# An Efficient Optimization Method for Extreme Learning Machine Using Artificial Bee Colony

Chao Ma
College of Digital Media, Shenzhen Institute of Information Technology
Shenzhen, Guangdong 518172, China
billmach@163.com

*Journal of Digital Information Management*

***ABSTRACT:*** *Recently extreme learning machine (ELM) was proposed as a new learning method for single hidden-layer feedforward neural networks (SLFNs), it is not the same as traditional gradient based learning algorithm strategies as it can achieve good generalization performance as well as extremely fast learning speed. However, ELM may require large number of hidden neurons due to the random determination of the input weights and hidden biases, and there may exist a set of non-optimal parameters which lead ELM not be able to reach the global optimum in some cases. With the help of ideas that using a hybrid approach which takes advantage of the optimization method and ELM to train SLFNs, this study proposes a novel hybrid approach based on artificial bee colony (ABC) optimization method to optimize the ELM parameters, where the optimal input weights and biases of ELM are specified by the ABC approach and Moore-Penrose (MP) generalized inverse to analytically determine the output weights. The proposed algorithm, named ABC-ELM, is rigorously compared with the original ELM and other evolutionary ELM methods in different classification datasets. The obtained results clearly confirm that the proposed approach is more suitable for classification problems that we investigated, and it can not only achieve better generalization performance but be more robust with much more compact networks.*

**Subject Categories and Descriptors**
[**F.1.1 Models of Computation neural networks**] [**I.5 Patterm Recognition**]; Neural nets [**I.1.2 Algorithms**] [**G.1.6 Optimization**]

**General Terms**
Learing algorithms, Learning Machines, Neural Networks

**Keywords:** Single hidden-layer feedforward neural networks (SLFNs), Extreme learning machine, Artificial bee colony algorithm, Hybrid method

**Received:** 20 January 2017, Revised 14 March 2017, Accepted 22 March 2017

## 1. Introduction

Traditional learning algorithms with gradient descent based technique, such as back-propagation (BP) and its variant Levenberg-Marquardt (LM) have been widely used in the training of multilayer feedforward neural networks. The gradient descent based algorithm may converge usually slower than required time in training, since many iterative learning step are needed by such learning algorithm, and difficult setting of learning parameters such as stopping criteria, learning rate, learning epochs, which make it easily cause the over-fitting or trap in local minimum. Moreover, the activation functions adopted in these gradient descent based tuning methods need to be differentiable

Recently, a new and fast learning algorithm for single hidden-layer feedforward neural networks (SLFNs) called extreme learning machine (ELM) was proposed by Huang *et al*. [10]. In ELM, the input weights (connecting the input layer to the hidden layer) and hidden biases are generated randomly, and the output weights (connecting the hidden

layer to the output layer) are analytically determined by using Moore-Penrose (MP) generalized inverse. Because of the fast training speed and high generalization, ELM method has been applied to various learning domains [8, 11]. It increases the learning speed by means of randomly assigned the input weights and hidden biases rather than iteratively adjusting network parameters which is usually used by gradient descent based methods. However, due to randomly selected input weights and hidden biases, in such way ELM tends to require more hidden neurons than traditional tuning-based algorithms in many cases, which may make ELM respond slowly to unknown testing data. Besides, too small/large hidden neurons lead to the under-fitting/over-fitting, especially, an extremely large network brings about longer prediction response, unnecessary memory requirement and high cost. For ELM, the input weights and hidden biases given in the first training step is the most important part, and the impact of these parameters on the performance is the essence of ELM.

In order to deal with the above problems, some improved efforts have been proposed. In the literature, evolutionary ELM (E-ELM) was proposed by Zhu [28], which used the differential evolutionary algorithm to choose the optimal input weights and hidden biases of the SLFN by means of iteratively adjusting in the training process of the SLFN, repeated until the stopping criteria were satisfied. The evolutionary ELM was able to achieve good generalization performance with more compact networks. Xu *et al*. [26] introduced a hybrid particle swarm optimization (PSO) and ELM algorithm, and PSO was used to optimize the input weights and hidden biases of the SLFN to solve some prediction problems, which primarily encoded the boundary conditions into PSO to improve the performance of ELM. In the method proposed by Zhao [7], an improved ELM was used to select the input weights for ELM with linear hidden neurons. Recently, Sundararajan *et al*. [21] combined Integer Coded Genetic Algorithm (ICGA) and PSO, coupled with ELM for gene selection and cancer classification, where the algorithm (ICGA-PSO-ELM) was developed to handle sparse data and sample imbalance. Wang *et al*. [25] made a study on the impact of random weights between the input and hidden layer in ELM, and proved that the randomly assigned input weights do have some effects on the learning, and for many classification or regression tasks, these effects were positive.

In this paper, to obtain satisfactory parameters for ELM, an attempt was tried to use the artificial bee colony (ABC) for parameters training process of ELM algorithm. The reason for the ABC algorithm as the optimized tool is that it possesses the global search ability by introducing neighborhood source production mechanism, and it does not require external parameters such as cross over rate and mutation rate as in case of differential evolution, these parameters are hard to determine in prior. For optimizing some multidimensional numeric problems, the ABC algorithm outperforms other meta-heuristic algorithms including the genetic algorithm (GA), particle swarm algorithm (PSO) and evolutionary algorithm (EA) [13]. The advantage of using the ABC algorithm over other techniques is that it possesses memory, local searches, and solution improvement mechanisms, so it can be applied to identify a high quality optimal solution and offer a balance between complexity and performance for optimizing approximate effectiveness. Therefore, the integration of ABC and ELM should be promising for SLFNs training the ABC algorithm is used to optimize a set of parameters, including input weights and biases, which maybe obtain global optimal solutions to achieve higher classification rates in some classification cases. In this study, different from the previous works such as E-ELM and PSO-ELM, the proposed approach uses ABC method and the k-fold cross validation (CV) scheme to search for the optimal input weights and hidden biases, while the MP generalized inverse is used to analytically calculate the output weights. As demonstrated in the experimental results, compared with original ELM, PSO-ELM and E-ELM algorithm, the ABC is found to be a better training method, and the proposed algorithm achieves better performance for overcoming the above mentioned problems of ELM with much more compact and more robust networks.

The rest of the paper is organized as follows. In Section 2 and Section 3, we give a brief description of ELM and ABC method respectively. Section 4 proposes our algorithm ABC-ELM, including the whole optimization process of this hybrid method. In Section 5, the detailed experimental design is presented, and describes all the empirical results and discussion in Section 6. Finally, conclusions are summarized in Section 7.

## 2. Extreme Learning Machine

This section gives a brief description on extreme learning machine (ELM) algorithm. ELM is an algorithm originally developed for training single hidden-layer feedforward neural networks (SLFNs), and then extended to the "generalized" SLFNs [9]. It has been attracting the attentions from more and more researchers and has been applied into many fields, such as human face recognition [4, 19, 29], medical diagnosis [6, 16], credit scoring [15], time-series forecasting [20], etc.

