Multimedia Tools and Applications
https://doi.org/10.1007/s11042-024-18579-0

# Bitterling fish optimization (BFO) algorithm

Lida Zareian[1] · Javad Rahebi[2] · Mohammad Javad Shayegan[1]

Received: 1 March 2023 / Revised: 28 December 2023 / Accepted: 3 February 2024

**Abstract**
The bitterling fish is a prime example of intelligent behavior in nature for survival. The bitterling fish uses the oyster spawning strategy as their babysitter. The female bitterling fish looks for a male fish stronger than other fish to find the right pair. In order to solve optimization issues, the Bitterling Fish Optimization (BFO) algorithm is modeled in this manuscript based on the mating behavior of these fish. The bitterling fish optimization algorithm is more accurate than the gray wolf optimization algorithm, whale optimization algorithm, butterfly optimization algorithm, Harris Hawks optimization algorithm, and black widow optimization algorithm, according to experiments and implementations on various benchmark functions. Data mining and machine learning are two areas where meta-heuristic techniques are frequently used. In trials, the MLP artificial neural network and a binary version of the BFO algorithm are used to lower the detection error for intrusion traffic. The proposed method's accuracy, precision, and sensitivity index for detecting network intrusion are 99.14%, 98.87%, and 98.85%, respectively, according to experiments on the NSL KDD data set. Compared to machine learning approaches like NNIA, DT, RF, XGBoot, and CNN, the proposed method is more accurate at detecting intrusion. The BFO algorithm is used for feature selection in the UNSW-NB15 dataset, and the tests showed that the accuracy of the proposed method is 96.72% in this dataset. The proposed method of the BFO algorithm is also used to improve Kmeans clustering, and the tests performed on the dataset of covid 19, diabetes, and kidney disease show that the proposed method performs better than iECA*, ECA*, GENCLUST + +(G + +) methods. Deep has KNN, LVQ, SVM, ANN, and KNN.

**Keywords** Optimization problems · Meta-heuristic algorithm · Bitterling Fish optimization(BFO) algorithm · Data mining

✉ Javad Rahebi
cevatrahebi@topkapi.edu.tr

Lida Zareian
lidazrn@gmail.com

Mohammad Javad Shayegan
shayegan@usc.ac.ir

[1] Department of Computer Engineering, University of Science and Culture, Tehran, Iran

[2] Software Engineering Department, Istanbul Topkapi University, Istanbul, Turkey

Published online: 21 February 2024

## 1 Introduction

Most of the problems around us are of a nature that maximizes or minimizes a goal function. Optimization problems have a maximization or minimization approach for engineering and computer science problems [1]. Optimization problems arise in many applications, such as image processing [2], data mining [3], machine learning [4], bioinformatics [5], engineering [6], and computer networks [7]. Solving an optimization problem is finding several optimal solutions from among the available solutions. Optimization problem-solving methods are divided into definite and indefinite categories. Definitive methods such as gradients can solve a limited range of optimization problems despite advantages such as high accuracy. Unlike deterministic methods, some optimization problem-solving methods are less specific but can be used to solve various continuous and discrete optimization problems. Meta-heuristic algorithms are a practical example of nondeterministic problem-solving methods and, in most cases, provide solutions close to the original solution. Meta-heuristic algorithms are modeled on modeling the behavior of animals, plants, biological phenomena, human behaviors, or physical and mathematical phenomena [8].

Modeling the efforts of living things and animals for survival is one of the critical methods of modeling meta-heuristic algorithms. Organisms and animals have complex and compelling behaviors for survival. Given that the behavior of living things has evolved over millions of years, they are brilliant. Particle swarm optimization algorithm [9], ant clone optimization algorithm [10], and firefly algorithm [11] are among the meta-heuristic algorithms. These algorithms are modeled on the behavior of birds, arthropods, and insects and are successful examples of meta-heuristic algorithms. In recent years, more meta-heuristic algorithms have inspired the swarm-hunting approach. Most of the meta-heuristic algorithms introduced in the last few years have formulated most of the hunting aspects of living things [12]. In these behaviors, the prey position is optimal, and other population members attack the optimal solution.

For example, the bait attack mechanism was used in the whale optimization algorithm in 2016 [13] and Harris Hawk's optimization algorithm in 2019 [14]. In the WOA algorithm, there are three behaviors of rotational motion: spiral, random, and hunting or searching. The HHO algorithm is too complex and has a large number of relationships. The gray wolf algorithm [15], the wolf leadership hierarchy, is used in the attack. In the GWO algorithm, if the three alpha, beta, and delta solutions are located near the local optimal, the entire population is directed to the optimal solution. The spotted hyena optimization algorithm [16] considers the optimal point as prey. In the SHO algorithm, the optimal solution is searched by hyenas, and the global space of the problem needs to be searched better. Spiders inspire the black widow optimization algorithm [17] in hunting and cannibalism. In the BWO algorithm, although inappropriate solutions are eliminated over time, this process will likely reduce population diversity. Some other meta-heuristic algorithms emulate the laws of evolution, such as genetic and differential evolution algorithms. GA and DE algorithms are prone to rapid convergence to optimal local solutions in early iterations. Meta-heuristic algorithms based on human behavior must be better modeled despite their intelligence because human behavior is complex. Some meta-heuristic algorithms are inspired by physical phenomena, such as the gravitational and atom search algorithms [18]. The challenge of physical algorithms is the lack of high intelligence, which can make their error significant. According to the meta-heuristic algorithms discussed and similar examples, it is necessary to introduce new algorithms that eliminate or modify the challenges of the previous meta-heuristic algorithms.

In this manuscript, a meta-heuristic algorithm called the bitterling fish optimization (BFO) algorithm is formulated and introduced based on the behavior of cunning fish. The purpose of the BFO algorithm is to perform a global search on initial iterations to escape local optimizations. Doing a local search in the last iterations to find a more accurate solution is another strategy of the BFO algorithm. The proposed approach introduces a distributed search that increases the proposed method's ability to discover the global optimal.

The purpose of this manuscript is to present a meta-heuristic algorithm based on bitterling fish optimization (BFO) behavior. The aim is to provide a new meta-heuristic method that is strong in its search for exploitation and exploration and can perform acceptably in various optimization problems. Instead of being evaluated only on benchmark functions and classical engineering problems, the proposed algorithm in the manuscript version is evaluated on several more practical problems. The third manuscript evaluates BFO algorithms on benchmark functions, engineering optimization problems, network traffic classification problems, network intrusion detection, phishing attack detection, clustering, and disease detection. Another innovation of this manuscript is the simultaneous presentation of the binary version of BFO algorithms for feature selection.

