Cogn Comput (2017) 9:275–284
DOI 10.1007/s12559-017-9451-y

Dolphin Swarm Extreme Learning Machine

Tianqi Wu[1] · Min Yao[1] · Jianhua Yang[1]

Received: 4 October 2016 / Accepted: 26 January 2017 / Published online: 27 February 2017

Abstract As a novel learning algorithm for a single hidden-layer feedforward neural network, the extreme learning machine has attracted much research attention for its fast training speed and good generalization performances. Instead of iteratively tuning the parameters, the extreme machine can be seen as a linear optimization problem by randomly generating the input weights and hidden biases. However, the random determination of the input weights and hidden biases may bring non-optimal parameters, which have a negative impact on the final results or need more hidden nodes for the neural network. To overcome the above drawbacks caused by the non-optimal input weights and hidden biases, we propose a new hybrid learning algorithm named dolphin swarm algorithm extreme learning machine adopting the dolphin swarm algorithm to optimize the input weights and hidden biases efficiently. Each set of input weights and hidden biases is encoded into one vector, namely the dolphin. The dolphins are evaluated by root mean squared error and updated by the four pivotal phases of the dolphin swarm algorithm. Eventually, we will obtain an optimal set of input weights and hidden biases. To evaluate the effectiveness of our method, we compare the proposed algorithm with the standard extreme learning machine and three state-of-the-art methods, which are the particle swarm optimization extreme learning machine, evolutionary extreme learning machine, and self-adaptive evolutionary extreme learning machine, under 13 benchmark datasets obtained from the University of California Irvine Machine Learning Repository. The experimental results demonstrate that the proposed method can achieve superior generalization performances than all the compared algorithms.

✉ Min Yao
myao@zju.edu.cn

1 School of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China

Keywords Single hidden-layer feedforward neural network · Extreme learning machine · Dolphin swarm algorithm · Hybrid algorithm

Introduction

Inspired by cognitive science, the extreme learning machine (ELM) was proposed to overcome challenging issues (such as slow convergence and local minima problem) faced by the conventional gradient-based approaches in a single hidden-layer feedforward neural network (SLFN) [1]. Since its inception, the ELM has attracted increasing attention from researchers all over the world. One of the salient features of the ELM is that the input weights and hidden biases are generated randomly instead of being tuned iteratively. The output weights are computed based on the prefixed input weights and hidden biases using the Moore-Penrose (MP) generalized inverse only once [2, 3]. Another characteristic of the ELM is that almost any nonlinear piecewise continuous random hidden nodes can be used in it [4, 5]. Compared with the conventional gradient-based approaches, the ELM has a significantly lower computational time and presents better generalization performances. Due to its efficiency and universal approximation capabilities, the ELM has been demonstrated on various problems in different fields, such as classification [6–8], recognition [9–12], imbalanced leaning [13, 14], steganalysis [15], and so on. However, there exist some sets of non-optimal input weights and hidden biases which may influence the performance in minimizing the cost function. Due to the randomness, the input weights and hidden biases generated by the ELM may be non-optimal. Since the output weights highly depend on the input weights and hidden biases, the non-optimal input weights and hidden biases may have a negative impact on the final results. Moreover, the ELM may require more hidden neurons than the conventional gradient-based algorithms in many cases [16–18].

To address the above ELM's drawbacks and find a proper set of input weights and hidden biases, Huang et al. proposed a hybrid form of the differential evolutionary algorithm and ELM method called the evolutionary ELM (E-ELM) [19]. Besides the E-ELM, some other bio-inspired methods and global search techniques are also adopted in the ELM during the past several years. Cao et al. proposed an improved learning algorithm named the self-adaptive evolutionary ELM (SaE-ELM) and evaluated the performances on regression and classification problems [20]. Xu and Shu introduced a hybrid particle swarm optimization (PSO) and ELM algorithm for a prediction problem [21]. Saraswathi et al. presented a PSO-driven ELM, combined with the integer-coded genetic algorithm (ICGA), to solve gene selection and cancer classification [22]. Silva et al. combined the ELM with group search optimization (GSO) and valuated the influence of four different forms of handling members that fly out of the search space bounds [23]. More contemporary soft computing techniques, such as ant colony optimization [24–26], artificial bee colony algorithm [27–29], firefly algorithm [30, 31], binary-coded PSO [32], and multi-subswarm PSO [33], are also promising to be applied in the ELM.

Recently, inspired by the biological characteristics and living habits shown in the dolphins' predatory process, Wu et al. presented a novel evolutionary algorithm named dolphin swarm algorithm (DSA) to solve optimization problems [34]. By simulating the dolphins' predatory process, DSA can make use of the behavior rules of dolphins and the interactions between dolphins to produce changes and achieve certain goals at the group level. With the help of effective strategies, the DSA has shown many promising features, such as fast periodic convergence, local-minima-free, no specific demand on the cost function, and so on [34]. In comparison with the classical evolutionary algorithms, the DSA has better global search ability, better stability, and higher convergence speed, which have been proven through some benchmark functions with different properties [35]. Therefore, compared with the conventional evolutionary algorithms, the DSA is a preferable method to address the ELM's drawbacks caused by the non-optimal input weights and hidden biases.