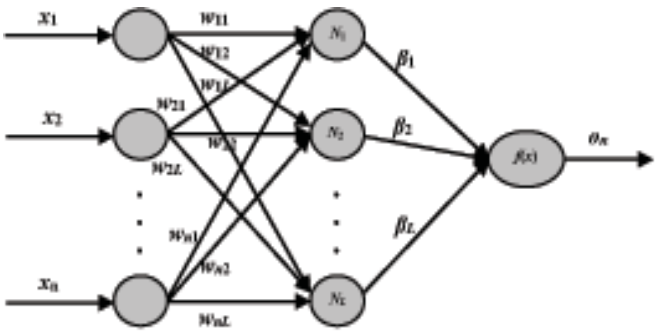

Figure 1. Structure of ELM

In ELM, the parameters of hidden neurons in network are randomly generated instead of tuned, and then fix the

nonlinearities of the network without iterations. Through the training process, the input space is mapped onto a new space by the hidden layer of ELM network with nonlinear transformation. A linear combination is performed on the new space. The parameters that have to be solved are the output weights, which are calculated by using the least-squares method. Fig. 1 shows the learning procedure and structure of ELM.

For $N$ arbitrary distinct samples $(\boldsymbol{x}_i, \boldsymbol{t}_i)$, $i = 1,..., N$, where $\boldsymbol{x}_i = [x_{i1}, x_{i2},..., x_{id}]^T \in R_n$ $\boldsymbol{t}_i = [t_{i1}, t_{i2},..., t_{im}]^T \in R_m$, are the $i$-th input pattern and associated target respectively. For a SLFN having $L$ nodes in the hidden neurons and activation function $g(x)$ are the mathematical modeled as

$$\sum_{j=1}^{L} \beta_j g\,(w_j . x_i + b_j) = o_i \qquad (1)$$

where $\boldsymbol{w}_j = [w_{j1}, w_{j2},..., w_{jn}]^T$ $(j = 1, 2,..., L)$ denotes the weight vector connecting $j$-th hidden neuron and input neurons, and $b_j$ is the bias of $j$-th hidden neuron, $w_j . x_i$ denotes the inner product of $w_j$ and $x_i$, $\beta_j = [\beta_{j1}, \beta_{j2},...\beta_{jL}]^T$ is the output weight connecting the $j$-th hidden neuron and the output neurons. $o_i$ is the corresponding actual output of $x_i$ in the network. The activation function $g(x)$ for additive nodes can be any bounded nonconstant piecewise continuous functions.

Huang had shown that standard SLFNs with $L$ hidden neurons can approximate these given $N$ samples with zero error, that is, $\sum_{i=1}^{N} ||o_i - t_i|| = 0$, it implies that there exist parameters $\boldsymbol{w}_i$, $\boldsymbol{x}_i$ and $b_i$ such that

$$\sum_{j=1}^{L} \beta_j g\,(w_j . x_i + b_j) = t_j, \qquad j = 1,2,...,L \qquad (2)$$

The above Eq. (2) can be written a linear system as

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

where $\boldsymbol{T} = [t_1, t_2,..., t_m]^T$ is the matrix of target (desired output).

$$H = H\,(w_1, w_2,..., w_L, b_1, b_2,..., b_L, x_1, x_2,..., x_n)$$

$$= \begin{bmatrix} g\,(w_1 . x_1 + b_1) & \cdots & g\,(w_L . x_1 + b_L) \\ \vdots & \cdots & \vdots \\ g\,(w_1 . x_N + b_1) & \cdots & g\,(w_L . x_N + b_L) \end{bmatrix}_{N \times L} \qquad (4)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{N \times m} \qquad \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \qquad (5)$$

where $\boldsymbol{H} = \{h_{ij}\}$ $(i = 1,..., N$ and $j = 1,..., L)$ is called the hidden layer output matrix of the neural network. The $j$-th column of $\boldsymbol{H}$ is $j$-th hidden neuron output vector with respect to inputs $x_1, x_2,..., x_n$ and the $i$-th row of $\boldsymbol{H}$ is the output vector of the hidden layer with respect to input $x_i$.

In most cases, the number of the hidden neurons is much less than the number of training samples, that is, $L \square N$.

There may not exit $w_i$, $b_i$ and $\beta_i$ $(i = 1,..., N)$ such that $H\beta = T$. Therefore, the determination of the output weights is computed by means of the least-square method. The minimum norm least-square solution to the linear system is

$$\beta' = H + T \qquad (5)$$

where $H^+$ is the Moore-Penrose generalized inverse [18] of the hidden layer output matrix $\boldsymbol{H}$. The solution $\beta'$ obtained by Eq. (5) is one of the least-squares solutions of system (3), which is unique and has the smallest norm over all the least-square solutions. As stated in [17], solution $\beta'$ leads to the smallest training error as well as a good generalization performance.

So the ELM algorithm can be summarized which only consists of three steps as follows:

ELM algorithm:
Given a training sample $N = \{(\boldsymbol{x}_i, \boldsymbol{t}_i) \mid \boldsymbol{x}_i \in R_n, \boldsymbol{t}_i \in Rm, i = 1, 2, ..., N\}$, activation function $g(x)$, and hidden neuron number $L$,

(1) Assign hidden neurons by randomly generating parameters $(\boldsymbol{w}_i, b_i)$, $i = 1, 2,..., L$;

(2) Compute the hidden layer output matrix $\boldsymbol{H}$ by Eq. (4);

(3) Calculate the output weight $\beta$: $\beta' = H^+ T$

### 3. Artificial Bee Colony

Artificial Bee Colony (ABC) algorithm was proposed by Karaboga in 2005 [12]. The algorithm simulates the intelligent foraging behavior of honey bee swarms. It is a population based stochastic optimization algorithm. In ABC algorithm, there are a set of computational agents called honey bees. The colony of honey bees consists of three groups of bees: employed bees, onlookers and scout bees. The employed bees and onlookers perform the exploitation process in the search food source position, and the scout bees control the exploration process.

The main steps of ABC algorithm are described for simple as follows:

1: cycle = 1;

2: Initialize the population of solutions $x_i$, $i = 1, 2, \ldots, SN$;

3: Evaluate the nectar amount of food sources;

4: repeat

5: Employed bees' phase;

6: Onlookers' phase;

7: Scout bees' phase;

8: Memorize the best solution obtained so far;

9: until cycle = *MCN*;

At the first step, the ABC algorithm creates a randomly distributed initial population $P$ of $i$ solutions $(i = 1, 2,..., NP)$, where $i$ denotes the size of population, and $NP$ is the number of the employed bees. Each solution $x_i$ is a $D$ -

dimensional vector. Here, $D$ is the number of optimization parameters. The position of food source represents a possible solution to the optimization problem, and the nectar amount of a food source stands for the quality (fitness value) of the solution. After initialization, the positions (solutions) are subjected to repeated cycle, $C$ =1, 2,..., $MCN$, of the search processes of the employed bees, the onlookers and the scout bees, where $MCN$ is the maximum cycle number of search process. An employed bee produces a modification on the position (solution) in the memory depending on the local information and evaluates the nectar amount (fitness value) of the new source (modified solution). Then the greedy selection criterion is used for the procedure: Suppose that the nectar amount of the new one is higher than that of the previous one, the bee memorizes the new position and forgets the old one, otherwise it keeps the position of the previous one in the memory. After all employed bees finish the search process, they share their position and the nectar information of the food sources with the onlookers in the hive's dance area. After evaluating the nectar information shared by all employed bees, an onlooker bee selects a food source according to a probability related to its nectar amount. Then the position modification and selection criterion applied to onlookers are the same as used in the employed bees procedure.

