Hindawi
Journal of Sensors
Volume 2022, Article ID 9620555, 14 pages
https://doi.org/10.1155/2022/9620555

Research Article

Feature Selection and Training Multilayer Perceptron Neural Networks Using Grasshopper Optimization Algorithm for Design Optimal Classifier of Big Data Sonar

Houman Kosarirad,[1] **Mobin Ghasempour Nejati**,[2] **Abbas Saffari**,[3] **Mohammad Khishe**,[3] **and Mokhtar Mohammadi**[4]

[1]*Durham School of Architectural Engineering and Construction, University of Nebraska-Lincoln, 122 NH, Lincoln, NE 68588, USA*
[2]*Department of Industrial Engineering, Univeristy of Tehran, Iran*
[3]*Department of Electrical Engineering, Imam Khomeini Marine Science University, Nowshahr, Iran*
[4]*Department of Information Technology, College of Engineering and Computer Science, Lebanese French University, Kurdistan Region, Iraq*

Correspondence should be addressed to Abbas Saffari; abbas.saffari@birjand.ac.ir

Received 27 January 2022; Revised 25 September 2022; Accepted 25 October 2022; Published 14 November 2022

Academic Editor: Sheng Du

The complexity and high dimensions of big data sonar, as well as the unavoidable presence of unwanted signals such as noise, clutter, and reverberation in the environment of sonar propagation, have made the classification of big data sonar one of the most interesting and applicable topics for active researchers in this field. This paper proposes the use of the Grasshopper Optimization Algorithm (GOA) to train Multilayer Perceptron Artificial Neural Network (MLP-NN) and also to select optimal features in big data sonar (called GMLP-GOA). GMLP-GOA hybrid classifier first extracts the features of experimental sonar data using MFCC. Then, the most optimal features are selected using GOA. In the last step, MLP-NN trained with GOA is used to classify big data sonar. To evaluate the performance of GMLP-GOA, this classifier is compared with MLP-GOA, MLP-GWO, MLP-PSO, MLP-ACO, and MLP-GSA classifiers in terms of classification rate, convergence rate, local optimization avoidance power, and processing time. The results indicated that GMLP-GOA achieved a classification rate of 98.12% in a processing time of 3.14 s.

1. Introduction

Nowadays, big data analysis and classification are highly valuable [1, 2]. The reason is that as the data increase, the need for more accurate data analysis and classification also increases [3, 4]. The more precise and accurate analysis, the more secure our decision-making will be. Better decisions mean more practicality and less cost. Sonar data is one type of data that is regarded to be part of the big data family [5, 6].

Concerning the complex physical characteristics of sonar purposes, classifying original purposes and avoiding unreal purposes has developed into a critical practical area for active researchers and craftsmen [7, 8]. Due to the complexity and heterogeneity of sound circulation in saltwater, several parameters for categorization and differentiation of sonar purposes should be extracted. As the dimensions of the feature vectors grow, the data dimensions also grow.

There are two distinct ways of categorizing high-dimensional data [9]. The first is to employ the Deterministic Approach [10]. Because this approach is so reliable, it almost always results in the best response; nonetheless, the method encounters difficulties as data dimensions rise, which is followed by an increase in spatial and temporal complexity [11]. Furthermore, this strategy is inapplicable to the data classified as big data [12, 13]. The stochastic method is the second approach [14]. These methodologies yield a near-optimal solution [15]. Additionally, they are less complicated in terms of spatial and temporal dimensions than deterministic methods

[16, 17]. Artificial Neural Networks (ANN) is one of the most effective stochastic methods utilized in the actual world of big data.

Neural networks have the ability to learn [18]. Learning here means, these networks are the basics of all neural networks, which may be parted into two groups of supervision learning [19] and without supervision learning [20, 21]. Most appliances are optimized for Multilayer Artificial Neural Networks, optimized [22] or standard [10, 23]. Backpropagation algorithm is used as a learning method that is considered among the family of supervised learning. The backpropagation algorithm is on an incline basis which has some problems such as gradual convergence [24] and appliance in a limit area [25, 26]. Thus, they are unreliable for functional appliances.

The eventual purpose for the process of learning in neural networks is to acquire the best structure of weighted edges and their bios. Such a way that the least number of errors may occur in network training and test specimens [27, 28]. The reference [29] demonstrates that metaheuristic optimization methods may be substituted with gradient-based learning algorithms, since the stochastic character of these algorithms prevents them from being trapped in a local optimum, increases the convergence rate, and decreases classification errors.

Some of the metaheuristic methods which have been recently used for training neural networks, are genetic algorithm (GA) [30], simulated annealing (SA) [31], biogeography-based optimization (BBO) [32], Magnetic Optimization Algorithm (MOA) [33], Artificial Bee Colony Algorithm (ABC) [34], Gray Wolf Optimizer (GWO) [35], Social Spider Algorithm (SSA) [36, 37], Particle Swarm Optimization and Gravity Search Algorithm (PSOGSA) [7], and so on. GA and SA decrease the possibility of getting stuck in the local optimum, but their low convergence rate. This shortage leads to a weak performance when the need for an immediate process exists. ABC acts properly dealing with small problems and data with low dimensions, but when the problem dimensions increase, the time for training increases greatly as well. MOA has an unsuitable performance and low accuracy, facing nonlinear data. BBO requires lengthy computations. Despite its simplicity and speed of convergence, GWO becomes trapped in the local optimum and so is not ideal for situations with a global optimization. Numerous adjustment parameters and a high level of complexity are SSA's flaws. PSOGSA is formed by a combination of PSO and GSA which leads to an increase in the spatial and temporal complexity.

One of the commonalities between metaheuristic algorithms and other search algorithms is the split of the search region into two phases: exploration and exploitation [38–40]. The first phase occurs concurrently with the algorithm's attempt to examine the most dependable areas of the search region [15, 41]. During the exploration phase, the population is subjected to abrupt alterations in order to properly investigate the whole region of the problem. The exploitation phase happens when the algorithm is converged toward a reliable answer. At this stage, the population is undergoing very small changes.

In most cases, given the random nature of evolutionary algorithms, there is no specified boundary betwixt these 2 phases [18, 42]. In other words, the lack of balance betwixt these two phases causes the algorithm to get stuck in the local optimum. This problem is intensified, dealing with data with high dimensions. By adjusting the displacement behavior betwixt these 2 phases, the probability of getting stuck in the local optimum can be reduced. As proved in the reference [43], GOA can properly recognize the border between exploration and exploitation phases [44, 45]. Thus, the algorithm converges toward more reliable answers.

On the other hand, any system that performs data classification consists of three main parts: data acquisition, feature extraction, and classifier design. The novelty of this article occurred in the feature extraction section. In general, all extracted features are not useful and may contain useless or duplicate information. Feature selection can be seen as the process of identifying useful features and removing useless and repetitive features. The goal of feature selection is to obtain a subset of features that solve problems well with minimal performance degradation. The goal of feature selection is to obtain a subset of features that solve problems well with minimal performance degradation.

This theory is mentioned here: No Free Lunch (NFL) [46, 47]. This proposition demonstrates logically that no metaheuristic method exists that is capable of resolving all optimization problems. In other words, one metaheuristic technique may perform admirably and predictably on one set of issues while failing miserably on another set of problems [48, 49]. NFL stimulates this field of study and contributes to the development of new methodologies and the formulation of new metaheuristic methods on an annual basis [50]. Taking into mind the described theory, the aforementioned issues, and GOA's capacity to cope with big data, this approach may be utilized to train Multilayer Perceptron Neural Networks (MLP-NN) and, subsequently, to classify sonar data.

On the other hand, any system that performs data classification consists of three main parts: data acquisition, feature extraction, and classifier design. The novelty of this article occurred in the feature extraction section. In general, all extracted features are not useful and may contain useless or duplicate information. Feature selection can be seen as the process of identifying useful features and removing useless and repetitive features. The goal of feature selection is to obtain a subset of features that solve problems well with minimal performance degradation. The NFL theorem and the ability of GOA to find the boundary between the two phases of exploration and extraction in the search space is a strong motivation to investigate GOA for the problem of feature selection. Therefore, in this paper, in addition to GOA being used as a neural network training algorithm, GOA is used to select optimal features (GMLP-GOA).