The contributions of the authors in this manuscript are as follows:

- Presenting a new meta-heuristic algorithm based on the behavior of bitterling fish.
- Presenting a discrete binary version of the BFO algorithm.
- Presenting an intrusion detection system based on the BFO + ANN algorithm.
- Providing a phishing attack detection system based on BFO + ANN.
- Presenting a new clustering method by optimizing cluster centers using the BFO algorithm.

This paper is prepared in several parts. In the Section 2, related works are reviewed. In the Section 3, the BFO algorithm is modeled and then the BFO algorithm is used to improve the multilayer neural network in attack detection. In the Section 4, the proposed method is implemented and analyzed. Finally, in the Section 5, the results of the examination and future works are expressed.

## 2 Related works

Meta-heuristic methods for solving optimization problems can emulate different mechanisms. The authors' classification of these algorithms is shown in Fig. 1. This classification divides meta-heuristic algorithms into categories based on the Trajectory [19] and population-based algorithms [20]. In Trajectory-based algorithms, there is only one solution to each algorithm iteration. In this group of algorithms, a solution searches the problem space. One of these algorithms is the Simulated annealing (SA) algorithm [21]. The challenge of trajectory-based algorithms is the high convergence rate to local optimizations. Most meta-heuristic algorithms are population-based. In this algorithm category, several solutions at each stage search the problem space in parallel. The advantage of population-based algorithms is that if one or more solutions are caught in the local optimization, the other solutions can converge to the global optimization.

The meta-heuristic algorithms are divided into population-based approaches to evolutionary algorithms [22], swarm intelligence algorithms [23], physical algorithms [24], and algorithms based on biological behavior [25]. In swarm intelligence algorithms, the swarm behavior of living things is used for survival. This algorithm considers most of the social nature

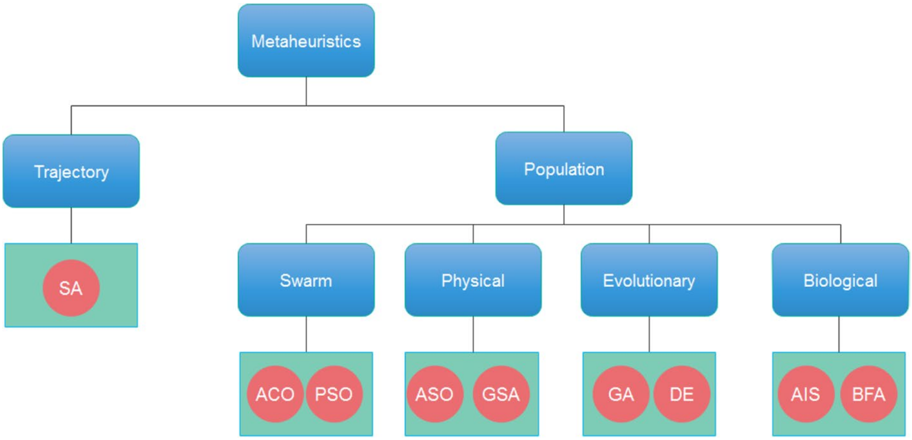


**Fig. 1** Classification of meta-heuristic algorithms

and group behavior of living things. The New Caledonian (NC) crow learning algorithm [26] or the water strider algorithm (WSA) [27] is a swarm intelligence algorithm. Evolutionary algorithms use the principles of evolution and competence to find the optimal solution. In this category of algorithms, it is assumed that reasonable solutions have a better chance of survival and wrong solutions disappear over time. Evolutionary algorithms include genetic and biogeography-based optimization (BBO) [28]. Physical algorithms are inspired by a physical phenomenon, such as a big bang in the Big Bang–Big Crunch (BB–BC) algorithm [29] or water waves in the water wave optimization algorithm [30]. Biological patterns are another example of meta-heuristic algorithms that try to solve a problem based on a biological approach. In these algorithms, a biological phenomenon is modeled, such as plants or bacteria, or the human immune system, such as the bacterial feeding algorithm and the immune system algorithm. The algorithms in question have their advantages and disadvantages. For example, a genetic algorithm can eliminate inappropriate solutions, or an algorithm such as WSA can optimize the relationship between solutions. None of these algorithms uses distributed global and local search capabilities. Using distributed local search makes search space more efficient.

Meta-heuristic algorithms are used in many machine learning and deep learning applications. The role of meta-heuristic algorithms in combination with deep learning and machine learning methods is to optimize the learning process and discover patterns. For example, in [32], a red deer algorithm method is used for data clustering. The results of the experiments show that in most of the experiments, the proposed method in data clustering is less sensitive to the increase in the number of clusters and data dimensions than similar clustering methods.

In [33], the performance of the evolutionary clustering algorithm STAR for clustering heterogeneous datasets is investigated. This paper compares the Evolutionary Clustering Algorithm (ECA*) star against five traditional and modern clustering algorithms. In this paper, the performance of ECA* is compared with the Genetic Algorithm for Clustering (GENCLUST++), Learning Vector Quantization (LVQ), EM, K-means++(KM++), and K-means (KM). Experiments showed that the performance of the ECA* method is more successful than that of other clustering algorithms. In [34], they presented a sine–cosine firefly algorithm with chaos theory for practical problems. Implementing their algorithm on a set of engineering and practical problems shows the proposed algorithm's efficiency, robustness, and effectiveness compared to similar meta-heuristic methods such as the firefly, particle swarm optimization, and ant colony algorithms. In [35], a deep neural network using hybrid optimization algorithms is presented for brain tumor detection. This research optimizes deep convolutional neural networks with HHO and GWO algorithms. In their method, Otsu's thresholding is used for segmentation in brain tumor diagnosis. The tumor detection method based on DCNN-G-HHO has an accuracy of about 97%. In [36], the star algorithm of the improved evolutionary clustering algorithm is used to diagnose COVID-19 patients. Experiments showed that iECA* was more effective than other algorithms in clustering selected medical disease datasets based on cluster validation criteria. iECA* has less running time and memory consumption for clustering all datasets. The results showed that iECA* performed best in clustering all medical datasets.

In [37], the reduction of formal context in extracting the hierarchy of concepts from objects is presented using the adaptive evolutionary clustering algorithm star. The results show that the adaptive ECA* performs the concept network faster than the other mentioned competitive techniques at different filling ratios. In [38], applications of the Shuffled Frog Leaping Algorithm(SFLA) in different fields are reviewed. In this study, a general review of SFLA is done, and then the types of improved algorithms based on the improvement and combination approach are stated. The primary motivation of this research is to provide helpful information about SFLA to researchers interested in improving or applying the frog-leap algorithm.

