Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(4): 7594–7632.
DOI: 10.3934/mbe.2023328
Received: 15 November 2022
Revised: 25 December 2022
Accepted: 27 December 2022
Published: 20 February 2023

Research article

Single dendritic neural classification with an effective spherical search-based whale learning algorithm

Hang Yu[1], Jiarui Shi[2], Jin Qian[1,*], Shi Wang[1] and Sheng Li[1]

[1] College of Computer Science and Technology, Taizhou University, Taizhou 225300, China
[2] Department of Engineering, Wesoft Company Ltd., Kawasaki-shi 210-0024, Japan

* **Correspondence:** Email: qianjin@tzu.edu.cn.

Abstract: McCulloch-Pitts neuron-based neural networks have been the mainstream deep learning methods, achieving breakthrough in various real-world applications. However, McCulloch-Pitts neuron is also under longtime criticism of being overly simplistic. To alleviate this issue, the dendritic neuron model (DNM), which employs non-linear information processing capabilities of dendrites, has been widely used for prediction and classification tasks. In this study, we innovatively propose a hybrid approach to co-evolve DNM in contrast to back propagation (BP) techniques, which are sensitive to initial circumstances and readily fall into local minima. The whale optimization algorithm is improved by spherical search learning to perform co-evolution through dynamic hybridizing. Eleven classification datasets were selected from the well-known UCI Machine Learning Repository. Its efficiency in our model was verified by statistical analysis of convergence speed and Wilcoxon sign-rank tests, with receiver operating characteristic curves and the calculation of area under the curve. In terms of classification accuracy, the proposed co-evolution method beats 10 existing cutting-edge non-BP methods and BP, suggesting that well-learned DNMs are computationally significantly more potent than conventional McCulloch-Pitts types and can be employed as the building blocks for the next-generation deep learning methods.

Keywords: evolutionary algorithms; optimization methods; whale optimization algorithm; back-propagation; artificial neural networks; deep learning; dendritic neuron model

1. Introduction

Artificial neural networks have been studied for more than half a century [1]. Among which, deep learning [2], over the last two decades, has dominated the area of artificial intelligence research as a result of the advancement of computer technology. Speech recognition has benefited greatly from deep learning [3]. Visual classification and recognition [4], unmanned vehicles [5, 6] as well as

protein structure prediction [7] also achieve fundamental technical evolution with the aid of deep learning. Nowadays, the popular generative adversarial networks [8], neural network architecture search [9], migration learning [10], federation learning [11], multitask learning [12], meta-learning [5] and attention models [13] are all built based on neural networks.

Traditional neural networks usually contain one or two hidden layers, where each "neuron" is a very simple computational unit [14]. A neuron receives input signals from others, which are amplified by the connection rights, and when they reach a neuron, if the total amount exceeds a certain threshold, the current neuron is "activated" and passes its output signals outwards. In fact, each neuron is a very simple computational equation, and a so-called neural network is a mathematical system obtained by nesting and iterating many such computational equations. The basic computational units of a neural network are continuously differentiable. In the past, the sigmoid function was used as an activation function of the neuron model, which was continuously differentiable. Recently, the rectified linear units (ReLU) is used as an activation function in deep neural networks, which is also continuously differentiable [15]. Thus, it is relatively easy to calculate the "gradient" of the whole system. Therefore, the BP algorithm is used to train the neural network by gradient descent optimization [16]. Before the work of Hinton et al. [17], it was not possible to train a neural network with more than five layers of the meridians because the neural network suffered from "gradient disappearance" during training. When the BP algorithm back-propagates the output layer of the neural network to the part far from it, a "zero" adjustment may be derived, so that the part of the network far from the output layer cannot be adjusted according to the output error, thus making the training fail. This is a huge technical hurdle to develop deep neural networks based on traditional neural networks, and Hinton et al. demonstrate that deep neural networks may be trained using "joint fine-tuning after layer-by-layer training" to avoid gradient disappearance [17].

The complexity of a machine learning model is related to its capacity, which has a significant impact on its learning ability [18, 19]. If the complexity of a learning model can be increased, then its learning ability can also be improved. There are two ways to do this for neural network models: make the models "deeper" or "wider" [20–22]. From the perspective of improving the complexity of the models, "deepening" is more effective because, in simple terms, only the amount of computational units is increased through "widening" and then that of basis functions; whereas through "deepening", the amount of not only basis functions but also layers of function nesting is increased, so the general function expression ability will be stronger. Therefore, in order to improve the complexity, the network should be "deepened" [23].

However, instead of building neural networks in increasing depth, some researchers have reconsidered the problem of modeling neurons from scratch [24]. The McCulloch-Pitts neuron has long been criticized for being overly simplified, using only weight to represent the strength of the connection between two neurons [25, 26]. The complexity of a genuine biological neuron allows it to exhibit both temporal and spatial characteristics. Some spiking neuron models have been presented, which were motivated by the capacity of neurons to process temporal information [27–30]. As basic units, due to their rich temporal dynamics, the new generations of neural units created as a consequence of the integration of spiking neuron models into deep learning [31] offer a great lot of promise for handling challenging recognition and prediction tasks. On the other hand, the construction of neuronal models based on their spatial functionality and large tree-like structures have also received a lot of research [32]. However, it is traditionally believed that this requires multiple

layers of McCulloch-Pitts neurons [33], and that individual neurons with large dendritic structures remain a challenge to model [34, 35].

Evolutionary optimization and brain-like inspired deep learning are at the core of artificial intelligence, both of which are inspired by the knowledge of processing and natural evolution in the brain to obtain optimal solutions and advance artificial intelligence [36]. Evolutionary computation is influenced by the natural selection mechanism of "superiority and inferiority" during biological evolution based on the law of transmission of genetic information, which is simulated by iterative programs that are used to solve the problems as an environment and seek optimal solutions through a natural evolution in a population of possible solutions. In 2005, Julie Greensmith [37] proposed that hazard theory doctrines and dendritic cell work mechanisms inspired immune algorithms that could be used to exploit prior knowledge to construct new algorithms that were fault-tolerant, immune, robust and evolutionary, and that optimal solutions closer to the dynamics in real-world problems should be sought. With the deepening of brain-like perception and cognition, the understanding and study of brain biomechanics as well as the sparsity, learning, selectivity, directionality, knowledge and diversity of brain neurons, deep learning networks are now being built to simulate these properties [32]. Thereby, the feedback connections between dendrites and dendrites, dendrites and synapses, layers and layers as well as neurons and neurons are less studied and utilized [32].

A neuron contains 10^4 dendritic interconnections on average, which leads to a broad range of neuronal morphology. The nonlinearity of dendrites explains the dynamics of various neurons, which mechanistically demonstrates the role of biological neurons in various circuits with behavioral significance [38]. Peripheral evidence has provided strong support for the robustness of dendritic computation, such as auditory coincidence detection [39], direction selectivity of the local computation of dendritic branches [40] and logical operations [41]. Most recently, studies have even shown that complex exclusive OR (XOR) logical computations can be performed through dendrites. Several dendritic neuron models have been proposed [42–44]. Logic ANDs and ORs models were employed as synapses inside the dendritic tree for interaction in Koch and Segev's logical neuron model [45]. In a two-layer model of a single neuron given by Poirazi et al. [46], synaptic subunits experienced nonlinear information processing in the sigma style. Legenstein and Maass [47] offered a mathematical model that supported the system's capacity to be self-organized in mixing various input qualities. These models, albeit used to take advantage of synapses' nonlinear properties, are unable to resolve complicated issues, notably those involving nonlinear separation [47]. Recently, it was shown that k-tree neuron models, which depicted dendritic structures as entire binary trees, excel at various challenging machine learning tasks including creating and expanding MNIST [48]. However, as the authors have pointed out [48], the model is biologically infeasible for presenting an unstructured input from mathematical representations and one-dimensional computational activities. By articulating synaptic processing in terms of sigmoid functions and dendritic interactions inside dendrites, which are performed by multiplicative operations, the development of a new physiologically realistic dendritic neuron model (DNM) has been made [49–51].

DNM has proved its capacity to solve difficult challenges as a single neuron model [50, 52–54]. Its huge dendritic connection structure is considered, so as to construct individual neuron learning models under the width structure, make full use of the nonlinear information processing mechanism on the synapses and achieve an automatic deletion [55] on the neuron structure. The intelligent algorithm of multi-objective evolutionary learning [56, 57] is used to complete the neurons in the

idiosyncratic function module, hyper-parameters, network structure and learning algorithm auto-evolution on itself. Evolutionary computation is combined with deep networks [58], in which weight learning is used for weight optimization of neural networks, dynamic structure learning for neural network structure search, and super-parametric learning for the optimization of neural network parameters. The properties of the standard problem of deep learning networks are used to maintain the diversity of populations, deep learning networks are used to replace the expensive process of fitness evaluation, and deep neural networks are used to learn the solution selection mechanism of evolutionary computation. Six evolutionary algorithms are presented to learn a novel dendritic neuron model with successful applications to classification, fitting and prediction problems [50, 59]. In addition, the authors use the "orthogonal experiment method" to systematically study the algorithm parameters and obtain optimal parameter combinations. With continuous research on evolutionary algorithms, the necessity of combining evolutionary computation with deep networks is becoming more and more prominent. A training dendritic neuron model learning method is proposed based on the differential evolutionary algorithm of scale-free networks, and the trained model shows very a high prediction accuracy in real PV power prediction problems, through which successful applications to classification and fitting problems are also achieved [60]. The classification and fitting issues have been satisfactorily addressed using the trained model. The experimental findings demonstrate that the learning method suggested in the research, which is based on the globally optimum differential evolution algorithm, may be utilized to significantly enhance the overall performance of the dendritic neuron model, are also successfully applied to complex function optimization and many practical problems to train dendritic neuron models [61]. Gao extended the original dendritic neural network to a full-complex dendritic neuron model (CDNM), deriving its error backpropagation algorithm in the complex domain and comprehensively discussing its initialization conditions, with nine different analytic functions as its activation functions [54]. The effectiveness of CDNM is successfully verified for complex XOR, non-minimum phase balance problems, and complex wind energy prediction problems. In addition, the combination of evolutionary computation and deep networks has shown outstanding results in dealing with practical problems [58].

Although various evolutionary algorithms have been proposed to learn DNMs, creating a cutting-edge DNM learning algorithm is necessary and challenging. Mirjalili introduced the whale optimization algorithm (WOA) [62] in 2016 to optimize the solution of the objective function by imitating the foraging behavior of whale schools in the ocean. WOA offers benefits such as straightforward operation, fewer parameters to specify, and strong performance in finding the optimum. Nevertheless, it has flaws including weak convergence precision and a tendency to quickly enter local optimum. To address these shortcomings, the hybrid algorithm is also quite hot recently, which compensates for their shortcomings through other algorithms. Depending on the spherical search algorithm (SS) [63, 64] and WOA, we propose a novel hybrid algorithm, namely SSWOA, to learn DNM. In SSWOA, the SS algorithm is used to improve the low convergence problem of WOA, which tends to fall into local optima. The local exploitation of the search space is the main goal of the WOA algorithm. WOA optimization can speed up convergence while preventing a local optimum by modifying non-convergence factors, establishing adaptive position weights, and setting adaptive thresholds. To our best knowledge, no hybrid algorithm was employed to understand the single dendritic neuron framework.

The contributions made in this study are listed as follows: 1) We innovatively employed a dynamic

mixing of two separate algorithms via an intricate co-evolutionary process to test if the hybrid algorithm's performance can be improved further while learning single dendritic neuron models. 2) The approach outperforms the original evolutionary algorithm, a derivative of the WOA algorithm, as well as several other methods to train neural models, according to extensive tests conducted in roughly three directions. The SSWOA algorithm may be used to more effectively resolve challenging practical problems from the standpoint of useful functions.

The remainder of the article is structured as follows: the details of DNM is elaborated in Section 2. SSWOA for learning DNM are discussed in Section 3. Section 4 provides comparison findings on 11 common classification tasks. Finally, Section 5 provides a summary and future projections.

2. Model architecture

There are 10^{11} neurons in the brain, with a total of nearly 10^{15} neural connections among the neurons. There is an average of 10^4 interconnections for a neuron upon dendrites, causing a vast range of different neuronal morphology. Nonlinearity of dendrites accounts for the dynamics of various neurons, which mechanistically explains the role of biological neurons in various circuits with behavioral significance [38]. Each neuron is capable of unequally handling thousands of different synaptic inputs. Moreover, the large dendritic structure of dendrites [32] can largely help explain why biological neural networks are able to handle quite complex tasks but consume little energy. Not only the postsynaptic signals are nonlinearly integrated in dendrites, but irrelevant background information is also filtered out there [65]. A physiologically valid single-neuron model was presented in light of the dendrites' capacity for nonlinear information processing, that is, DNM [50, 51].

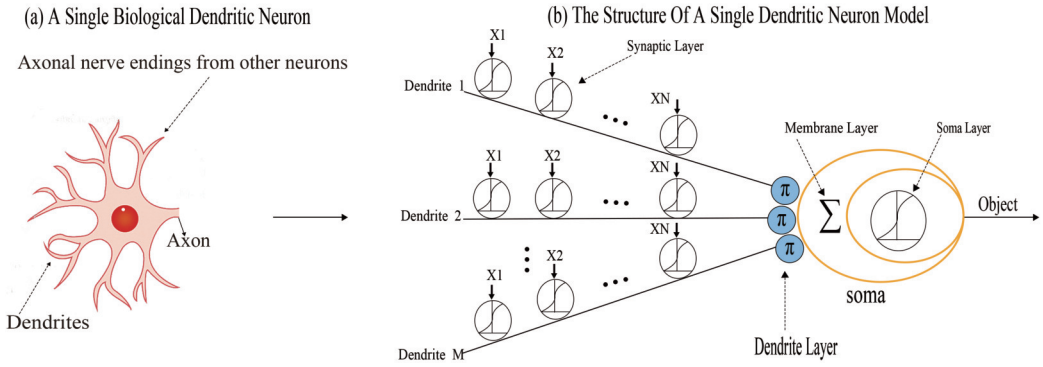


Figure 1. Dendritic neuron models: (a) the structure of neurons in biology, (b) the structure of the theoretical DNM.

DNM include four layers, i.e., the synaptic layer, dendritic layer, membrane layer, and somatic layer. The method by which dendritic neurons in living organisms acquire information is seen in Figure 1(a). A biological model of dendrites and extracellular inputs from other neurons is shown in Figure 1(b). In contrast, specifically, information from other neurons (i.e., stimuli) is collected in the synaptic layer. Information collected from other neurons (i.e., stimuli) is collected in the synapses and then transferred to the dendritic layer, where all of the data is multiplicatively processed, and the

outcomes are subsequently sent to the membrane layer for accumulative processing. All of the results are then added together and sent to the soma layer, where the outcome is processed using another sigmoid function.

2.1. Dendritic layer

Synapses enable neurons to receive signals from axons or dendrites of other synapses by joining them to other dendrites or axons. Each input signal has a unique weight or threshold that is connected to a branch of the dendrite on the axon. The relationship between the ith ($i = 1, 2, ..., N$) synapse and the jth ($j = 1, 2, ..., M$) synaptic layer, which receives input signals, is shown by the formula:

$$Y_{ij} = \frac{1}{1 + e^{-k\left(w_{ij}x_i - \theta_{ij}\right)}} \tag{2.1}$$

In this equation, x_i is the ith dendritic input, x_i is 0 or 1, Y_{ij} is the ith synaptic input to the jth output, and k is a parameter that describes the strength of presynaptic and postsynaptic connections among neurons. The connection parameters w_{ij} and θ_{ij} [50] indicate the weighting as well as threshold of synapses, respectively. A learning method will be used to teach these two sorts of parameters.

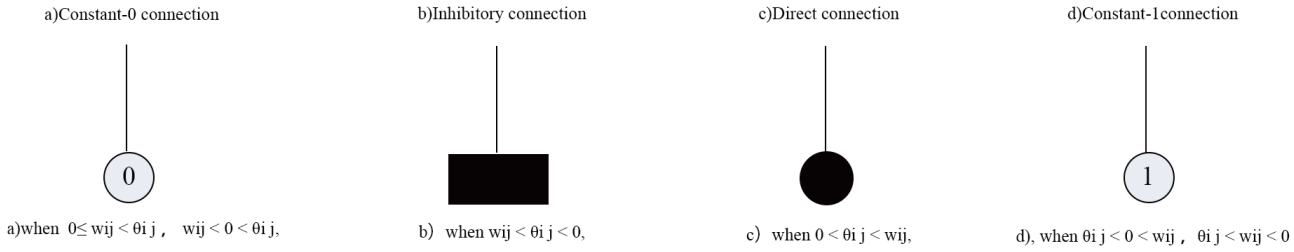


Figure 2. Four connection states of the synapse layers.

After a synapse is activated through the sigmoid function, there are four connectivity scenarios [49], as is shown in Figure 2, replying on different scores of w_{ij} and θ_{ij}: direct-connecting state is also known as being excited to connect (●), the opposite-connecting state is also known as an inhibitory connection (▬), constant-1 connection (①) and constant-0 connection (⓪). Furthermore, these four states can be separated into the six situations depicted in Figure 3. Presynaptic neural input is represented by the horizontal axis, while synaptic layer output is represented by the vertical axis.

1) As is shown in Figure 3a(1), when $0 \leq w_{ij} < \theta_{ij}$, for example, if $w_{ij} = 1.0$ and $\theta_{ij} = 1.5$, then it indicates that there is never a connection. Regardless of its input value, the result is a constant score of 0. As is shown in Figure 3a(2), when $w_{ij} < 0 < \theta_{ij}$, for example, if $w_{ij} = -1.0$ and $\theta_{ij} = 0.5$, the output score is 0 regardless of the input.

2) As is shown in Figure 3(b), when $w_{ij} < \theta_{ij} < 0$, a backward connection is established, for example, if $w_{ij} = -1.0$ and $\theta_{ij} = -0.5$, it indicates that regardless of whenever the input goes from 0 to 1, the potential is always inversely proportional to the input.

3) As is shown in Figure 3(c), when $0 < \theta_{ij} < w_{ij}$,there is a direct link created, which is a fascinating relationship. For example, if $w_{ij} = 1.0$ and $\theta_{ij} = 0.5$, the input is proportional to the output anytime when the input is changed from 0 to 1.