The probability $P_i$ with the food source located at $i$ will be chosen by an onlooker can be expressed as follows:

$$P_i = \frac{fitness_i}{\sum_{i=1}^{NP} fitness_i} \quad (6)$$

where $fitness_i$ denotes the fitness value of the solution represented by food source $i$. Clearly, a good food source (solution) will have more chance to be selected by the onlookers than a bad one, and then they produce a neighbor food source position $i+1$ to the selected one $i$, comparing the nectar amount (fitness value) of the neighbor $i+1$ position with the previous $i$ position. The greedy selection criterion is also used for the onlookers. The sequence is repeated until all onlookers are distributed. What is more, if a solution $i$ do not improve by a predetermined number of trials (*limit*), this solution would be abandoned by its employed bee and the associated employed bee becomes a scout and randomly searches for a new food source position entirely. When the new position is obtained, the next cycle ($MCN$) starts. The same procedures are repeated until the stopping criteria are satisfied.

To produce a candidate food position from the old one in memory, change one selected parameter randomly and keep the remaining parameters unaltered. It means that by adding to the current chosen parameter value the product of the uniform variant [-1, 1] and the difference between the chosen parameter value and other solution parameter value, the position of the selected neighbor food source is calculated as following:

$$v_{ij} = x_{ij} + u_{ij}(x_{ij} - x_{kj}) \quad (7)$$

Where $j$ and $k \in \{1,2,...D\}$ are randomly produced index, $k$ has to be different from $i$. The multiplier $u_{ij}$ is a uniformly distributed real random number between [-1, 1].

The food source position is replaced with a new food source by a scout when the nectar is abandoned by a bee. The scout produces an entirely new food source position can be defined

$$x_{ij} = x_j^{min} + r\,(x_j^{min} - x_j^{min}) \quad (8)$$

where $x_j^{max}$ is the upper bound of the food source position in dimension $j$ and $x_j^{min}$ is the lower bound of the food source position in dimension $j$, $r$ is a uniformly distributed real random number in the range of [-1, 1]. Notice that if the value of a parameter generated by using (7) and (8) exceeds its control limit, the parameter will be assigned to the boundary value associated with it.

## 4. The Proposed ABC-ELM Model

This study proposes a novel ABC-ELM model for parameter optimization problem of ELM. In the proposed model, the parameter optimization for ELM are dynamically conducted by implementing ABC algorithm and the k-fold CV scheme, then the obtained optimal parameters are taken by the ELM classifier model to perform the classification task.

As the input weights and hidden biases in the ELM are generated randomly and they are unchanged during the training process, the training time spending on tuning these parameters is reduced, but the output weight is calculated by using Eq. (5) based on the predetermined the input weights and hidden biases initially, there may exist a set of non-optimal or unnecessary input weights and hidden biases and many redundant hidden neurons. As a result, more hidden neurons may be required by the ELM algorithm than traditional tuning-based learning algorithm in some applications, which could make the ELM algorithm respond slowly to unknown testing data and cause the network to over-fit. Therefore, more compact networks with better generalization performance are desired.

In this section, a hybrid model named ABC-ELM based on ABC and ELM is proposed, where the ABC algorithm is used to find the best set of the network parameters including the input weights and hidden biases, meanwhile, *CV* approach is used in this process such that the estimated generalization error is small, and the MP generalized inverse is adopted to get the output weights. In $k$-fold *CV*, the training dataset is divided into $k$ equal subsets (of approximately equal size). ELM algorithm is called $k$ times, each time leaving one of the subsets from training, and the generalization accuracy is calculated based on the omitted subset. Here, the parameter value of $k$ is 5. After selecting the optimal input weights and hidden biases, the classifier model is developed by using

training set.

First, the population of the initially generation is randomly created. Each individual in the population is a set of the input weights and hidden biases defined as:

$$\theta=\{w_{11},w_{12},...,w_{1L},w_{21},w_{22},...,w_{2L},...,w_{n1},w_{n2},...,w_{nL},b_1,b_2,...,b_L\}$$

where the input weight $w_{ij}$ and hidden bias b$j$ are randomly initialized with the range [-1, 1]. Second, the output weight corresponding to each individual (a set of weights and biases) is determined by the MP generalization inverse. Then the fitness value of each individual is calculated on the training set using Eq. (9). Meanwhile, the 5-fold CV is used for determining the optimal parameters to avoid the over-fitting.

$$f = 1 - \frac{\sum_{i=1}^{n} MissClassificationRate_i}{n} \quad i = 1, 2, ..., n \qquad (9)$$