In this paper, we introduce a new hybrid method named dolphin swarm extreme learning machine (DS-ELM) for SLFN. In the proposed algorithm, several sets of input weights and hidden biases are used. Each set is encoded into one vector, namely the dolphin. These dolphins are evaluated by root mean squared error (RMSE) [19–23] and updated by the four pivotal phases of DSA. After the four pivotal phases, the set of input weights and hidden biases represented by the best dolphin is selected. Then, the output weights are derived by the chosen set. To evaluate the effectiveness of our method, we compare the proposed algorithm with the standard ELM and three state-of-the-art methods (PSO-ELM, E-ELM, and SaE-ELM) under 13 benchmark datasets (the newest or the most popular datasets) obtained from the University of California Irvine (UCI) Machine Learning Repository [36]. The experimental results show that DS-ELM can achieve superior generalization performances than all the compared algorithms.

The remaining structures of this paper are as follows. The "Preliminaries" presents the standard ELM and DSA methods. After that, the DS-ELM is proposed in "Dolphin Swarm Extreme Learning Machine (DS-ELM)." "Experiments" is made up of certain experiments. The DS-ELM will be compared with the standard ELM, PSO-ELM, E-ELM, and SaE-ELM under six regression benchmark datasets and seven classification benchmark datasets obtained from the UCI Machine Learning Repository [36]. A summary of this paper will be given in the last section.

Preliminaries

Extreme Learning Machine (ELM)

The ELM was originally proposed for SLFN by Huang et al. [1]. Then, it was extended to the generalized SLFN where the hidden layer need not be neuron like [5]. The main concept of the ELM is that the input weights and hidden biases are generated randomly instead of being tuned iteratively. Moreover, the output weights are computed by the MP generalized inverse only once [2, 3]. Therefore, the ELM has a significantly lower computational time compared with the conventional gradient-based approaches. Besides, ELM theories have shown that almost any nonlinear piecewise continuous random hidden nodes can be used in the ELM [4, 5], and the resultant networks have universal approximation capabilities [37–39].

The output function of the ELM for generalized SLFNs with L hidden nodes is

$$f_L(x) = \sum_{i=1}^{L} \beta_i g_i(x) = \sum_{i=1}^{L} \beta_i G(a_i, b_i, x), x \in R^d, \beta_i \in R^m \quad (1)$$

where a_i and b_i are the input weight and hidden bias of the ith hidden node respectively; β_i denotes the linked weight between the ith hidden node and the output neurons; and $g_i(x)$ denotes the output function $G(a_i, b_i, x)$ of the ith hidden node. The output functions $G(a, b, x)$ are a nonlinear piecewise continuous function satisfying ELM universal approximation capability theorems [4, 5]. For additive nodes with activation function $g(x)$, $g_i(x)$ is defined as

$$g_i(x) = G(a_i, b_i, x) = g(a_i \cdot x + b_i), a_i \in R^d, b_i \in R \quad (2)$$

For radial basis function (RBF) nodes with activation function $g(x)$, $g_i(x)$ is defined as

$$g_i(x) = G(a_i, b_i, x) = g(b_i \| x - a_i \|), a_i \in R^d, b_i \in R^+ \quad (3)$$

Suppose we are training the generalized SLFN with L hidden neurons to learn N arbitrary samples (x_i, t_i). Since the input weights and hidden biases are generated randomly, the nonlinear system can be converted to a linear model:

$$H\beta = T \tag{4}$$

where

$$H = \begin{bmatrix} G(a_1, b_1, x_1) & \cdots & G(a_L, b_L, x_1) \\ \vdots & \ddots & \vdots \\ G(a_1, b_1, x_N) & \cdots & G(a_L, b_L, x_N) \end{bmatrix}_{N\times L} \tag{5}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L\times m} \text{ and } T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N\times m} \tag{6}$$

H is the hidden-layer output matrix; $\beta = [\beta_1, \beta_2, \ldots \beta_K]^{\mathrm{T}}$ is the output weights; $T = [t_1, t_2, \ldots t_N]^{\mathrm{T}}$ is desired outputs. The input

weights and hidden biases are generated randomly. Thus, the

generalized SLFN is converted into finding the least-square (LS) solution to the given linear model. The minimum norm LS solution to the linear model Eq. (4) is

$$\beta = H^{\dagger} T \tag{7}$$

where $H^{\dagger}$ is the MP generalized inverse of the matrix H. By utilizing such MP generalized inverse method,

mances [2, 3].

In summary, the ELM algorithm is presented in algorithm 1.

```
Algorithm 1 Extreme Learning Machine
    Step 1 Randomly generate the input weights $a$ and hidden biases $b$
    Step 2 Calculate the hidden-layer output matrix $H$
    Step 3 Compute the output weights $\beta = H^{\dagger}T$
```

Dolphin Swarm Algorithm (DSA)

The DSA was introduced by Wu et al. as an efficient global search method to solve varied optimization problems [34]. It is mainly implemented by simulating the biological characteristics and living habits shown in the dolphin's actual predatory process, such as echolocation, information exchanges, cooperation, and division of labor. By simulating the dolphin's actual predatory process, the DSA can make use of the behavior rules of dolphins and the interactions between dolphins to produce changes at the group level and achieve certain goals. With the help of effective strategies, the DSA has shown many promising features, such as fast periodic convergence, local minima free, no specific demand on the cost function, and so on [34]. Compared with the conventional evolutionary algorithms, the DSA has better global search ability, better stability, and higher convergence speed, which have been proven through some benchmark functions with different properties [35].