The main contribution of this paper is as follows:

(i) Obtaining and collecting experimental data sets

(ii) Feature extraction using the MFCC method

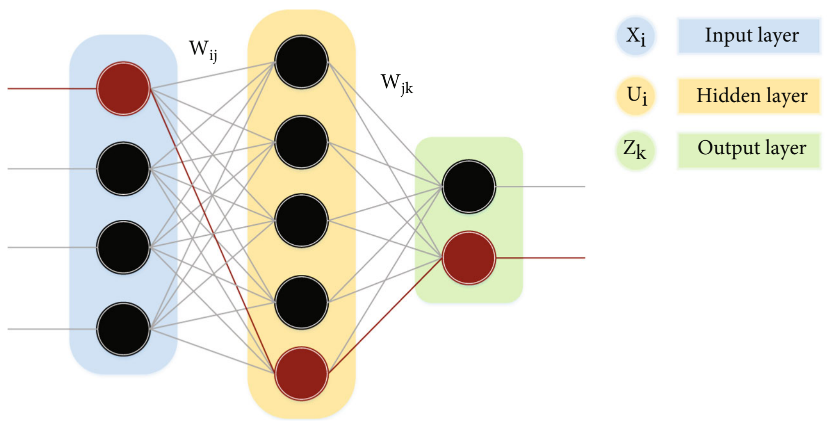


Figure 1: An artificial multilayer neural network.

(iii) Feature selection using GOA

(iv) Designing an optimal GMLP-GOA hybrid classifier and classification of big data sonar

(v) Data classification using MLPs trained with five population-based metaheuristic algorithms

This paper is organized as section two will introduce the MLP-NN. Section 3 explains general issues for GOA. Section 4 will describe how the outcoming GOA as a training algorithm for metaheuristic methods in MLP-NNs is applied. Section 5 will present the dataset and feature selection. Section 6 presents experimental results and discussion. References used are provided in Section 7.

2. Multilayer Perceptron Neural Network

Figure 1 displays an MLP-NN where m, l, and s, respectively, stand for the number of input nodes, hidden nodes, and output nodes [51, 52]. As observed, there is a one-sided junction betwixt the nodes of MLP-NN, which is among the group of neural networks (FNN) [53, 54]. MLP-NN output is computed by

$$U_j = \sum_{i=1}^{m} \left(W_{ij} \bullet X_i \right) - \theta_j . j = 1.2. \cdots .l. \quad (1)$$

In this relation W_{ij} stands for the weight of the edge which connects i-th node (input layer) to j-th node (hidden layer), X_i stands for the input to i-th node (input layer), θ_j stands for the bios of j-th node (hidden layer), and m stands for the number of input nodes. Any hidden node's output is acquired as a relation (2) to a sigmoid function.

$$U_j = \text{sigmoid}\left(U_j\right) = \frac{1}{\left(1 + \exp\left(-s_j\right)\right)} . j = 1.2. \cdots .h, \quad (2)$$

$$Z_k = \sum_{j=1}^{l} \left(W_{jk} \bullet U_j \right) - \theta'_k . k = 1.2. \cdots .s, \quad (3)$$

$$Z_k = \text{sigmoid}(Z_k) = \frac{1}{(1 + \exp(-Z_k))} . k = 1.2. \cdots .s. \quad (4)$$

After calculating the hidden nodes, it is possible to define the final outputs as follows.

In which W_{jk} stands for the weight of the edge which θ'_k stands for the bios of the node k -th and connects the node j −th (hidden layer) to the node k −th (output layer). The most essential factors of an MLP-NN, are the weight for edges and their bios. As seen in the above relations, edges weigh, and bios have defined the ultimate output. Training an MLP-NN, consist of detecting the best optimal output out of certain outputs.

3. Grasshopper Optimization Algorithm

Grasshoppers are an insect species. They are classified as pestilences owing to the harm they do to agricultural crops [55–57]. Although grasshoppers seem to be alone in nature, they are part of one of the biggest animal groups on the planet. They, sometimes, are a threat to farmers. One of their unique features is their social behavior which can be seen both in their childhood and their maturity. Millions of their kids jump and roll-like rollers and eat almost all the plants along the way. Slow movements and short steps are the main features of grasshoppers. Short and sudden movement is one feature of a mature grasshopper community. An important feature of their community is the search for food resources [58]. GOA being inspired by nature, logically, divides the searching process into 2 phases of exploration, and exploitation.

While seeking agents are encouraged to make abrupt moves during the exploration phase, they prefer to make local movements during the exploitation phase. The mathematical model for simulating this grasshopper social behavior is as follows [43]:

$$X_i = A_i + S_i + G_i, \quad (5)$$

where X_i is the location of the i -th grasshopper, A_i denotes the wind, S_i denotes social interaction, and G_i denotes the gravity placed on the i -th grasshopper. To include randomness, the equation is modified as follows:

$$X_i = r_1 S_i + r_2 G_i + r_3 A_i, \tag{6}$$

where r_1, r_2 and r_3 are random numbers between [0,1].

$$S_i = \sum_{\substack{j=1 \\ j\neq i}}^{N} s(d_{ij})\widehat{d_{ij}}, \qquad d_{ij} = |x_j - x_i|, \tag{7}$$

where d_{ij} is the distance between the i-th and j-th grasshoppers and is calculated using the relationship (8). S is a function that is used to define social power. As seen in equation (9) and in the relationship below, $\widehat{d_{ij}}$ is a unit vector extending from the i-th grasshopper to the j-th grasshopper.

$$\widehat{d_{ij}} = \frac{x_j - x_i}{d_{ij}}. \tag{8}$$

The function s shows social power as follows:

$$S(r) = f e^{-r/l} - e^{-r}. \tag{9}$$

f is the intensity of absorption and l is absorption length scale. The function S is not able to impose strong powers between faraway grasshoppers. The G component of the relation (6) is computed as follows:

$$G_i = -g\widehat{e_g}, \tag{10}$$

where g denotes the gravitational constant and $\widehat{e_g}$ denotes a unit vector pointing toward the earth's center. Component A in relation (1) is obtained by

$$A_i = u\widehat{e_w}, \tag{11}$$

where u denotes a constant displacement and $\widehat{e_w}$ denotes a unit vector perpendicular to the wind direction.

Because grasshopper larvae lack wings, their motions are entirely dependent on the direction of the wind. After placing S and G values in the equation (1), this equation can be expanded as

$$X_i = \sum_{\substack{j=1 \\ j\neq i}}^{N} s(|x_j - x_i|)\frac{x_j - x_i}{d_{ij}} - g\widehat{e_g} + u\widehat{e_w}, \tag{12}$$

where the relation (9) and N are equal to the number of grasshoppers. Their children's location on the ground should not be lower than the threshold. We will not, however, utilize this equation to simulate the grasshopper group and the optimization algorithm in order to prevent the algorithm from exploring and exploiting the search space around the solutions. The mathematical model is capable of simulating the grasshopper community in 2- and 3-dimensional as well as multidimensional spaces. However, this mathematical model cannot directly be used for solving optimization problems. The main reason for this is the rapid growth of grasshopper in the area of inertia. As a result, this group cannot converge on a single point. A reformed version of this equation is presented as follows for the purpose of addressing optimization problems:

$$X_i^d = \mathrm{c}\left(\sum_{\substack{j=1 \\ j\neq i}}^{N} c\frac{ub_d - lb_d}{2}\bullet s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right) + \widehat{T_d}, \tag{13}$$

where ub_d is the upper limit on the d dimension, $lb_d$is the purpose value of the d dimention (the best answer so far). Relation (9) and also $\widehat{T_d}$ is a constant decreasing coefficient to reduce the area of inertia, absorption, and desorption. It should be considered that S is almost similar to S in the relation (1). However, we disregard the linear trend and assume that the wind component is always ideal (purpose value).