4) As is shown in Figure 3d(1), when $\theta_{ij} < 0 < w_{ij}$, for example, if $w_{ij} = 1.0$ and $\theta_{ij} = -0.5$, it signifies that the output is a linked constant score of 1, which is independent of the input value. As is shown in Figure 3 d(2), when $\theta_{ij} < w_{ij} < 0$, for example, if $w_{ij} = -1.0$ and $\theta_{ij} = -1.5$, 1 is the output value, which is unrelated to the input.

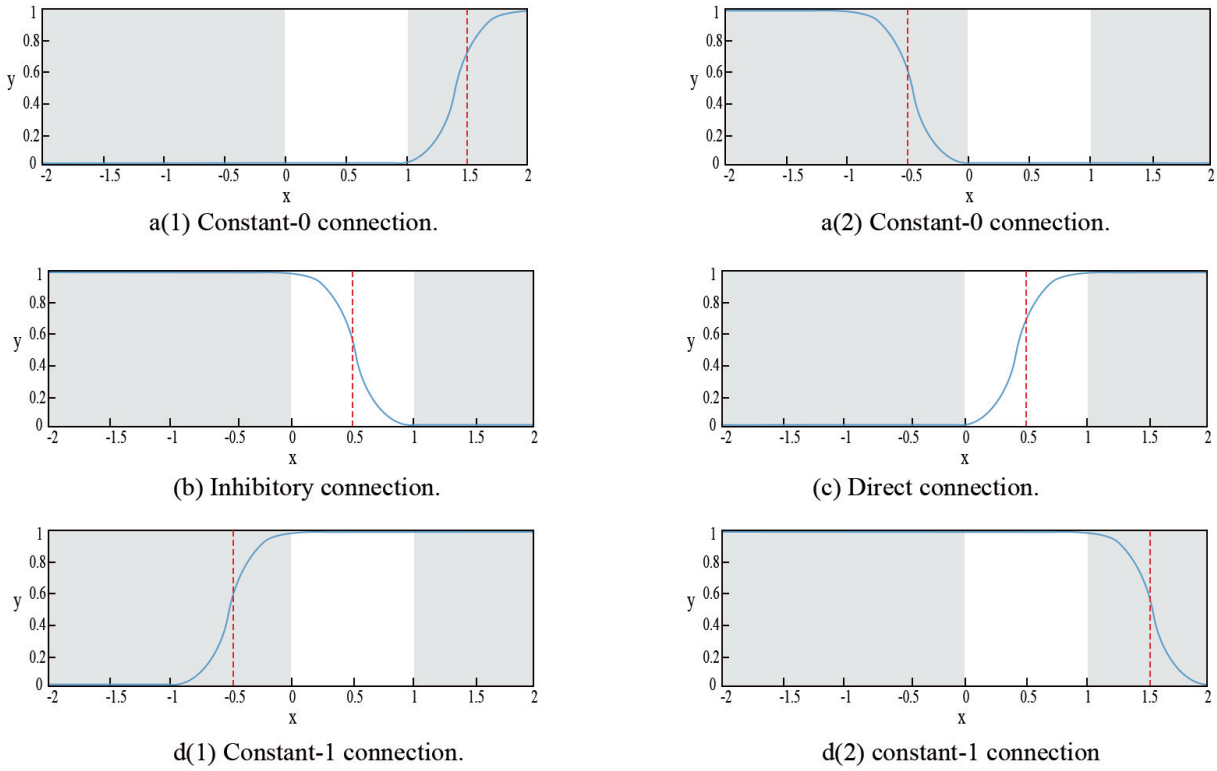


Figure 3. Six kinds of parameter settings which result in four kinds of connections in DNM.

2.2. *Dendrite layer*

Impulses are sent from the synaptic layer to the dendritic layer, which receives them. Because synapses are nonlinear, that is, nonlinear signals are processed at the dendritic layer using the multiplication operation, which is the same as logical AND, and can be explained as follows:

$$Z_j = \prod_{i=1}^{N} Y_{ij} \tag{2.2}$$

2.3. Membrane layer

Each dendrite sends signals to the membrane layer, which adds the results, before sending the processed information to the soma bodies. A logical OR operation, which can be expressed as the term for this summing process:

$$V = \sum_{j=1}^{M} Z_j \tag{2.3}$$

2.4. Soma layer

After layer-by-layer signal processing is finished, the soma layer receives membrane layer signals, and the final result is determined using a sigmoid function, which is expressed as follows:

$$O = \frac{1}{1 + e^{-k_s(V-\theta_s)}} \tag{2.4}$$

where θ_s has a score range of $[0, 1]$ and k_s is a distance score that represents the distance between the dendrites and the soma.

2.5. Neuronal pruning mechanism

Nodes and weights are eliminated through a pruning mechanism, which is the most important feature of DNM. Some dendrites and synapses can be unnecessary since the DNM has a complete connectivity pattern at first. The DNM can be pruned during training by recognizing superfluous synapses and duplicated dendrites. For many practical challenges, a simpler and unique neural structure can be generated. The neural pruning process consists of two parts: dendritic pruning and synaptic pruning. For synaptic pruning, the synaptic output is always when the synaptic layer receiving the axonal input is in a constant-1 connection. Because of the multiplication used in the dendritic layer, any arbitrary score multiplied by 1 will equal itself in the dendritic layer, and none of the results will change. There is clearly a constant-1 connection in which the synaptic input has the least effect on the output of the dendritic layer. Thus, those synapses with a constant-1 connectivity have such little impact on the dendritic layer's output that it is possible to disregard them entirely. Therefore, throughout the computation we may completely disregard these synaptic layers. For pruning of dendrites, the output is always 0 when the synaptic layer receiving the input signal is in a constant-0 connection, regardless of the input value. When one synaptic output is 0, the output of all other synapses is also 0 due to the multiplication operation. With a synapse's output being 0, the entire dendritic branch's outcome will also be 0. They should be removed because they barely affect the outcome of the adjacent soma. Thus, such dendritic layers ought to be eliminated.

DNM can also be used to implement neuronal pruning. Figure 4 shows the simulation for synaptic and dendritic pruning. First, the DNM has with two dendrites and four synapses, as shown in Figure 4(a). From the Figure 4(b) and (c), it can be seen that synaptic and dendritic pruning has occurred. According to the pruning mechanism of the neuron, the first synaptic layer needs to be discarded because it belongs to the case of constant-1 connection. The fourth synapse on the second dendrite belongs to the constant-0 connection case, so this dendrite should also be eliminated. Finally, the morphology of the neuron after DNM pruning is shown in Figure 4(d).

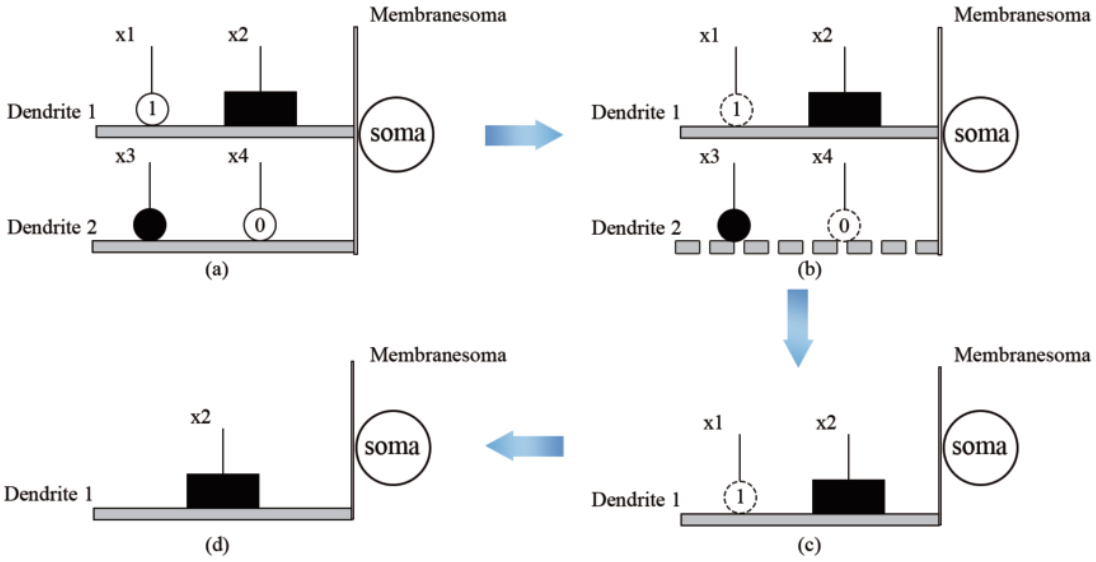


Figure 4. Simulation of synaptic and dendritic pruning.

3. Learning algorithm SSWOA

To train DNMs, an amount of training methods have been developed. The stochastic gradient descent method BP has been widely employed [51, 52, 66], but it suffers from the local trapping problems, which influences the performance of models with high inductive power. An effective global search optimization method, biogeography-based optimization (BBO) [50] outperforms ant colony optimization (ACO) [59], genetic algorithm (GA), evolutionary strategy (ES), particle swarm optimization (PSO), and BP by a wide margin, showing to be a promising DNM training method. In addition, states of matter search algorithm (SMS) [67] has also shown its superior performance in training DNMs. Recently, DNM classification [68] and wind speed prediction [69] have both employed differential evolution (DE) and one of its variations, LSHADE [70]. The learning capacity of DNMs has been significantly enhanced by the most recent state-of-the-art information feedback-based differential evolution algorithm (IFDE) [71]. Given that the learning algorithm significantly influences how well DNMs perform, single algorithms have been used to train DNMs for a long time, which motivates us to further enhance performance by utilizing several algorithms to address a single algorithm's faults. Therefore, the major objective of this work is to provide the best learning method for DNMs.

The mathematical optimization challenge of minimizing the objective function can be used to represent the learning problem of DNMs, which is represented as:

$$f(X) = MSE(X_k) = \frac{1}{P}\sum_{p=1}^{P}(T_p - O_p)^2 \tag{3.1}$$

where the SSWOA is the mean square error (MSE) between the computed target value and the output of the DNM for individual X_i. P stands for the quantity of training samples, while T_p and O_p stand for the pth target value and the DNM's actual output, respectively. Individual solutions represented by the

following are among the learnable parameters in the DNM's synaptic layer:

$$X_g = \{w_{11}, w_{12}, ..., w_{nM}, q_{11}, q_{12}, ..., q_{nM}\} \tag{3.2}$$

where M is the number of DNM dendrites and n is the number of external inputs. As a result, the dimensionality of the DNM learning problem as it has been defined is $D = 2nM$.

3.1. Characteristics of whale optimization and spherical search

The SS algorithm [63] is a vast, undirected search method that maintains a reasonable balance between the expansion and exploration of the search space, which can be used to compute spherical boundaries and develop innovative experimental answers for spherical boundary surfaces for non-linear global optimization problems with constrained boundaries. SS algorithm starts with half of the better solutions concentrating on development and the other half of the worse solutions concentrating on exploration, in contrast to the majority of algorithms, which usually begin with comprehensive exploration and then begin development, leading to increased diversity in the better one half of the solutions and increased adaptability in the other half of the worse solutions. Nonetheless, the search mechanism of SS algorithms leads to relatively poor exploration and often settles for local optima.

WOA [62] is a multi-application optimization strategy that mimics the preference behavior of humpback whales. Nonetheless, the ability to leverage the whale optimization technique is heavily dependent on the distance between the search unit and the current best solution, and a good enough current optimal solution is required. In this work, the benefits of the SS algorithm are combined with the WOA approach to generate a hybrid whale algorithmo-based search algorithm (SSWOA). One aspect of the SSWOA remains the computation of the search direction, thus the SS algorithm is used to approximately establish the orientation of the ideal solution, and then the whale optimization method is used to discover the optimal solution, thereby improving its performance.

The standard WOA has disadvantages such as slow convergence, the tendency to fall into local optima and low convergence accuracy, which will affect the overall performance. Hence, in this work, two improvements are innovatively made to the WOA: a non-linear convergence coefficient and a co-evolutionary control strategy.

3.2. Spherical search optimization algorithm

Proposed by Abhishek Kumar [63] in 2019, a group-based meta-heuristic approach is the spherical search (SS) method, which can contribute to solving nonlinear restricted global optimization problems. The modified SS algorithm solution is shown in Figure 5 as a two-dimensional search space. The surface of the spherical boundary is used in the SS method to generate the suitable next solution for each individual solution. The method concentrates on global exploration while the spherical boundary is small, while it specializes on local advancement when the spherical boundary is large. As a result, the SS algorithm may in certain cases aid in achieving a balance between the advancement and exploration of the search area. The solution initialization in the SS algorithm is denoted as follows:

$$X_{ij}^0 = X_{lj} + (X_{hj} - X_{lj}) * rand(0, 1] \tag{3.3}$$

where X_{hj} and X_{lj} denote the jth element's upper and lower boundaries, respectively. Within the limit, $rand(0, 1]$ generates random numbers from a uniform distribution in $(0, 1]$.

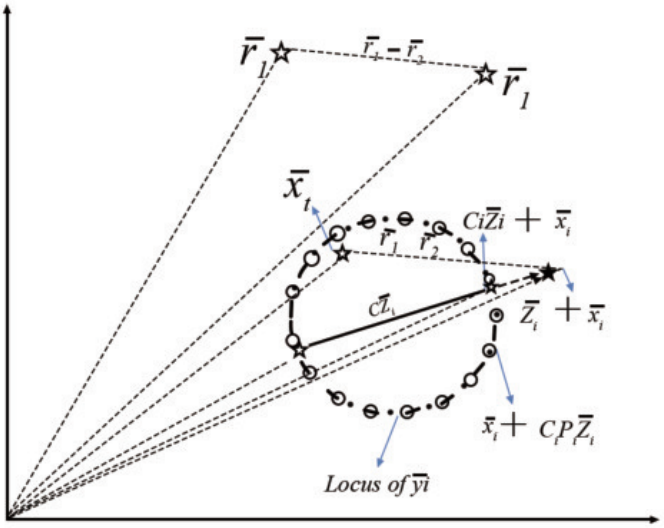


Figure 5. SS algorithm in 2-D search space for solution update solution space.

The SS method uses the following formula to provide the appropriate test for the ith solution:

$$\epsilon = \phi + C_i^k P_i^k \varphi \tag{3.4}$$

The projection matrix P_i determines the score of ϵ on the spherical edge of (D-1). For a given solution, ϕ corresponds to multiple different P_i^k values and gives several ϵ values. The range of the step length control vector C_i^k is $[0.5, 0.7]$. The search direction is φ, which is produced by utilizing two random solutions as target locations. With "towards-rand" having stronger exploration skills and "towards-best" having better local exploitation capabilities, the SS method uses both of these features to determine the search direction. As a result, in the exploration space, the balance between the advancement and exploration is maintained. Through the instilling of the variety in a superior solution as well as inferior options, adaptation is promoted.

The direction of the search is φ. It is necessary to calculate the search direction as follows, using $r1$ and $r2$:

$$\varphi = (\bar{X}_t^{\,k} + r_1^{\,k} - r_2^{\,k}) - \phi \tag{3.5}$$

The desired point is ϕ. In Eq (3.5), from the present set of solutions, $r1$ and $r2$ are chosen at random in the population.

The search direction, φ for the kth iteration of the ith solution in "towards-rand" is determined as:

$$\varphi = \bar{X}_{pi}^{\,k} + \bar{X}_{qi}^{\,k} - \bar{X}_{ri}^{\,k} - \phi \tag{3.6}$$

The search direction, φ, for the kth iteration of the ith solution is determined in "towards-best" as:

$$\varphi = \bar{X}_{pbesti}^{\,k} + \bar{X}_{qi}^{\,k} - \bar{X}_{ri}^{\,k} - \phi \tag{3.7}$$

The population of i solutions is updated and filled with a new set in the subsequent iteration using a greedy selection strategy that adheres to the following guidelines: If the test solution's score $f(\epsilon)$ is less than that of the solution to the goal function $f(\phi)$, therefore, y_i substitutes x_i, expressed as:

$$\vec{Xi}^{(k+1)} = \begin{cases} \epsilon & \text{if } f(\epsilon) \leq f(\phi) \\ \phi & \text{otherwise} \end{cases} \tag{3.8}$$

3.3. *Whale optimization algorithm*

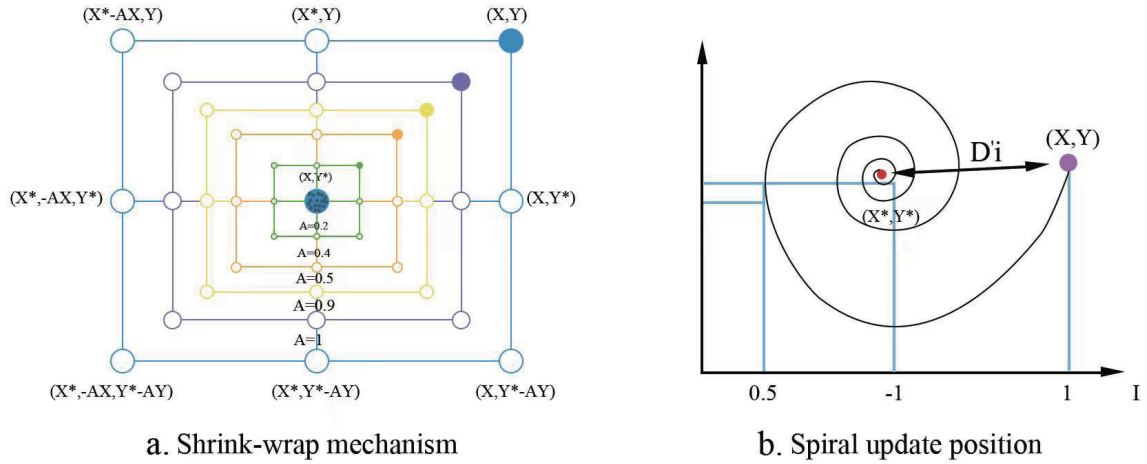


Figure 6. The WOA's bubble net search mechanism: (a) a shrinking envelope mechanism and (b) a spiral update position.