The dolphin's actual predatory process consists of three stages. In the first stage, each dolphin independently takes advantage of sounds to search for nearby prey and to evaluate the surrounding environment by echoes. In the second stage, dolphins exchange their information. Those dolphins that find large prey call other dolphins for help. And those dolphins that have received information move toward the prey and surround it along with other dolphins. In the last stage, the prey is surrounded by the dolphins and then what the dolphins need to do is to take turns to enjoy the food, which means that the predatory process is accomplished [34].

In the DSA, a swarm of N dolphins is kept. In the D dimensional search space of optimization problems, the dolphins are defined as $\mathrm{Dol}_i = [x_1, x_2, \ldots x_D]^T$, ($i = 1, 2, \ldots, N$), where $x_j (j = 1, 2, \cdots, D)$ is the component of each dimension to be optimized. For each Dol_i, there are two corresponding variables $L_i (i = 1, 2, \cdots, N)$ and $K_i (i = 1, 2, \cdots, N)$, where L_i represents the optimal solution Dol_i finds in a single time and K_i represents the optimal solution of what Dol_i finds by itself or gets from others. The fitness function represented by fitness(X) is the basis for judging whether the position is better. There are three kinds of distances used in total. One is the distance between Dol_i and Dol_j

DK_i, and the other one is the distance between L_i and K_i named

DKL_i. Moreover, exchanging processes are maintained by an $N \times N$ order matrix named transmission time matrix TS, where the term $\mathrm{TS}_{i,j}$ represents the remaining time of the sound spreading from Dol_j to Dol_i.

Generally, DSA can be divided into six phases. In each phase, dolphins have distinct work to do. Besides the initialization phase and termination phase, four pivotal phases of DSA including search phase, call phase, reception phase, and predation phase need to be expounded for a better understanding.

Search Phase

In the search phase, each dolphin searches its nearby area by making sounds $V_i = [v_1, v_2, \cdots, v_D]^T$, ($i = 1, 2, \cdots, M$) toward M random directions, where $v_j (j = 1, 2, \cdots, D)$ is the component of each dimension. Besides, the sounds satisfy $\|V_i\| = \mathrm{speed}$, $(1 = 1, 2, \cdots, M)$, where speed is a constant. Within the maximum search time T, the sound V_j that Dol_i makes at time t will search a new solution X_{ijt}:

$$X_{ijt} = \mathrm{Dol}_i + V_j \times t \tag{8}$$

For the new solution X_{ijt} that Dol_i gets, calculate its fitness E_{ijt}:

$$E_{ijt} = \mathrm{fitness}(X_{ijt}) \tag{9}$$

If

$$E_{iab} = \min_{j=1,\ldots,M;t=1,\ldots,T} E_{ijt} = \min_{j=1,\ldots,M;t=1,\ldots,T} \text{fitness}\left(X_{ijt}\right) \quad (10)$$

then L_i is determined as

$$L_i = X_{iab} \quad (11)$$

If

$$\text{fitness}(L_i) < \text{fitness}(K_i) \quad (12)$$

then replace K_i by L_i. Otherwise K_i does not change.

Call Phase

In the call phase, each dolphin will make sounds to inform other dolphins of its result in the search phase. And the transmission time matrix TS needs to be updated as follows:

For K_i, K_j, and $\text{TS}_{i,j}$, if

$$\text{fitness}(K_i) > \text{fitness}\left(K_j\right) \quad (13)$$

and

$$\text{TS}_{i,j} > \frac{DD_{i,j}}{\text{speed}} \quad (14)$$

then update $\text{TS}_{i,j}$ as

$$\text{TS}_{i,j} = \frac{DD_{i,j}}{\text{speed}} \quad (15)$$

Reception Phase

When DSA enters the reception phase, all the terms $\text{TS}_{i,j}$ subtract one to indicate that the sounds spread a unit of time, then check all the terms $\text{TS}_{i,j}$, if

$$\text{TS}_{i,j} = 0 \quad (16)$$

then it means that the sound spreading from Dol_j to Dol_i can be received. Compare K_i with K_j. If it satisfies Eq. (13), then replace K_i by K_j. Otherwise K_i does not change. Then set $\text{TS}_{i,j}$ as

$$\text{TS}_{i,j} = \left\lceil \frac{\text{range}}{\text{speed}} \right\rceil \quad (17)$$

where range is the length of the search space.

Predation Phase

In the predation phase, each dolphin needs to get a new position according to its known information and it can be discussed in two cases.