Various meta-heuristic algorithms have been presented in the last few years, and in the following section, several related works are reviewed. Meta-heuristic algorithms are widely used in applications such as intrusion detection systems and the detection of phishing attacks and diseases. Therefore, some of these applications are also reviewed.

In [49], the Siberian tiger optimization algorithm is presented to solve optimization problems. In the Siberian tiger algorithm, two main mechanisms of prey search and the fight between bear and tiger are modeled. The advantage of the Siberian tiger optimization algorithm is its simplicity. The main challenge of the Siberian tiger algorithm is the imbalance between heuristic and local search. Another challenge of the Siberian tiger algorithm is the need for more tests on practical applications.

In [50], the optimization algorithm of African vultures is presented based on group hunting between African vultures. In this algorithm, two worthy vultures are responsible for guiding other vultures. The high complexity of the African vulture optimization algorithm(AVOA) is one of the disadvantages of this algorithm. One of the advantages of the AVOA algorithm is that it is more accurate than conventional meta-heuristic methods.

In [51], the white shark optimization (WSO) algorithm is presented based on the behavior of white sharks and hunting them. The white shark optimization algorithm is presented based on the search based on the sense of smell, hearing, and sound sensed by sharks. The white shark algorithm has different phases and uses the velocity vector to update the solutions like the PSO algorithm. One of the challenges of this algorithm is to keep the global-to-local search strategy the same according to the repetition of the WSO algorithm. The rate of convergence to the local optimum in this algorithm can be significant due to the location of the current optimum near the local optimum. This algorithm does not have robust modeling and only tries to find the optimal solution based on the speed of sharks.

In [52], the behavior of coatis in hunting and escaping from predators is used to model the coatis optimization algorithm(COA). The COA algorithm divides the population into two groups, and this strategy prevents all solutions in the same area from being searched. The need for more accurate modeling in the coatis escape phase is one of this algorithm's challenges. Dividing the population into two groups has increased the algorithm's time complexity.

In [53], they presented a feature selection method based on improved hybrid modification optimization for intrusion detection. The proposed approach is implemented on the CEC2021 benchmark dataset, UCI, and security datasets NSL-KDD, WUSTL-IIOT, and HAI datasets. Experimental results show that their attack detection and feature selection method increases the accuracy of attack detection.

In [54], intrusion detection using hybrid meta-heuristic techniques is presented. This paper presents a sine–cosine-whale algorithm as a feature selection algorithm combined with XgBoost for detecting attacks. Experimental results on UNSW-NB15 and CICIDS datasets show that the XGBoost and sine–cosine whale algorithm's accuracy is higher than the standard WOA algorithm.

In [55], they presented a spam detection method with feature selection based on the Sandpiper optimization algorithm. In this paper, after feature selection, Radial Bias Neural Network (RBNN) classifies emails as ham and spam. This paper uses the Enron email dataset and the Spam Assassin dataset to evaluate their method. Experiments showed that the method performs significantly better than existing optimization methods in feature selection.

In [56], a method based on multiple neural networks of improved WOA optimization algorithms is presented to predict COVID-19. In this paper, the improved Whale optimization algorithm uses the learning weights with sinusoidal operators—the SCWOA method used with high accuracy to predict Covid-19.

In [57], the optimization of Harris Hawks is presented for diagnosing COVID-19 based on lung CT scan images. They used the Harris-Hawks optimization(HHO) technique to find the optimal threshold values in Otsu's method. The proposed approach reduces the computational cost function and convergence time (Table 1).

The review of related works shows that various meta-heuristic algorithms are presented to find the optimal solution to engineering problems. Swarm intelligence algorithms have a special place among meta-heuristic methods. In most works related to meta-heuristic algorithms, benchmark functions, and classical engineering problems are used for evaluation. Unlike related works, in the proposed method, for the first time, the behavior of Bitterling fish is modeled to solve optimization problems. The BFO algorithm presents various types of searches to find the optimal solution. Unlike other papers, to evaluate the proposed method for several practical problems in today's world, the criterion for measuring the efficiency of the BFO algorithm is placed. The proposed method solves problems such as network attack detection, phishing attack detection, clustering, and disease detection. The simultaneous presentation of continuous and binary versions of the BFO algorithm for feature selection applications is a distinctive aspect of the proposed method.

## 3 Bitterling fish optimization (BFO)

In this section, the bitterling fish optimization (BFO) is modeled. In this section, the behavior of this fish is described, and then, in the continuation, the behavior of this algorithm is formulated.

### 3.1 Inspire

Fish reproduction occurs as early as possible in the animal kingdom. In mating, male and female species approach each other and then release their sperm and eggs into the water, but this method has a big drawback. Young fish are exposed to various external hazards and may easily fall prey to other animals. In other words, only a few of these eggs turn

**Table 1** Comparison and analysis of related works

| Reference | Method | Problem | Advantage | Disadvantages | Domain |
|---|---|---|---|---|---|
| [32] | Red deer algorithm | Data clustering | High accuracy | More complexity than Kmeans | Clustering + Optimization |
| [33] | Evolutionary Clustering Algorithm Star (ECA*) | Data clustering | More accurate than K-means + + | More complexity than Kmeans | Clustering + Optimization |
| [34] | Sine–cosine firefly algorithm with chaos theory | Engineering problems | Better than PSO,FA,ABC, etc | Test on classic problems | Optimization |
| [35] | DCNN-G-HHO | Deep Learning + Optimization | More accurate than CNN | High complexity | Tumor detection + Optimization |
| [36] | iECA* | Diagnosis of covid 19 | High accuracy | Unbalanced data set | Clustering algorithm + Optimization |
| [37] | ECA* | Data analysis | High speed | Unbalanced data set | Optimization |
| [38] | Shuffled Frog Leaping Algorithm(SFLA) | A review paper | Comprehensive review | - | Optimization |
| [49] | Siberian tiger optimization algorithm | Benchmark function + Engineering problems | Low complexity | imbalance between heuristic and local search | Optimization |
| [50] | African vulture optimization algorithm(AVOA) | Benchmark function + Engineering problems | Strong modeling | High complexity | Optimization |
| [51] | white shark optimization (WSO) algorithm | Benchmark function + Engineering problems | Weak modeling | More accuracy than GA, PSO algorithm | Optimization |
| [52] | Coatis optimization algorithm(COA) | Benchmark function + Engineering problems | Division of work between solutions | oor modeling in the escape phase | Optimization |
| [53] | Feature selection method based on optimization + IDS | Network traffic classification | Reducing traffic dimensions and optimizing errors | Training time overhead | Intrusion detection system + Optimization |
| [54] | Sine–Cosine-WOA + XGBoost | Network traffic classification | More accurate diagnosis than WOA | Unbalanced data set | Intrusion detection system + Optimization |
| [55] | Sandpiper optimization algorithm + RBNN | Network traffic classification | More detection accuracy than RBNN | Unbalanced data set | Intrusion detection system + Optimization |
| [56] | WOA + ANN | Predict COVID-19 | More detection accuracy than WOA | Error rate | Machine learning + Optimization |
| [57] | Harris-Hawks optimization(HHO) + Otsu | Diagnosis of covid 19 | More accurate than Otsu | Medium accuracy | Image Processing + Optimization |