In 2016, Seyedali Mirjalili [62] introduced the whale optimization algorithm as a population-based search strategy, as well as a program that mimics the hunting behavior of humpback whales via bubble nets. Three parts make up the algorithm: encircling the target, unleashing a bubble net assault, and hunting for prey. Figure 6 provides a rough idea of the bubble net search mechanism implemented in WOA: Figure 6(a) is the shrink-wrap mechanism and Figure 6(b) is the spiral update position. Considering that the intended prey is the best option for the existing population, other whales in the population, after the selection of prey, would update their locations based on the current position of the prey. Suppose that the ideal solution in the current population is the target prey, other whales in the population, after the selection of prey, would update their locations based on the current position of the prey, with the mathematical model as follows:

$$\vec{D} = \left| \vec{C} \cdot \vec{X^t_{best}} - \vec{X^t_i} \right| \tag{3.9}$$

$$\vec{X^{t+1}_i} = \vec{X^t_{best}} - \vec{A} \cdot \vec{D} \tag{3.10}$$

where t stands for the current iteration count, and $\vec{X^t_{best}}$ represents the optimum solution's position in the present group vector. $\vec{X^t_i}$ shows where the whale is at the moment, $\vec{A} \cdot \vec{D}$ indicates enclosing measure,

and $\vec{A}$ and $\vec{C}$ relate to the coefficient vectors which are represented as:

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \tag{3.11}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{3.12}$$

where $\vec{a}$ decreases linearly in the iterative procedure from 2 to 0, $\vec{r_1}$ and$\vec{r_2}$ are random vectors produced using a uniform distribution in the [0, 1] range. Equation (3.12) states that the search unit (whale) modifies its location in accordance with the location of the ideal solution (prey). By changing the values of the A and C vectors, the whale's area surrounding its prey may be managed.

According to Eq (3.13), reducing the score of a variable in Eq (3.11) results in the contraction and enclosing behavior of the whale:

$$\vec{a} = 2 - t \cdot \frac{2}{MaxIter} \tag{3.13}$$

where t denotes the number of permissible iterations and $MaxIter$ the maximum number of allowed iterations. For a spiral route replica, the distance between the search unit $\vec{X_i^t}$ and the best response so far $\vec{X_{best}^t}$ is computed first, followed by the spiral equation as in Eq (3.14). Following that, is used to establish the position of the nearest search unit:

$$\vec{X_i^{t+1}} = \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X_{best}^t} \tag{3.14}$$

$$\vec{D} = \left| \vec{C} * \vec{X_{rand}^t} - \vec{X_i^t} \right| \tag{3.15}$$

$$\vec{X_i^{t+1}} = \vec{X_{rand}^t} - \vec{A} \cdot \vec{D} \tag{3.16}$$

where $\vec{D}$ is the distance between the ith Whale and its prey, b is a constant defining the logarithmic spiral form, and l is a random value between [-1, 1]. This is the best response so far. The whale must hunt following the spiral pattern the prey takes as it shrinks the area around it. So, according to the WOA algorithm, there is a 0.5 chance that the whale would use two different hunting strategies. Here is the mathematical model:

$$\vec{X_i^{t+1}} = \begin{cases} \vec{X_{best}^t} - \vec{A} \cdot \vec{D} & \text{if } P < 0.5 \\ \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X_{best}^t} & \text{if } P \geq 0.5 \end{cases} \tag{3.17}$$

3.4. Nonlinear convergence factor

Global search and local advancement are two actions performed by swarm intelligence algorithms. If the two are not appropriately timed, the algorithm will enter local optimization or suffer from poor performance and slower convergence. Population will remain diverse if the global search strength is strong enough to prevent the algorithm from settling into the local optimum. The speed of the algorithm's convergence will increase whether local advancement can be ensured by the local advancement strength. The WOA algorithm illustrates that the global search and local advancement capabilities of WOA are dependent on parameter A. According to Eqs (3.10), (3.14), and (3.15), the parameter "$\vec{A}$" is reliant on the convergence factor "$\vec{A}$" in the phase of comparing WOA. If the convergence factor $\vec{a}$ is larger, the algorithm's ability for global exploration will be strengthened, preventing it from settling into the local optimum. On the other hand, if the convergence factor $\vec{a}$ is

less, the algorithm will be more capable of global exploration. The algorithm's capacity for local growth will be enhanced, and will have a faster convergence rate. As the amount of iterations grows in WOA, $\vec{a}$ decreases linearly. Based on the literature, the following formula must be employed to ensure global exploration and local advancement capabilities, as well as to increase the algorithm's convergence rate:

$$\vec{a} = \left(2 - t.\frac{2}{maxIter}\right).\left(1 - \frac{t^3}{maxIter^3}\right) \tag{3.18}$$

$$\vec{a} = \left(-1 - t.\frac{-1}{maxIter}\right).\left(1 - \frac{t^3}{maxIter^3}\right) \tag{3.19}$$

Equation (3.16) should be used when $\vec{a}$ is between 2 and 0, while Eq (3.17) should be used when $\vec{a}$ is between -1 and -2. Among them, t which indicates the current iteration count and *maxIter*, which stands for the maximum iteration count. The first phases $\vec{a}$ will improve the capability of global investigation and quicken the velocity of algorithm convergence when the effects of the convergence factor $\vec{a}$. Later on, it will be more biased, with a focus on local advancement.

3.5. Co-evolutionary control strategy

WOA's predatory behavior targeting humpback whale bubble nets employs two strategies: spiral renewal and contraction circular processes. To choose a foraging strategy, the randomly generated P in $[0, 1]$ is compared to the algorithm's assumed threshold of 0.5 by modeling the two processes (contracting and enclosing prey and spiral paths) as illustrated in the calculation. Repeated testing on the data set given in the image has made it clearly evident that WOA's convergence time is unnecessarily slow, making the local optimization issue easy to introduce. The possibility of altering the threshold in WOA has been demonstrated in [72]. The score of the threshold will have an effect on the algorithm's findings while attempting to find the best answer. If the score is too high, the algorithm's optimization effect will be very little, but if the score is too small, the algorithm's capacity to evolve locally will be hindered. The feasibility of adaptive parameters is supported by references [72, 73]. WOA's original 0.5 threshold is replaced with the adaptive threshold. It will change when the algorithm is adjusted repeatedly to discover the best mechanism for both local and global advancement. Based on the above considerations, the threshold is modeled in the following way:

$$\gamma = 1 - \left[\frac{1}{\alpha + \beta}.(\alpha.\frac{t^\alpha}{maxIter^\alpha} + \beta.\frac{t^\beta}{maxIter^\beta})\right] \tag{3.20}$$

Among them, γ represents the threshold in WOA, while t and *maxIter* stand for the current and maximum iterations, respectively. α and β denote the control parameters, respectively. The updating rule of individuals changes to the following equation when $\alpha = 3$ and $\beta = 2$.

$$\vec{X_i^{t+1}} = \begin{cases} \vec{X_{best}^t} - \vec{A} \cdot \vec{D} & \text{if} P < \gamma \\ \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X_{best}^t} & \text{if} P \geq \gamma \end{cases} \tag{3.21}$$

3.6. Adaptive position weights

Algorithm 1: Learn the implementation procedures of DNM by SSWOA.

Initially, set the iteration number $t = 1$, the maximum function evaluation number $MaxFES = 30000$.
begin
  Create the whale population from scratch X_i (i = 1, 2, ..., n).
  Evaluate the population using the fitness function defined in Eq (3.1).
  Determine each search agent's fitness.
  $c_i \leftarrow rand\,(0, 1]$.
  while $nFES \leq MaxFES$ **do**
    $A \leftarrow ComputeOrthogonalMatrix\,()$.
    for $i = 1$ *to* N **do**
      $diag\,(b_i) \leftarrow ComputeBinaryVector\,()$.
      if $nFES < 1/2FES$ **then**
        if $i < 0.5*N$ **then**
          SS is used to update the population's towards-rand.
        else
          Improved WOA with adaptive thresholds is used to update the population's best-case scenario.
      else
        Update the towards-rand part of the population Using SS.
    Examine whether any search agents move beyond the search space.
    Evaluate the population using the fitness function defined in Eq (3.1).
    Determine each search agent's fitness.
    Update if there is a better solution.
    $nFES = nFES + 1$.
    $\overrightarrow{x_i} \leftarrow Selection\left(\overrightarrow{x_i}, \overrightarrow{y_i}\right)$
Output the best solution y_i in the population which consists of optimal weights w_{ij} and thresholds q_{ij} for DNM.
This is followed by morphological transformation, dendritic and synaptic pruning, and hardware implementation for DNM.

The position vector of the randomly chosen whales, *veca* and *vecA*, is the primary factor that may affect the WOA location update process in addition to the WOA algorithm's location update process. i.e., the population $X^{\vec{t}}_{best}$'s position vector that was randomly chosen [74]. Moreover, taking into consideration the numerous interactions between randomly selected whales and their prey at various times during the algorithm's iterative phase, adaptive parameters are utilized as weight coefficients and paired with the aforementioned threshold to rewrite the locations of WOA as follows:

$$\omega = \frac{t^3}{maxIter^3} \tag{3.22}$$

$$\vec{X_i^{t+1}} = \vec{X_i^t} \cdot \omega - \vec{A} \cdot \vec{D},\ |A| \geq 1, P < \gamma \tag{3.23}$$

$$\vec{X_i^{t+1}} = \vec{X_{best}^t} \cdot \omega - \vec{A} \cdot \vec{D},\ |A| < 1, P < \gamma \tag{3.24}$$

$$\vec{X_i^{t+1}} = \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X_{best}^t} \cdot (1 - \omega), P \geq \gamma \tag{3.25}$$

where ω is the prey adaptive weight, which is chosen between [0,1]. If ω increases as the number of iterations increases, so will the optimal solution of the prey chosen in each iteration, i.e., for the present population. In Eqs (3.10), (3.16), and (3.17), the weight coefficient increases as the number of iterations increases. While updating the location of the whale in Eqs (3.23)–(3.25), one may investigate the best solution around the prey using the algorithm's rapid convergence and optimization accuracy Eq (3.25), thereby improving the local advancement ability. The whole implementation of SSWOA to learn DNM is summarized in Algorithm 1.

4. Experimental results

4.1. Experimental setup

To comprehensively study of learning performance of SSWOA on DNM, a number of representative optimization techniques have been compared, which are summarized within three categories: the variants of SSWOA (including its components), some other evolutionary algorithms, and some other machine learning techniques. These include the latest variant of the WOA algorithm, i.e., the spatial bound whale optimization algorithm (SBWOA) [75] proposed in 2021, the cellular automata-based whale optimization algorithm (CAWOA) [76] proposed in 2020, the nonlinear adaptive weight and golden sine operator-based whale optimization algorithm (NGS-WOA) [77] proposed in 2020, the hybrid improved whale optimization algorithm (HIWOA) [78] proposed in 2019, IFDE [71] proposed in 2021, which is currently regarded as the most effective method for training DNM, BBO [50], SMS [67], the most effective version of SS, i.e., SASS [63], BP, K-nearest neighbor (KNN), and support vector machines (SVM). All experiments were conducted using MATLAB (R2019b) and implemented on a machine with a 32 GB RAM and Intel(R) Core i7-10700K 5.1 GHz processor. To better verify the difference between the hybrid SSWOA algorithm and its peer, we conduct the experiment from two aspects, namely the benchmark function in IEEE CEC2017 [79], and the artificial dendritic neural model training problem (DNM). The following evaluation tools were used to assess the performance of SSWOA.

4.2. Verifying the performance of SSWOA on IEEE CEC2017

To assess the performance of the SSWOA approach, single-peaked functions (F1–F3), basic multi-peaked functions (F4–F10), hybrid functions (F11–F20), and combination functions (F21–F30) taken from IEEE CEC2017 are used. The studies involved a comparison of SSWOA with IFDE, SS, SMS, BBO, WOA, CAWOA, HIWOA, NGSWOA, and SBWOA. Instead, the commonly used parameters are set as follows: the number of function evaluations with a population size of 100, the maximum number of function evaluations of $10^4 \times D$, where D is the number of dimensions. Each algorithm to obtain statistics was executed 51 times for each function separately.

1) $W/T/L$: T represents the amount of functions where SSWOA performs similarly to other algorithms. L represents the amount of functions for which SSWOA performs noticeably worse. W represents the amount of functions for which SSWOA surpasses the other algorithms.

2) The convergence curve represents the most recent optimum history recorded at each repeat. The function evaluation numbers are shown on the x-axis, while the average error score is shown on the y-axis.

3) The box-and-whisker graph includes boxes and peaks. Maximum values are indicated by the line above the blue box, and minimum values are indicated by the line below the blue box. The box's upper and lower edges correspond to the first and third quartiles, respectively. The red "+" sign represents the extreme extremes, while the red line represents the median. Furthermore, the greater the discrepancy between the highest and minimum numbers, the more unstable the method's performance.

4.3. Verifying the performance of SSWOA for learning DNM

Additionally, 11 frequently used classification problems from the UCI Machine Learning Repository were used to verify the performance of the proposed SSWOA for learning DNM, including the BreastEW, Australia, CongressEW, Exactly, Heart, German, KrVsKpEW, Ionosphere, SpectEW, TIC-tac-toe, and Vote datasets. Additionally, the number of attributes (features other than class labels), the total sample size and the learning space dimension of the DNM for each problem (i.e., D) are summarized in Table 1.

Table 1. DNM' parameters setting in UCI classification datasets.

Classification datasets	Attributes	Instances	Dimension of learning space
BreastEW	30	568	300
Australia	14	690	280
CongressEW	16	435	96
Exactly	13	1000	260
German	24	1000	48
KrVsKpEW	36	3196	72
Ionosphere	34	351	204
SpectEW	22	267	44
TIC-tac-toe	9	8	90
Vote	16	300	96
Heart	13	270	200

Samples including missing values were removed for a fair comparison or because of the classifier used. The samples for each dataset were randomly divided. On the same computer, each method for these comparisons was ran 30 times separately with the parameters set to recommended levels in the relevant literature [80]. The amount of function evaluations for each of the algorithms under comparison was made to be the same in order to provide a generally fair performance comparison, namely, $MaxFES = 3000$. 70% of the samples were used for training, with the remaining 30% as test samples.

First, we normalize all the values to prevent small numerical attributes from being dominated by large numerical attributes. Therefore, the variables for which input is performed are normalized.

Specifically, all attributes are normalized to the range [0, 1] based on a min-max normalization rule. The equation can be expressed as follows:

$$X_{normalized} = \frac{X_{original} - X_{min}}{X_{max} - X_{min}} \tag{4.1}$$

Table 2. Orthogonal experimental design ($L_{25}(5^3)$) method for hyper-parameters in DNM.

No.	M	K	Q	No.	M	K	Q
#1	1	1	0.1	#14	5	15	0.1
#2	1	5	0.3	#15	5	20	0.3
#3	1	10	0.5	#16	10	1	0.7
#4	1	15	0.7	#17	10	5	0.9
#5	1	20	0.9	#18	10	10	0.1
#6	3	1	0.3	#19	10	15	0.3
#7	3	5	0.5	#20	10	20	0.5
#8	3	10	0.7	#21	20	1	0.9
#9	3	15	0.9	#22	20	5	0.1
#10	3	20	0.1	#23	20	10	0.3
#11	5	1	0.5	#24	20	15	0.5
#12	5	5	0.7	#25	20	20	0.7
#13	5	10	0.9				

Table 3. DNM' parameters setting in UCI classification datasets.

data set	M	K	θ
BreastEW	5	5	0.7
Australia	10	15	0.3
CongressEW	3	5	0.5
Exactly	10	5	0.9
Heart	10	10	0.1
German	1	10	0.5
KrVsKpEW	1	5	0.3
Ionosphere	3	5	0.5
SpectEW	1	5	0.3
TIC-tac-toe	5	5	0.7
Vote	3	5	0.5

Furthermore, since three hyper-parameters of DNM need to be predetermined, Taguchi's orthogonal experimental design technique was used to determine the ideal arrangement of hyper-parameter settings. This contained dendrite number M, firing parameter Q, and distance parameter K [50, 81]. As Taguchi's method is not a full factor analysis, but tests only partial orthogonal combinations between factors and levels, with the promise of minimal experimental testing and the most accurate statistical calculations of the implementation-related parameters. In

DNM, each factor (i.e., hyper-parameter) was set to five levels as the following: $M \in \{1, 3, 5, 10, 20\}$, $Q \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, and $K \in \{1, 5, 10, 15, 20\}$. In order to completely find the most promising set of parameters, a thorough factor analysis would thus need $5^3 = 125$ trials, which is clearly time-consuming. Taguchi technique can better go about this task of counting by using only an orthogonal array $L_{25}(53)$ containing 25 combinations, as shown in Table 2. Table 3 shows the optimal settings of the three hyper-parameters for each classification problem.

In our studies, we employed a wide range of performance criteria in order to further precisely assess our model. True positives (TP) are cases that were both expected to be positive and were in fact positive. The term "true negatives" (TN) refers to cases that are both predicted and actual negative. False positive instances (FP) are those that are projected to be positive but are actually negative. False negatives (FN) are instances where a negative forecast was made but the result was actually positive. Accuracy (Acc), which is calculated using the formula below, is the proportion of accurate predictions to all samples.

$$Acc = \frac{TP + TN}{TP + FP + FN + TN} \tag{4.2}$$

The true positive rate is defined as the ratio of genuine positive cases in the positive class predicted by the classifier to all positive instances (TPR), also known as sensitivity. This ratio is determined by the following formula:

$$TPR = \frac{TP}{TP + FN} \tag{4.3}$$

Rate of False Positives (FPR) FPR is the ratio of real negative instances to all negative instances in the positive class that the classifier predicted. The following formula yields:

$$FPR = \frac{FP}{FP + TN} \tag{4.4}$$

The true negative rate (TNR), also known as specificity, is the ratio of genuine negative cases predicted by the classifier to all actual negative instances. It is calculated using the following formula:

$$TNR = \frac{TN}{FP + TN} \tag{4.5}$$

The proportion of individuals who are labeled as negative but are actually positive is known as the false negative rate (FNR). The following formula yields:

$$FNR = \frac{FN}{TP + FN} \tag{4.6}$$

The bigger the FPR, the more negative categories are really in the expected positive categories (horizontal axis of ROC FPR: 1-TNR, 1-Specificity). TPR sensitivity (positive category coverage), the higher the TPR, the more positive categories are predicted in the positive category. The ideal goal with $TPR = 1$ and $FPR = 0$, which corresponds to the $(0, 1)$ point on the graph; hence, the closer the ROC curve is to the $(0, 1)$ point, the larger the divergence from the 45-degree diagonal, and the higher the sensitivity and specificity values. As the ROC curve is generally above the straight line $y = x$, its value is between 0.5 and 1. The reason for using AUC as an evaluation metric is that the ROC curve

usually does not give a clear indication of which classifier is more effective. It is given by the following equation.

$$AUC = \int_0^1 TPR(FPR)dFPR \tag{4.7}$$

5. Performance comparison

5.1. Algorithm performance comparison

Tables 4–6 summarize the classification findings for the mean test error and standard deviation for all comparison methods (std). We evaluated and compared several methods in dimensions 30, 50, and 100 at IEEE CEC2017, as well as examined the performance change of SSWOA from medium to large dimensions. It is important to note that F2 was left in place despite the function's unreliable performance in CEC2017, particularly at high dimensions due to the guaranteed data integrity. Compared to IFDE, SS, SMS, BBO, WOA, CAWOA, HIWOA, NGSWOA, and SBWOA, the number of wins for SSWOA in Table 4 is 4, 19, 27, 21, 29, 29, 29, 29, 29, 29, 29, 29, in Table 5 are 4, 17, 24, 17, 28, 28, 29, 28, 28, 28, and in Table 6 are 4, 25, 27, 16, 27, 28, 27, 27, 27, 27, respectively. It is shown that SSWOA outperforms all algorithms except IFDE in the 30, 50 and 100 dimensions of CEC2017, indicating that SSWOA performs well in the medium, high and large dimensions of the function.