- For DK_i, if

$$DK_i \le T \times \text{speed} \quad (18)$$

we can get Dol_i's new position newDol_i:

$$\text{newDol}_i = K_i + w \times (\text{Dol}_i - K_i) \times \left(1 - \frac{2}{e}\right) \quad (19)$$

where e is a constant which is greater than 2 and w is an arbitrary unit vector.

- For DK_i, if

$$DK_i > T \times \text{speed} \quad (20)$$

we can get Dol_i's new position newDol_i:

$$\text{newDol}_i = K_i + w \times \left(1 - \frac{DK_i \times \frac{1}{\text{fitness}(K_i)} + (DK_i - DKL_i) \times \frac{1}{\text{fitness}(K_i)}}{e \times DK_i \times \frac{1}{\text{fitness}(K_i)}}\right) \times DK_i \quad (21)$$

After all the Dol_i get their new position newDol_i, compare newDol_i with K_i in fitness. If

$$\text{fitness}(\text{newDol}_i) < \text{fitness}(K_i) \quad (22)$$

then replace K_i by newDol_i. Otherwise K_i does not change.

If the end condition is satisfied, DSA enters the termination phase. Otherwise, DSA enters the search phase again.

Although there are many parameters used in DSA, only a few of them are user-specified, including the number of dolphins N, the number of sounds M, the speed of sounds speed, maximum search time T, and the constant e used in the predation phase.

In summary, the DSA is presented in algorithm 2.

Dolphin Swarm Extreme Learning Machine (DS-ELM)

The standard ELM generates the input weights and hidden biases randomly. However, there exist some sets of non-optimal input weights and hidden biases which may inevitably influence the performance in minimizing the cost function and have a negative impact on the final results. There are two feasible ways to avoid these non-optimal input weights and hidden biases. One is to simply increase the number of hidden nodes which may weaken the performances of the ELM. The other one is to find a proper set of input weights and hidden biases which can achieve a favorable performance with less hidden nodes and more compact networks. In this section, we present our new hybrid approach DS-ELM to improve the performance of the ELM in the latter way by taking advantages of the DSA's great global search ability.

Algorithm 2 Dolphin Swarm Algorithm

```
Step 1 Initialization
    Randomly generate the initial dolphin swarm $Dol = \{Dol_1, \cdots, Dol_N\}$ in the $D$ dimensional space. Calculate the fitness for each dolphin, and get $Fit_K = \{Fit_{K,1}, \cdots, Fit_{K,N}\}$.
Step 2 Starting Loop
while the end condition is not satisfied do
    Step 2.1 Search Phase
        $E_{ijt} = Fitness\left(Dol_i + V_j \times t\right)$
        $Fit_L = \{\min\{E_{1jt}\}, \cdots, \min\{E_{Njt}\}\}$
        $Fit_{K,i} = \begin{cases} Fit_{L,i}, & if\ Fit_{L,i} < Fit_{K,i} \\ Fit_{K,i}, & otherwise \end{cases}$
    Step 2.2 Call Phase
        $TS_{i,j} = \begin{cases} \left\lceil \frac{DD_{i,j}}{speed} \right\rceil, & if\ Fit_{K,j} < Fit_{K,i}\ and\ TS_{i,j} > \left\lceil \frac{DD_{i,j}}{speed} \right\rceil \\ TS_{i,j}, & otherwise \end{cases}$
    Step 2.3 Incept Phase
        $TS$ counts down for 1 unit time.
        $Fit_{K,i} = \begin{cases} Fit_{K,j}, & if\ TS_{i,j} = 0\ and\ Fit_{K,j} < Fit_{K,i} \\ Fit_{K,i}, & otherwise \end{cases}$
    Step 2.4 Hunt Phase
        Calculate $DK_i$ and $DKL_i$
        $newDol_i = \begin{cases} Eq.(14), & if\ DK_i \le T \times speed \\ Eq.(16), & otherwise \end{cases}$
        Calculate $newDol_i$'s fitness and updates $Fit_{K,i}$.
end while
Step 3 Termination
    Output the best dolphin and its fitness.
```

Table 1 Parameters used in the DS-ELM, PSO-ELM, and E-ELM

DS-ELM	PSO-ELM	E-ELM
Speed = 0.1;	$V = 0.1$;	$F = 1$
$T = 3$;	C1 = 2;	CR = 0.8
M = 2;	C2 = 2	
$E = 3$	$W = 0.8$	

In the DS-ELM, a swarm of N dolphins is kept. Each dolphin represents one set of input weights and hidden biases, namely $\mathrm{Dol}_i = [w_{11}, w_{12}, \ldots, w_{1K}, w_{21}, w_{22}, \ldots, w_{M1}, w_{M2}, \ldots, w_{MK}, b_1, b_2, \ldots, b_K]^T, (i = 1, 2, \ldots, N)$. All the input weights and hidden biases are randomly initialized within the range of [−1, 1]. For each dolphin, the corresponding output weights β are computed by Eq. (7).

Then, we utilize the RMSE as the fitness function [19–23]:

$$\mathrm{fitness}(\mathrm{Dol}_i) = \sqrt{\frac{\sum_{j=1}^{M}\left\|\sum_{j=1}^{K}\beta g\left(w_i \cdot x_j + b_j\right) - t_j\right\|_2^2}{M}} \tag{23}$$

where $T = [t_1, t_2, \ldots, t_M]^T$ is the desired output.