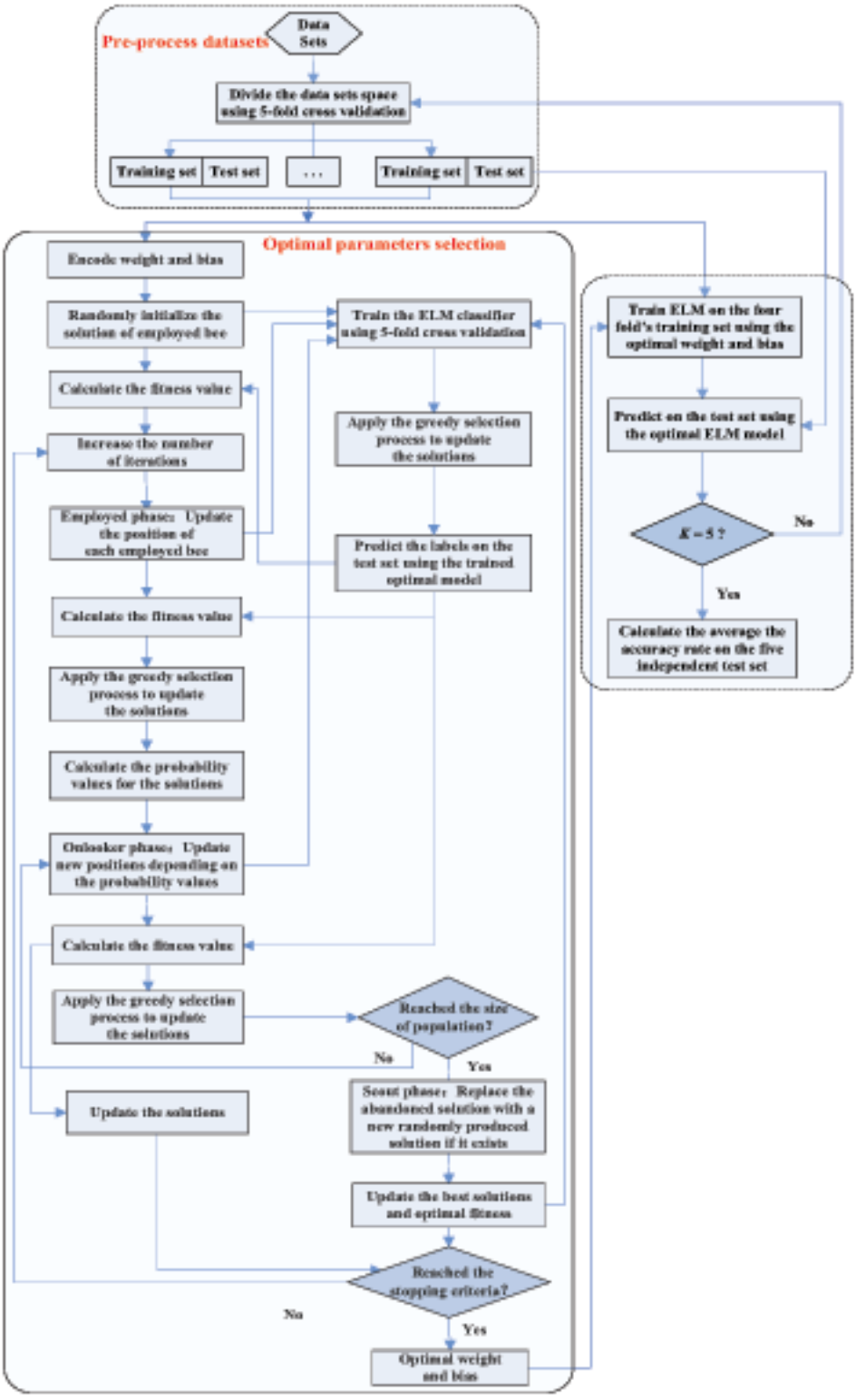


Figure 2. Flowchart of the ABC-ELM model

where $n$ is the number of the training sets, the accuracy rate $f$ on the training sets is adopted as the fitness.

After the fitness values of all individuals in the population are obtained, the employed bees' phase, the onlookers' phase and the scout bees' phase are performed in sequence, the populations of the new generation are generated, until the optimal goal or the maximum iterations is reached.

The flowchart of the ABC-ELM model is constructed through the following main steps as shown in Fig. 2.

**Step 1:** Divide the transformed data into 5 subsets using 5-fold cross validation method. Each subset contains 4 training sets and 1 test dataset;

**Step 2:** Encode the solution with $L \times (n+1)$ dimensions. The first $L \times n$ dimensions are the input weight values. The remaining $L$ dimensions are the bias values. They are both continuous values;

**Step 3:** Initialize the individuals of the populations (food source positions) with random numbers. Meanwhile, the ABC parameters including the size of the population, the lower and the upper bounds of the population, the number of iterations and the value of the limit, etc;

**Step 4:** Train the ELM classifier with the weight and the bias obtained in Step 2, Notice that the solution used for optimization is a vector, but they are 2-dimensional matrix in training process. For this reason, an extra step is used to change the solution vector to 2-dimensional matrix before the solution is evaluated;

**Step 5:** The fitness value is calculated according to Eq. (9);

**Step 6:** Increase the number of iterations: $iter = iter + 1$;

**Step 7:** In employed phase, update the position of each employed bee using Eq. (7) in each solution;

**Step 8:** Train the ELM classifier with the weight and the bias obtained in Step 6 and calculate the fitness value of each solution using Eq. (9);

**Step 9:** Apply the greedy selection process to improve the fitness value and solution by comparing the current fitness with the old fitness stored in the memory. If the current fitness is dominated by the old one stored in the memory, otherwise, replace the old one with the current fitness value and position;

**Step *10:*** Calculate the probability value of each solution. It means that the better fitness value of solution achieved, the more opportunity the solution been selected;

**Step 11:** In onlooker phase, update new positions depending on the probability values obtained in Step 9;

**Step 12:** Train the ELM with the weight and the bias obtained in Step 10 and calculate the fitness value of each solution using Eq. (9);

**Step 13:** Apply the greedy selection process, the same as Step 8;

**Step 14:** If the size of population is reached, then go to Step 11, otherwise, go to Step 10;

**Step 15:** In scout bee phase, if the fitness value can not be improved further through a predetermined number of iterations, abandon this solution, and randomly generate a new solution using Eq. (8), and then calculate the fitness of this solution;

**Step 16:** Update the global optimal fitness and global optimal solution by comparing the fitness with the optimal fitness from the whole population. If the current optimal fitness is dominated by the fitness stored in the memory, then keep the fitness and solution in the memory, otherwise, replace the fitness and solution with the current optimal fitness and corresponding solution from the whole population;

**Step 17:** If the stopping criteria are satisfied, then go to Step 17, otherwise, go to Step 5. The termination criterion is that the iteration number achieves the maximum number of iterations;

**Step 18:** Get the optimal weight and bias from the best solution.

## 5. Experimental Design

### 5.1 Data Description

To evaluate the performance of the proposed approach in different classification problems, eight real world datasets (Thyroid, WDBC, Wine, Bupa Liver, Cleveland Heart, Australian, Breast Cancer and Parkinson), provided by the StatLog and the UCI Machine Learning databases [2], are evaluated in this investigation. All of the eight datasets have often been used as benchmarks for classification performance comparison that can be easily found in data mining and pattern recognition literature. Bupa Liver dataset contains 345 instances with 6 attributes which are classified into 6 categories. WDBC dataset has 768 instances describing 2 classes based on 8 features. Wine dataset is the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. This set contains 178 instances with 13 continuous features. Thyroid dataset contains 3 classes of 215 instances, each instance has 5 features. Cleveland Heart and Parkinson datasets are composed of 303 instances and has 13 features, 195 instances and has 22 features for each other. Iris, Australian and Breast cancer datasets have the missing values, the missing categorical attributes are replaced by the mode of the attributes, and the missing continuous ones are replaced by the mean of the attributes.

The detail characteristics of these datasets are listed in Table 1.

Normalization is a necessary process to avoid the values in greater numerical ranges dominating those in smaller numerical ranges and reduce the numerical difficulties during the calculation. Generally, data is normalized by

| No. | Dataset | #Classes | #Instances | #Features | Missing Value |
|---|---|---|---|---|---|
| 1 | Bupa Liver | 2 | 345 | 6 | No |
| 2 | WDBC | 2 | 768 | 8 | No |
| 3 | Wine | 3 | 178 | 13 | No |
| 4 | Australian | 2 | 690 | 14 | Yes |
| 5 | BreastCancer | 2 | 699 | 9 | Yes |
| 6 | Thyroid | 3 | 215 | 5 | No |
| 7 | ClevelandHeart | 2 | 303 | 13 | Yes |
| 8 | Parkinson | 2 | 195 | 22 | No |

Table 1. Datasets from StatLog and the UCI Databases

scaling them into the interval of [-1, 1], and we choose the range of [-1, 1] using the Eq. (10), where $x$ is the original value, $x'$ is the scaled value, $min_n$ is minimum value of $n$ and $max_n$ is maximum value of $n$.

$$x' = \left( \frac{x - min_n}{max_n - min_n} \right) * 2 - 1 \qquad (10)$$

In order to get an unbiased estimate of the generalization accuracy, the 5-fold CV was used to evaluate the classification accuracy. Each time, four of the 5 subsets are put together to form a training set and the other subset is used as the test set. Then the average result across all 5 trials is computed. The advantage of this method is that all of the test sets are independent and the reliability of the results could be improved. It's worth nothing that only one repetition of the 5-fold CV will not generate enough classification accuracies for comparison. Because of the arbitrary partition of the dataset, it is not necessarily the same for the predicted accuracy of a model each iteration. Meanwhile, we design our experiment making use of two loops, which is also adopted in [3, 23]. The inner loop is used for determining the optimal fitness and best parameters for the ELM classifier. The outer loop is used to estimate the performance of the ELM classifier. So, the 5-fold CV is used for the inner loop and another 5-fold CV is employed as the outer loop.