Equation (13) demonstrates that the position of the grasshopper is determined in terms of its present location, the position of the best solution, and the position of all grasshoppers in the group. It is worth noting that the first component of this equation examines the current location of the grasshopper in relation to the positions of other grasshoppers. To determine the placement of search agents around the purpose, we assessed the state of all grasshopper positions. This is in contrast to the particle swarm algorithm. Each particle in the particle mass algorithm has two vectors: a location vector and a velocity vector.

However, in the grasshopper algorithm, each search agent is represented by a single vector. Another significant distinction between the two methods is that the particle swarm algorithm modifies its location depending on the particle's current position, the particle's best position, and the group's best response. Whereas in the Grasshopper Algorithm, the location of the search agent is modified based on its current position, the best response, and the positions of all the particles in the group. This implies that none of the other groups in the particle swarm algorithm engage in updating a particle's location, but the Grasshopper Algorithm needs all search agents to participate in deciding each agent's next position.

The parameter C is utilized twice in equation (13) for the following reasons. The first C on the left is fairly similar to the particle swarm algorithm's (w) weighted inertia. This setting reduces the grasshoppers' movements in the vicinity of the objective spot. In other words, this parameter optimizes the balance between the exploration (search) and exploitation stages of the input population. The second parameter in the equation decreases grasshopper absorption, inertia, and desorption. Consider the component $c(ub_d - lb_d)/2\bullet s(|x_j^d - x_i^d|)$ in the equation (13), the component $c(ub_d - lb_d)/2$

linearly reduces the space that grasshoppers should explore and exploit. The component $s(x_j - x_i)/d_{ij}$ indicates the grasshopper's absorption to the purpose or the grasshopper's desorption from the optimal location.

Internal C decreases absorption and desorption forces among grasshoppers as the number of repetitions increases, but external C decreases the coverage area around the ideal response as the number of repetitions increases. In summary, the first statement of equation (13) takes into account the total of the positions of the other grasshoppers and applies the grasshoppers' natural interaction. $\widehat{T_d}$ replicates the grasshoppers' hunger for food in the second sentence. Additionally, parameter C replicates the decline in the grasshoppers' acceleration to and intake of the food source. To increase the randomness of the behavior and as a substitute, both phrases of equations (13) might be multiplied by a random value. Single sentences can also be multiplied by random values to model the grasshopper's random behavior in interaction with each other as well as the tendency toward the food source. The mathematical approach offered here is capable of exploring and exploiting the search space. However, a mechanism must exist to transition candidates from the exploration stage to the exploitation stage. Naturally, grasshoppers look for food locally initially, since they lack wings throughout their infancy. They then fly freely across the air, discovering new regions. Unlike this, in stochastic optimization techniques, the exploration phase is conducted first to determine the permissible regions of the search space. Following the discovery of permitted areas, the exploitation phase forces the search agents to locate an accurate approximation of the optimal answer location on a local level.

To balance the two phases of exploration and operation, parameter C must be reduced according to the number of repetitions. This mechanism increases efficiency when the number of repetitions increases. The area of inertia is reduced in proportion to the number of repetitions and is computed as follows:

$$\mathrm{c} = c\max - 1\frac{c\max - c\min}{L}, \tag{14}$$

where $c\max$ is the maximum value, $c\min$ is the minimum value, l is the current repetition count, and L is the maximum repetition count. These parameters were assigned values of 1 and 0.00001 in this study. The appropriate purpose chase by the group is due to the effect of the last sentence of equation (12) that the grasshoppers tend to be attracted to the purpose value. The more interesting pattern is the gradual convergence of the grasshoppers toward the purpose with increasing repetition, which is again due to the decrease in the parameter C. This behavior helps the GOA algorithm not to quickly converge to the optimal answer and thus not get stuck in the local optimal. Therefore, in the latter phases of optimization, the grasshoppers approach as closely as possible to the objective, which is important in the exploitation space.

The preceding discussion demonstrates that the suggested mathematical model motivates grasshoppers to progress toward the goal with increasing repetitions. However, in a true search space, there are no objectives, since it is not quite evident what the best and most significant objective is. As a result, each optimization phase requires us to assign a purpose to each collection of grasshoppers. The Grasshopper Algorithm makes the assumption that the best or purpose value is the most suitable grasshopper (response vector) throughout the optimization process. This will help the algorithm store the most appropriate answer vector in each repetition and in the search space and direct the grasshopper group toward that purpose value. This is done in the goal of discovering a more precise and superior purpose that serves as the best approximation for the overall and true optimization of the search space.

The Grasshopper Algorithm flowchart utilized in the neural network is seen in Figure 2. The GOA method begins by generating a random beginning population. Agents of search revise their positions in light of connections (13). Each iteration has updated the best answer so far. Additionally, factor c is determined using equation (14), and the distance between grasshoppers is normalized to a value between one and four. Updating the grasshopper position has been repeatedly performed to reach the criterion of terminating the algorithm. The position and value of the objective function of the optimal answer, as the best approximation of the overall optimal answer, is finally obtained.

4. Training a Multilayer Neural Network Using the Grasshopper Algorithm

In general, there are three ways for training MLP-NN using evolutionary algorithms. The first is to utilize evolutionary networks to determine the optimal mix of edge weight and node bias in an MLP-NN. The second is the use of evolutionary networks to determine the optimal arrangement of MLP-NNs in a given situation, and the third is the use of evolutionary networks to determine the learning rate and amount of movement of the gradient-based learning algorithm. The Grasshopper Optimization Algorithm is evaluated against an MLP-NN utilizing the 1-th approach in this research. To appropriately represent the weights of edges and nodes in a training procedure for MLP-NN networks, the weights of edges and nodes must be properly represented.

Generally, three methods are used to express the weight of edges and the bias of nodes: vector, matrix, and binary. Each element is represented as a vector, matrix, or string of binary bits in the vector, matrix, and binary methods. Each of these strategies has a number of benefits and downsides that may be advantageous in certain situations. Figure 3 shows how to train a neural network using GOA.

While it is straightforward to convert elements to vectors, matrices, or strings of binary bits using the first technique, the process of retrieving them is more complex. As a result, this technique is often utilized in rudimentary neural networks. In the second technique, it is simpler to recover than it is to encode components in complicated networks. This approach is particularly well-suited for developing algorithms for generic neural networks. The

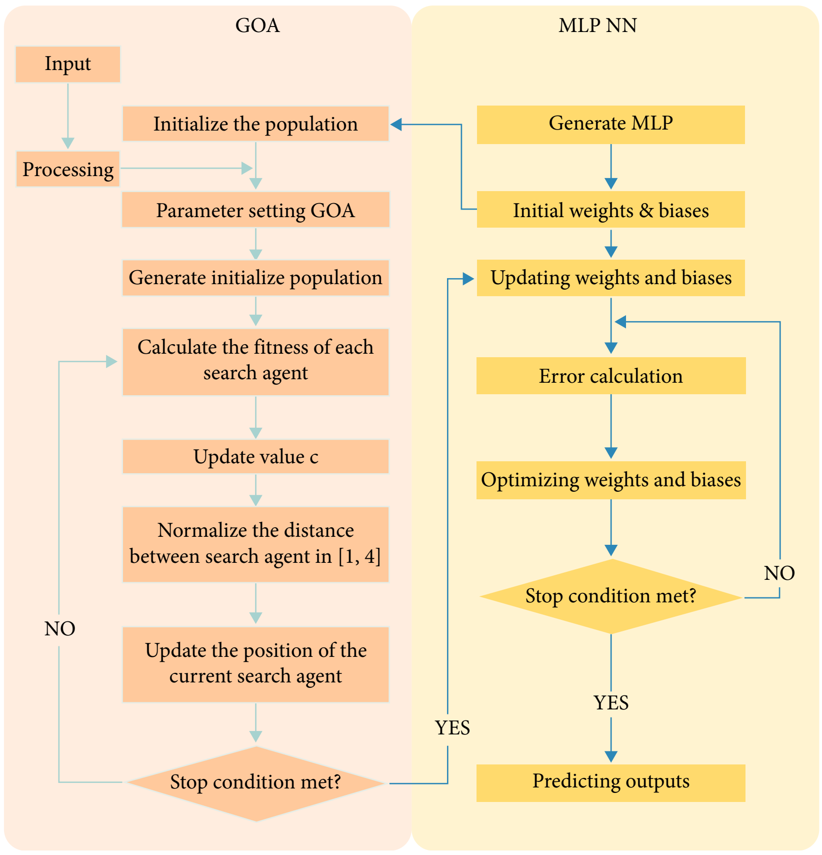

GOA
MLP NN
Input
Initialize the population
Processing
Parameter setting GOA
Generate initialize population
Calculate the fitness of each search agent
Update value c
Normalize the distance between search agent in [1, 4]
Update the position of the current search agent
Stop condition met?
NO
YES
Generate MLP
Initial weights & biases
Updating weights and biases
Error calculation
Optimizing weights and biases
Stop condition met?
NO
YES
Predicting outputs