With N dolphins and the fitness function, we are able to adopt the DSA to our DS-ELM. After the search phase, the call phase, the reception phase, and the predation phase described in "Dolphin Swarm Algorithm (DSA)," an optimal set of input weights and hidden biases will be obtained eventually.

In summary, the DS-ELM is presented in algorithm 3.

Experiments

In this section, the DS-ELM is compared with the standard ELM and three state-of-the-art methods, including the PSO-ELM, E-ELM, and SaE-ELM. The experiments are composed of two parts. In the first part, these five algorithms are tested under six regression benchmark datasets. And seven classification benchmark datasets are employed in the second part. Specifically, these 13 benchmark datasets are obtained from the UCI Machine Learning Repository [36]. All these simulations are conducted in Matlab 9.1 environment (the latest version in 2016) with an Intel Core i7, 2 GHz CPU and 8 GB RAM.

In our experiment, each dataset is divided into two parts, namely the training set and the testing set. We adopt a tenfold cross-validation method in the training set and use the validation set to determine the suitable parameters of the five algorithms. The number of hidden nodes is gradually increased in a preset region [5, 10, 15…200], and the one with the lowest validation error is selected. Moreover, the number of individuals (dolphins in the DS-ELM, particles in the PSO-ELM, and population in the E-ELM and SaE-ELM) is 5, and the times of invoking the ELM is equally set as 1000 for the DS-ELM,

Table 2 Specifications of the six regression benchmark datasets

Dataset	Data		Attributes
	Training	Testing	
Servo	84	83	4
Cancer	99	99	32
Housing	253	253	13
Airfoil	752	751	3
Wine	2449	2449	11
Air	4679	4679	15

Table 3 Results of the five algorithms under the six regression benchmark datasets

Dataset	Algorithm	Testing accuracy		Training time (s)	Hidden nodes
		Means	StDev		
Servo	DS-ELM	*8.679e^{-2}*	*2.106e^{-2}*	16.154	20
	PSO-ELM	1.010e^{-1}	2.760e^{-2}	16.185	20
	E-ELM	1.013e^{-1}	3.787e^{-2}	16.416	20
	SaE-ELM	9.383e^{-2}	2.705e^{-2}	16.526	20
	ELM	1.246e^{-1}	3.576e^{-2}	0.004	40
Cancer	DS-ELM	*2.557e^{-1}*	*1.948e^{-2}*	15.917	20
	PSO-ELM	2.734e^{-1}	2.542e^{-2}	16.050	20
	E-ELM	2.746e^{-1}	2.677e^{-2}	16.147	20
	SaE-ELM	2.665e^{-1}	2.350e^{-2}	16.482	20
	ELM	2.845e^{-1}	4.070e^{-2}	0.006	35
Housing	DS-ELM	*8.528e^{-2}*	*1.116e^{-2}*	17.119	20
	PSO-ELM	9.577e^{-2}	2.057e^{-2}	17.183	20
	E-ELM	9.664e^{-2}	1.680e^{-2}	17.399	20
	SaE-ELM	8.974e^{-2}	2.001e^{-2}	17.707	20
	ELM	1.053e^{-1}	2.483e^{-2}	0.008	45
Airfoil	DS-ELM	*9.527e^{-2}*	*4.070e^{-3}*	20.996	25
	PSO-ELM	1.005e^{-1}	5.241e^{-3}	21.136	25
	E-ELM	1.034e^{-1}	5.203e^{-3}	21.294	25
	SaE-ELM	9.975e^{-2}	5.223e^{-3}	21.524	25
	ELM	1.068e^{-1}	6.723e^{-3}	0.006	50
Wine	DS-ELM	*1.188e^{-1}*	*4.308e^{-3}*	69.755	75
	PSO-ELM	1.216e^{-1}	5.023e^{-3}	69.905	75
	E-ELM	1.237e^{-1}	6.414e^{-3}	70.344	75
	SaE-ELM	1.229e^{-1}	9.514e^{-3}	70.815	75
	ELM	1.286e^{-1}	1.055e^{-2}	0.190	155
Air	DS-ELM	*2.326e^{-2}*	*2.422e^{-4}*	78.062	50
	PSO-ELM	2.944e^{-2}	2.465e^{-4}	78.819	50
	E-ELM	2.623e^{-2}	3.989e^{-4}	78.695	50
	SaE-ELM	2.530e^{-2}	3.803e^{-4}	79.098	50
	ELM	3.273e^{-2}	4.133e^{-4}	0.324	100