### 5.2 Methods for Comparison

For presenting the superiority of the proposed ABC-ELM algorithm, we select some previous optimization techniques in the literature for algorithm comparisons. First we choose the original ELM algorithm as a method to be compared because it is a famous approach, proved to produce good generalization performance and learn thousands of times faster than traditional learning algorithm for feedforward neural networks, and has been widely applied in many kinds of fields. Moreover, as a global optimization based methodology, the ABC-ELM algorithm will be compared with the E-ELM and PSO-ELM.

Differential evolution (DE) is designed to iteratively try to improve a candidate solution with regard to a given measure of quality, which returns an optimal solution for optimization problems [24]. Zhu presented an instance of DE to search for the optimal input weights and hidden biases for ELM automatically [28]. DE imitates the mechanisms by choosing solutions based on vectors and iteratively updating vectors based on the solution quality.

Particle swarm optimization (PSO) is also a global optimization approach, which simulates bird flocking or fish schooling behavior to achieve a self-evolution process [14]. PSO can search automatically the *pbest* and *gbest* of data sets by optimizing the objective function. In the literature [26], PSO is also used to find suitable parameters of ELM for some prediction problems.

In Section 6, we will set up a series of experiments to describe method comparisons between ABC-ELM, original ELM, DE-ELM and PSO-ELM algorithms.

### 5.3 Experimental Setup

The proposed ABC-ELM model and other three methods were carried out on the platform of MATLAB 7.0. For ELM, the implementation by Huang available from *http://www3.ntu.edu.sg/home/egbhuang* was used. The computer is Intel Dual-Core TM with 2.0 GHz CPU and 2 GB RAM, the operating system is Windows XP.

The detailed parameter setting for ABC-ELM is as follows. The number of the iterations and population size were set to 250 and 40 respectively. The number of the employed bees (*SN*) is equal to the onlookers, that is to say, the number of the employed bees or the onlookers is 20. The value of *"limit"* is set to $SN \times D$, where *D* is the dimension of the problem [1]. The input weights and the biases are obtained into the range [-1, 1]. The sigmoid function $g(x) = 1/(1 + exp(-x))$ is adopt as ELM activation function to compute hidden layer output matrix.

For PSO [22], the searching range for the solution is set as follows: $[-pop_{max}, pop_{max}]$ is set as [-1, 1]. $v_{max}$ is set about $50\%$ of $pop_{max}$, and $v_{min}$ is set to $-v_{max}$. The acceleration constants $c_1$ and $c_2$ are set as follows: $c_1 = 2$, $c_2 = 2$. According to our preliminary experiment, *wmax* and *wmin* are set to 0.9 and 0.4 respectively. The inertia weight

$w$ is calculated using Eq. (11).

$$w = w_{max} - i * (w_{max} - w_{min})/ MCN \quad (11)$$

where $i$ denotes $i$-th iteration, and $MCN$ is the total number of iterations.

For DE, the roulette wheel selection method was used. The parameter $F$ and $CR$ is set to 1 and 0.8 for each other. In order to make a fair comparison, the same computational effort is used in ABC, ELM, PSO and DE. That is, the maximum generation, population size and searching range of the parameters in PSO, DE and ELM are the same as parameters in ABC-ELM.

## 6. Experimental Results and Discussions

To evaluate the effectiveness of the proposed model, we compare the ABC-ELM model with other three reference models (original ELM, PSO-ELM, DE-ELM) in this experiment.

The performance of the ABC-ELM algorithm is tested with the number of hidden neurons incremented by a step equal to 5, ranging from 5 to 40, the reason why we choose this range is that the performance of ELM algorithm would induce the over-fitting with gradually increasing number of hidden neurons, while due to ABC method optimizes the input weights and biases, which makes optimal

| Datasets | Algorithms | Accuracy (%) | | Hidden Neurons |
|---|---|---|---|---|
| | | Training | Testing | |
| **Bupa Liver** | ABC-ELM | 76.54 | **72.23±4.99** | **20** |
| | ELM | 76.58 | 71.30±5.14 | 30 |
| | PSO-ELM | 77.38 | 71.54±5.26 | 25 |
| | DE-ELM | 76.26 | 71.19±5.70 | 20 |
| **WDBC** | ABC-ELM | 97.86 | **96.42±1.23** | 30 |
| | ELM | 96.43 | 96.13±1.64 | 30 |
| | PSO-ELM | 98.49 | 96.28±1.60 | 20 |
| | DE-ELM | 98 | 96.10±1.93 | 20 |
| **Wine** | ABC-ELM | 99.97 | **98.43±1.12** | **15** |
| | ELM | 99.86 | 97.98±2.08 | 25 |
| | PSO-ELM | 1 | 97.63±2.27 | 15 |
| | DE-ELM | 1 | 98.30±1.69 | 25 |
| **Australian** | ABC-ELM | 88.14 | **86.38±2.91** | **15** |
| | ELM | 87.50 | 86.35±3.20 | 30 |
| | PSO-ELM | 89.37 | 86.14±2.31 | 40 |
| | DE-ELM | 89.51 | 86.03±2.71 | 20 |
| **Breast Cancer** | ABC-ELM | 97.54 | **96.51±1.19** | **20** |
| | ELM | 97.42 | 96.45±1.02 | 60 |
| | PSO-ELM | 97.16 | 96.51±1.25 | 35 |
| | DE-ELM | 97.88 | 96.45±1.67 | 35 |
| **Thyroid** | ABC-ELM | 97.79 | **94.79±3.04** | **20** |
| | ELM | 96.84 | 92.93±3.98 | 30 |
| | PSO-ELM | 97.92 | 94.14±3.67 | 30 |
| | DE-ELM | 99.10 | 92.74±3.02 | 40 |
| **Cleveland Heart** | ABC-ELM | 86.84 | **83.22±4.20** | **10** |
| | ELM | 85.60 | 82.45±4.32 | 20 |
| | PSO-ELM | 88.75 | 82.39±6.00 | 10 |
| | DE-ELM | 89.11 | 82.45±3.99 | 20 |
| **Parkinson** | ABC-ELM | 92.12 | **88.31±4.30** | **30** |
| | ELM | 92.28 | 86.15±5.79 | 40 |
| | PSO-ELM | 94.66 | 87.59±4.70 | 30 |
| | DE-ELM | 95.17 | 87.08±5.70 | 35 |

Table 2. The detailed results obtained by four investigated models via 5-fold on the eight datasets.

parameters more suitable for associated network, it can achieve good results just required a few of number of hidden neurons, at the same time, the results are stable.