Figures 7 and 8 provide the experimental findings to further demonstrate the advantages of SSWOA. It is clear from the convergence plots that SSWOA's convergence ability and speed are superior to those of comparable methods, with the exception of IFDE. It is evident from Figure 7 that the SSWOA convergence curve in F20 at $D = 30$ is very different from other algorithms.

To demonstrate the benefits of SSWOA's performance in training DNM models for classification, several comparison algorithms were adopted simultaneously to train DNM models and classify datasets. The comparison results are presented in Table 7, where the criteria for comparison contain accuracy and standard deviation, and the data of optimal classification results is shown in bold. Although some are inferior to the non-algorithmic categories in terms of results, four results stand out in comparison to the algorithm and outperform the other algorithms. This also demonstrates the effectiveness and applicability of the SSWOA algorithm.

To better verify whether the hybrid algorithm SSWOA is the optimal choice for learning DNM, the Wilcoxon symbolic ranking test [82] was conducted to identify the differences between each set of data, and to determine if there were any significant differences between each pair of learning algorithms, and to figure out whether the differences between algorithms were at the significance level of $\alpha = 0.05$. The p-values obtained, which are all below than the significance threshold, are included in Table 8 as well. Taking Tables 7 and 8 into account, it can prove that the hybrid algorithm SSWOA outperforms other comparative algorithms in terms of classification accuracy.

Table 4. Experiment results on IEEE CEC 2017 on 30 dimensions.

	SSWOA		IFDE		SS		SMS		BBO	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	1.000E+02	1.025E-14	1.000E+02	0.000E+00	1.000E+02	6.962E-15	8.091E+06	3.047E+06	4.859E+04	1.970E+04
F2	2.000E+02	8.637E-10	2.000E+02	9.749E-15	2.000E+02	6.713E-10	5.111E+24	3.575E+25	1.229E+03	3.497E+03
F3	3.000E+02	4.890E-14	3.000E+02	0.000E+00	3.000E+02	4.823E-14	1.882E+04	5.335E+03	3.268E+02	9.881E+00
F4	4.638E+02	3.233E+01	4.188E+02	2.667E+01	4.545E+02	3.868E+01	6.025E+02	7.778E+01	4.817E+02	2.666E+01
F5	6.642E+02	8.277E+00	5.102E+02	1.981E+00	6.648E+02	9.606E+00	6.527E+02	3.673E+01	5.498E+02	1.363E+01
F6	6.000E+02	1.044E-04	6.000E+02	3.272E-08	6.000E+02	1.314E-03	6.422E+02	6.368E+00	6.001E+02	2.226E-02
F7	8.764E+02	8.389E+00	7.400E+02	2.077E+00	8.922E+02	7.987E+00	9.969E+02	6.078E+01	7.719E+02	1.233E+01
F8	9.491E+02	9.015E+00	8.111E+02	2.025E+00	9.634E+02	9.682E+00	9.205E+02	3.207E+01	8.556E+02	1.586E+01
F9	9.000E+02	1.254E-02	9.000E+02	0.000E+00	9.000E+02	9.007E-02	3.639E+03	9.918E+02	9.022E+02	5.301E+00
F10	7.496E+03	2.669E+02	2.752E+03	1.715E+02	7.928E+03	2.409E+02	5.284E+03	6.973E+02	4.251E+03	6.285E+02
F11	1.149E+03	2.411E+01	1.114E+03	1.691E+01	1.157E+03	2.673E+01	1.371E+03	8.786E+01	1.240E+03	5.599E+01
F12	2.794E+03	1.505E+03	2.239E+03	3.353E+02	3.736E+03	3.032E+03	9.534E+07	9.445E+07	6.112E+05	3.795E+05
F13	1.371E+03	1.234E+01	1.315E+03	6.847E+00	1.423E+03	5.015E+01	1.454E+05	1.169E+05	8.160E+03	5.435E+03
F14	1.455E+03	4.242E+00	1.422E+03	1.303E+00	1.462E+03	5.248E+00	6.136E+04	6.097E+04	3.330E+04	3.707E+04
F15	1.527E+03	3.430E+00	1.504E+03	1.481E+00	1.537E+03	9.508E+00	5.959E+04	3.872E+04	5.466E+03	4.934E+03
F16	2.734E+03	1.476E+02	1.802E+03	8.986E+01	2.779E+03	1.485E+02	3.217E+03	3.639E+02	2.584E+03	2.998E+02
F17	1.887E+03	2.571E+01	1.737E+03	5.908E+00	1.954E+03	5.906E+01	2.320E+03	2.098E+02	2.043E+03	1.719E+02
F18	1.828E+03	2.404E+00	1.823E+03	1.859E+00	1.832E+03	3.741E+00	9.864E+05	1.017E+06	2.357E+05	2.384E+05
F19	1.923E+03	1.462E+00	1.905E+03	1.686E+00	1.926E+03	2.268E+00	4.610E+06	2.972E+06	6.832E+03	4.802E+03
F20	2.378E+03	9.271E+01	2.050E+03	9.671E+00	2.412E+03	9.034E+01	2.524E+03	1.332E+02	2.507E+03	1.754E+02
F21	2.440E+03	8.854E+00	2.309E+03	1.784E+00	2.456E+03	8.907E+00	2.451E+03	2.845E+01	2.351E+03	1.285E+01
F22	2.300E+03	0.000E+00	2.300E+03	0.000E+00	2.300E+03	6.431E-14	2.339E+03	5.025E+01	3.294E+03	1.665E+03
F23	2.775E+03	3.339E+01	2.645E+03	3.585E+00	2.803E+03	1.392E+01	2.936E+03	7.343E+01	2.716E+03	1.957E+01
F24	2.899E+03	6.007E+01	2.822E+03	3.502E+00	2.959E+03	3.457E+01	3.109E+03	8.489E+01	2.889E+03	2.192E+01
F25	2.887E+03	1.563E-02	2.887E+03	1.275E+00	2.887E+03	3.726E-02	2.961E+03	3.008E+01	2.888E+03	3.993E+00
F26	4.374E+03	6.709E+02	3.125E+03	3.111E+02	4.156E+03	7.602E+02	6.097E+03	1.009E+03	4.475E+03	2.937E+02
F27	3.194E+03	8.725E+00	3.207E+03	7.417E+00	3.197E+03	1.002E+01	3.462E+03	6.493E+01	3.257E+03	1.657E+01
F28	3.117E+03	3.946E+01	3.111E+03	3.266E+01	3.144E+03	5.576E+01	3.347E+03	5.401E+01	3.210E+03	1.259E+01
F29	3.499E+03	4.767E+01	3.340E+03	8.677E+00	3.604E+03	8.944E+01	4.428E+03	3.207E+02	3.701E+03	1.961E+02
F30	5.237E+03	1.442E+02	5.075E+03	6.767E+01	5.352E+03	2.348E+02	1.275E+07	1.257E+07	7.939E+03	1.880E+03
W/T/L	-/-/-		4/4/22		19/10/0		27/0/3		21/1/8	
	WOA		CAWOA		HIWOA		NGSWOA		SBWOA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	2.280E+06	1.670E+06	1.270E+09	3.430E+09	2.930E+10	4.940E+09	1.110E+09	1.370E+09	2.890E+06	1.590E+06
F2	5.830E+22	3.160E+23	3.000E+41	2.110E+42	7.140E+38	3.690E+39	6.250E+30	4.460E+31	1.350E+22	5.460E+22
F3	1.500E+05	5.610E+04	9.700E+04	5.970E+04	5.650E+04	4.970E+03	1.150E+05	4.500E+04	1.700E+05	6.860E+04
F4	5.320E+02	3.470E+01	7.420E+02	5.030E+02	4.880E+03	6.300E+02	6.430E+02	1.760E+02	5.500E+02	4.100E+01
F5	7.760E+02	4.330E+01	8.340E+02	5.720E+01	8.260E+02	3.300E+01	7.740E+02	6.430E+01	7.710E+02	5.160E+01
F6	6.670E+02	1.190E+01	6.730E+02	9.700E+00	6.710E+02	7.480E+00	6.680E+02	9.540E+00	6.660E+02	1.100E+01
F7	1.220E+03	1.030E+02	1.700E+03	2.340E+02	1.300E+03	4.690E+01	1.210E+03	8.590E+01	1.210E+03	9.360E+01
F8	1.000E+03	5.480E+01	1.070E+03	5.340E+01	1.030E+03	3.700E+01	1.010E+03	5.380E+01	9.930E+02	4.800E+01
F9	7.500E+03	2.320E+03	8.630E+03	2.310E+03	7.810E+03	1.020E+03	8.530E+03	2.620E+03	8.310E+03	2.830E+03
F10	6.070E+03	8.180E+02	6.590E+03	1.150E+03	6.860E+03	8.000E+02	5.770E+03	7.910E+02	5.930E+03	8.050E+02
F11	1.500E+03	1.620E+02	2.410E+03	3.090E+03	4.500E+03	5.180E+02	2.240E+03	1.080E+03	1.490E+03	1.290E+02
F12	4.200E+07	2.750E+07	2.710E+08	8.540E+08	5.960E+09	2.140E+09	6.710E+07	1.040E+08	3.750E+07	2.820E+07
F13	1.510E+05	9.320E+04	2.650E+08	9.110E+08	1.590E+09	6.950E+08	5.640E+06	3.910E+07	1.470E+05	8.460E+04
F14	6.730E+05	6.910E+05	9.340E+05	1.590E+06	2.600E+06	1.270E+06	6.860E+05	8.620E+05	6.180E+05	7.130E+05
F15	8.520E+04	6.210E+04	1.810E+04	1.440E+04	1.990E+08	1.210E+08	3.380E+06	1.070E+07	8.000E+04	4.210E+04
F16	3.530E+03	4.610E+02	3.380E+03	5.440E+02	4.310E+03	4.160E+02	3.440E+03	4.730E+02	3.560E+03	3.890E+02
F17	2.520E+03	2.340E+02	2.920E+03	3.870E+02	3.300E+03	4.480E+02	2.450E+03	2.190E+02	2.530E+03	2.930E+02
F18	2.960E+06	3.170E+06	1.150E+06	1.730E+06	3.590E+06	3.790E+06	4.070E+06	5.120E+06	2.590E+06	2.840E+06
F19	2.490E+06	2.400E+06	3.110E+05	1.100E+06	8.830E+07	5.060E+07	1.570E+06	1.920E+06	2.690E+06	2.410E+06
F20	2.730E+03	2.000E+02	2.930E+03	2.000E+02	2.760E+03	1.670E+02	2.710E+03	1.900E+02	2.720E+03	1.540E+02
F21	2.570E+03	5.330E+01	2.580E+03	5.130E+01	2.620E+03	3.240E+01	2.570E+03	5.790E+01	2.550E+03	5.910E+01
F22	6.240E+03	2.240E+03	7.460E+03	1.850E+03	6.410E+03	1.550E+03	6.960E+03	1.690E+03	6.340E+03	2.250E+03
F23	3.050E+03	1.120E+02	3.220E+03	1.190E+02	3.150E+03	9.630E+01	3.040E+03	1.060E+02	3.040E+03	1.120E+02
F24	3.170E+03	9.760E+01	3.330E+03	1.160E+02	3.260E+03	8.790E+01	3.160E+03	9.520E+01	3.150E+03	7.580E+01
F25	2.940E+03	3.150E+01	3.020E+03	9.090E+01	3.530E+03	7.180E+01	2.990E+03	4.840E+01	2.950E+03	2.820E+01
F26	7.440E+03	1.230E+03	8.460E+03	9.830E+02	8.620E+03	5.180E+02	7.410E+03	1.200E+03	7.590E+03	1.200E+03
F27	3.360E+03	8.800E+01	3.590E+03	2.120E+02	3.610E+03	1.630E+02	3.370E+03	9.270E+01	3.380E+03	9.360E+01
F28	3.300E+03	4.330E+01	3.530E+03	5.360E+02	5.050E+03	1.240E+02	3.400E+03	6.540E+01	3.320E+03	1.040E+02
F29	4.750E+03	3.910E+02	5.700E+03	7.390E+02	5.010E+03	3.920E+02	4.730E+03	4.260E+02	4.850E+03	4.930E+02
F30	9.630E+06	8.070E+06	7.290E+05	2.060E+06	6.960E+08	4.890E+08	7.030E+06	4.420E+06	1.060E+07	6.980E+06
W/T/L	29/0/1		29/0/1		29/0/1		29/0/1		29/0/1	

Table 5. Experiment results on IEEE CEC 2017 on 50 dimensions.