Algorithm 3 Dolphin Swarm Extreme Learning Machine

```
Step 1 Initialization
  Randomly generate the initial dolphin swarm $Dol = \{Dol_1, \cdots, Dol_N\}$ where
  $Dol_i = [w_{11}, w_{12}, \cdots, w_{1K}, w_{21}, \cdots, w_{M1}, \cdots, w_{MK}, b_1, \cdots, b_K]^T, (i = 1, 2, \cdots, N)$. Calculate the fitness of
  $Dol_i$ by Eq.(7) and Eq.(23), and get $Fit_K = \{Fit_{K,1}, \cdots, Fit_{K,N}\}$.
Step 2 Starting Loop
while the end condition is not satisfied do
    Step 2.1 Search Phase
        Compute the fitness of $(Dol_i + V_j \times t)$ by Eq.(7) and Eq.(23), and get $E_{ijt}$.
        $Fit_L = \{\min\{E_{1jt}\}, \cdots, \min\{E_{Njt}\}\}$
        $Fit_{K,i} = \begin{cases} Fit_{L,i}, & if\ Fit_{L,i} < Fit_{K,i} \\ Fit_{K,i}, & otherwise \end{cases}$
    Step 2.2 Call Phase
        $TS_{i,j} = \begin{cases} \left\lceil \frac{DD_{i,j}}{speed} \right\rceil, & if\ Fit_{K,j} < Fit_{K,i}\ and\ TS_{i,j} > \left\lceil \frac{DD_{i,j}}{speed} \right\rceil \\ TS_{i,j}, & otherwise \end{cases}$
    Step 2.3 Incept Phase
        $TS$ counts down for 1 unit time.
        $Fit_{K,i} = \begin{cases} Fit_{K,j}, & if\ TS_{i,j} = 0\ and\ Fit_{K,j} < Fit_{K,i} \\ Fit_{K,i}, & otherwise \end{cases}$
    Step 2.4 Hunt Phase
        Calculate $DK_i$ and $DKL_i$
        $newDol_i = \begin{cases} Eq.(14), & if\ DK_i \le T \times speed \\ Eq.(16), & otherwise \end{cases}$
        Calculate $newDol_i$'s fitness by Eq.(7) and Eq.(23) , and updates $Fit_{K,i}$.
end while
Step 3 Termination
  Calculate the hidden-layer output matrix $H$ of the best dolphin. Compute the output
  weights matrix by Eq.(7).
```

PSO-ELM, E-ELM, and SaE-ELM. Other parameters, suggested by [20, 21, 34], are shown in Table 1.

Regression

In this subsection, the five algorithms are compared under six regression benchmark datasets from the UCI Machine Learning Repository [36], which are Servo, Breast Cancer Wisconsin (Cancer), Housing, Airfoil Self-Noise (Airfoil), Wine Quality (Wine), and Air Quality (Air). The Air dataset is one of the newest regression datasets in the UCI Machine Learning Repository [36]. And the other five datasets are the most popular datasets. The specifications of the six datasets are shown in Table 2.

All the attributes have been normalized to the range of [−1, 1], and the desire outputs have been normalized to the range of [0, 1]. To get a better comparison and reduce the influence of accidental factors, all the experimental results are obtained by taking the average of 20 independent experiments. For each experiment, the training set and the testing set are randomly selected from the whole dataset, and the two sets have no overlap with each other to avoid overfitting. Moreover, the two sets are kept the same for each trial of the five algorithms. The results are shown in Table 3, and the best results are shown in italic font.

In terms of the training time, it is quite easy to see that the ELM is the fastest one since all the other five algorithms invoke it multiple times. Besides, there is not much difference in training time among the DS-ELM, PSO-ELM, E-ELM, and SaE-ELM, because invoking the ELM is time consuming and the times of invoking the ELM of these four algorithms are the same. But it can still be seen that the DS-ELM is slightly faster than the PSO-ELM, E-ELM, and SaE-ELM.

As for the testing accuracy, the DS-ELM, PSO-ELM, E-ELM, and SaE-ELM obtain better results with less hidden nodes than the ELM which means that the DS-ELM, PSO-ELM, E-ELM, and SaE-ELM can achieve better generalization performances with more compact networks. Compared with the PSO-ELM, E-ELM, and SaE-ELM, the DS-ELM has smaller means and standard deviations. Wilcoxon's signed-rank tests with 95% confidence interval show that the DS-ELM is significantly different from the ELM, PSO-ELM, E-ELM, and SaE-ELM which indicates that the DS-ELM is superior to the other four methods under these six regression benchmark datasets.

Classification

In this subsection, the performances of the five algorithms are evaluated under seven classification benchmark datasets from the UCI Machine Learning Repository [36]. The seven datasets are Iris, Heart, Pima Indians Diabetes (Diabetes), Image Segmentation (Image), Landsat Satellite (Satellite), Occupancy Detection (Occupancy), and Shuttle respectively. The Occupancy dataset is one of the newest classification datasets in the UCI Machine Learning Repository [36]. And the other six datasets are the most popular datasets. The specifications of the seven datasets are listed in Table 4.

All the attributes have been normalized to the range of [−1, 1]. The same as the experiments in "Regression," all the experimental results are obtained by taking the average of 20 independent experiments. For each experiment, the whole dataset is divided into two parts with no overlap, namely the training set and testing set. And the two sets are kept coincident for each trial of the five algorithms. The results are shown in Table 5, and the best results are emphasized with italic font.

Speaking of the training time, the same conclusion can be made that the ELM runs fastest and the other four methods put in nearly the same amount of time to train the networks which is theoretically reasonable as analyzed in "Regression."