The following tables and figures report the average accuracies (in percentage) of five independent runs. In each run, we use 5-fold CV method in partitioning training dataset and test dataset to have more reliable results. The last column of the table shows the smallest number of hidden neurons to be used in order to achieve the best results. Table 2 shows the results achieved with all four investigated models (ABC-ELM, original ELMÀÛÈÞPSO-ELM, DE-ELM) for the eight benchmarks classification datasets based on five trails. It is well known that the higher the accuracy rate is and the fewer the number of hidden neurons are needed, the better the model is said to be. At the same time, the test accuracy, the nearly optimal number of hidden neurons for these algorithms and the standard deviation are shown in the table in the form of mean ± standard deviation. From Table 2 we can easily find that ABC-ELM outperforms other three methods for all of the eight classification problems. It reveals that the optimal network structure tuned by the ABC algorithm contributes a lot in reducing the hidden neurons in the models and achieving a reasonable generalization performance for these benchmarks datasets.

For the Bupa Liver dataset, the proposed method outperforms other three methods in terms of classification accuracy by almost 1%, 0.69% and 1.04% respectively. ELM requires 30 hidden neurons to achieve the best testing accuracy, and PSO-ELM offers the best testing accuracy when the number of hidden neurons is 25, while for ABC-ELM and DE-ELM, only 20 hidden neurons are needed. Moreover, ABC-ELM obtains best testing accuracy in this case. Although the training accuracy of ABC-ELM is slightly lower than PSO-ELM, the ABC-ELM offers better testing accuracy than all other three approaches empirically with reduced number of hidden neurons, and the standard deviation for the acquired performance by the proposed method is also smaller, which indicates consistency and stability of the proposed model. Fig. 5(a) also shows the classification accuracy of four methods achieved is changed with the number of hidden neurons on the Bupa Liver dataset. In particular, when the number of hidden neurons is smaller than 10, the accuracy of ELM is quite poor, the reason is that the parameters are selected randomly.

For the WDBC dataset, ABC-ELM achieves better testing accuracy than original ELM, PSO-ELM, DE-ELM and ABC-ELM obtains smaller standard deviation. Although ABC-ELM uses 30 hidden neurons to achieve the best accuracy, PSO-ELM and DE-ELM use 20 hidden neurons to get the best accuracy for each other, ABC-ELM still obtains the accuracy of 96.35% which is better than other algorithms with 20 hidden neurons. The results evaluated on the WDBC dataset are shown in Fig. 5(b).

For the Wine dataset, the proposed method outperforms all other cases, the result is very close to that of DE-ELM, but only uses 15 hidden neurons, and the standard deviation obtained by the proposed method is less than those of others. Fig. 5(c) illustrates the results on the Wine dataset which confirms the accuracy results of Table 2.

For the Australian dataset, the ABC-ELM method performs better than all other cases with the smallest number of hidden neurons, moreover, the results of the proposed method is more stable. These results are also confirmed in Fig. 5(d).

For the results obtained for the Breast Cancer dataset indicates the optimal weights and hidden biases obtained by the ABC algorithm lead the network to be more accurate and stable. In particular, the best accuracy of ELM with 60 hidden neurons is less than that of the ABC-ELM. Fig. 5(e) shows the accuracy obtained by four methods on Breast Cancer dataset. For the Thyroid dataset, in general, the ABC-ELM performs better than all other cases, improving by more then 2% compared to the DE-ELM. In addition, by increasing the number of hidden neurons (from 5 to 20, in our experiments), the network accuracy

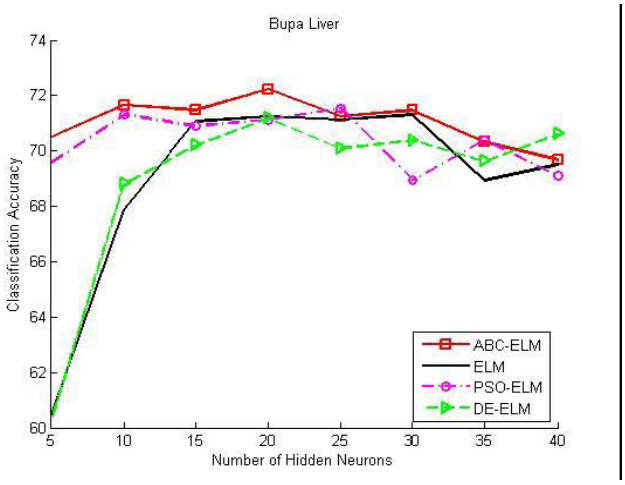


Figure 5(a)

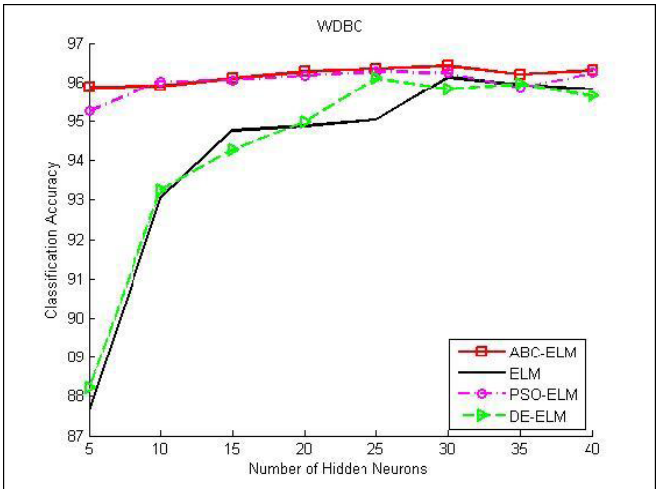


Figure 5(b)

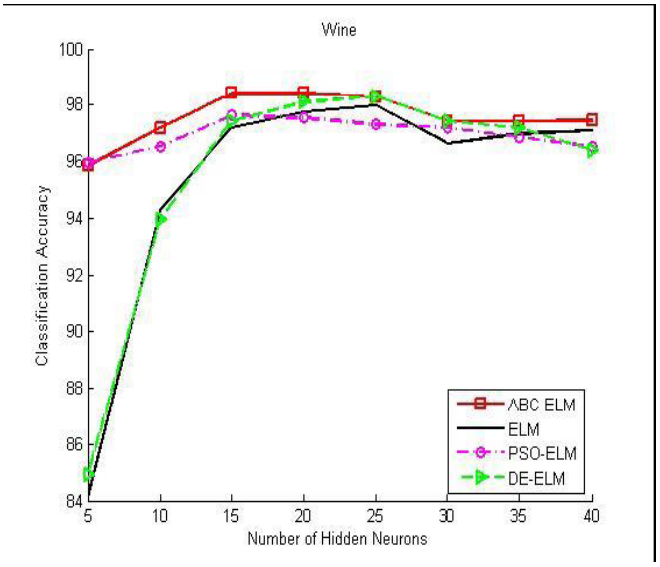


Figure 5(c)

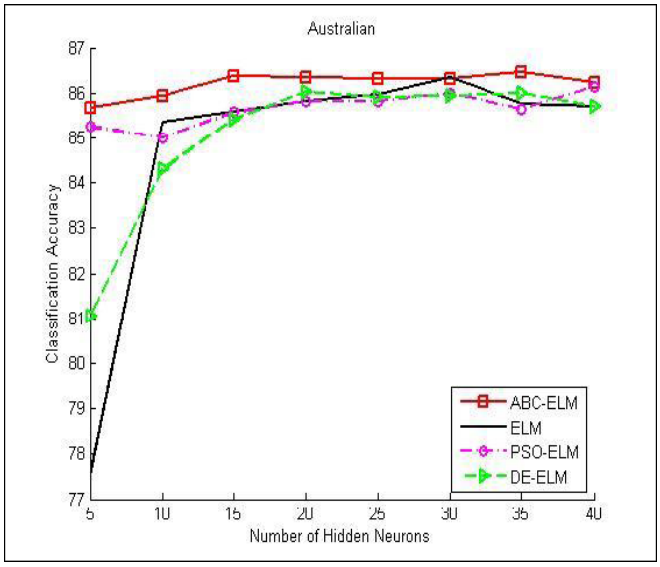


Figure 5(d)