FIGURE 2: Flowchart related to the Grasshopper Algorithm used in a neural network.

variables must be supplied in binary form for the third technique. When the network structure becomes intricate in this item, the length of each element likewise increases. As a result, the coding and decoding processes will be very difficult.

In this paper, since we do not deal with complex multilayer neural networks, the vector method is used. The MATLAB generic toolbox will not be used to reduce the time of the multilayer neural network operation. As an example of this coding method, the final vector of the multilayer neural network shown in Figure 4 is given in.

$$\text{Position} = [w_{13}\, w_{23} w_{14} w_{24} w_{15} w_{25} w_{36} w_{46} w_{56} \theta_1 \theta_2 \theta_3 \theta_4]. \tag{15}$$

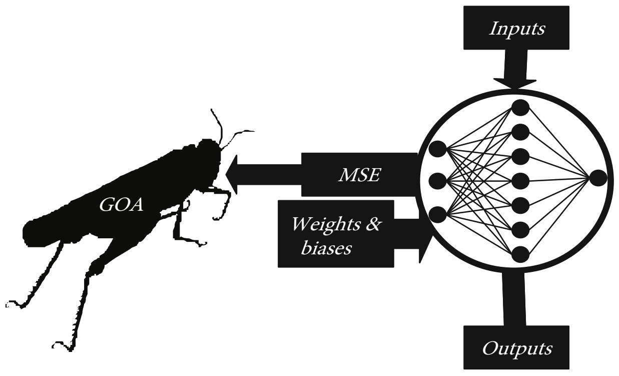


FIGURE 3: How to train a neural network using GOA.

5. Data Set

This chapter uses one of the most challenging engineering problems in the real world to prove GOA's capability. The chosen issue is the classification of sonar data, which is one of the challenges and concerns of engineers and scientists, working in this field.

5.1. Scenario Test Design and Experimental Data Formation. Since our goal is to obtain a reliable and realistic set of high-dimensional sonar data, a real experiment was designed and implemented. The experiment was conducted using the tunnel cavitation model NA-10, made in England. In the first phase, three types of impellers were produced in classes A, B, and C. The Class A impeller has three blades that can be used to pick up sound from a boat, and small passenger ship. The Class B impeller has four blades that are used to get the sound from a container ship, ocean liner, and small oil tanker. The Class C impeller has five blades and is used to extract sound from the aircraft carrier and large oil tanker. In this experiment, the impellers are evaluated at different speeds to simulate different operating conditions. During these experiments, the sound (acoustic noise) of the various impellers was stored on a computer using the

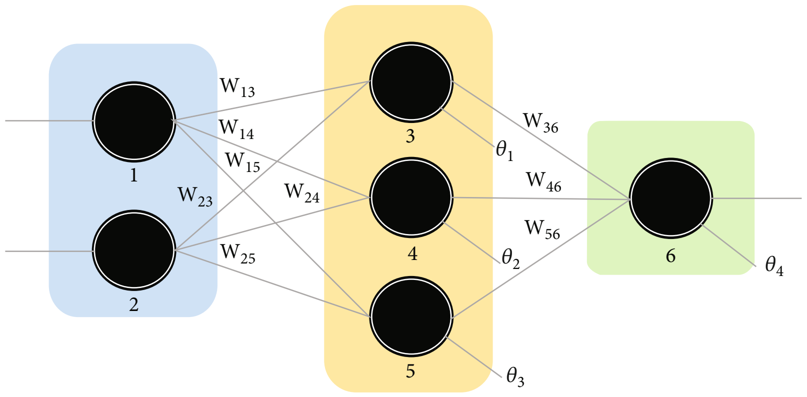


FIGURE 4: The structure of the multilayer neural network with the structure of 2-3-1.

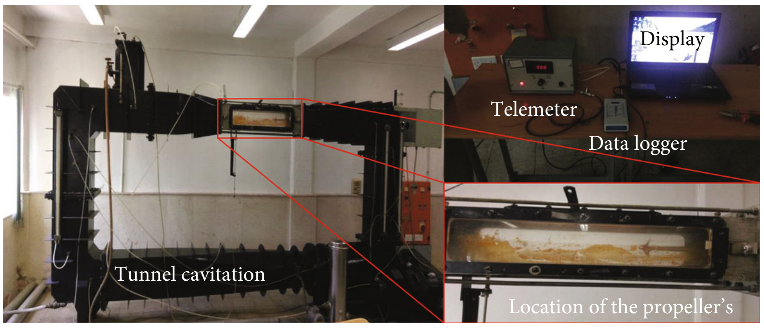


FIGURE 5: An overview of the proposed test scenario.

B&K 8103 hydrophone and Data-Logger of the UDAQ_Lite model.

The proposed test scenario is shown in Figure 5. Propulsion velocity in free water is expressed as a number without J dimension, proportional to rotation speed N(RPM) or rotation speed per second (RPS) and impeller diameter D (m) and v the water flow velocity:

$$J = \frac{v}{nD}. \tag{16}$$

At all experiments, the atmospheric pressure of 100 kPa and pressure inside the tunnel were considered concerning the depth of impeller placement in that floating class. The water flow rate inside the tunnel is also 4 m/s. One of the hydrophones is mounted next to the propeller at a distance of 10 cm and the other 50 cm from the first hydrophone.

In this section, the noise of the designed impellers is measured in four steps. In the first step, after the water flow slows down, the noise is received by the hydrophones and then received and stored by the MATLAB software and Data-Logger. Secondly, by turning on the impeller and without the impeller wheel, the engine noise is also obtained in several stages, so that we can obtain a reasonable estimate of this noise. In the third stage, the impeller rotates at different rotations (depending on the type of the model float) to obtain the impeller rotation noises for the different floating classes. In the fourth step, by turning on the water rotation pump and the bubble discharging pump into the discharge tunnel, the impeller motor is activated and the sound is collected by the Data-Logger and the MATLAB software, in the computer. At all stages, all the actual data, without amplifying the values, are stored in the computer for later use.

5.1.1. Drawing Noise Curves for Model propeller's. According to the standard reference [30, 31] the power is calculated in dB related to the water acoustic reference power (1μPa). Figure 6 shows noise curves at the hydrophone surface, Fourier transforms, and dB power spectrum, respectively, for different classes of impellers.

Generally, the relation (17) is used to obtain the fundamental frequency by RPM.