When it comes to the testing accuracy, it can be easily seen that the DS-ELM achieves the highest mean testing accuracy and the lowest standard deviation in all the classification datasets. In addition, Wilcoxon's signed-rank tests with 95% confidence interval show that the DS-ELM is significantly different from the ELM, PSO-ELM, E-ELM, and SaE-ELM, which indicates that the DS-ELM outperforms the other four approaches under these seven classification benchmark datasets.

Conclusion

In this paper, we firstly introduced the standard ELM and DSA. Then, we proposed a brand new hybrid approach named dolphin swarm extreme learning machine for the SLFN. In our new algorithm, the DSA is used to optimize the input weights and hidden biases of the ELM, and the ELM is employed to calculate the output weights. In the experiment part of this paper, our method is compared

Table 4 Specifications of the seven classfication benchmark datasets

Dataset	Data		Attributes	Classes
	Training	Testing		
Iris	75	75	4	3
Heart	135	135	13	2
Diabetes	384	384	8	2
Image	1155	1155	19	7
Satellite	3218	3217	36	7
Occupancy	10,280	10,280	7	2
Shuttle	29,000	29,000	9	5

Table 5 Results of the five algorithms under the seven classification benchmark datasets

Dataset	Algorithm	Testing accuracy (%)		Training	Hidden
		Means	StDev	Time (s)	Nodes
Iris	DS-ELM	*97.798*	*3.057*	2.255	20
	PSO-ELM	95.732	4.422	2.290	20
	E-ELM	95.200	4.269	2.374	20
	SaE-ELM	96.798	3.335	2.449	20
	ELM	94.666	4.988	0.004	35
Heart	DS-ELM	*82.593*	*6.425*	1.307	20
	PSO-ELM	80.743	7.551	1.309	20
	E-ELM	80.371	7.115	1.318	20
	SaE-ELM	81.482	8.921	1.343	20
	ELM	78.889	8.615	0.006	45
Diabetes	DS-ELM	*79.160*	*2.963*	2.616	20
	PSO-ELM	77.079	3.111	2.631	20
	E-ELM	77.217	3.020	2.671	20
	SaE-ELM	77.861	3.368	2.702	20
	ELM	75.007	5.446	0.004	40
Image	DS-ELM	*95.239*	*1.113*	23.122	100
	PSO-ELM	94.112	1.712	23.124	100
	E-ELM	94.028	1.751	23.316	105
	SaE-ELM	94.805	1.240	23.497	105
	ELM	93.723	2.261	0.060	190
Satellite	DS-ELM	*88.716*	*1.072*	38.625	100
	PSO-ELM	87.475	1.549	37.709	100
	E-ELM	87.398	1.203	38.206	100
	SaE-ELM	87.615	1.490	38.180	100
	ELM	86.422	1.682	0.060	190
Occupancy	DS-ELM	*99.055*	*0.156*	137.515	50
	PSO-ELM	98.963	0.171	138.836	50
	E-ELM	98.878	0.164	139.097	50
	SaE-ELM	98.990	0.167	139.670	50
	ELM	98.905	0.184	0.423	100
Shuttle	DS-ELM	*99.609*	*0.101*	247.797	50
	PSO-ELM	99.266	0.140	248.340	50
	E-ELM	99.277	0.145	249.372	50
	SaE-ELM	99.481	0.179	251.368	50
	ELM	99.079	0.225	0.875	100

with the standard ELM and three state-of-the-art methods, which are the PSO-ELM, E-ELM, and SaE-ELM under six regression benchmark datasets and seven classification benchmark datasets obtained from the UCI Machine Learning Repository. Experimental results show that our method has better generalization performances with more compact networks than the standard ELM. Moreover, our algorithm can achieve better testing results (smaller RMSE on regression and higher accuracy on classification). According to Wilcoxon's signed-rank tests, the dolphin swarm extreme learning machine is superior to the other four methods both on the six regression datasets and the seven classification datasets in the experiments. However, the DS-ELM is time consuming compared with the standard ELM. Besides, it does not reduce the number of hidden nodes compared with the state-of-the-art methods. Future research works will be concentrated on how to get faster as well as how to make the network more compact.

Acknowledgements This paper is the partial achievement of Project 61272261 supported by the National Natural Science Foundation of China.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