into fish—unfortunately, most fish are hunted by other aquatic animals. Bitterling fish have different mechanisms of reproduction. This fish begins with oysters. The task of finding suitable shells for spawning is the responsibility of the male species. They need to find larger oysters and have more space to hold their eggs. When these fish find one or more victims(oysters), they stand against their enemies. The reason for the aggressive behavior of the male species is that some other male species also tend to use the oysters for themselves. In combat mode, the body color of the male fish (the one that first found the host oyster) darkens and uses everything it can to defend its territory. In the mating stage, a phenomenon called sexual selection takes place. During sexual selection, the female species chooses the male fish according to color and physical strength. According to Fig. 2, female fish usually bring males with larger bodies and more beautiful body colors. At this time, the female fish lays her eggs in the oyster, and then the male fertilizes them with sperm:.

Male fish try to show off their fitness by attracting female species by selecting the oyster they want to reproduce. At this stage, the female species inserts its egg-shaped tubular organ into the oyster to transport them to the host's body space, which is very strange among fish and aquatic animals. After this process, the male enters the operation and transfers his sperm into the oyster body to be fertilized there. When the eggs and larvae hatch, their vulnerability is slightly reduced. The oyster transports the floating food particles in the water into its body. At this stage, baby fish eat water-soluble solids before oysters. The larvae of Bitterling fish also steal oxygen from their hosts and, therefore, have parasitic behavior. These small fish stay in this space long enough to feed on oyster food to prepare for entering the outside world.

## 3.2 Modeling

The following supposition is considered to model the BFO algorithm in solving optimization problems:

- Each solution is a tricky fish.
- Oysters are a function of the target (objective function), and an oyster is more efficient if the objective function displays a more efficient value for it.
- Solutions to the problem of fish are distributed and evaluated in oysters.

Fig. 2 Mating of male and female bitterling fish in oysters

- Each oyster can eject several fish, which can be a local and global search.
- Oyster laurels kill some lesser-known fish or pathogens.

### 3.3 Initial solutions

Each solution to the problem is a bitterling fish or egg, as defined in Eq. (1). We can consider several bitterling fish populations that are produced as a random population, as in Eq. (2):

$$F_i = [F_i^1, F_i^2, F_i^3, \dots, F_i^D] \tag{1}$$

$$F = \begin{bmatrix} F_1^1 & F_1^2 & \dots & F_1^D \\ F_2^1 & F_2^2 & \dots & F_2^D \\ \vdots & \vdots & \vdots & \vdots \\ F_n^1 & F_n^2 & \dots & F_n^D \end{bmatrix} \tag{2}$$

The value of D is the number of dimensions or decision variables of each solution. The value of i is also a solution to the problem. The value of F is a matrix of prime populations of n. $F_i^j$ represents the dimension j of the i-th solution. Population created in the interval [l, u] as in Eq. (3) to create the initial population:

$$F_i^j = l + (u - l).r \tag{3}$$

r is a random number between zero and one.

### 3.4 Evaluate solutions

Each solution to the problem is in the BFO algorithm as a fish. To evaluate each solution, we can pay attention to its oyster selection. Each fish has merit based on the suitable oyster it chooses and attracts other female fish. A target function such as f is used to evaluate each fish or solution. In Eq. (4), the competency of each solution is defined in the competency matrix:

$$Fittness = \begin{bmatrix} f(F_1^1 & F_1^2 & \dots & F_1^D) \\ f(F_2^1 & F_2^2 & \dots & F_2^D) \\ \vdots & \vdots & \vdots & \vdots \\ f(F_n^1 & F_n^2 & \dots & F_n^D) \end{bmatrix} \tag{4}$$

### 3.5 Search and seize oysters

Any solution or fish can search the problem space and find suitable mating oysters. In the proposed method, each fish qualified to be located in an area with more optimal shells. A fish can roam to find an oyster, in which case it targets an oyster and moves towards it. In this case, the target oyster is not noticed by another fish and is captured by this fish. Equation (5) is used to formulate the state of oyster possession:

$$F_i^{t+1} = \begin{cases} J.F_i^t + \left(F^+ - J.F_i^t\right).\delta \ \ r \le P \\ J.\,F_i^t + \left(F^* - J.F_i^t\right).\delta \ \ r > P \end{cases} \tag{5}$$

In this Equation, $F_i^t$ and $F_i^{t+1}$ are a fish's current and new positions in the iteration of t and t + 1, respectively, for the fish or the i-th solution. $F^*$ is also the best oyster or optimal solution. $F^+$ is one of the oysters' worthies of the population that is randomly selected. δ and r is a random number between zero and one. J is the number of steps or the rate at which the fish moves to escape or approach the oyster, and this factor is reduced by iterating the algorithm. The reason for the decrease in parameter J is that after a while, the activity of the male fish decreased because they succeeded in mating. Decreasing J causes global search to decrease to local search over time. To reduce the parameter J, Eq. (6) is used:

$$J(t) = \left( J(1) - \frac{J(1).t}{Maxt} \right).U(t) \tag{6}$$

J (1) is each fish's step and jump value in the initial iterations. The values of t and MaxIt are the current and maximum iteration numbers, respectively. J (t) is the step and jump of each fish to find the oyster in repetition t. The random function U uses Eq. (7) to generate random sequences.

$$U(t+1) = \cos(t \times cos^{-1}(U(t))) \tag{7}$$

U (1) can be considered equal to one. Combining Eqs. (6) and (7) results in Eq. (8).

$$J(t+1) = \left( J(1) - \frac{J(1).t}{Maxt} \right).\cos(t \times cos^{-1}(U(t))) \tag{8}$$

Equation (8) is represented in the diagram of Fig. 3 in terms of the iteration of the proposed algorithm. In the GFO algorithms, the optimal solution is searched over time and in terms of more space repetition around the optimal solution. Therefore, it is essential to reduce the parameter P over time to the second criterion of Eq. (5), with the condition r > P, which is most likely to be done. A function similar to the arctangent behavior, such as Eq. (9), is used for the parameter P. The graph of this function is shown in Fig. 3:

$$P = \left| 1 - \frac{t}{\sqrt{1+t^2}} \right| + \frac{rand}{t^a} \tag{9}$$

The value of t is the repetition counter, and a is equal to the power of reduction. Figure 3 displays the P value based on three powers of 0.1, 0.5, and 0.9. The smaller the power, the lower the P value for iteration. The probability of two types of searches in Eq. (5) decreases with less difference.