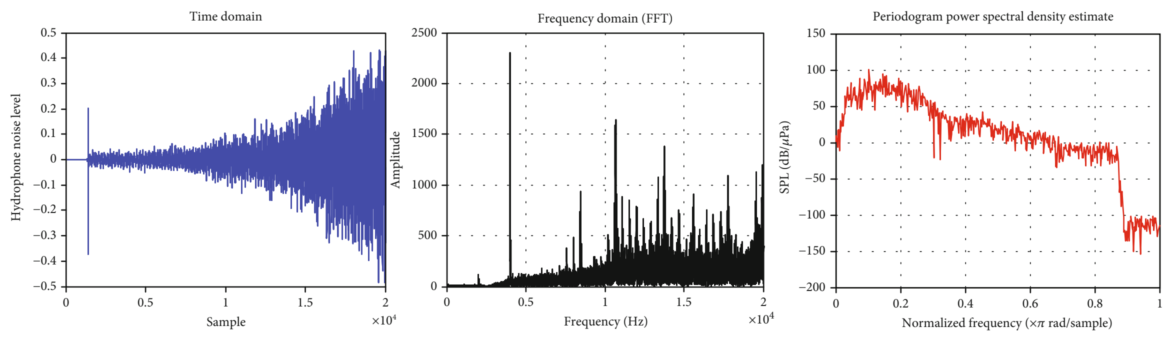
$$f_{\text{Fundamental}} = \frac{\text{RPM} \times \text{Number of blade}}{60} \text{Hz}. \tag{17}$$

In this section, 500 samples with different propeller and a number of rotations were obtained.

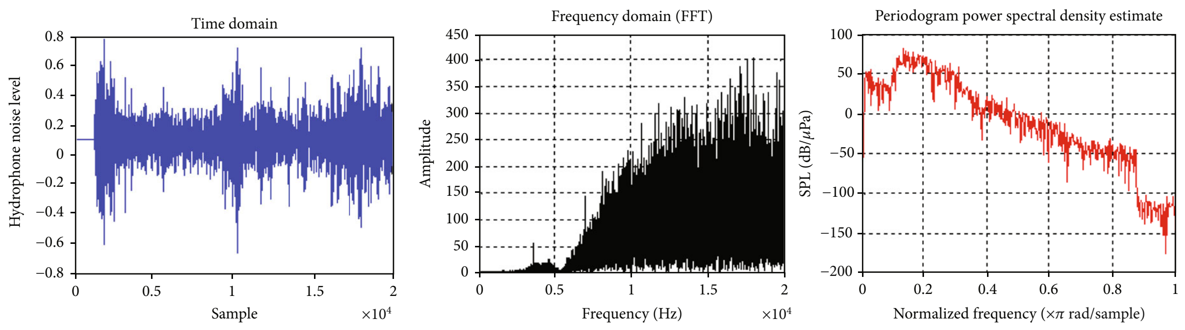
5.2. Feature Extraction

5.2. Feature Extraction. After the preprocessing section after receiving the detected frames containing the audio matching to the received signals, the detected sounds are provided to the feature extraction section with the effects of synthetic phenomena eliminated and converted to the frequency domain (called $S(k)$). At this point, the signal spectrum's energy is calculated using the following;

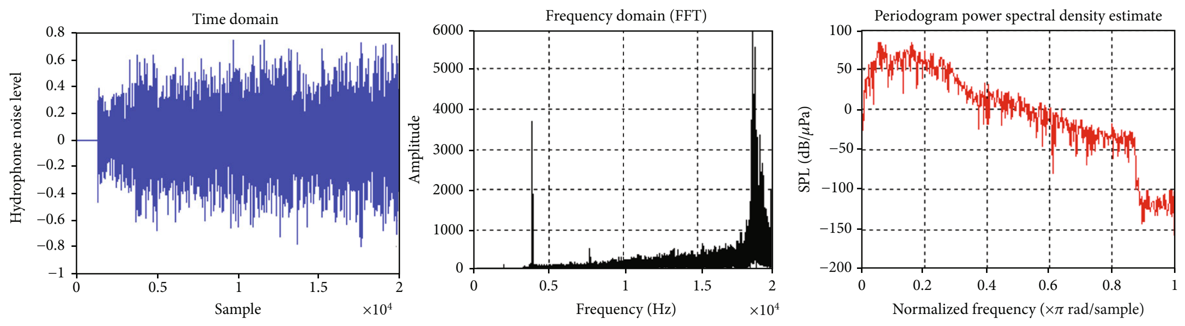
$$|S(k)|^2 = S_r^2(k) + S_i^2(k). \tag{18}$$



(a) Class A propeller (three blades)



(b) Class B propeller (four blades)



(c) Class C propeller (five blades)

Figure 6: The received noise curves at the hydrophone surface, the Fourier transform, and the power spectrum in dB relative to the acoustic water reference power (1 Pa) for the different model floats.

$S_r(k)$ and $S_i(k)$ denote the real and imaginary components of the detected signal's Fourier transform, respectively. After that, Mel-scaled triangle filters are used to filter the spectral energy of $|S(k)|^2$. The relationship between the output energy of the l filter and (20).

$$E(l) = \sum_{k=0}^{N-1} |S(k)|^2 H_l(k). \tag{19}$$

N is the number of discrete frequencies utilized in the FFT conversion of the preprocessing phase, and $H_l(k)$ is a filtered transfer functions, where $l = 0$. The logarithm function compresses the dynamic range of the Mel-Filtered Energy Spectrum.

$$E(l) = \log\left(E(l)\right). \tag{20}$$

Eventually, the relation (21) and discrete cosine transform are used to convert the Mel-Frequency Cepstral Coefficients (MFCC) to the time domain (DCT).

$$\mathrm{C(n)} = \sum_{l=1}^{M} e(l)\cos\left(n\left(l - \frac{1}{2}\right)\frac{\pi}{M}\right). \tag{21}$$

The feature vector will be as the relation in this situation and for any explicit purpose.

$$X_m = [c(0).c(1).\cdots.c(P-1)]^T. \tag{22}$$

Figure 7 shows a block diagram of the procedures involved in the classification steps.

This section contains 140 extracted characteristics. Given 500 samples, the data set will be 500×140 in size, with 140 representing the number of input nodes (n) in the neural network and 281 being the number of neurons in the hidden layer. Thus, despite the vast data sets,

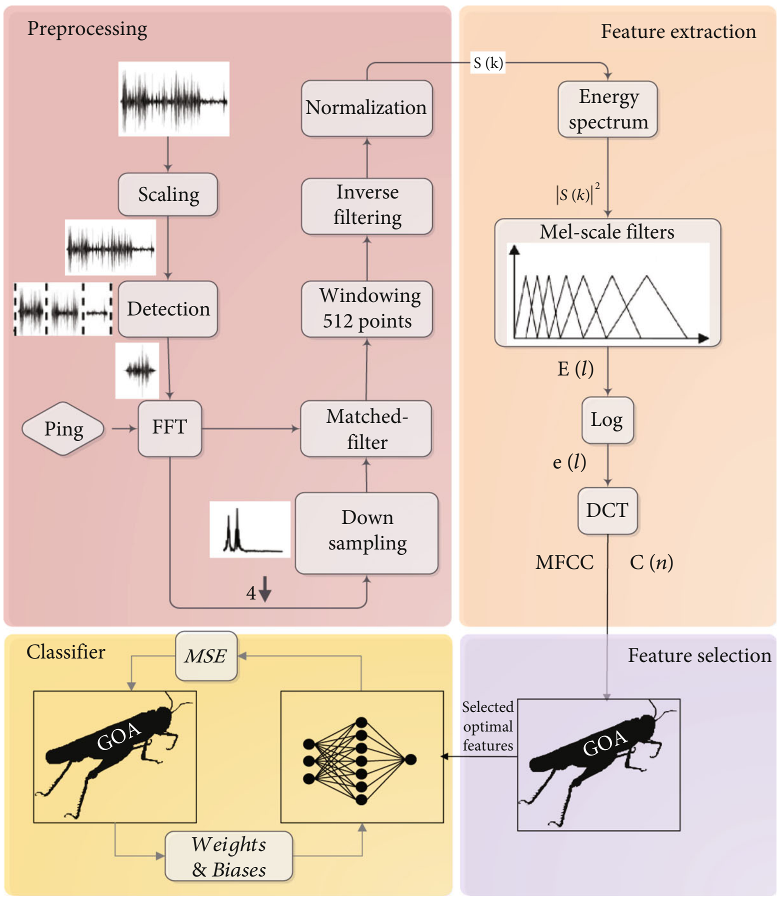


FIGURE 7: Block diagram of the procedures involved in classification steps.

TABLE 1: Different feature modes.