	SSWOA		IFDE		SS		SMS		BBO	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	1.000E+02	1.670E+06	1.000E+02	5.970E-15	1.000E+02	1.810E-04	2.890E+08	1.910E+08	3.150E+06	6.530E+05
F2	2.000E+02	3.160E+23	2.000E+02	2.980E-14	2.000E+02	8.090E-08	6.920E+52	4.570E+53	3.510E+12	1.520E+13
F3	3.000E+02	5.610E+04	3.000E+02	5.570E-14	3.000E+02	2.890E-01	1.090E+05	1.600E+04	4.780E+02	3.110E+01
F4	4.080E+02	3.470E+01	4.340E+02	4.490E+01	4.530E+02	6.600E+01	1.010E+03	2.290E+02	5.430E+02	5.710E+01
F5	8.160E+02	4.330E+01	5.260E+02	5.010E+00	8.350E+02	1.220E+01	7.920E+02	5.520E+01	6.010E+02	2.350E+01
F6	6.000E+02	1.190E+01	6.000E+02	4.390E-03	6.000E+02	5.760E-02	6.590E+02	5.890E+00	6.010E+02	5.130E-01
F7	1.060E+03	1.030E+02	7.770E+02	3.400E+00	1.080E+03	1.110E+01	1.400E+03	9.070E+01	8.630E+02	1.910E+01
F8	1.110E+03	5.480E+01	8.270E+02	4.270E+00	1.130E+03	1.200E+01	1.090E+03	5.840E+01	9.050E+02	2.240E+01
F9	9.010E+02	2.320E+03	9.000E+02	5.960E-14	9.020E+02	1.860E+00	1.360E+04	3.400E+03	1.190E+03	3.100E+02
F10	1.350E+04	8.180E+02	4.930E+03	3.680E+02	1.390E+04	3.690E+02	9.870E+03	1.050E+03	6.760E+03	1.020E+03
F11	1.150E+03	1.620E+02	1.160E+03	9.940E+00	1.200E+03	5.360E+01	2.290E+03	2.590E+02	1.320E+03	5.860E+01
F12	9.750E+04	2.750E+07	3.340E+03	5.440E+02	3.240E+05	2.360E+05	4.130E+08	2.290E+08	2.780E+06	9.970E+05
F13	3.040E+03	9.320E+04	1.360E+03	2.610E+01	4.260E+03	1.220E+03	2.170E+06	1.740E+06	3.090E+03	1.320E+03
F14	1.530E+03	6.910E+05	1.430E+03	4.850E+00	1.540E+03	9.410E+00	1.180E+06	1.010E+06	7.530E+04	5.830E+04
F15	1.770E+03	6.210E+04	1.550E+03	1.610E+01	1.770E+03	5.860E+01	1.460E+05	3.350E+05	5.530E+03	3.510E+03
F16	3.900E+03	4.610E+02	2.080E+03	9.220E+01	3.990E+03	1.950E+02	4.550E+03	5.850E+02	3.140E+03	4.760E+02
F17	3.300E+03	2.340E+02	2.090E+03	8.390E+01	3.260E+03	1.550E+02	3.630E+03	3.580E+02	2.920E+03	3.050E+02
F18	1.910E+03	3.170E+06	1.850E+03	1.780E+01	1.970E+03	4.410E+01	6.140E+06	4.320E+06	6.340E+05	4.530E+05
F19	1.970E+03	2.400E+06	1.940E+03	1.300E+01	1.980E+03	1.050E+01	4.990E+06	4.850E+06	1.810E+04	6.880E+03
F20	3.260E+03	2.000E+02	2.320E+03	1.050E+02	3.390E+03	1.730E+02	3.250E+03	2.690E+02	3.130E+03	2.790E+02
F21	2.630E+03	5.330E+01	2.330E+03	3.750E+00	2.630E+03	1.350E+01	2.630E+03	5.940E+01	2.400E+03	2.180E+01
F22	2.540E+03	2.240E+03	2.610E+03	9.600E+02	4.580E+03	4.980E+03	1.110E+04	2.530E+03	8.560E+03	8.120E+02
F23	3.020E+03	1.120E+02	2.730E+03	7.480E+00	3.030E+03	5.750E+01	3.470E+03	1.270E+02	2.890E+03	3.080E+01
F24	2.910E+03	9.760E+01	2.900E+03	6.010E+00	3.000E+03	1.270E+02	3.720E+03	1.620E+02	3.050E+03	3.710E+01
F25	3.010E+03	3.150E+01	3.030E+03	3.320E+01	3.020E+03	3.690E+01	3.330E+03	9.560E+01	3.070E+03	2.810E+01
F26	3.830E+03	1.230E+03	3.500E+03	4.890E+02	4.560E+03	1.110E+03	1.030E+04	1.050E+03	5.430E+03	3.850E+02
F27	3.230E+03	8.800E+01	3.240E+03	9.240E+00	3.230E+03	1.350E+01	4.560E+03	3.110E+02	3.620E+03	8.190E+01
F28	3.280E+03	4.330E+01	3.280E+03	2.370E+01	3.280E+03	2.590E+01	3.820E+03	2.070E+02	3.320E+03	1.790E+01
F29	4.020E+03	3.910E+02	3.280E+03	1.290E+01	4.020E+03	1.630E+02	6.610E+03	7.960E+02	3.990E+03	2.650E+02
F30	9.580E+05	8.070E+06	6.110E+05	3.020E+04	1.130E+06	1.580E+05	1.920E+08	6.670E+07	9.130E+05	1.180E+05
W/T/L	-/-/-		4/0/26		17/10/3		24/2/4		17/2/11	
	WOA		CAWOA		HIWOA		NGSWOA		SBWOA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	6.970E+06	6.390E+06	4.160E+09	5.340E+09	5.060E+10	5.690E+09	3.770E+09	2.750E+09	9.140E+06	6.190E+06
F2	5.640E+54	3.890E+55	2.120E+74	1.520E+75	8.310E+79	5.330E+80	1.710E+61	1.220E+62	2.130E+55	1.480E+56
F3	7.080E+04	3.140E+04	1.430E+05	5.810E+04	1.460E+05	1.180E+04	9.610E+04	2.730E+04	8.990E+04	3.980E+04
F4	6.710E+02	5.830E+01	1.310E+03	6.820E+02	1.440E+04	2.860E+03	1.140E+03	5.530E+02	6.840E+02	6.020E+01
F5	9.260E+02	9.860E+01	1.090E+03	9.790E+01	9.980E+02	5.900E+01	9.580E+02	8.230E+01	9.070E+02	6.600E+01
F6	6.750E+02	1.030E+01	6.810E+02	8.930E+00	6.810E+02	6.320E+00	6.770E+02	1.060E+01	6.760E+02	7.820E+00
F7	1.680E+03	1.260E+02	2.770E+03	4.230E+02	1.800E+03	6.010E+01	1.720E+03	1.090E+02	1.690E+03	1.040E+02
F8	1.200E+03	8.440E+01	1.360E+03	9.610E+01	1.240E+03	3.940E+01	1.240E+03	9.350E+01	1.210E+03	9.330E+01
F9	2.080E+04	6.290E+03	2.320E+04	4.350E+03	2.620E+04	5.030E+03	1.950E+04	4.540E+03	2.030E+04	6.370E+03
F10	9.820E+03	1.080E+03	1.050E+04	1.630E+03	1.100E+04	1.420E+03	9.580E+03	1.220E+03	9.900E+03	1.440E+03
F11	1.590E+03	1.280E+02	2.250E+03	1.160E+03	1.050E+04	1.350E+03	2.970E+03	1.410E+03	1.600E+03	1.070E+02
F12	1.930E+08	8.900E+07	2.630E+09	3.500E+09	2.850E+10	8.880E+09	5.140E+08	6.300E+08	1.990E+08	1.010E+08
F13	2.080E+05	1.120E+05	1.030E+09	2.720E+09	9.770E+09	5.780E+09	4.330E+07	7.970E+07	1.930E+05	1.570E+05
F14	8.500E+05	7.030E+05	2.050E+06	3.680E+06	1.770E+07	8.570E+06	1.010E+06	1.140E+06	6.240E+05	4.910E+05
F15	7.190E+04	4.750E+04	6.410E+07	4.020E+08	3.200E+09	9.510E+08	1.360E+07	5.940E+07	8.440E+04	6.080E+04
F16	4.780E+03	6.170E+02	4.770E+03	1.020E+03	5.870E+03	6.290E+02	4.720E+03	6.570E+02	4.880E+03	6.070E+02
F17	3.990E+03	4.900E+02	4.870E+03	6.830E+02	5.350E+03	5.920E+02	3.950E+03	4.000E+02	4.050E+03	3.600E+02
F18	6.200E+06	4.600E+06	4.160E+06	4.650E+06	1.000E+08	5.600E+07	6.640E+06	5.010E+06	5.270E+06	4.090E+06
F19	2.450E+06	1.590E+06	7.030E+07	3.380E+08	6.840E+08	4.940E+08	3.670E+06	1.380E+07	2.790E+06	2.020E+06
F20	3.650E+03	3.150E+02	3.990E+03	3.140E+02	3.460E+03	2.650E+02	3.650E+03	3.430E+02	3.640E+03	3.350E+02
F21	2.860E+03	9.980E+01	2.920E+03	1.060E+02	2.960E+03	6.630E+01	2.860E+03	1.010E+02	2.870E+03	1.160E+02
F22	1.160E+04	1.330E+03	1.240E+04	1.890E+03	1.270E+04	1.100E+03	1.150E+04	1.030E+03	1.190E+04	1.200E+03
F23	3.590E+03	1.920E+02	3.950E+03	2.380E+02	3.800E+03	1.040E+02	3.540E+03	1.540E+02	3.630E+03	1.710E+02
F24	3.690E+03	1.670E+02	4.040E+03	2.050E+02	3.840E+03	1.660E+02	3.660E+03	1.390E+02	3.720E+03	1.610E+02
F25	3.140E+03	3.370E+01	3.640E+03	5.270E+02	6.700E+03	4.480E+02	3.470E+03	1.970E+02	3.140E+03	3.760E+01
F26	1.330E+04	1.860E+03	1.520E+04	2.460E+03	1.370E+04	5.490E+02	1.250E+04	1.560E+03	1.260E+04	1.660E+03
F27	4.260E+03	4.060E+02	5.090E+03	6.690E+02	5.590E+03	5.960E+02	4.320E+03	4.150E+02	4.220E+03	4.810E+02
F28	3.440E+03	6.640E+01	4.790E+03	1.180E+03	6.810E+03	8.960E+02	4.120E+03	3.470E+02	3.420E+03	5.460E+01
F29	6.910E+03	6.690E+02	8.370E+03	1.600E+03	1.600E+04	4.700E+03	6.620E+03	7.270E+02	7.110E+03	6.790E+02
F30	8.090E+07	2.370E+07	6.070E+06	7.830E+06	2.000E+09	6.460E+08	7.290E+07	2.860E+07	9.370E+07	3.530E+07
W/T/L	28/0/2		28/0/2		29/0/1		28/0/2		28/0/2	

Table 6. Experiment results on IEEE CEC 2017 on 100 dimensions.

	SSWOA		IFDE		SS		SMS		BBO	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	4.890E+03	5.380E+03	1.000E+02	0.000E+00	6.570E+03	6.330E+03	2.130E+10	3.780E+09	8.880E+07	8.550E+06
F2	2.000E+02	1.720E-03	2.100E+02	3.600E+01	2.000E+02	2.840E-05	2.780E+140	1.410E+141	1.240E+44	6.690E+44
F3	1.560E+05	1.840E+04	3.000E+02	0.000E+00	1.550E+05	1.870E+04	3.510E+05	3.910E+04	2.410E+03	6.260E+02
F4	4.320E+02	4.540E+01	4.580E+02	6.730E+01	4.960E+02	8.830E+01	4.290E+03	8.590E+02	7.020E+02	4.360E+01
F5	1.280E+03	1.770E+01	5.880E+02	1.490E+01	1.300E+03	2.250E+01	1.440E+03	1.130E+02	8.080E+02	4.260E+01
F6	6.020E+02	7.470E-01	6.000E+02	3.780E-02	6.020E+02	7.320E-01	6.760E+02	5.840E+00	6.040E+02	1.220E+00
F7	1.620E+03	1.920E+01	8.970E+02	8.750E+00	1.640E+03	2.630E+01	2.950E+03	2.010E+02	1.400E+03	3.930E+01
F8	1.590E+03	1.500E+01	8.850E+02	1.330E+01	1.600E+03	2.380E+01	1.810E+03	1.110E+02	1.110E+03	4.340E+01
F9	9.760E+02	3.020E+01	9.010E+02	6.090E-01	1.030E+03	6.550E+01	4.180E+04	6.370E+03	6.300E+03	1.450E+03
F10	1.760E+03	1.310E+02	1.350E+04	4.940E+02	3.040E+04	4.150E+02	2.600E+04	1.990E+03	1.520E+04	1.260E+03
F11	7.400E+05	1.260E+06	1.610E+03	6.680E+01	1.730E+03	1.090E+02	8.080E+04	1.270E+04	2.520E+03	2.010E+02
F12	3.560E+04	8.650E+03	2.410E+04	7.790E+03	9.960E+05	1.560E+06	5.650E+09	1.970E+09	2.960E+07	7.740E+06
F13	2.890E+04	5.570E+03	2.140E+03	5.950E+02	3.370E+04	7.670E+03	6.090E+07	4.270E+07	7.550E+03	2.070E+03
F14	2.860E+03	3.290E+02	1.650E+03	2.410E+01	3.840E+03	2.070E+03	8.910E+06	3.370E+06	8.450E+05	4.090E+05
F15	2.240E+04	3.470E+03	1.740E+03	3.840E+01	2.680E+04	7.030E+03	1.220E+07	5.040E+07	2.740E+03	1.270E+03
F16	7.890E+03	8.460E+02	3.400E+03	3.240E+02	8.410E+03	4.550E+02	9.760E+03	1.340E+03	5.160E+03	6.630E+02
F17	5.910E+03	2.740E+02	3.180E+03	2.090E+02	6.130E+03	2.930E+02	6.980E+03	8.980E+02	4.470E+03	5.120E+02
F18	4.910E+04	1.950E+04	2.000E+03	4.260E+01	9.080E+04	3.490E+04	1.060E+07	4.980E+06	1.830E+06	8.420E+05
F19	2.780E+04	9.170E+03	2.080E+03	2.100E+01	4.240E+04	2.510E+04	2.480E+07	1.980E+07	3.120E+03	1.350E+03
F20	6.330E+03	2.170E+02	4.040E+03	2.090E+02	6.520E+03	2.530E+02	5.640E+03	6.120E+02	5.170E+03	3.770E+02
F21	3.100E+03	2.190E+01	2.390E+03	1.360E+01	3.130E+03	2.220E+01	3.680E+03	1.250E+02	2.650E+03	5.200E+01
F22	3.070E+04	5.820E+03	1.490E+04	2.440E+03	3.110E+04	5.900E+03	2.930E+04	1.750E+03	1.790E+04	1.270E+03
F23	2.950E+03	1.930E+01	2.890E+03	1.300E+01	2.990E+03	8.820E+01	5.090E+03	2.110E+02	3.280E+03	4.860E+01
F24	3.390E+03	2.590E+01	3.280E+03	1.200E+01	3.420E+03	2.600E+01	7.570E+03	5.820E+02	3.890E+03	8.220E+01
F25	3.190E+03	3.790E+01	3.240E+03	4.630E+01	3.240E+03	6.040E+01	5.310E+03	3.960E+02	3.340E+03	5.090E+01
F26	6.820E+03	2.280E+02	6.090E+03	8.240E+02	7.150E+03	3.100E+02	3.140E+04	2.800E+03	1.160E+04	9.240E+02
F27	3.370E+03	1.990E+01	3.350E+03	2.100E+01	3.400E+03	3.820E+01	7.330E+03	7.850E+02	3.860E+03	9.910E+01
F28	3.320E+03	2.820E+01	3.330E+03	3.050E+01	3.340E+03	5.980E+01	7.690E+03	9.880E+02	3.450E+03	3.420E+01
F29	6.060E+03	1.080E+03	4.240E+03	2.070E+02	6.850E+03	8.370E+02	1.430E+04	1.800E+03	6.670E+03	4.430E+02
F30	1.660E+06	7.930E+05	5.520E+03	1.830E+02	1.480E+06	6.080E+05	8.600E+08	3.900E+08	6.510E+04	2.010E+04
W/T/L	-/-/-		4/0/26		25/4/1		27/0/3		16/0/14	
	WOA		CAWOA		HIWOA		NGSWOA		SBWOA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	3.870E+07	1.730E+07	2.900E+10	1.140E+10	1.580E+11	1.890E+10	2.030E+10	8.880E+09	4.040E+07	1.330E+07
F2	3.200E+132	1.270E+133	6.080E+147	3.050E+148	6.560E+156	2.270E+133	8.690E+141	4.610E+142	2.130E+135	1.320E+136
F3	6.000E+05	1.740E+05	4.700E+05	9.200E+04	2.970E+05	1.770E+04	4.470E+05	1.800E+05	6.180E+05	1.880E+05
F4	9.910E+02	8.560E+01	4.300E+03	1.450E+03	4.580E+04	8.280E+03	2.700E+03	6.830E+02	1.010E+03	7.650E+01
F5	1.450E+03	1.220E+02	1.900E+03	2.000E+02	1.600E+03	5.710E+01	1.490E+03	1.010E+02	1.460E+03	1.150E+02
F6	6.790E+02	8.180E+00	6.900E+02	7.770E+00	6.860E+02	6.690E+00	6.830E+02	8.390E+00	6.800E+02	8.700E+00
F7	3.280E+03	1.580E+02	5.870E+03	9.520E+02	3.530E+03	1.120E+02	3.310E+03	1.770E+02	3.260E+03	1.900E+02
F8	1.870E+03	1.010E+02	2.280E+03	1.980E+02	2.050E+03	7.770E+01	1.970E+03	1.250E+02	1.890E+03	1.150E+02
F9	3.550E+04	7.650E+03	5.060E+04	9.130E+03	4.850E+04	6.610E+03	4.050E+04	1.130E+04	3.990E+04	1.140E+04
F10	1.980E+04	2.670E+03	2.480E+04	3.100E+03	2.350E+04	2.420E+03	2.010E+04	2.120E+03	2.080E+04	2.420E+03
F11	8.560E+03	6.140E+03	7.400E+04	4.590E+04	1.070E+05	1.340E+04	3.970E+04	1.630E+04	8.010E+03	2.650E+03
F12	5.900E+08	2.610E+08	8.610E+09	1.100E+10	8.380E+10	1.610E+10	2.970E+09	2.770E+09	7.340E+08	2.570E+08
F13	2.130E+05	8.380E+05	4.160E+08	9.260E+08	1.750E+10	5.280E+09	1.820E+08	3.950E+08	8.900E+04	3.800E+04
F14	1.900E+06	8.350E+05	6.990E+06	8.900E+06	6.220E+06	1.730E+06	4.360E+06	2.400E+06	1.900E+06	8.480E+05
F15	6.230E+04	2.760E+04	3.730E+08	1.020E+09	7.650E+09	2.940E+09	5.000E+07	2.320E+08	9.570E+04	1.290E+05
F16	1.010E+04	1.490E+03	9.210E+03	2.430E+03	1.610E+04	1.420E+03	9.990E+03	1.560E+03	9.640E+03	1.550E+03
F17	7.030E+03	8.310E+02	2.460E+04	3.340E+04	3.270E+04	4.100E+04	7.970E+03	3.290E+03	6.980E+03	8.450E+02
F18	2.290E+06	9.530E+05	9.090E+06	2.040E+07	1.210E+07	8.980E+06	3.190E+06	1.740E+06	2.130E+06	8.640E+05
F19	1.380E+07	6.270E+06	3.070E+08	1.330E+09	6.730E+09	2.410E+09	4.650E+07	1.560E+08	1.470E+07	7.700E+06
F20	6.380E+03	5.190E+02	6.900E+03	7.970E+02	6.090E+03	5.080E+02	6.210E+03	5.390E+02	6.210E+03	6.190E+02
F21	3.910E+03	2.230E+02	4.130E+03	1.890E+02	4.080E+03	1.550E+02	3.880E+03	2.190E+02	3.900E+03	1.880E+02
F22	2.320E+04	2.310E+03	2.820E+04	3.140E+03	2.690E+04	2.450E+03	2.290E+04	2.420E+03	2.390E+04	2.560E+03
F23	4.740E+03	2.270E+02	5.570E+03	3.220E+02	5.090E+03	2.790E+02	4.740E+03	3.090E+02	4.680E+03	3.290E+02
F24	6.020E+03	4.710E+02	7.810E+03	6.060E+02	6.750E+03	6.390E+02	5.910E+03	4.530E+02	5.960E+03	3.500E+02
F25	3.630E+03	8.140E+01	6.140E+03	1.470E+03	1.310E+04	1.620E+03	4.930E+03	5.090E+02	3.620E+03	6.150E+01
F26	3.070E+04	3.190E+03	3.840E+04	4.520E+03	3.430E+04	1.470E+03	3.030E+04	3.160E+03	3.130E+04	3.380E+03
F27	5.050E+03	6.420E+02	6.950E+03	1.510E+03	6.850E+03	1.150E+03	4.950E+03	5.470E+02	5.040E+03	6.460E+02
F28	3.710E+03	6.380E+01	8.540E+03	2.460E+03	1.030E+04	6.460E+02	5.460E+03	8.260E+02	3.730E+03	6.140E+01
F29	1.390E+04	1.950E+03	2.450E+04	1.580E+04	4.060E+04	1.620E+04	1.310E+04	1.570E+03	1.370E+04	1.880E+03
F30	1.910E+08	8.620E+07	5.650E+08	1.260E+09	1.370E+10	4.390E+09	2.630E+08	3.050E+08	2.160E+08	7.530E+07
W/T/L	27/1/2		28/0/2		27/0/3		27/1/2		27/1/2	

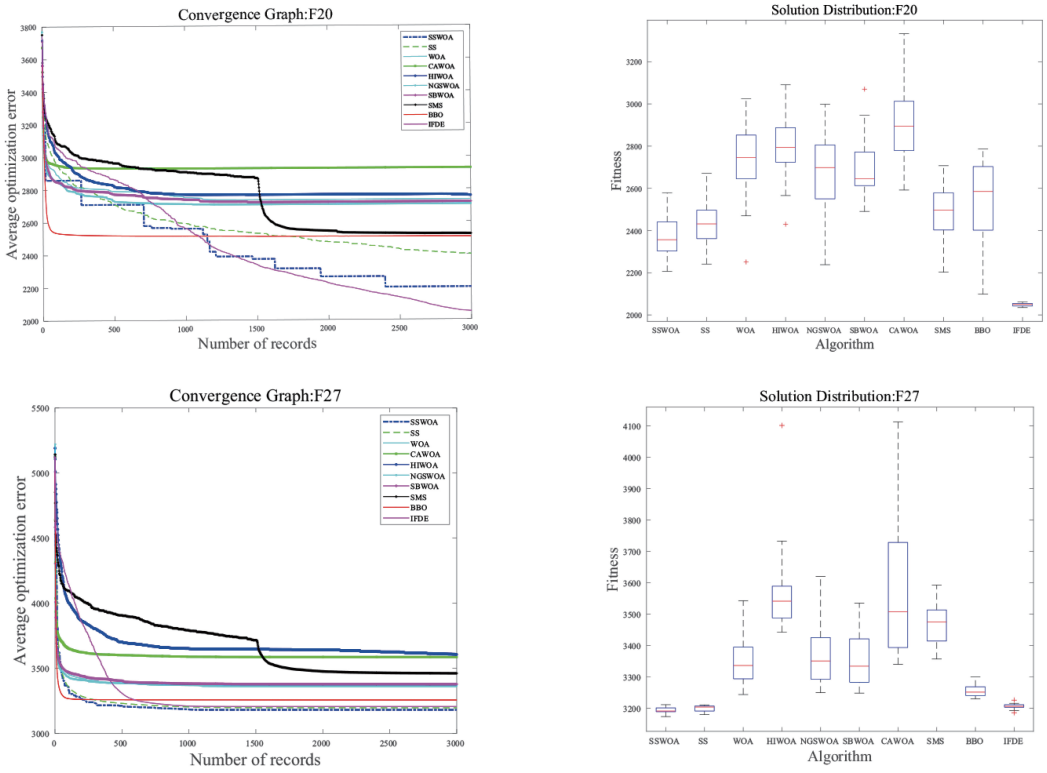


Figure 7. Convergence and Box-and-whisker of D30.