Informed Consent Informed consent was not required as no human or animal was involved.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Huang, Guang-Bin, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. Neural Networks, 2004. Proceedings. 2004 I.E. International Joint Conference on. Vol. 2. IEEE, 2004.
2. Huang G-B, Siew C-K. Extreme learning machine with randomly assigned RBF kernels. Int J Inf Technol. 2005;11(1):16–24.
3. Huang, Guang-Bin, and Chee-Kheong Siew. Extreme learning machine: RBF network case. Control, Automation, Robotics and Vision Conference, 2004. ICARCV 2004 8th. Vol. 2. IEEE, 2004.
4. Huang G-B, Chen L, Siew CK. Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Transactions on Neural Networks. 2006;17(4): 879–92.
5. Huang G-B, Chen L. Convex incremental extreme learning machine. Neurocomputing. 2007;70(16):3056–62.
6. Duan L, et al. A voting optimized strategy based on ELM for improving classification of motor imagery BCI data. Cogn Comput. 2014;6.3:477–83.
7. Akusok A, et al. A two-stage methodology using K-NN and false-positive minimizing ELM for nominal data classification. Cogn Comput. 2014;6(3):432–45.
8. Cao K, et al. Classification of uncertain data streams based on extreme learning machine. Cogn Comput. 2015;7.1:150–60.
9. Zhao Z, et al. A class incremental extreme learning machine for activity recognition. Cogn Comput. 2014;6(3):423–31.
10. Zhang S, et al. Fast image recognition based on independent component analysis and extreme learning machine. Cogn Comput. 2014;6.3:405–22.
11. He B, et al. Fast face recognition via sparse coding and extreme learning machine. Cogn Comput. 2014;6(2):264–77.
12. Xie SJ, et al. Feature component-based extreme learning machines for finger vein recognition. Cogn Comput. 2014;6.3:446–61.

13. Vong C-M, et al. Imbalanced learning for air pollution by meta-cognitive online sequential extreme learning machine. Cogn Comput. 2015;7.3:381–91.
14. Xia S-X, et al. A kernel clustering-based possibilistic fuzzy extreme learning machine for class imbalance learning. Cogn Comput. 2015;7.1:74–85.
15. Sachnev V, et al. A cognitive ensemble of extreme learning machines for steganalysis based on risk-sensitive hinge loss function. Cogn Comput. 2015;7.1:103–10.
16. Huang G-B, Zhu Q-Y, Siew C-K. Extreme learning machine: theory and applications. Neurocomputing. 2006;70(1):489–501.
17. Hagan MT, Menhaj MB. Training feedforward networks with the Marquardt algorithm. IEEE transactions on Neural Networks. 1994;5(6):989–93.
18. Levenberg K. A method for the solution of certain non-linear problems in least squares. Q Appl Math. 1944;2(2):164–8.
19. Zhu Q-Y, et al. Evolutionary extreme learning machine. Pattern Recogn. 2005;38(10):1759–63.
20. Cao J, Lin Z, Huang G-B. Self-adaptive evolutionary extreme learning machine. Neural Process Lett. 2012;36(3):285–305.
21. Xu, You, and Yang Shu. Evolutionary extreme learning machine–based on particle swarm optimization. International Symposium on Neural Networks. Springer Berlin Heidelberg, 2006.
22. Saraswathi S, et al. ICGA-PSO-ELM approach for accurate multiclass cancer classification resulting in reduced gene sets in which genes encoding secreted proteins are highly represented. IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2011;8.2:452–63.
23. Silva, Danielle NG, Luciano DS Pacifico, and Teresa Bernarda Ludermir. An evolutionary extreme learning machine based on group search optimization. 2011 I.E. Congress of Evolutionary Computation (CEC). IEEE, 2011.
24. Drigo, M., V. Maniezzo, and A. Colorni. The ant system: optimization by a colony of cooperation agents. IEEE Trans Syst, Man, Cybernet Part B. 1996: 29–41.
25. Dorigo M, Gambardella LM. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput. 1997;1.1:53–66.
26. Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Comput Intell Mag. 2006;1(4):28–39.
27. Karaboga D, Basturk B. Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. International Fuzzy Systems Association World Congress. Springer Berlin Heidelberg, 2007.
28. Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput. 2008;8(1):687–97.
29. Karaboga D, Akay B. A comparative study of artificial bee colony algorithm. Appl Math Comput. 2009;214(1):108–32.
30. Yang X-S. Firefly algorithm, stochastic test functions and design optimisation. International Journal of Bio-Inspired Computation. 2010;2(2):78–84.
31. Yang, Xin-She Nature-inspired metaheuristic algorithms. Luniver Press. Beckington. 2008.
32. Taormina R, Chau K-W. Data-driven input variable selection for rainfall–runoff modeling using binary-coded particle swarm optimization and extreme learning machines. J Hydrol. 2015;529: 1617–32.
33. Zhang J, Chau K-W. Multilayer ensemble pruning via novel multi-sub-swarm particle swarm optimization. Journal of Universal Computer Science. 2009;15(4):840–58.
34. Tian-qi WU, Min YAO, Jian-hua YANG. Dolphin swarm algorithm. Frontiers of Information Technology & Electronic Engineering. 2016;707–729
35. Yao X, Liu Y, Lin G. Evolutionary programming made faster. IEEE Trans Evol Comput. 1999;3(2):82–102.
36. A. Frank and A. Asuncion, UCI Machine Learning Repository, Univ. California, Sch. Inform. Comput. Sci., Irvine, CA, 2011 [Online]. Available: http://archive.ics.uci.edu/ml.
37. Huang G-B. An insight into extreme learning machines: random neurons, random features and kernels. Cogn Comput. 2014;6(3): 376–90.
38. Huang G-B. What are extreme learning machines? Filling the gap between Frank Rosenblatt's dream and John von Neumann's puzzle. Cogn Comput. 2015;7.3:263–78.
39. Huang G-B, et al. Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2012;42.2:513–29.