### 3.6 Escape and not seize the oyster

In the Escape and not seize the oyster behavior, another fish takes care of the oyster and causes the fish to refrain from approaching the oyster and choose another position. Equation (10) used to escape or accidentally search for a fish that has not succeeded in capturing oysters:

$$F_i^{t+1} = \begin{cases} J.F_i^t + (F^* - J.M).\delta & r \le 0.5 \\ l + (u - l).\delta & r > 0.5 \end{cases} \tag{10}$$

The value of M is equal to the average position of the fish swarm. In this regard, since the escape, a bitterling fish can search the space between the average and the

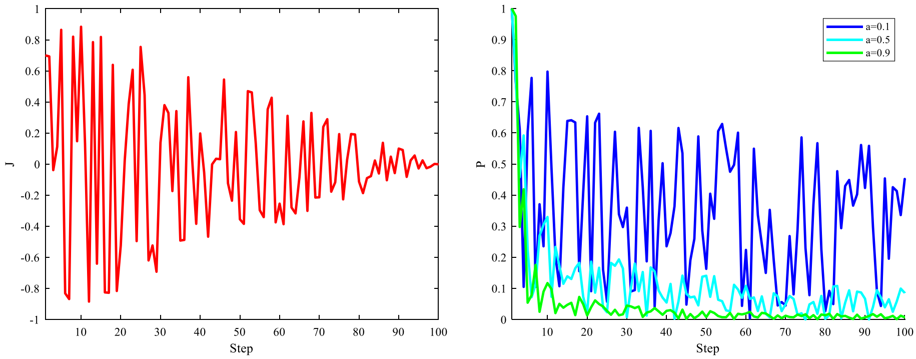


**Fig. 3** From left to right, decrease the value of parameter J and parameter P according to the iteration of the proposed algorithm

optimal or search the random space problems. Equation (11) used to calculate the population gravity point:

$$M = \frac{\sum_{i=1}^{n} F_i^t}{n} \tag{11}$$

## 3.7 Production

After finding an oyster, a male fish can attract the female fish and lay eggs in the oyster, and the male can fertilize the eggs to create new fish around the current fish, as in Equation (12):

$$F_i^{t+1} = F_i^t + R * rand(0,1) \tag{12}$$

R is the radius of distribution of the fish around the shell inside them. This radius has an initial value in the range [0,2]. Usually, the value of this parameter is initially considered equal to 2 and decreases over time according to the iteration of the BFO algorithm.

## 3.8 Hunting fish

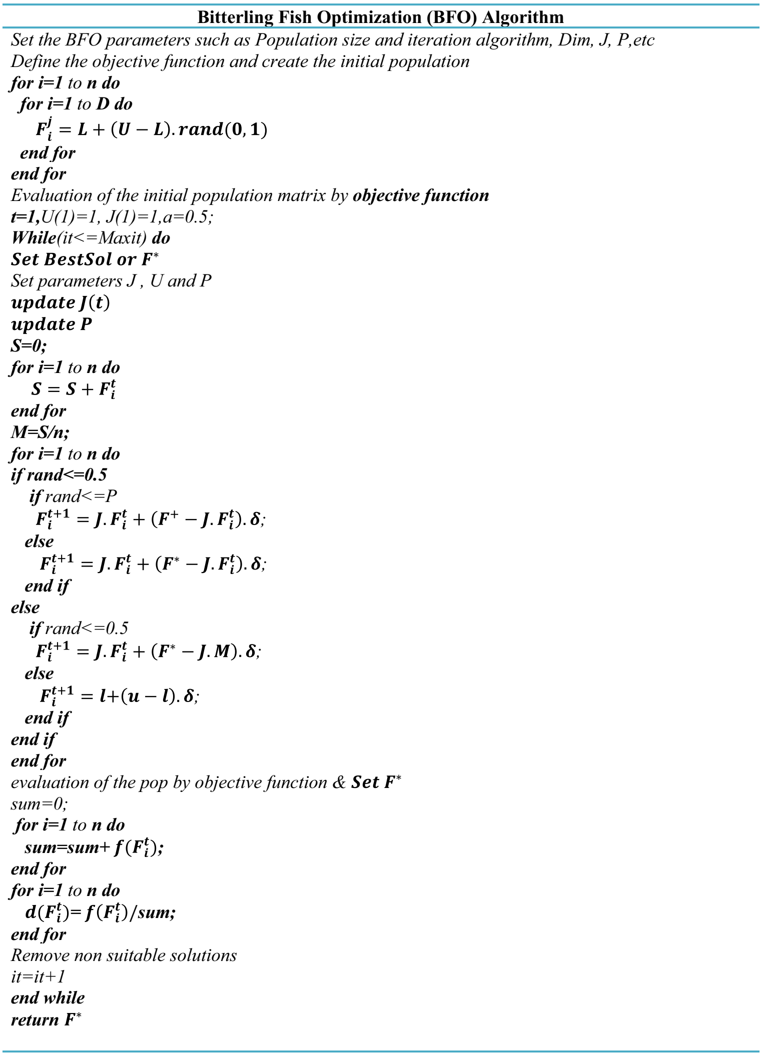
An oyster can hold several baby fish; some weaker ones can be hunted and killed by oysters or predators. The probability of losing a solution can be considered inversely proportional to its merit. Suppose the problem is of the minimization type, and the probability of losing a fish is equal to Eq. (13):

$$d(F_i^t) = \frac{f(F_i^t)}{\sum_{i=1}^{n} f(F_i^t)} \tag{13}$$

The probability of eliminating the solution $F_i^t$ is equal to $d(F_i^t)$, and $f(F_i^t)$ is the objective function value for this solution.