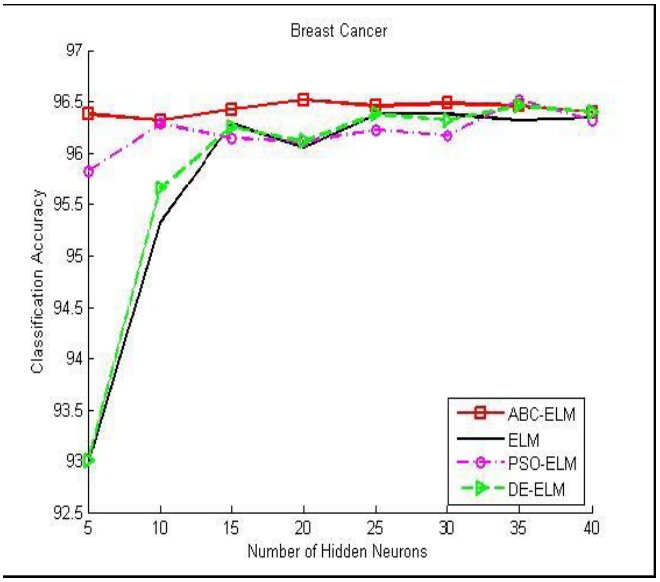


Figure 5(e)

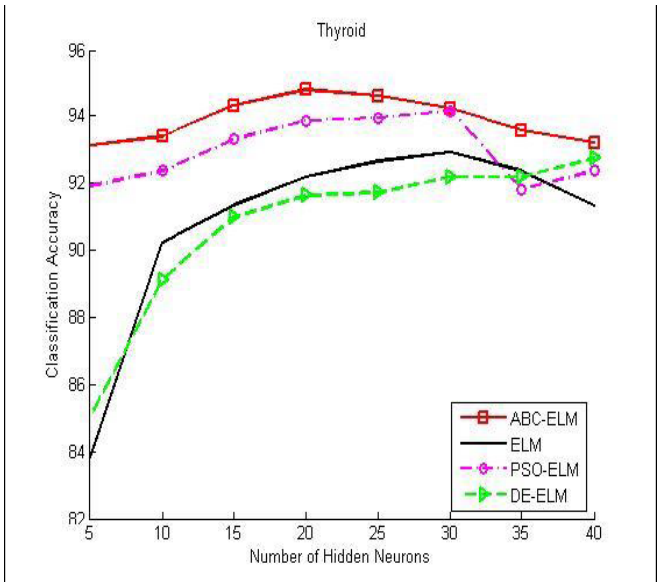


Figure 5(e)

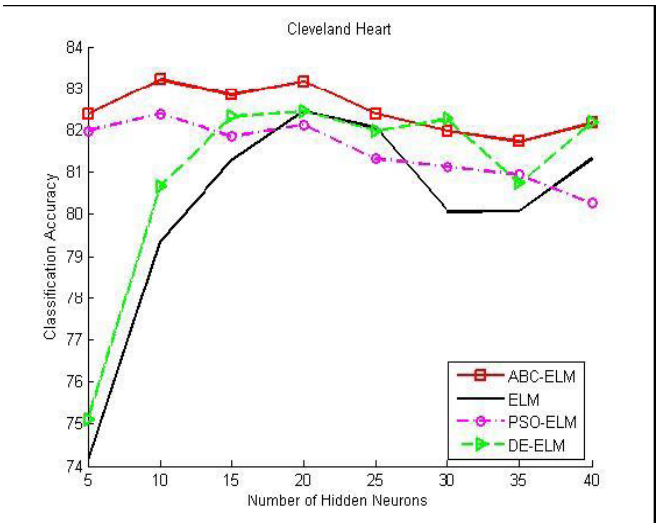


Figure 5(g)

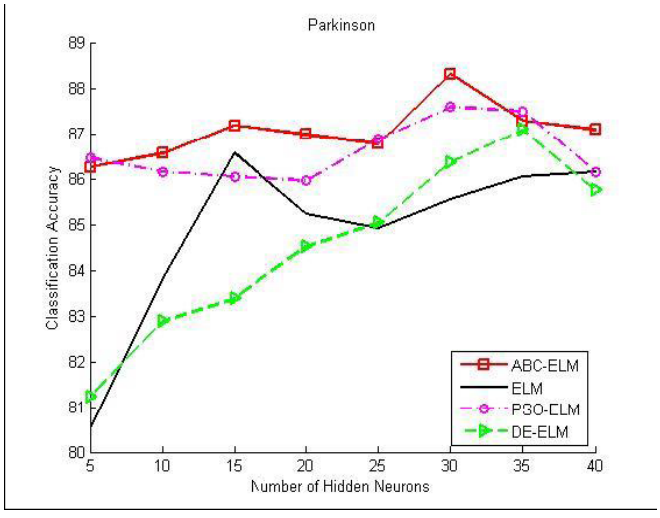


Figure 5(h)

increase, however, the number of hidden neurons increasing too large, the results of ELM will fall down rapidly, it induces the over-fitting. The proposed method has relatively more stable results. The results of the Thyroid dataset are shown in Fig. 5(f).

For the Cleveland Heart dataset and Parkinson dataset respectively, which indicates the proposed method outperforms others in all cases, as shown in Fig. 5(g) and Fig. 5(h).

In order to evaluate whether the proposed method is statistically significant than others, in this paper, a non-parametric statistical hypothesis test called Wilcoxon signed ranks test [5] for comparing two related sample has been conducted at the 5% significance level. It ranks for differences in performances of two classifiers for each dataset, to analyze whether the average difference between the exist algorithm and the proposed algorithm performance over various datasets is significantly different. Table 3 shows the *P* values calculated by Wilcoxon signed ranks test for comparison of the accuracy of two groups (the left group corresponding to competitor algorithm and the right group corresponding to ABC-ELM) at a time. In most cases, the *P* values reported are less than 0.05 (5% significance level) except that in the Parkinson dataset, the difference of classification rate between ABC-ELM and PSO-ELM is not obvious. In sum, it is evidence that the better performance produced by ABC-ELM is statistically significant.