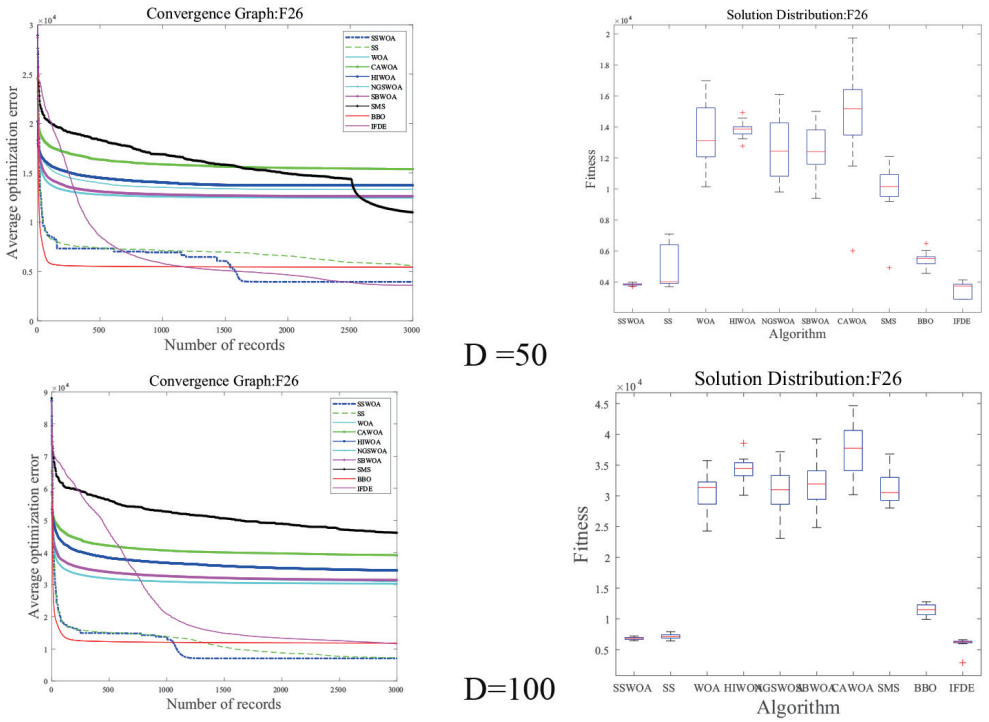


Figure 8. Convergence and Box-and-whisker of D50 and D100.

Table 7. Performance of SSWOA compared with other algorithms in testing the mean and standard deviation of classification accuracy.

	SSWOA		WOA		SASS		NGSWOA		CAWOA		HIWOA		SBWOA	
	test	std	test	std	test	std	test	std	test	std	test	std	test	std
BreastEW	0.9247	0.0103	0.7008	0.1799	0.9114	0.0608	0.6424	0.1730	0.8498	0.0159	0.5903	0.1523	0.7364	0.1609
Australia	0.8488	0.0116	0.7382	0.1225	0.8523	0.0098	0.8124	0.1290	0.6520	0.0455	0.6353	0.0000	0.7425	0.1229
Heart	**0.9036**	0.0106	0.8836	0.1363	0.8991	0.0129	0.8838	0.1277	0.7849	0.1238	0.6582	0.0604	0.8287	0.1559
CongressEW	**0.9690**	0.0096	0.5241	0.1764	0.9631	0.0098	0.5953	0.1379	0.5393	0.1098	0.4227	0.1642	0.4963	0.1705
Exactly	**0.7645**	0.0503	0.4724	0.2041	0.7502	0.0414	0.5930	0.1803	0.4173	0.1575	0.3180	0.1219	0.4568	0.1939
German	0.7667	0.0040	0.6251	0.1492	0.7630	0.0073	0.7111	0.1423	**0.8052**	0.0303	0.4798	0.1130	0.6306	0.1726
KrVsKpEW	**0.7961**	0.0667	0.5252	0.3101	0.7792	0.0683	0.7596	0.2645	0.2084	0.0779	0.3453	0.2462	0.5759	0.3196
Ionosphere	**0.9433**	0.0033	0.5908	0.1139	0.9422	0.0030	0.5990	0.1166	0.5158	0.0847	0.4744	0.0100	0.5661	0.1061
SpectEW	0.8241	0.0016	0.8501	0.0614	0.8234	0.0010	0.8367	0.0534	0.8542	0.0346	**0.8927**	0.0446	0.8529	0.0612
TIC-tac-toe	0.7736	0.0269	0.5310	0.1393	0.7462	0.0249	0.5936	0.1295	0.6662	0.0238	0.3759	0.0888	0.5666	0.1459
Vote	0.9307	0.0176	0.7839	0.1550	0.9361	0.0106	0.8644	0.1308	0.7578	0.0972	0.6372	0.0742	0.7631	0.1590
	IFDE		SMS		BBO		BP		KNN		SVM		RandomForest	
	test	std	test	std	test	std	test	std	test	std	test	std	test	std
BreastEW	0.9291	0.0098	0.8225	0.0537	0.9031	0.0205	0.6353	0.0000	0.9412	0.0000	**0.9706**	0.0000	0.9616	0.0001
Australia	**0.8507**	0.0109	0.7804	0.1046	0.8446	0.0305	0.8525	0.0122	0.8357	0.0000	0.8454	0.8454	**0.8588**	0.0002
Heart	0.8629	0.0598	0.8782	0.0130	0.8850	0.0151	0.8948	0.0145	0.7416	0.0000	0.8427	0.0000	**0.9858**	0.0133
CongressEW	0.9605	0.0129	0.6923	0.0009	0.9628	0.0078	0.8108	0.1639	0.9154	0.0000	0.9538	0.0000	0.9633	0.0001
Exactly	0.7359	0.0401	0.5239	0.1363	0.6926	0.0079	0.7073	0.0423	0.7000	0.0000	0.6925	0.0000	0.7508	0.0005
German	0.7589	0.0275	0.5671	0.1522	0.7456	0.0225	0.4281	0.2165	0.7200	0.0000	0.7633	0.0000	0.7351	0.0004
KrVsKpEW	0.7715	0.0684	0.6766	0.0548	0.7317	0.0621	0.4726	0.0000	0.9452	0.0000	0.9499	0.0000	**0.9791**	0.0000
Ionosphere	0.9411	0.0020	0.6919	0.0716	0.9411	0.0128	0.1788	0.0000	0.9139	0.0000	0.9338	0.0000	0.9402	0.0003
SpectEW	0.8226	0.0049	0.7670	0.0350	0.8139	0.0125	0.8848	0.0669	0.6471	0.0000	0.7326	0.0000	0.7701	0.0008
TIC-tac-toe	**0.7848**	0.0332	0.7232	0.0293	0.7159	0.0326	0.6943	0.0883	0.8198	0.0000	0.6580	0.0000	**0.8982**	0.0003
Vote	0.9255	0.0092	0.9389	0.0088	**0.9419**	0.0056	0.8750	0.1217	0.8917	0.0000	0.9250	0.0000	0.9372	0.0001

Table 8. *p*-values obtained by the Wilcoxon sign-ranked test for SSWOA.

SSWOA vs.	WOA	SASS	NGSWOA	CAWOA	HIWOA
	0.0051	0.0294	0.0039	0.0039	0.0039
SBWOA	IFDE	SMS	BBO	BP	KNN
0.0039	0.0144	0.0039	0.011	0.0144	0.1424

Table 9. Running times of SSWOA and its competitors.

	SSWOA	IFDE	SASS	WOA	CAWOA	HIWOA	NGSWOA
Australia	41.46	42.18	53.36	40.79	38.46	38.64	42.49
BreastEW	34.81	35.30	13.63	35.43	33.23	32.21	36.70
CongressEW	15.77	15.73	16.63	14.91	13.55	14.06	16.77
Exactly	47.15	48.72	47.74	46.00	43.79	44.69	0.68
German	19.39	20.28	19.50	17.98	17.95	18.21	21.65
Heart	22.22	23.27	22.95	39.47	20.66	20.52	20.61
Ionosphere	16.18	16.52	16.57	14.88	13.82	14.47	17.16
KrVsKpEW	56.58	57.81	57.46	54.46	51.88	53.51	59.04
SpectEW	4.23	5.45	4.22	3.81	3.98	3.36	7.33
TIC-tac-toe	25.86	26.62	26.55	24.32	24.14	24.54	27.38
Vote	10.47	10.62	11.73	9.15	9.05	9.18	12.18
Ave.	26.74	27.50	26.39	27.38	24.59	24.85	23.82
	SBWOA	SMS	BBO	BP	KNN	SVM	RandomForest
Australia	42.66	40.85	14.09	357.67	1.55	1.02	1.42
BreastEW	37.53	34.18	16.36	306.28	0.84	0.81	0.83
CongressEW	18.07	16.12	10.83	264.42	0.83	0.78	0.83
Exactly	48.36	47.72	16.38	403.77	0.87	0.83	0.84
German	21.05	19.94	22.19	78.23	0.87	0.85	0.93
Heart	20.87	22.26	9.43	49.23	0.79	0.75	0.81
Ionosphere	17.98	16.20	12.07	152.62	0.71	0.87	0.91
KrVsKpEW	59.66	57.71	62.68	13752.18	0.87	1.04	0.88
SpectEW	6.22	5.19	5.86	154.23	0.87	0.87	0.89
TIC-tac-toe	28.55	26.15	14.25	261.07	0.93	1.05	0.86
Vote	11.95	10.70	8.46	48.75	1.36	1.74	1.52
Ave.	28.45	27.00	17.51	1438.95	0.95	0.97	0.97

5.2. Algorithm training model performance comparison

In order to precisely visualize the performance of SSWOA, we decided to employ two different graphs, namely the box-and-whisker plots graphs (Figures 9 and 10) and the convergence graph, i.e., Figure 11. Figures 9 and 10 depict two typical problems, namely the CongressEW, Exactly. This box-and-line diagram includes two evaluation metrics, MSE and accuracy. The box-and-whisker plot includes the maximum, minimum, first quartile, and median. Figure 10 summarizes the maximum, minimum, first quartile, median, third quartile and extreme values. The distribution of solutions shown in Figure 10 indicates that the difference between the lowest and highest numbers shows how

consistently the algorithm performs. Additionally, the reduced height in terms of MSE suggests that the algorithm might provide a superior answer during training.

SSWOA has the best median and lowest values for the Exactly and CongressEW issues, as can be observed in Figures 9 and 10, which indicates that SSWOA exhibits the optimal effect in training DNM. It is inevitable that some of the evolutionary algorithms may experience problems with excesses in the training process. Nevertheless, SSWOA presents the best classification accuracy for all problems tested compared to its peers.

Figure 11 introduces the convergence plots for all compared algorithms. It can be noticed from the graphs that SSWOA has a reasonable convergence rate compared to other learning algorithms. Therefore, combining Figures 10–11 and the time displayed by Table 9, we can conclude that SSWOA is a hybrid algorithm that combines high classification accuracy, good robustness and high convergence. SSWOA is exactly the DNM's learning system that shows the greatest promise.

In addition, we compared the ROC curves and AUC values of the models trained by both evolutionary and non-evolutionary algorithms, as shown in Figure 12. From the Congress dataset we can observe that the AUC values of SSWOA are slightly larger than those of the evolutionary algorithms IFDE, SASS, which is much better than WOA and its variants. Also, in evolutionary algorithms, the AUC values of SSWOA are also slightly larger than those of non-evolutionary algorithms. This is even more evident in the dataset Exactly, where SSWOA has a larger computed area under the ROC curve than other evolutionary and non-evolutionary algorithms. This is also true for other important metrics. Based on these experimental results, we can easily conclude that the hybrid algorithm has better classification performance than other single algorithms. To a certain extent, SSWOA improves the classification accuracy, precision and has good robustness and high convergence on the basis of one single algorithm.

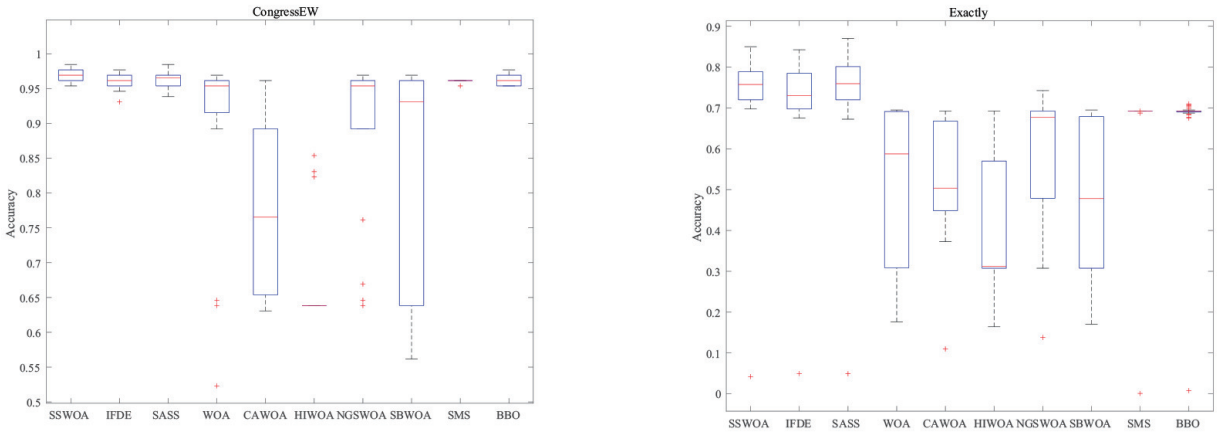


Figure 9. Box-and-whisker plots of the classification effects of SSWOA and its comparative algorithms for the CongressEW and Exactly datasets (Acc).

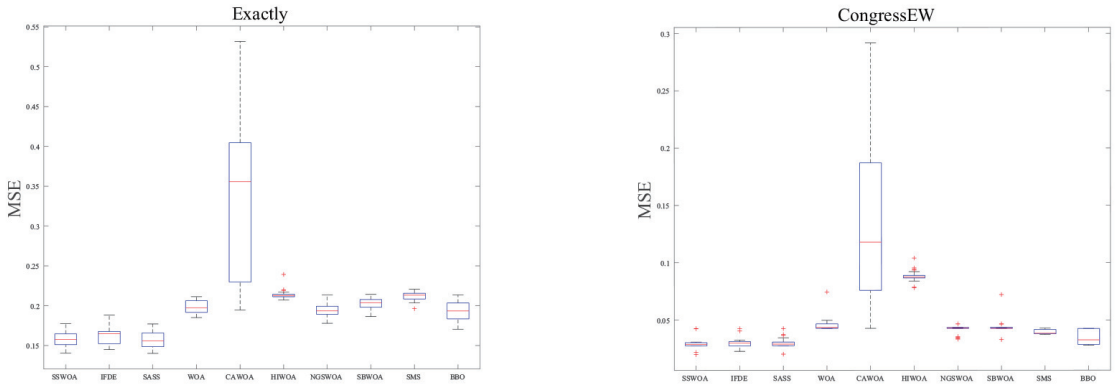


Figure 10. Box-and-whisker plots of the classification effects of SSWOA and its comparative algorithms for the CongressEW and Exactly datasets (MSE).

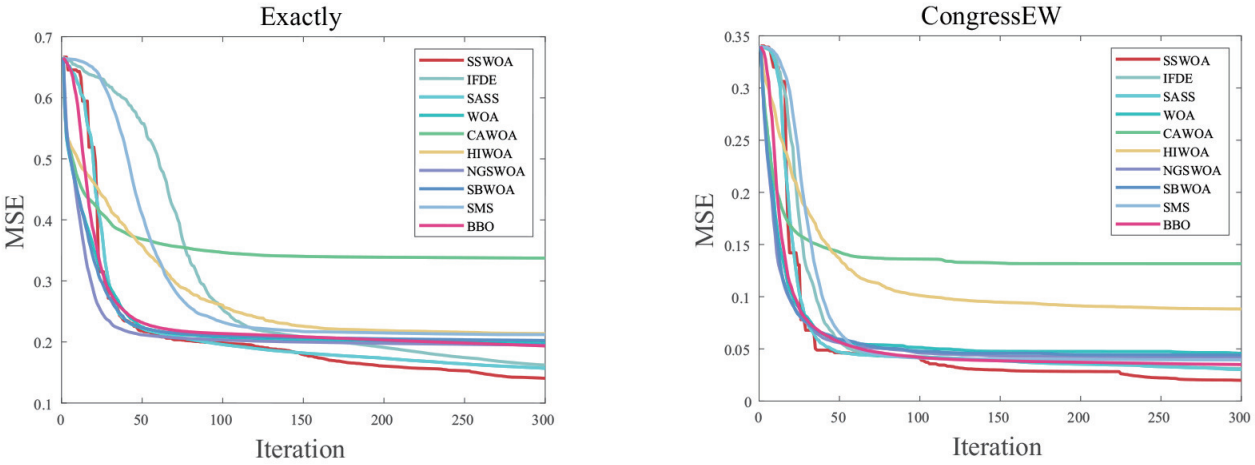


Figure 11. Convergence plots of the classification effects of SSWOA and its comparative algorithms on the CongressEW, Exactly datasets.