### 3.9 Pseudo-code of the proposed method

The pseudo-code of the BFO algorithm for finding the optimal solution is shown in Fig. 4. In the BFO algorithm, several solutions are created in the random problem space, and by the BFO algorithm, these solutions are sent to the optimal to achieve the optimal solution. In the proposed method, phases include creating the initial population, updating the parameters, evaluating solutions, searching and capturing oysters, escaping from other fish, moving towards fish accumulation, production, reproduction of fish, and extinction of undesired fish.



```
Bitterling Fish Optimization (BFO) Algorithm
Set the BFO parameters such as Population size and iteration algorithm, Dim, J, P,etc
Define the objective function and create the initial population
for i=1 to n do
 for i=1 to D do
   $F_i^j = L + (U - L).rand(0,1)$
 end for
end for
Evaluation of the initial population matrix by objective function
t=1,U(1)=1, J(1)=1,a=0.5;
While(it<=Maxit) do
Set BestSol or $F^*$
Set parameters J , U and P
update $J(t)$
update P
S=0;
for i=1 to n do
   $S = S + F_i^t$
end for
M=S/n;
for i=1 to n do
if rand<=0.5
   if rand<=P
   $F_i^{t+1} = J.F_i^t + (F^+ - J.F_i^t).\delta;$
  else
   $F_i^{t+1} = J.F_i^t + (F^* - J.F_i^t).\delta;$
  end if
else
   if rand<=0.5
   $F_i^{t+1} = J.F_i^t + (F^* - J.M).\delta;$
  else
   $F_i^{t+1} = l+(u - l).\delta;$
  end if
end if
end for
evaluation of the pop by objective function & Set $F^*$
sum=0;
 for i=1 to n do
  sum=sum+ $f(F_i^t)$;
end for
for i=1 to n do
  $d(F_i^t)= f(F_i^t)/sum;$
end for
Remove non suitable solutions
it=it+1
end while
return $F^*$
```

Fig. 4 Pseudo-code of Bitterling Fish Optimization (BFO)

### 3.10 Proposed intrusion detection system

One application of meta-heuristic algorithms is to use them in feature selection and reduce input, which increases the accuracy of learning methods. In the proposed method to reduce the error of distinguishing normal from abnormal traffic in the network, a binary version of the BFO algorithm with the artificial neural network is present in this section. Penetration into computer networks is a communication challenge. One way to deal with intrusion is to use network intrusion detection systems (IDS). This section presents an intrusion detection system to deal with attacks using the BFO algorithm and MLP neural network.

#### 3.10.1 Proposed framework for IDS

The framework of the proposed IDS for detecting network intrusion is shown in Fig. 5:

In the proposed IDS, the optimal feature vector for network traffic selection. The optimal feature vector for learning artificial neural networks is considered a classification tool. The role of the bitterling fish optimization(BFO) algorithm is to select features and update feature vectors to reduce the error of the MLP neural network. The steps of the proposed IDS in detecting network intrusion using the BFO algorithm are as follows:

- Network traffic is pre-processed and normalized.
- Network traffic is divided into two categories: training and testing. Training traffic is used to train the neural network, and feature selection and test traffic are used to evaluate the proposed method.
- Balancing the dataset makes the number of attack samples equal to the number of standard samples. Balancing the dataset increases the accuracy of the learning model in detecting attacks. Game theory and the GAN [58] are used to balance the data set in the proposed method.
- Feature vector coding of a member of the BFO algorithm.
- Each feature vector has zero and one component, indicating not selecting a feature.
- Several random feature vectors are generated as members of the BFO algorithm.
- The binary map of each feature vector is applied to the network traffic data set, and the neural network classification input is reduced.
- Each feature vector is evaluated using the average error of distinguishing normal from abnormal traffic and the number of selected features.
- Any network traffic-related feature vector that minimizes the objective function is more qualified to detect intrusion.
- The BFO algorithm updates the feature vectors in each iteration.
- Each retrieval vector is converted from continuous to discrete space with transfer functions.
- Feature vectors are updated in each iteration, and the optimal feature vector is updated in each iteration.
- In the last iteration, the optimal feature vector is used to reduce the dimensions of network traffic for classifying traffic.

#### 3.10.2 Network traffic preprocessing

The network traffic dataset is considered as the input of the proposed method. The output error of the proposed method depends on the input type, so the proposed method's traffic input is pre-processed. The pre-processed data and network traffic reduce classification errors