## 7. Conclusions and future work

This work has explored a novel hybrid learning algorithm named ABC-ELM, which makes use of the advantages of both ABC and ELM method. As far as we know it is the first application to combine ABC optimization algorithm with ELM, meanwhile, ABC algorithm is applied to optimize the ELM with a satisfactory training process which helps to achieve better performance. In addition, double loop of the 5-fold CV method is adopted to avoid the over-fitting. As can be perceived from the experimental results, ABC-ELM performs better than other three state-of-the-art methods (original ELM, PSO-ELM and DE-ELM) and more compact neural network on real world classification datasets. It is also confirmed that due to the optimal parameters to be selected, the desired results are more stable with only just a few of number of hidden neurons. Through the use of non-parametric Wilcoxon signed ranks test, we demonstrate that the ABC-ELM method could

| **Datasets** | **Algorithm Comparison** | | ***P* Value** |
|---|---|---|---|
| **Bupa Liver** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.012<br>0.036<br>0.025 |
| **WDBC** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.012<br>0.012<br>0.012 |
| **Wine** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.012<br>0.017<br>0.018 |
| **Australian** | ELM<br>**PSO - ELM**<br>DE-ELM | ABC-ELM | 0.017<br>0.012<br>**0.012** |
| **Breast Cancer** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.012<br>0.025<br>0.027 |
| **Thyroid** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.012<br>0.012<br>0.012 |
| **Cleveland Heart** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.012<br>0.012<br>0.036 |
| **Parkinson** | ELM<br>PSO-ELM<br>DE-ELM | ABC-ELM | 0.017<br>0.093<br>0.017 |

Table 3.
P values produced by Wilcoxon signed ranks test comparing three existing algorithms with the ABC-ELM on eight datasets

outperform above three algorithms in a statistically significant fashion. Hence, it can be concluded the developed ABC-ELM method can be a feasible and effective alternative algorithm for neural network optimization.

The future investigation will pay much attention to finding the optimal number of hidden neurons and the parameters of the weights and biases at the same time, because many existing methods usually find the number of hidden neurons by trial and error rather than self-adjusting. In addition, the application of the proposed method for solving various domain problems will also be studied.

## 7. Acknowledgements

This research is supported by the Natural Science Foundation of China (NSFC) under Grant Nos.61170092, 61133011, 60973088, 60973089, 61103091.

## References

[1] Akay, B., Karaboga, D. (2009). Parameter Tuning for the Artificial Bee Colony Algorithm. *In*: Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems. Vol. 5796, p. 608-619.

[2] Asuncion, A. (2010). UCI machine learning repository. University of California, Center for Machine Learning and Intelligent Systems., http://archive.ics.uci.edu/ml.

[3] Chen, H. L., Yang, B., Wang, G., Liu, J., Xu, X., Wang, S. J., Liu, D. Y. (2011). A novel bankruptcy prediction model based on an adaptive fuzzy k-nearest neighbor method. *Knowledge-Based Systems*, 24, 1348-1359.

[4] Chen, X., Liu, W., Lai, J., Li, Z., Lu, C. (2012). Face recognition via local preserving average neighborhood margin maximization and extreme learning machine. Soft Computing - A Fusion of Foundations, Methodologies and Applications.

[5] Demsar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research*, 7, 1-30.

[6] Huang, F., H., D. S., Zhu, Z. H. (2006). The forecast of the postoperative survival time of patients suffered from non-small cell lung cancer based on PCA and extreme learning machine. *International Journal of Neural Systems*, 16, 39-46.

[7] Guopeng, Z., Zhiqi, S., Chunyan, M., Zhihong, M. (2009). On improving the conditioning of extreme learning machine: A linear case. *In*:ICICS'09 Proceedings of the 7th International conference on Information, Communications and Signal Processing (p. 234-238).

[8] Huang, G.-B., Wang, D., Lan, Y. (2011). Extreme learning machines: A survey. *International Journal of Machine Learning and Cybernetics*, 2, 107-122.

[9] Huang, G.-B., Zhou, H., Ding, X., Zhang, R. (2012). Extreme Learning Machine for Regression and Multiclass Classification. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*., 42, 513-529.

[10] Huang, G.-B., Zhu, Q.-Y., Siew, C.-K. (2004). Extreme learning machine: A new learning scheme of feedforward neural networks. *In*: Proceedings of International Joint Conference on Neural Networks (IJCNN2004). 6. Vol. 2, p. 985-990.

[11] Huang, G.-B., Zhu, Q.-Y., Siew, C.-K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing,* 70, 489-501.

[12] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical Report - TR06, Erciyes University.

[13] Karaboga, D., Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing*, 8, 687-697.

[14] Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. *In*: IEEE International Conference on Neural Networks, Washington, DC, USA. Vol. 4, p. 1942-1948.

[15] Li, F. C., Wang, P. K., Wang, G. E. (2009). Comparison of the primitive classifiers with extreme learning machine in credit scoring. *In*: Industrial Engineering and Engineering Management. (IEEM 2009) p. 685-688.

[16] Li, L.-N., Ouyang, J.-H., Chen, H.-L., Liu, D.-Y. (2012). A Computer Aided Diagnosis System for Thyroid Disease Using Extreme Learning Machine. *Journal of Medical Systems*, 1-11.

[17] Li, M.-B., Huang, G.-B., Saratchandran, P., & Sundararajan, N. (2005). Fully complex extreme learning machine. *Neurocomputing*, 68, 306-314.

[18] Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A. (2010). OP-ELM: Optimally Pruned Extreme Learning Machine. *IEEE Transactions on Neural Networks,* 21, 158-162.

[19] Mohammed, A. A., Minhas, R., Wu, Q. M. J., Sid-Ahmed, M. A. (2011). Human face recognition based on multidimensional PCA and extreme learning machine. Pattern Recognition, 44, 2588-2597.

[20] Pan, F., Zhang, H., Xia, M. (2009). A Hybrid Time-Series Forecasting Model Using Extreme Learning Machines. *In*: Intelligent Computation Technology and Automation, 2009. (ICICTA '09) (Vol. 1, p. 933-936).

[21] Saraswathi, S., Sundaram S. (2011). ICGA-PSO-ELM Approach for Accurate Multiclass Cancer Classification Resulting in Reduced Gene Sets in Which Genes Encoding Secreted Proteins Are Highly Represented. *IEEE Transactions on Computational Biology and Bioinformatics,* 8, 452-463.

[22] Shi, Y., Eberhart, R. (1998). A modified particle swarm optimizer. In IEEE World Congress on Computational Intelligence, Evolutionary Computation Proceedings, p. 69-73.

[23] Statnikov, A., Tsamardinos, I., Dosbayev, Y., Aliferis, C. F. (2005). GEMS: A system for automated cancer diagnosis and biomarker discovery from microarray gene expression data. *International Journal of Medical Informatics*, 74, 491-503.

[24] Storn, R., Price, K. (1997). Differential Evolution - A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11, 341-359.

[25] Wang, R., Kwong, S., Wang, X. (2012). A study on random weights between input and hidden layers in extreme learning machine. Soft Computing - A Fusion of Foundations, Methodologies and Applications.

[26] Xu, Y., Shu, Y. (2006). Evolutionary Extreme Learning Machine-Based on Particle Swarm Optimization. Advances in Neural Networks. p. 644-652.

[27] Zhao, X.-g., Wang, G., Bi, X., Gong, P., Zhao, Y. (2011). XML document classification based on ELM. *Neurocomputing,* 74, 2444-2451.

[28] Zhu, Q.-Y., Qin, A. K., Suganthan, P. N., Huang, G.-B. (2005). Evolutionary extreme learning machine. *Pattern Recognition*, 38, 1759-1763.

[29] Zong, W., Huang, G.-B. (2011). Face recognition based on extreme learning machine. *Neurocomputing*, 74, 2541-2551.