Feature vector states	f_1	f_2	f_3	$\cdots$	f_{138}	f_{139}	f_{140}
1	0	0	0	$\cdots$	0	0	0
2	1	0	0	$\cdots$	0	0	0
3	1	1	0	$\cdots$	0	0	0
4	1	1	1	$\cdots$	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
2^{140}-2	1	1	1	$\cdots$	1	0	0
2^{140}-1	1	1	1	$\cdots$	1	1	0
2^{140}	1	1	1	$\cdots$	1	1	1

TABLE 2: Assumed values for an initial population of 209.

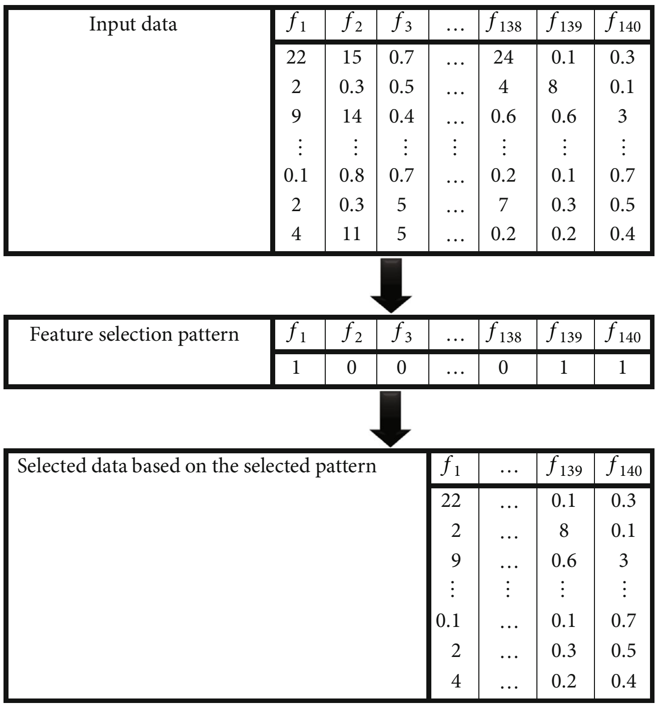
Initial population	f_1	f_2	f_3	$\cdots$	f_{138}	f_{139}	f_{140}
1	0	1	0	$\cdots$	0	0	0
2	0	0	1	$\cdots$	0	1	0
3	1	0	0	$\cdots$	0	1	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
207	0	0	0	$\cdots$	1	0	1
208	1	1	1	$\cdots$	0	1	0
209	1	0	1	$\cdots$	0	0	1

computational and deterministic approaches have a high time complexity, and random methods are regarded as the optimal answer for this kind of problem.

5.3. Feature Extraction. As discussed in the previous subsection, the dimension of the feature matrix is 500×140. All extracted features are not useful and may contain useless or duplicate information. As shown in Table 1, there are 2^{140} states for the obtained feature matrix. The binary version of GOA is responsible for selecting the optimal features.

It is assumed that the initial population is 209. Table 2 shows the hypothetical values for the initial population of 209.

In Table 2, each row is used as a feature selection pattern. By using these patterns, the entire educational input data is selected. In this paper, accuracy is used as a fitness function. In the following, MLP-GOA is used to calculate the fitness function. Therefore, for each selected pattern, the accuracy is calculated using MLP-GOA (the accuracy value is exactly the fit value of each pattern). Assuming that the initial population is 209, the length of the fitness vector will also be



Input data	f_1	f_2	f_3	…	f_{138}	f_{139}	f_{140}
	22	15	0.7	…	24	0.1	0.3
	2	0.3	0.5	…	4	8	0.1
	9	14	0.4	…	0.6	0.6	3
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	0.1	0.8	0.7	…	0.2	0.1	0.7
	2	0.3	5	…	7	0.3	0.5
	4	11	5	…	0.2	0.2	0.4

Feature selection pattern	f_1	f_2	f_3	…	f_{138}	f_{139}	f_{140}
	1	0	0	…	0	1	1

Selected data based on the selected pattern	f_1	…	f_{139}	f_{140}
	22	…	0.1	0.3
	2	…	8	0.1
	9	…	0.6	3
	⋮	⋮	⋮	⋮
	0.1	…	0.1	0.7
	2	…	0.3	0.5
	4	…	0.2	0.4

Figure 8: Feature selection from the most optimal selection pattern.

Table 3: Parameters and initial values of training algorithms.

Algorithm	Parameter	Value
GWO	Population size	209
	The number of Gray Wolf	13
PSO	Population size	208
	Cognitive constant (C1)	1.1
	Social constant (C2)	1.1
	Local constant (W)	0.4
ACO	Population size	209
	ACO primary pheromone ($\tau 0$)	0.000001
	Pheromone updating constant (Q)	20
	Pheromone constant (q0)	1.1
	Decreasing rate of the overall pheromone (Pg)	0.8
	Decreasing rate of local pheromone (Pt)	0.6
	Pheromone sensitivity (a)	2
	Observable sensitivity (β)	6
GSA	Population size	209
	Coefficient (α)	21
	Limit down	-31
	Limit up	31
	Gravitational constant (G°)	1
	The initial speed of the masses	[0, 1]
	The initial value of the acceleration	0
	The initial value of mass	0
GOA	Population size	209
	Highest value (c max)	1
	Lowest value (c min)	0.00001

Table 4: Results of applying different training algorithms in designing sonar purpose classifier.

Classifier	MSE ($AVE \pm STD$)	P values	Classification rate (%)	Processing time (s)
GMLP-GOA	**0 .1055 ± 3.4180*e* − 01**	**N/A**	**98.1276**	**3.14**
MLP-GOA	$0.1283 \pm 8.2720e-04$	N/A	95.6667	6.24
MLP-GWO	0.1519 ± 0.0269	0.0039	94.3522	7.39
MLP-GSA	0.3149 ± 0.2965	6.2149e-04	69.6633	10.44
MLP-ACO	0.2527 ± 0.1744	7.2798e-12	75.3333	7.54
MLP-PSO	0.2011 ± 0.2076	0.2239e-03	92.8222	8.78

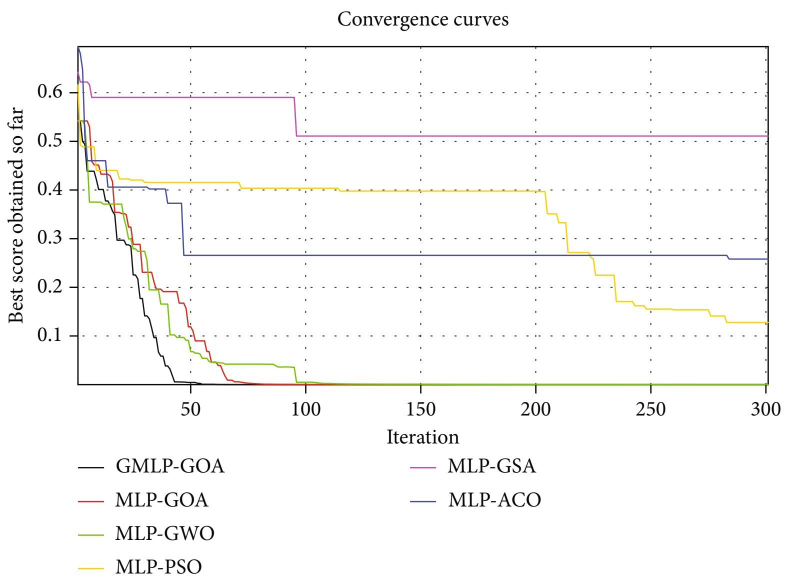


Figure 9: Convergence diagram of different training algorithms.

equal to 209. Figure 8 shows how to select the feature from the most optimal selection pattern.

If the stop condition (reaching 100% accuracy or reaching the maximum number of iterations) occurs, the program ends, and the data in the best pattern (selected and reduced features) is selected for classification with MLP-GOA.