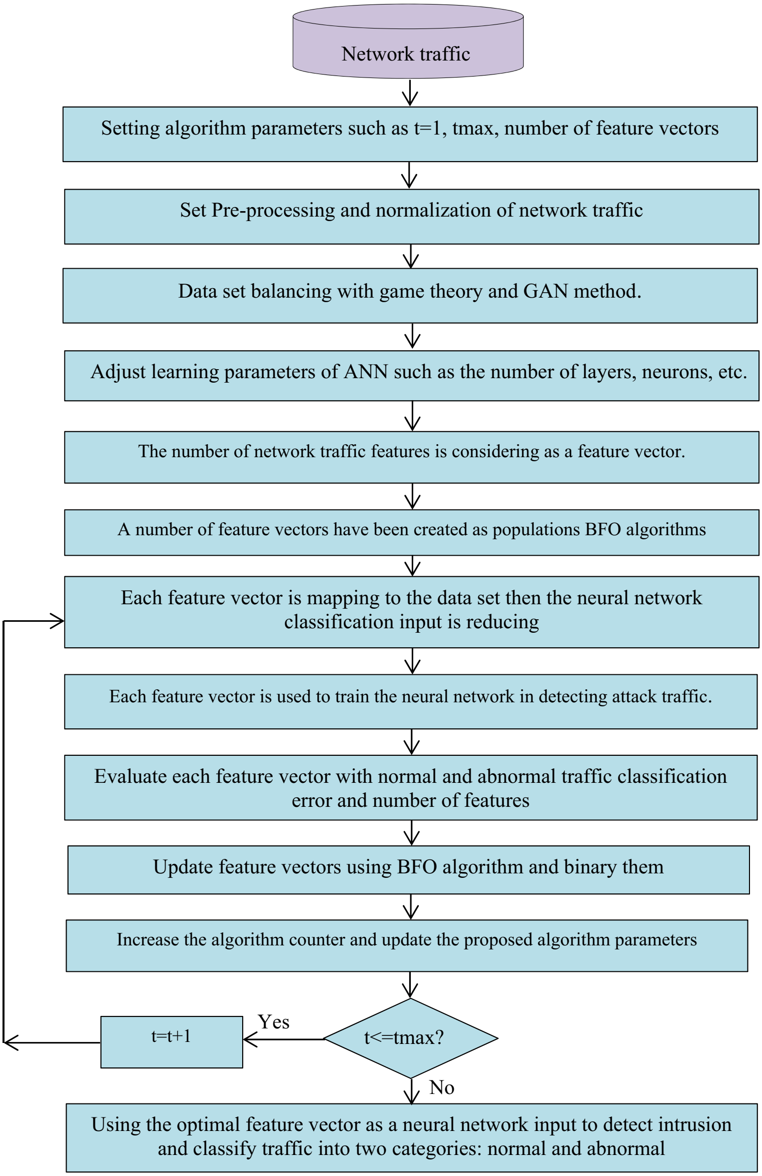

Network traffic
Setting algorithm parameters such as t=1, tmax, number of feature vectors
Set Pre-processing and normalization of network traffic
Data set balancing with game theory and GAN method.
Adjust learning parameters of ANN such as the number of layers, neurons, etc.
The number of network traffic features is considering as a feature vector.
A number of feature vectors have been created as populations BFO algorithms
Each feature vector is mapping to the data set then the neural network classification input is reducing
Each feature vector is used to train the neural network in detecting attack traffic.
Evaluate each feature vector with normal and abnormal traffic classification error and number of features
Update feature vectors using BFO algorithm and binary them
Increase the algorithm counter and update the proposed algorithm parameters
t<=tmax?
Yes
t=t+1
No
Using the optimal feature vector as a neural network input to detect intrusion and classify traffic into two categories: normal and abnormal

**Fig. 5** Proposed framework for detecting network intrusion

in data mining and machine learning. Normalization is the primary step in the pre-processing of input data. Normalization causes the amount of change in all features used in a data set to change at a fixed interval. The range of changes in the features of a network traffic data set is varied and needs to be normalized. Some features of the network traffic data set have a small range of changes, which causes their role in learning and classification by the artificial neural network to be ignored. Equation (14) in the range [a, b] normalizes a network traffic feature. For normalization in the range [0,1] and [-1, + 1], Eqs. (15) and (16) are used:

$$N\left(F_i\right) = a + \frac{F_i - min}{max - min}(b - a) \tag{14}$$

$$N\left(F_i\right) = \frac{F_i - min}{max - min} \tag{15}$$

$$N\left(F_i\right) = 2 \times \frac{F_i - min}{max - min} - 1 \tag{16}$$

An abnormal value is a network traffic property equal to $F_i$. The normalized value of a traffic property is $N\left(F_i\right)$. A network traffic feature's maximum and minimum values, such as i-th traffic, are max and min, respectively.

### 3.10.3 Reduce dimensions and select features

Network traffic data are the input of the proposed IDS. This paper uses the NSL_KDD dataset. The data set is pre-processing in the pre-learning phase. In the next phase, selecting the feature and reducing the traffic dimensions is necessary. In Eq. (15), the proposed objective function, which includes the mean error and the number of selected features, is used to evaluate the feature vectors:

$$Cost(F_i) = \alpha.E(F_i) + \beta.\frac{\|F_i\|}{D} \tag{17}$$

$E(F_i)$ is the error of detecting network intrusion or error of classifying normal to abnormal traffic of a feature vector such as $F_i$ and is calculated according to Eq. (16). $\|F_i\|$ is the size of a feature vector. D is the number of possible features in a data set. $Cost(F_i)$ is the objective function of an attribute vector. α and β are two random numbers between zero and one, and their sum, like Eq. (18), is one:

$$\alpha + \beta = 1 \tag{18}$$

Each vector is a feature of a member of the BFO algorithm. Several random feature vectors are considered members of the initial population of the BFO algorithm. Each of them evaluates with an objective function such as Eq. (19):

$$Cost\begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{bmatrix} = \begin{bmatrix} Cost\,(F_1^1 & F_1^2 & \dots & F_1^D) \\ Cost(F_2^1 & F_2^2 & \dots & F_2^D) \\ \vdots & \vdots & \vdots & \vdots \\ Cost(F_n^1 & F_n^2 & \dots & F_n^D) \end{bmatrix} \tag{19}$$

The role of the BFO algorithm in the proposed vector update method is the optimal feature for the MLP classification technique.

## 4 Analysis

Several evaluation functions are used for analysis to measure the accuracy of the BFO algorithm. The BFO algorithm is compared with several meta-heuristic methods with a swarm approach using benchmark functions. In the third phase, the proposed method is used in an operational application such as intrusion detection systems and compared with similar methods.

## 4.1 Analysis of BFO algorithm

In this section, using the benchmark functions of the BFO algorithm, the evaluation and its error are calculated and compared with similar methods.

### 4.1.1 Benchmark functions

The valuation functions used in this section are cost functions that aim to find their global minimum. Tables 2 and 3 show a set of uni-model, complex, and hybrid evaluation functions for evaluating the proposed method. The uni-model functions are less complex than multi-model functions. Some of these functions have local optimizations that may be involved.

### 4.1.2 Parameters

In implementations, the initial population size of the algorithms is 15, and the maximum number of iterations is 100. Each experiment repeats 30 times. Table 4 shows the parameters for implementing the BFO algorithm and other algorithms.

### 4.1.3 Convergence analysis

Suppose that the number of repetitions and the population size of the proposed method are equal to t and n, respectively. In the proposed method, there are two loops: one for updating the population and one for updating the steps of the BFO algorithm. According to the proposed code's network, the algorithm's time complexity is n*n + t.(n + n + n + n). In other words, the complexity of the proposed method to find optimal solutions is O(n*n + t.(n + n + n + n)), which is equivalent to O(n*n + 4t.n).