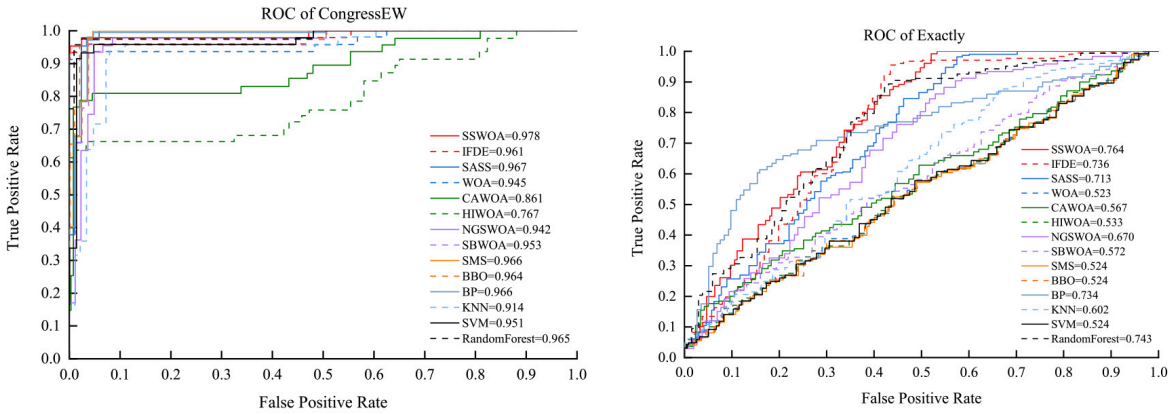


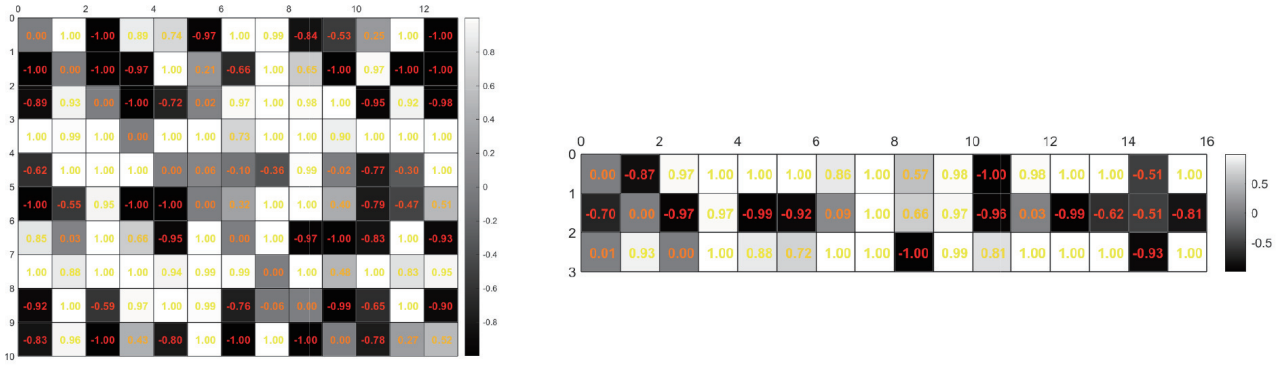
Figure 12. ROC and AUC of CongressEW and Exactly.

5.3. *Morphology and logical circuits realization*

In training the DNM with the evolutionary algorithm, it is important to determine the appropriate values of w_{ij} and q_{ij}. As described in Section 2, w_{ij} represents the state that the synapse is in, i.e., excitation or inhibition, while q_{ij} represents a threshold value regarding the presence or absence of the synapse. Then, in Figure 13, we represent synaptic weights obtained from the DNM model of synapses on both Exactly and CongressEW datasets, suggesting that dendrites impose a strong influence on excitation types and synaptic plasticity.

These state transitions are based on mechanisms previously used in morphology and include here four types of connected states, i.e., direct, inverse, constant 1 and constant 0. Once the pruning operation is complete, this morphological representation can be converted to a logic circuit on the hardware. The ultimate morphology may be acquired after dendritic and synaptic pruning. The precise information is shown in Figure 14. A complex and large unpruned morphology can be seen in the Exactly dataset, whereas only four dendritic layers were retained after pruning and up to seven attributes in the dataset were also removed from the data by the learned DNM, with only $X1$–$X5$, $X7$, $X9$ and $X11$ needing to be involved in information processing. Figure 15 shows the neuromorphology of the DNM on the CongressEW dataset. Attribute $X1$ has been removed, which is functionally equivalent to a feature extraction, and more than half of the attributes have been removed. This drastic simplification produces a short and clear logic circuit.

It is also easy and convenient to implement inside hardware. It is important to note that typical neural networks take a significant amount of design work to build their complete models, whether they are implemented digitally, analogously, or on FPGA-based circuits [83,84], Even silicon photonic chips are required for the hardware implementation of some particular neural networks [85]. Compared to these efforts, the hardware implementation of DNMs is relatively simple, as only logic circuits are utilized, which brings DNMs greater potential and broader prospects in real-world applications.



Exactly CongressEW

Figure 13. Learning result of the synaptic weight on the Exactly and CongressEW datasets.

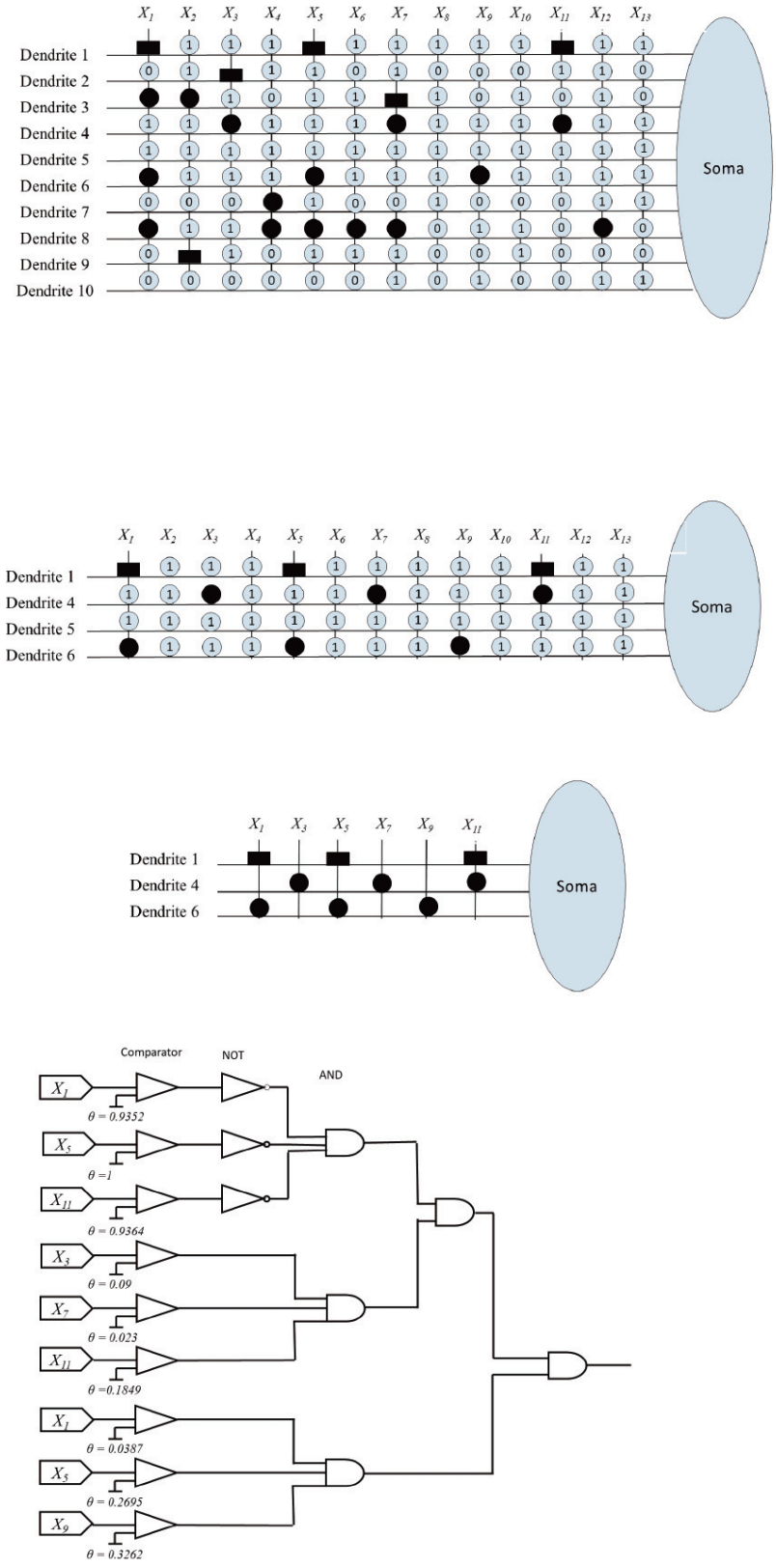


Figure 14. Morphology and logical circuit of DNM on the Exactly dataset.

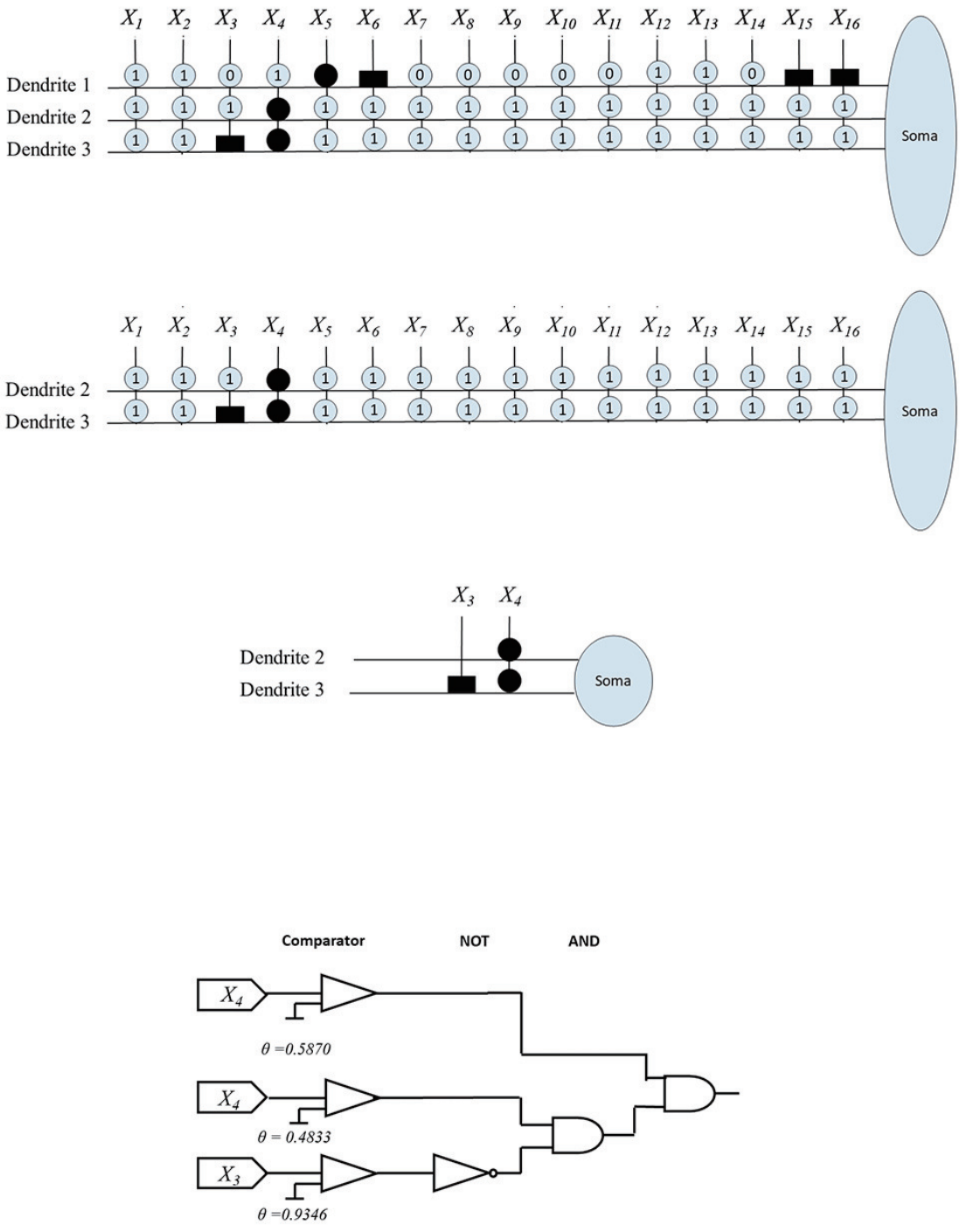


Figure 15. Morphology and logical circuit of DNM on the CongressEW dataset.

6. Conclusions and future works

A hybrid algorithm (SSWOA) was innovatively proposed in the paper to train the single dendritic neuron model. SSWOA was constructed by fusing the benefits of the SS algorithm and the WOA approach. The SS algorithm was employed to roughly determine the direction of the best option, followed by the advancement capabilities of the whale optimization algorithm were used to find the optimal solution, thus improving the capability of the search algorithm. The specific solution is to use SS to exploit exploration in the direction-Rand and WOA to exploit in the other portion towards-best, allowing faster convergence in the best region.

Twelve common classification tasks were used in the experiments, and the findings show that SSWOA can perform much better than BP and ten other non-BP advanced learning methods. The findings also verified that the performance of the hybrid algorithm can be further enhanced when learning single dendritic neuron models.

However, the proposed SSWOA has the following limitations: 1) it is less impressive in terms of time consumption and outcomes compared to other machine learning techniques, 2) the DNM learned by SSWOA still suffers from the curse of dimensionality, thus limiting its applicability on large-scale applications, e.g., image recognition of ImageNet, videos information processing, etc.

The future work of this study include: 1) There is no one global optimization procedure that is the optimal for all optimization problems, according to the renowned "no free lunch" theory. In training DNMs, a variety of heuristic optimization strategies have shown good results, although they still have drawbacks. It is still unclear whether we need to take into account the effectiveness of other heuristic optimization methods, e.g., the farmland fertility algorithm [86–88], African vultures optimization algorithm [89], mountain gazelle optimizer [90], artificial gorilla troops optimizer [91], that co-evolve to ultimately improve the DNM.

2) DNM can identify neuromorphology for a particular task, implying that for a neuron with organized dendrites, it is possible for DNM to determine the amount of dendrites required, the location of the presynaptic axon terminals connecting the dendrites, and the nature of the connections, i.e., excitatory or inhibitory. If we can utilize this local spatial information for visual orientation detection, as Taylor [92] suggests, we will have a better understanding of how a neuron is structured when executing a particular task.

3) Given the spatial dendritic structure, the promising single-neuron model DNM should be able to serve as the foundation for multi-layer deep learning models, enhancing its capacity to resolve challenging real-world problems.

4) More bio-experimental data, such as auditory coincidence detection data, may be obtained with DNM since it can conduct neural morphology detection. The exact rules and methods that biologists and neurologists may utilize as teacher signals while examining neurons, synapses, and dendrites.

Acknowledgments

This research was partially supported by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Nos. 19KJB520059, 19KJB520058, 19KJB520057), and the project of Talent Development of Taizhou University (No. QD2016061).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. C. Lee, H. Hasegawa, S. Gao, Complex-valued neural networks: a comprehensive survey, *IEEE/CAA J. Autom. Sin.*, **9** (2022), 1406–1426. https://doi.org/10.1109/JAS.2022.105743
2. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature*, **521** (2015), 436–444. https://doi.org/10.1038/nature14539
3. Z. Zhang, J. Geiger, J. Pohjalainen, A. E. D. Mousa, W. Jin, B. Schuller, Deep learning for environmentally robust speech recognition: an overview of recent developments, *ACM Trans. Intell. Syst. Technol.*, **9** (2018), 1–28. https://doi.org/10.1145/3178115
4. Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, M. S. Lew, Deep learning for visual understanding: a review, *Neurocomputing*, **187** (2016), 27–48. https://doi.org/10.1016/j.neucom.2015.09.116
5. D. Guo, M. Zhong, H. Ji, Y. Liu, R. Yang, A hybrid feature model and deep learning based fault diagnosis for unmanned aerial vehicle sensors, *Neurocomputing*, **319** (2018), 155–163. https://doi.org/10.1016/j.neucom.2018.08.046
6. J. Cheng, M. Ju, M. Zhou, C. Liu, S. Gao, A. Abusorrah, et al., A dynamic evolution method for autonomous vehicle groups in a highway scene, *IEEE Internet Things J.*, **9** (2021), 1445–1457. https://doi.org/10.1109/JIOT.2021.3086832
7. J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, et al., Highly accurate protein structure prediction with AlphaFold, *Nature*, **596** (2021), 583–589. https://doi.org/10.1038/s41586-021-03819-2
8. I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, et al., Generative adversarial networks, *arXiv preprint*, (2014), arXiv:1406.2661. https://doi.org/10.48550/arXiv.1406.2661
9. B. Zoph, Q. V. Le, Neural architecture search with reinforcement learning, *arXiv preprint*, (2016), arXiv:1611.01578. https://doi.org/10.48550/arXiv.1611.01578
10. S. J. Pan, Q. Yang, A survey on transfer learning, *IEEE Trans. Knowl. Data Eng.*, **22** (2009), 1345–1359, https://doi.org/10.1109/TKDE.2009.191
11. P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, et al., Advances and open problems in federated learning, *Found. Trends Mach. Learn.*, **14** (2021), 1–210. https://doi.org/10.1561/2200000083
12. Y. Zhang, Q. Yang, A survey on multi-task learning, *IEEE Trans. Knowl. Data Eng.*, **34** (2022), 5586–5609. http://doi.org/10.1109/TKDE.2021.3070203
13. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, *Adv. Neural Inf. Process. Syst. (NIPS)*, **30** (2017), 5998–6008.
14. F. Han, J. Jiang, Q. H. Ling, B. Y. Su, A survey on metaheuristic optimization for random single-hidden layer feedforward neural network, *Neurocomputing*, **335** (2019), 261–273. https://doi.org/10.1016/j.neucom.2018.07.080

15. D. Yarotsky, Error bounds for approximations with deep ReLU networks, *Neural Networks*, **94** (2017), 103–114. http://doi.org/10.1016/j.neunet.2017.07.002

16. S. Ruder, An overview of gradient descent optimization algorithms, *arXiv preprint*, (2016), arXiv:1609.04747. https://doi.org/10.48550/arXiv.1609.04747

17. G. E. Hinton, S. Osindero, Y. W. Teh, A fast learning algorithm for deep belief nets, *Neural Comput.*, **18** (2006), 1527–1554. http://doi.org/10.1162/neco.2006.18.7.1527

18. S. Sun, Z. Cao, H. Zhu, J. Zhao, A survey of optimization methods from a machine learning perspective, *IEEE Trans. Cybern.*, **50** (2019), 3668–3681. https://doi.org/10.1109/TCYB.2019.2950779

19. M. A. Ferrag, L. Shu, O. Friha, X. Yang, Cyber security intrusion detection for agriculture 4.0: machine learning-based solutions, datasets, and future directions, *IEEE/CAA J. Autom. Sin.*, **9** (2021), 407–436. https://doi.org/10.1109/JAS.2021.1004344

20. Z. Lu, H. Pu, F. Wang, Z. Hu, L. Wang, The expressive power of neural networks: a view from the width, *Adv. Neural Inf. Process. Syst. (NIPS)*, **30**, (2017).