6. Experimental Results and Discussion

For fair comparison and performance evaluation of GMLP-GOA classifier, five classifiers MLP-GOA, MLP-GWO, MLP-PSO, MLP-ACO, and MLP-GSA are used. The selection algorithms are all population based. GMLP-GOA and MLP-GOA classifiers have the same training by GOA. The only difference between these two classifiers is that in the GMLP-GOA classifier, GOA is used for feature selection. Table 3 contains the parameters and beginning values for these algorithms.

In the GMLP-GOA hybrid classifier, the optimal features obtained from GOA are used. If for other classifiers, a feature matrix with dimensions of 500×140 is used. Classifiers are evaluated in terms of classification rate, local minimization avoidance and convergence speed.

Table 4 shown the classification rate, mean and standard deviation of the smallest error, and P value for each method after it has been run 20 times. The classification rate indicates the correct recognition accuracy of the classifier, while the smallest error's mean values and standard deviation, as well as the P value, and show the algorithmic power in avoiding local optimization. Also shown in Figure 9, is a comprehensive comparison of the convergence rate and method and the final error of the classifiers.

As shown in Figure 9, GMLP-GOA has the best convergence rate and MLP-GSA has the worst convergence rate among the used classifiers. The results obtained in Table 4 show that in terms of classification rate, GMLP-GOA succeeded in classifying sonar big data with 98.12% accuracy, while MLP-GSA had the worst performance with a classification rate of 69.66%. In terms of processing time, GMLP-GOA had the fastest processing time with 3.14 s, while MLP-GSA required more time for processing than other classifiers with 10.44 s. As can be seen in Table 4 and the values of standard deviation and P value, the GMLP-GOA hybrid classifier performs optimally in terms of avoiding being trapped in the local minimum. One of the reasons for the success of GMLP-GOA can be mentioned the power of GOA in detecting the boundary between exploration and extraction phase. As shown in Figure 9, GMLP-GOA converged after 50 iterations, while MLP-GOA and MLP-GWO converged after 75 and 95 iterations, respectively. Therefore, according to the obtained results, GMLP-GOA showed a successful performance in dealing with sonar big data and is recommended for use in real-world problems.

7. Conclusion

In this paper, GOA is used to select optimal features and train MLP-NN in GMLP-GOA hybrid classifier to classify sonar big data. Also, to have a fair comparison, 5 classifiers MLP-GOA, MLP-GWO, MLP-PSO, MLP-ACO, and MLP-GSA were used, which are all based on population-based metaheuristic algorithms. As seen in the simulation results, GOA can correctly detect the boundary between exploration and exploitation phases. Therefore, it does not get stuck in local optima, and its ability to find global optima for solving high-dimensional problems such as big data sonar is well-proven. The results show that GMLP-GOA has the best performance for classifying sonar big data by reaching a classification rate of 98.12%. 5 classifiers MLP-GOA, MLP-GWO, MLP-PSO, MLP-ACO, and MLP-GSA have the most accurate classification accuracy by reaching values of 95.66, 94.35, 92.82, 75.33, and 69.66, respectively.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. He, B. Ai, A. F. Molisch et al., "Clustering enabled wireless channel modeling using big data algorithms," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 177–183, 2018.

[2] L. Yang, Z. Xiong, G. Liu, Y. Hu, X. Zhang, and M. Qiu, "An analytical model of page dissemination for efficient big data transmission of C-ITS," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16524–16533, 2022.

[3] G. Liu, "Data collection in MI-assisted wireless powered underground sensor networks: directions, recent advances, and challenges," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 132–138, 2021.

[4] X. Zenggang, Z. Mingyang, Z. Xuemin et al., "Social similarity routing algorithm based on socially aware networks in the big data environment," *Journal of Signal Processing Systems*, 2022.

[5] T. Berthold, A. Leichter, B. Rosenhahn, V. Berkhahn, and J. Valerius, "Seabed sediment classification of side-scan sonar data using convolutional neural networks," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, Honolulu, HI, USA, 2018.

[6] B. Zhu, Q. Zhong, Y. Chen et al., "A novel reconstruction method for temperature distribution measurement based on ultrasonic tomography," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 7, pp. 2352–2370, 2022.

[7] W. Qiao, Y. Wang, J. Zhang, W. Tian, Y. Tian, and Q. Yang, "An innovative coupled model in view of wavelet transform for predicting short- term PM10 concentration," *Journal of Environmental Management*, vol. 289, article 112438, 2021.

[8] A. Saffari, M. Khishe, and S. Zahiri, "Fuzzy-ChOA: an improved chimp optimization algorithm for marine mammal classification using artificial neural network," *Analog Integrated Circuits and Signal Processing*, vol. 111, no. 3, pp. 403–417, 2022.

[9] X. Wu, W. Zheng, X. Xia, and D. Lo, "Data quality matters: a case study on data label correctness for security bug report prediction," *IEEE Transactions on Software Engineering*, vol. 48, no. 7, pp. 2541–2556, 2022.

[10] W. Zhou, Y. Lv, J. Lei, and L. Yu, "Global and local-contrast guides content-aware fusion for rgb-d saliency prediction," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 6, pp. 3641–3649, 2021.

[11] R. Patgiri, "A taxonomy on big data: survey," 2018, http://arxiv.org/abs/1808.08474.

[12] P. Koturwar, S. Girase, and D. Mukhopadhyay, "A Survey of Classification Techniques in the Area of Big Data," *Computer Science*, vol. 1, no. 11, pp. 1–7, 2015.

[13] Y. Zhang, F. Liu, Z. Fang, B. Yuan, G. Zhang, and J. Lu, "Learning from a complementary-label source domain: theory and algorithms," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[14] Y. Shen, N. Ding, H. T. Zheng, Y. Li, and M. Yang, "Modeling relation paths for knowledge graph completion," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 11, pp. 3607–3617, 2021.

[15] D. Javaheri, P. Lalbakhsh, and M. Hosseinzadeh, "A novel method for detecting future generations of targeted and metamorphic malware based on genetic algorithm," *IEEE Access*, vol. 9, pp. 69951–69970, 2021.

[16] W. Qiao, Z. Li, W. Liu, and E. Liu, "Fastest-growing source prediction of US electricity production based on a novel hybrid model using wavelet transform," *International Journal of Energy Research*, vol. 46, no. 2, pp. 1766–1788, 2022.

[17] K. Shang, Z. Chen, Z. Liu et al., "Haze prediction model using deep recurrent neural network," *Atmosphere*, vol. 12, no. 12, p. 1625, 2021.

[18] X. Wu, Z. Liu, L. Yin et al., "A haze prediction model in Chengdu based on lstm," *Atmosphere*, vol. 12, no. 11, p. 1479, 2021.

[19] B. P. Howell, S. Wood, and S. Koksal, "Passive sonar recognition and analysis using hybrid neural networks," in *Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)*, pp. 1917–1924, San Diego, CA, USA, 2003.

[20] H. Xu, J. Zhou, P. G. Asteris, D. J. Armaghani, and M. M. Tahir, "Supervised machine learning techniques to the prediction of tunnel boring machine penetration rate," *Applied Sciences*, vol. 9, no. 18, pp. 3715–3719, 2019.

[21] C. Wu, M. Khishe, M. Mohammadi, S. H. Taher Karim, and T. A. Rashid, "Evolving deep convolutional neutral network by hybrid sine-cosine and extreme learning machine for real-time COVID19 diagnosis from X-ray images," *Soft Computing*, vol. 6, pp. 1–20, 2021.

[22] D. Devikanniga, K. Vetrivel, and N. Badrinath, "Review of meta-heuristic optimization based artificial neural networks and its applications," *Journal of Physics Conference Series*, vol. 1362, no. 1, article 012074, 2019.

[23] X. Cai, R. Tang, H. Zhou et al., "Dynamically controlling terahertz wavefronts with cascaded metasurfaces," *Advanced Photonics*, vol. 3, no. 3, pp. 1–10, 2021.

[24] L. Zhang, D. Wu, X. Han, and Z. Zhu, "Feature extraction of underwater target signal using mel frequency cepstrum coefficients based on acoustic vector sensor," *Journal of Sensors*, vol. 2016, Article ID 7864213, 11 pages, 2016.

