM algorithms. The system trained by the IPSO-ELM is well-conditioned. The proposed algorithm is more stable than the E-ELM, PSO-ELM, ELM and LM algorithms. Finally, the simulation results also verified the effectiveness and efficiency of the proposed algorithm. Future research works will include how to apply the new learning algorithm to resolve more complex problem such as data mining on gene expression data.

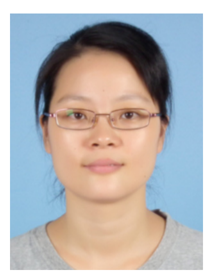
Acknowledgements

This work was supported by the National Natural Science Foundation of China (nos. 61271385, 60702056), Natural Science Foundation of Jiangsu Province (nos. BK2009197, BK2009199) and the Initial Foundation of Science Research of Jiangsu University (no. 07JDG033).

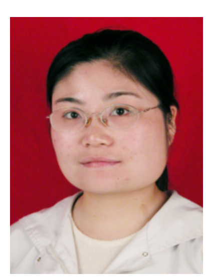
References

[1] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, A. Lendasse, OP-ELM: Optimally Pruned Extreme Learning Machine, IEEE Trans. Neural Networks 21 (1) (2010) 158–162.
[2] D.S. Huang, Radial basis probabilistic neural networks: model and application, Int. J. Pattern Recognition Artif. Intell. 13 (7) (1999) 1083–1101.
[3] S. Haykin, Neural Networks: A Comprehensive Foundation, second ed., Prentice-Hall, Englewood Cliffs, NJ, USA, 1999.
[4] D.S. Huang, Ji-Xiang Du, A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks, IEEE Trans. Neural Networks 19 (12) (2008) 2099–2115.
[5] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme Learning Machine: theory and applications, Neurocomputing 70 (1–3) (2006) 489–501.
[6] F. Han, D.S. Huang, Improved extreme learning machine for function approximation by encoding a priori information, Neurocomputing 69 (16–18) (2006) 2369–2373.
[7] D.S. Huang, A constructive approach for finding arbitrary roots of polynomials by neural networks, IEEE Trans. Neural Networks 15 (2) (2004) 477–491.
[8] D.S. Huang, Zheru Chi, Finding roots of arbitrary high order polynomials based on neural network recursive partitioning method, Sci. China Ser. F. Inf. Sci. 47 (2) (2004) 232–245.
[9] D.S. Huang, Horace H.S. Ip, Zheru Chi, A neural root finder of polynomials based on root moments, Neural Comput. 16 (8) (2004) 1721–1762.
[10] D.S. Huang, The local minima free condition of feedforward neural networks for outer-supervised learning, IEEE Trans on Syst. Man Cybern. 28 (3) (1998) 477–480.
[11] D.S. Huang, Horace H.S. Ip, K.C. Law, Zheru Chi, Zeroing polynomials using modified constrained neural network approach, IEEE Trans. Neural Networks 16 (3) (2005) 721–732.
[12] D.S. Huang, Horace H.S. Ip, Zheru Chi, H.S. Wong, Dilation method for finding close roots of polynomials based on constrained learning neural networks, Phys. Lett. A 309 (5–6) (2003) 443–451.
[13] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme Learning Machine: a new learning scheme of feedforward neural networks, 2004 International Joint Conference on Neural Networks (IJCNN'2004), Budapest, Hungary, July 25–29, 2004,pp. 985–990.
[14] Q.-Y. Zhu, A.K. Qin, P.N. Suganthan, G.-B. Huang, Evolutionary extreme learning machine, Pattern Recognition 38 (10) (2005) 1759–1763.
[15] G.P. Zhao, Z.Q. hen, C.Y. Miao, Z.H. Man, On improving the conditioning of extreme learning machine: a linear case, 7th International Conference on Information, Communications and Signal Processing, 2009 (ICICS2009), Maucu, China, December 8–10, 2009, pp. 1–5.
[16] Parrott Daniel, Li Xiaodong, Locating and tracking multiple dynamic optima by a particle swarm model using speciation, IEEE Trans. Evol. Comput. 10 (4) (2006) 440–458.
[17] W.B. Langdon, Riccardo Poli, Evolving problems to learn about particle swarm optimizers and other search algorithms, IEEE Trans. Evol. Comput. 11 (5) (2007) 561–578.
[18] You Xu, Yang Shu, Evolutionary extreme learning machine-based on particle swarm optimization, International Symposium on Neural Networks 2006 (ISNN2006), LNCS, vol. 3971, 2006, pp. 644–652.
[19] R.C. Eberhart, J. Kennedy, A new optimizer using particles swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human science, Nagoya, Japan, 1995, pp. 39–43.
[20] R.C. Eberhart, J. Kennedy, Particle swarm optimization, in: Proceeding of the IEEE International Conference on Neural Network, Perth, Australia, 1995, pp. 1942–1948.
[21] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of IEEE World Conference on Computation Intelligence, 1998, pp. 69–73.
[22] G.H. Golub, C.F. Van Loan, Matrix Computations, Johns Hopkins University Press, 1996.
[23] R. Ghosh, B. Verma, A hierarchical method for finding optimal architecture and weights using evolutionary least square based learning, Int. J. Neural Syst. 12 (1) (2003) 13–24.
[24] 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 Trans. Inf. Theory 44 (2) (1998) 525–536.

Fei Han received the M.A. degree from Hefei University of Technology in 2003 and the Ph.D. degree from University of Science and Technology of China in 2006. He is currently an Associate Professor of computer science at Jiangsu University. His principal research interests are intelligent computing and intelligent information processing, including neural networks, particle swarm optimization, and bioinformatics.



Hai-Fen Yao is a graduate student in the School of Computer Science and Telecommunication Engineering at Jiangsu University. She received the B.S degree from Suzhou University in 2000. Her research interests include neural networks, particle swarm optimization and intelligent computing.



Qing-Hua Ling received the B.S. degree from Nanjing Normal University in 2002 and the M.A. degree from Hefei Institute of Intelligent Machines, Chinese Academy of Sciences in 2005. She is currently a lecturer of computer science at Jiangsu University of Science and Technology. Her research interests include neural networks, particle swarm optimization and bioinformatics.