21. H. P. Beise, S. D. Da Cruz, U. Schröder, On decision regions of narrow deep neural networks, *Neural Networks*, **140** (2021), 121–129. https://doi.org/10.1016/j.neunet.2021.02.024

22. J. He, H. Yang, L. He, L. Zhao, Neural networks based on vectorized neurons, *Neurocomputing*, **465** (2021), 63–70. https://doi.org/10.1016/j.neucom.2021.09.006

23. S. Q. Zhang, W. Gao, Z. H. Zhou, Towards understanding theoretical advantages of complex-reaction networks, *Neural Networks*, **151** (2022), 80–93. https://doi.org/10.1016/j.neunet.2022.03.024

24. S. Ostojic, N. Brunel, From spiking neuron models to linear-nonlinear models, *PLoS Comput. Biol.*, **7** (2011), e1001056. https://doi.org/10.1371/journal.pcbi.1001056

25. T. Zhou, S. Gao, J. Wang, C. Chu, Y. Todo, Z. Tang, Financial time series prediction using a dendritic neuron model, *Knowl. Based Syst.*, **105** (2016), 214–224. https://doi.org/10.1016/j.knosys.2016.05.031

26. T. Zhang, C. Lv, F. Ma, K. Zhao, H. Wang, G. M. O'Hare, A photovoltaic power forecasting model based on dendritic neuron networks with the aid of wavelet transform, *Neurocomputing*, **397** (2020), 438–446. https://doi.org/10.1016/j.neucom.2019.08.105

27. S. Ghosh-Dastidar, H. Adeli, Spiking neural networks, *Int. J. Neural Syst.*, **19** (2009), 295–308. https://doi.org/10.1142/S0129065709002002

28. S. R. Kheradpisheh, T. Masquelier, Temporal backpropagation for spiking neural networks with one spike per neuron, *Int. J. Neural Syst.*, **30** (2020), 2050027. https://doi.org/10.1142/S0129065720500276

29. X. Wang, X. Lin, X. Dang, Supervised learning in spiking neural networks: a review of algorithms and evaluations, *Neural Networks*, **125** (2020), 258–280. https://doi.org/10.1016/j.neunet.2020.02.011

30. L. Deng, Y. Wu, X. Hu, L. Liang, Y. Ding, G. Li, et al., Rethinking the performance comparison between SNNs and ANNs, *Neural Networks*, **121** (2020), 294–307. https://doi.org/10.1016/j.neunet.2019.09.005

31. S. Woźniak, A. Pantazi, T. Bohnstingl, E. Eleftheriou, Deep learning incorporating biologically inspired neural dynamics and in-memory computing, *Nat. Mach. Intell.*, **2** (2020), 325–336. https://doi.org/10.1038/s42256-020-0187-0

32. P. Poirazi, A. Papoutsi, Illuminating dendritic function with computational models, *Nat. Rev. Neurosci.*, **21** (2020), 303–321. https://doi.org/10.1038/s42256-020-0187-0

33. A. Gidon, T. A. Zolnik, P. Fidzinski, F. Bolduan, A. Papoutsi, P. Poirazi, et al., Dendritic action potentials and computation in human layer 2/3 cortical neurons, *Science*, **367** (2020), 83–87. https://doi.org/10.1126/science.aax6239

34. R. L. Wang, Z. Lei, Z. Zhang, S. Gao, Dendritic convolutional neural network, *IEEJ Trans. Electron. Inf. Syst.*, **17** (2022), 302–304. https://doi.org/10.1002/tee.23513

35. J. Li, Z. Liu, R. L. Wang, S. Gao, Dendritic deep residual learning for COVID-19 prediction, *IEEJ Trans. Electron. Inf. Syst.*, **18** (2022) 297–299. https://doi.org/10.1002/tee.23723

36. Z. H. Zhan, L. Shi, K. C. Tan, J. Zhang, A survey on evolutionary computation for complex continuous optimization, *Artif. Intell. Rev.*, **55** (2022), 59–110. https://doi.org/10.1007/s10462-021-10042-y

37. J. Greensmith, U. Aickelin, S. Cayzer, Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection, in *International Conference on Artificial Immune Systems (ICARIS 2005)*, Springer, (2005), 153–167. https://doi.org/10.1007/11536444_12

38. M. London, M. Häusser, Dendritic computation, *Annu. Rev. Neurosci.*, **28** (2005), 503–532. https://doi.org/10.1146/annurev.neuro.28.061604.135703

39. H. Agmon-Snir, C. E. Carr, J. Rinzel, The role of dendrites in auditory coincidence detection, *Nature*, **393** (1998), 268–272. https://doi.org/10.1038/30505

40. T. Euler, P. B. Detwiler, W. Denk, Directionally selective calcium signals in dendrites of starburst amacrine cells, *Nature*, **418** (2002), 845–852. https://doi.org/10.1038/nature00931

41. C. Koch, T. Poggio, V. Torre, Nonlinear interactions in a dendritic tree: localization, timing, and role in information processing, *Proc. Natl. Acad. Sci.*, **80** (1983), 2799–2802. https://doi.org/10.1073/pnas.80.9.2799

42. S. Chavlis, P. Poirazi, Drawing inspiration from biological dendrites to empower artificial neural networks, *Curr. Opin. Neurobiol.*, **70** (2021), 1–10. https://doi.org/10.1016/j.conb.2021.04.007

43. J. Guerguiev, T. P. Lillicrap, B. A. Richards, Towards deep learning with segregated dendrites, *Elife*, **6** (2017), e22901. https://doi.org/10.7554/eLife.22901

44. T. Moldwin, M. Kalmenson, I. Segev, The gradient clusteron: a model neuron that learns to solve classification tasks via dendritic nonlinearities, structural plasticity, and gradient descent, *PLoS Comput. Biol.*, **17** (2021), e1009015. https://doi.org/10.1371/journal.pcbi.1009015

45. C. Koch, I. Segev, The role of single neurons in information processing, *Nat. Neurosci.*, **3** (2000), 1171–1177. https://doi.org/10.1038/81444

46. P. Poirazi, T. Brannon, B. W. Mel, Pyramidal neuron as two-layer neural network, *Neuron*, **37** (2003), 989–999. https://doi.org/10.1016/S0896-6273(03)00149-1

47. R. Legenstein, W. Maass, Branch-specific plasticity enables self-organization of nonlinear computation in single neurons, *J. Neurosci.*, **31** (2011), 10787–10802. https://doi.org/10.1523/JNEUROSCI.5684-10.2011

48. I. S. Jones, K. P. Kording, Might a single neuron solve interesting machine learning problems through successive computations on its dendritic tree, *Neural Comput.*, **33** (2021), 1554–1571. https://doi.org/10.1162/neco_a_01390

49. Y. Todo, H. Tamura, K. Yamashita, Z. Tang, Unsupervised learnable neuron model with nonlinear interaction on dendrites, *Neural Networks*, **60** (2014), 96–103. https://doi.org/10.1016/j.neunet.2014.07.011

50. S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, J. Wang, Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction, *IEEE Trans. Neural Networks Learn. Syst.*, **30** (2019), 601–614. https://doi.org/10.1109/TNNLS.2018.2846646

51. Y. Todo, Z. Tang, H. Todo, J. Ji, K. Yamashita, Neurons with multiplicative interactions of nonlinear synapses, *Int. J. Neural Syst.*, **29** (2019), 1950012. https://doi.org/10.1142/S0129065719500126

52. X. Luo, X. Wen, M. Zhou, A. Abusorrah, L. Huang, Decision-tree-initialized dendritic neuron model for fast and accurate data classification, *IEEE Trans. Neural Networks Learn. Syst.*, **33** (2022), 4173 – 4183. https://doi.org/10.1109/TNNLS.2021.3055991

53. J. Ji, Y. Tang, L. Ma, J. Li, Q. Lin, Z. Tang, et al., Accuracy versus simplification in an approximate logic neural model, *IEEE Trans. Neural Networks Learn. Syst.*, **32** (2020), 5194–5207. https://doi.org/10.1109/TNNLS.2020.3027298

54. S. Gao, M. Zhou, Z. Wang, D. Sugiyama, J. Cheng, J. Wang, et al., Fully complex-valued dendritic neuron model, *IEEE Trans. Neural Networks Learn. Syst.*, 1–14. https://doi.org/10.1109/TNNLS.2021.3105901

55. Y. Tang, J. Ji, S. Gao, H. Dai, Y. Yu, Y. Todo, A pruning neural network model in credit classification analysis, *Comput. Intell. Neurosci.*, **2018** (2018), 9390410. https://doi.org/10.1155/2018/9390410

56. Z. Lei, S. Gao, Z. Zhang, M. Zhou, J. Cheng, MO4: a many-objective evolutionary algorithm for protein structure prediction, *IEEE Trans. Evol. Comput.*, **26** (2022), 417–430. https://doi.org/10.1109/TEVC.2021.3095481

57. J. X. Mi, J. Feng, K. Y. Huang, Designing efficient convolutional neural network structure: a survey, *Neurocomputing*, **489** (2022), 139–156. https://doi.org/10.1016/j.neucom.2021.08.158

58. Z. H. Zhan, J. Y. Li, J. Zhang, Evolutionary deep learning: a survey, *Neurocomputing*, **483** (2022), 42–58. https://doi.org/10.1016/j.neucom.2022.01.099

59. Z. Wang, S. Gao, M. Zhou, S. Sato, J. Cheng, J. Wang, Information-theory-based nondominated sorting ant colony optimization for multiobjective feature selection in classification, *IEEE Trans. Cybern.*, 1–14. https://doi.org/10.1109/TCYB.2022.3185554

60. Y. Yu, Z. Lei, Y. Wang, T. Zhang, C. Peng, S. Gao, Improving dendritic neuron model with dynamic scale-free network-based differential evolution, *IEEE/CAA J. Autom. Sin.*, **9** (2022), 99–110. https://doi.org/10.1109/JAS.2021.1004284

61. Y. Yu, S. Gao, Y. Wang, Y. Todo, Global optimum-based search differential evolution, *IEEE/CAA J. Autom. Sin.*, **6** (2018), 379–394. https://doi.org/10.1109/JAS.2019.1911378

62. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

63. A. Kumar, R. K. Misra, D. Singh, S. Mishra, S. Das, The spherical search algorithm for bound-constrained global optimization problems, *Appl. Soft Comput.*, **85** (2019), 105734. https://doi.org/10.1016/j.asoc.2019.105734

64. K. Wang, Y. Wang, S. Tao, Z. Cai, Z. Lei, S. Gao, Spherical search algorithm with adaptive population control for global continuous optimization problems, *Appl. Soft Comput.*, **132** (2023), 109845. https://doi.org/10.1016/j.asoc.2022.109845

65. N. Takahashi, K. Kitamura, N. Matsuo, M. Mayford, M. Kano, N. Matsuki, et al., Locally synchronized synaptic inputs, *Science*, **335** (2012), 353–356. https://doi.org/10.1126/science.1210362

66. W. Chen, J. Sun, S. Gao, J. J. Cheng, J. Wang, Y. Todo, Using a single dendritic neuron to forecast tourist arrivals to Japan, *IEICE Trans. Inf. Syst.*, **100** (2017), 190–202. https://doi.org/10.1587/transinf.2016EDP7152

67. J. Ji, S. Song, Y. Tang, S. Gao, Z. Tang, Y. Todo, Approximate logic neuron model trained by states of matter search algorithm, *Knowl. Based Syst.*, **163** (2019), 120–130. https://doi.org/10.1016/j.knosys.2018.08.020

68. Z. Wang, S. Gao, J. Wang, H. Yang, Y. Todo, A dendritic neuron model with adaptive synapses trained by differential evolution algorithm, *Comput. Intell. Neurosci.*, **2020** (2020), 2710561. https://doi.org/10.1155/2020/2710561

69. Z. Song, Y. Tang, J. Ji, Y. Todo, Evaluating a dendritic neuron model for wind speed forecasting, *Knowl. Based Syst.*, **201** (2020), 106052. https://doi.org/10.1016/j.knosys.2020.106052

70. R. Tanabe, A. S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in *2014 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, (2014), 1658–1665. https://doi.org/10.1109/CEC.2014.6900380

71. Z. Xu, Z. Wang, J. Li, T. Jin, X. Meng, S. Gao, Dendritic neuron model trained by information feedback-enhanced differential evolution algorithm for classification, *Knowl. Based Syst.*, **233** (2021), 107536. https://doi.org/10.1016/j.knosys.2021.107536

72. R. Jiang, M. Yang, S. Wang, T. Chao, An improved whale optimization algorithm with armed force program and strategic adjustment, *Appl. Math. Modell.*, **81** (2020), 603–623. https://doi.org/10.1016/j.apm.2020.01.002

73. S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, M. Zhou, Chaotic local search-based differential evolution algorithms for optimization, *IEEE Trans. Syst. Man Cybern. Syst.*, **51** (2021), 3954–3967. https://doi.org/10.1109/TSMC.2019.2956121

74. W. Dong, L. Kang, W. Zhang, Opposition-based particle swarm optimization with adaptive mutation strategy, *Soft Comput.*, **21** (2017), 5081–5090. https://doi.org/0.1007/s00500-016-2102-5

75. J. Too, M. Mafarja, S. Mirjalili, Spatial bound whale optimization algorithm: an efficient high-dimensional feature selection approach, *Neural Comput. Appl.*, **33** (2021), 16229–16250. https://doi.org/10.1007/s00521-021-06224-y

76. Y. Gao, C. Qian, Z. Tao, H. Zhou, J. Wu, Y. Yang, Improved whale optimization algorithm via cellular automata, in *2020 IEEE International Conference on Progress in Informatics and Computing (PIC)*, IEEE, (2020), 34–39. https://doi.org/10.1109/PIC50277.2020.9350796

77. J. Zhang, J. S. Wang, Improved whale optimization algorithm based on nonlinear adaptive weight and golden sine operator, *IEEE Access*, **8** (2020), 77013–77048. https://doi.org/10.1109/ACCESS.2020.2989445

78. C. Tang, W. Sun, W. Wu, M. Xue, A hybrid improved whale optimization algorithm, in *2019 IEEE 15th International Conference on Control and Automation (ICCA)*, IEEE, (2019), 362–367. https://doi.org/10.1109/ICCA.2019.8900003

79. U. Škvorc, T. Eftimov, P. Korošec, CEC real-parameter optimization competitions: progress from 2013 to 2018, in *2019 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, (2019), 3126–3133. https://doi.org/10.1109/CEC.2019.8790158

80. T. Jiang, S. Gao, D. Wang, J. Ji, Y. Todo, Z. Tang, A neuron model with synaptic nonlinearities in a dendritic tree for liver disorders, *IEEJ Trans. Electr. Electron. Eng.*, **12** (2017), 105–115. https://doi.org/10.1002/tee.22350

81. J. F. Khaw, B. Lim, L. E. Lim, Optimal design of neural networks using the taguchi method, *Neurocomputing*, **7** (1995), 225–245. https://doi.org/10.1016/0925-2312(94)00013-I

82. J. Carrasco, S. García, M. Rueda, S. Das, F. Herrera, Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: practical guidelines and a critical review, *Swarm Evol. Comput.*, **54** (2020), 100665. https://doi.org/10.1016/j.swevo.2020.100665

83. J. Misra, I. Saha, Artificial neural networks in hardware: a survey of two decades of progress, *Neurocomputing*, **74** (2010), 239–255. https://doi.org/10.1016/j.neucom.2010.03.021

84. S. Mittal, A survey of FPGA-based accelerators for convolutional neural networks, *Neural Comput. Appl.*, **32** (2020), 1109–1139. https://doi.org/10.1007/s00521-018-3761-1

85. H. Zhang, M. Gu, X. Jiang, J. Thompson, H. Cai, S. Paesani, et al., An optical neural chip for implementing complex-valued neural network, *Nat. Commun.*, **12** (2021), 1–11. https://doi.org/10.1038/s41467-020-20719-7

86. H. Shayanfar, F. S. Gharehchopogh, Farmland fertility: a new metaheuristic algorithm for solving continuous optimization problems, *Appl. Soft Comput.*, **71** (2018), 728–746. https://doi.org/10.1016/j.asoc.2018.07.033

87. F. S. Gharehchopogh, B. Farnad, A. Alizadeh, A modified farmland fertility algorithm for solving constrained engineering problems, *Concurrency Comput. Pract. Exper.*, **33** (2021), e6310. https://doi.org/10.1002/cpe.6310

88. A. Hosseinalipour, F. S. Gharehchopogh, M. Masdari, A. Khademi, A novel binary farmland fertility algorithm for feature selection in analysis of the text psychology, *Appl. Intell.*, **51** (2021), 4824–4859. https://doi.org/10.1007/s10489-020-02038-y

89. B. Abdollahzadeh, F. S. Gharehchopogh, S. Mirjalili, African vultures optimization algorithm: a new nature-inspired metaheuristic algorithm for global optimization problems, *Comput. Ind. Eng.*, **158** (2021), 107408. https://doi.org/10.1016/j.cie.2021.107408

90. B. Abdollahzadeh, F. S. Gharehchopogh, N. Khodadadi, S. Mirjalili, Mountain gazelle optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems, *Adv. Eng. Software*, **174** (2022), 103282. https://doi.org/10.1016/j.advengsoft.2022.103282

91. B. Abdollahzadeh, F. Soleimanian Gharehchopogh, S. Mirjalili, Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems, *Int. J. Intell. Syst.*, **36** (2021), 5887–5958. https://doi.org/10.1002/int.22535

92. W. R. Taylor, S. He, W. R. Levick, D. I. Vaney, Dendritic computation of direction selectivity by retinal ganglion cells, *Science*, **289** (2000), 2347–2350. https://doi.org/10.1126/science.289.5488.2347

AIMS Press