[25] Y. Liu, J. Tian, W. Zheng, and L. Yin, "Spatial and temporal distribution characteristics of haze and pollution particles in China based on spatial statistics," *Urban Climate*, vol. 41, article 101031, 2022.

[26] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. Zhang, and L. Guibas, "GRASS: Generative Recursive Autoencoders for Shape Structures," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–14, 2017.

[27] W. Qiao, W. Liu, and E. Liu, "A combination model based on wavelet transform for predicting the difference between monthly natural gas production and consumption of US," *Energy*, vol. 235, article 121216, 2021.

[28] H. Zhao, C. Zhu, X. Xu, H. Huang, and K. Xu, "Learning practically feasible policies for online 3D bin packing," *Science China Information Sciences*, vol. 65, article 112105, 2022.

[29] P. Wang, X. Yu, and J. Lü, "Identification and evolution of structurally dominant nodes in protein-protein interaction networks," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 8, no. 1, pp. 87–97, 2014.

[30] Y. Wang, L. P. Yuan, M. Khishe, A. Moridi, and F. Mohammadzade, "Training RBF NN using sine-cosine algorithm for sonar target classification," *Archives of Acoustics*, vol. 45, no. 4, pp. 753–764, 2020.

[31] M. Khishe and A. Safari, "Classification of sonar targets using an MLP neural network trained by dragonfly algorithm," *Wireless Personal Communications*, vol. 108, no. 4, pp. 2241–2260, 2019.

[32] M. Khishe, M. R. Mosavi, and M. Kaveh, "Improved migration models of biogeography-based optimization for sonar dataset classification by using neural network," *Applied Acoustics*, vol. 118, pp. 15–29, 2017.

[33] S. Mirjalili and A. S. Sadiq, "Magnetic optimization algorithm for training multi layer perceptron," in *2011 IEEE 3rd International Conference on Communication Software and Networks*, Xi'an, China, 2011.

[34] D. Karaboga, B. Akay, and C. Ozturk, "Artificial Bee Colony (ABC) optimization algorithm for training feed-forward neural networks," in *Modeling Decisions for Artificial Intelligence. MDAI 2007*, V. Torra, Y. Narukawa, and Y. Yoshida, Eds., vol. 4617 of Lecture Notes in Computer Science, pp. 318–329, Springer, Berlin, Heidelberg, 2007.

[35] S. Mirjalili, S. Mohammad, and A. Lewis, "Advances in engineering software Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[36] A. Luque-Chang, E. Cuevas, F. Fausto, D. Zald, and M. Pérez, "Social spider optimization algorithm: modifications, applications, and perspectives," *Mathematical Problems in Engineering*, vol. 2018, Article ID 6843923, 29 pages, 2018.

[37] L. A. M. Pereira, D. Rodrigues, P. B. Ribeiro, J. P. Papa, and S. A. T. Weber, "Social-spider optimization-based artificial neural networks training and its applications for Parkinson's disease identification," in *2014 IEEE 27th International Symposium on Computer-Based Medical Systems*, pp. 14–17, New York, NY, USA, 2014.

[38] Y. Ling, Y. Zhou, and Q. Luo, "Lévy flight trajectory-based whale optimization algorithm for global optimization," *IEEE Access*, vol. 5, pp. 6168–6186, 2017.

[39] G. Dhiman and V. Kumar, "Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems," *Knowledge-Based Systems*, vol. 165, pp. 169–196, 2019.

[40] X. Wu, S. E. N. Zhang, W. Xiao, and S. Member, "The exploration/exploitation tradeoff in whale optimization algorithm," *IEEE Access*, vol. 7, pp. 125919–125928, 2019.

[41] J. Zhang, C. Zhu, L. Zheng, and K. Xu, "ROSEFusion: random optimization for online dense reconstruction under fast camera motion," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, 2021.

[42] W. Qiao, M. Khishe, and S. Ravakhah, "Underwater targets classification using local wavelet acoustic pattern and multi-layer perceptron neural network optimized by modified whale optimization algorithm," *Ocean Engineering*, vol. 219, article 108415, 2021.

[43] S. Saremi, S. Mirjalili, and A. Lewis, "Advances in engineering software grasshopper optimisation algorithm: theory and application," *Advances in Engineering Software*, vol. 105, pp. 30–47, 2017.

[44] M. Noroozi, H. Mohammadi, E. Efatinasab, A. Lashgari, M. Eslami, and B. Khan, "Golden search optimization algorithm," *IEEE Access*, vol. 10, pp. 37515–37532, 2022.

[45] K. Liu, F. Ke, X. Huang et al., "DeepBAN: a temporal convolution-based communication framework for dynamic WBANs," *IEEE Transactions on Communications*, vol. 69, no. 10, pp. 6675–6690, 2021.

[46] S. P. Adam, S. N. Alexandropoulos, P. M. Pardalos, and M. N. Vrahatis, "No free lunch theorem: a review," in *Approximation and Optimization. Springer Optimization and Its Applications*, vol. 145, pp. 57–82, Springer, Cham.

[47] K. H. Lai, Z. Zainuddin, and P. Ong, "A study on the performance comparison of metaheuristic algorithms on the learning of neural networks," *AIP Conference Proceedings*, vol. 1870, 2017.

[48] M. A. Simaan, "Simple explanation of the no-free-lunch theorem and its implications," *Journal of optimization theory and applications*, vol. 115, no. 3, pp. 549–570, 2003.

[49] H. Zheng and S. Jin, "A multi–source fluid queue based stochastic model of the probabilistic offloading strategy in a MEC system with multiple mobile devices and a single MEC server," *International Journal of Applied Mathematics and Computer Science*, vol. 32, no. 1, pp. 125–138, 2022.

[50] Y. Wang, X. Han, and S. Jin, "MAP based modeling method and performance study of a task offloading scheme with time-correlated traffic and VM repair in MEC systems," *Wireless Networks*, 2022.

[51] B. Cao, J. Zhao, X. Liu et al., "Multiobjective evolution of the explainable fuzzy rough neural network with gene expression programming," *IEEE Transactions on Fuzzy Systems*, 2022.

[52] W. Zheng, Y. Xun, X. Wu, Z. Deng, X. Chen, and Y. Sui, "A comparative study of class rebalancing methods for security bug report classification," *IEEE Transactions on Reliability*, vol. 70, no. 4, pp. 1658–1670, 2021.

[53] V. N. Kovalnogov, M. I. Kornilova, Y. A. Khakhalev, D. A. Generalov, T. E. Simos, and C. Tsitouras, “New family for Runge-Kutta-Nyström pairs of orders 6(4) with coefficients trained to address oscillatory problems,” *Mathematical Methods in the Applied Sciences*, vol. 45, no. 12, pp. 7715–7727, 2022.

[54] H. Wu, S. Jin, and W. Yue, “Pricing policy for a dynamic spectrum allocation scheme with batch requests and impatient packets in cognitive radio networks,” *Journal of Systems Science and Systems Engineering*, vol. 31, no. 2, pp. 133–149, 2022.

[55] A. Saffari, S. H. C. A. Zahiri, and M. Khishe, “Fuzzy Grasshopper Optimization Algorithm: a hybrid technique for tuning the control parameters of GOA using Fuzzy System for big data sonar classification,” *Iranian Journal of Electrical and Electronic Engineering*, vol. 18, no. 1, article 2131, 2022.

[56] Y. Meraihi, A. B. Gabis, and S. Mirjalili, “Grasshopper optimization algorithm: theory, variants, and applications,” *IEEE Access*, vol. 4, pp. 50001–50024, 2021.

[57] P. S. Optimization, “Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks,” in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706)*, Indianapolis, IN, USA, 2003.

[58] C. S. Koh and S. Y. Hahn, “Detection of magnetic body using artificial neural network with modified simulated annealing,” *IEEE Transactions on Magnetics*, vol. 30, no. 5, pp. 3644–3647, 1994.