**Table 2** Uni-model benchmark functions

| Function | Dim | Range | $f_{\min}$ |
|---|---|---|---|
| $F_1(X) = \sum_{i=1}^{n} X_i^2$ | 30 | [− 100,100] | 0 |
| $F_2(X) = \sum_{i=1}^{n} \lvert X_i \rvert + \prod_{i=1}^{n} \lvert X_i \rvert$ | 30 | [− 10,10] | 0 |
| $F_3(X) = \sum_{i=1}^{n} \left( \sum_{j-1}^{i} X_j \right)^2$ | 30 | [− 100,100] | 0 |
| $F_4(X) = max\left\{ \lvert X_i \rvert, 1 \le i \le n \right\}$ | 30 | [− 100,100] | 0 |
| $F_5(X) = \sum_{i=1}^{n-1} \left[ 100\left(X_{i+1} - X_i^2\right)^2 + \left(X_i - 1\right)^2 \right]$ | 30 | [− 30,30] | 0 |
| $F_6(X) = \sum_{i=1}^{n} \left( \left[ X_i + 0.5 \right] \right)^2$ | 30 | [− 100,100] | 0 |
| $F_7(X) = \sum_{i=1}^{n} i X_i^4 + \mathrm{random}[0, 1)$ | 30 | [− 1.28,1.28] | 0 |
| $F_8(X) = X_1^2 + 10^6 \sum_{i=2}^{n} X_i^2$ | 30 | [− 100,100] | 0 |
| $F_9(X) = \sum_{i}^{n} \lvert X_i \rvert^{i+1}$ | 30 | [− 100,100] | 0 |
| $F_{10}(X) = \sum_{i=1}^{n} X_i^2 + \left( \sum_{i=1}^{n} 0.5 X_i \right)^2 + \left( \sum_{i=1}^{n} 0.5 X_i \right)^4$ | 30 | [− 100,100] | 0 |

**Table 3** Multi-model benchmark functions

| Function | Dim | Range | $f_{\min}$ |
|---|---|---|---|
| $F_{11}(X) = \sum_{i=1}^{n} - X_i sin\left(\sqrt{\lvert X_i \rvert}\right)$ | 30 | [− 500,500] | -418*Dim |
| $F_{12}(X) = \sum_{i=1}^{n} \left[X_i^2 - 10cos\left(2\pi X_i\right) + 10\right]$ | 30 | [− 5.12,5.12] | 0 |
| $F_{13}(X) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} X_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos\left(2\pi X_i\right)\right) + 20 + e$ | 30 | [− 32,32] | 0 |
| $F_{14}(x) = \frac{1}{4000}\sum_{i=1}^{n} X_i^2 - \prod_{i=1}^{n} cos\frac{X_i}{\sqrt{i}} + 1$ | 30 | [− 600,600] | 0 |
| $F_{15}(X) = \frac{\pi}{n}\left\{10sin\left(\pi y_1\right) + \sum_{i=1}^{n-1}\left(y_i - 1\right)^2\left[1 + 10sin2\left(\pi y_{i+1}\right)\right] + \left(y_n - 1\right)^2\right\} + \sum_{i=1}^{n} u\left(X_i, 10, 100, 4\right)$ | 30 | [− 50,50] | 0 |
| $y_i = 1 + \frac{X_i + 1}{4}$ | | | |
| $u\left(X_i, a, k, m\right) = \begin{cases} k\left(X_i - a\right)^m x_i > a \\ 0 - a < x_i < a \\ k\left(-X_i - a\right)^m x_i < a \end{cases}$ | | | |
| $F_{16}(x) = 0.1\left\{sin2\left(3\pi x_1\right) + \sum_{i=1}^{n}\left(x_i - 1\right)^2\left[1 + sin2\left(3\pi x_i + 1\right)\right] + \left(x_n - 1\right)^2\left[1 + sin2\left(2\pi x_n\right)\right]\right\} + \sum_{i=1}^{n} u\left(X_i, 5, 100, 4\right)$ | 30 | [− 50,50] | 0 |
| $F_{17}(X) = \sum_{i=1}^{n-1}\left(100\left(X_i^2 - X_{i+1}\right)^2 + \left(X_i - 1\right)^2\right)$ | 30 | [− 100,100] | 0 |
| $F_{18}(X) = sin2\left(\pi w_1\right) + \sum_{i=1}^{n-1}\left(w_i - 1\right)^2\left[1 + 10sin2\left(\pi w_i + 1\right)\right] + \left(w_n - 1\right)^2\left[1 + sin2\left(2\pi w_n\right)\right]$ | 30 | [− 100,100] | 0 |
| $w_i = 1 + \frac{x_i - 1}{4}, \forall i = 1, \ldots, n$ | | | |

GWO, WOA, BOA, HHO, and BWO algorithms are used to implement and compare the BFO algorithm. The proposed algorithm and other swarm intelligence(SI) algorithms use benchmark functions as the objective function. The optimal calculation error is calculated and compared according to the iteration of the algorithms. The global optimal calculation error in the last iteration is used to compare the methods. Figures 6 and 7 show the two Ackley and Griewank functions in MATLAB, respectively. The purpose of displaying these benchmark functions is to show the complexity of benchmark functions for search by meta-heuristic algorithms. Figure 8 shows the error rate of the BFO algorithm and other algorithms on 6 sample benchmark functions. In the error calculations of the optimal solution in terms of iteration, it is observed that the BFO method is reducing the error compared to similar swarm methods. However, it is more than other methods in the BFO algorithm. The error reduction of the proposed algorithm increases in the last iterations. This examination shows that the population algorithm leads its solutions with more convergence toward optimal solutions. The placement of the convergence diagram of the cunning BFO algorithm below other convergence diagrams shows that the proposed method has a greater tendency to reduce the optimal calculation error in terms of iteration.

In the last iteration, the BFO algorithms have less global optimal calculation error than other swarm intelligence algorithms. In some experiments, the convergence diagrams do not decrease error after multi-step, and the slope of the diagram is close to zero. This state happens when the meta-heuristic algorithm is caught in the local optimum. Experiments show that the BFO is less caught in the local optimum, and in most experiments, it finds the global optimum.

### 4.1.4 Rank analysis in error calculation

In the Fig. 9, the rank of the proposed algorithm and other algorithms are compared:

One way to analyze meta-heuristic algorithms is to use their rankings to find the universal optimal calculation error. The BFO, GWO, WOA, BOA, HHO, and BWO algorithms are implemented on benchmark functions to calculate the ranking. In the next step, the rank of each algorithm in finding the error is obtained. In implementations, if an algorithm calculates the minimized optimal error calculation, it is assigned a ranking of one. If it has the worst optimal error calculation, its rank is equal to 6 because the number of algorithms compared is equal to 6. Rank tests are used in evaluations. The average rating is calculated on 18 benchmark functions. Ranking in BFO, BOA, GWO, HHO, BWO, and WOA algorithms is 1.86, 4.24, 2.87, 3.14, 3.88, and 2.34. Analysis and evaluation show that in most

**Table 4** List of parameters in implementations

| Algorithm | Parameter | Value | Parameter | Value |
|---|---|---|---|---|
| BFO | R | 0 | U(1) = J(1) | 1 |
| WOA | B | 1 | C | [0,2] |
| | | | l | [-1, + 1] |
| BOA | C | 0.1 | p | 0.01 |
| | E(0) | [0,1] | Escaping | [0,2] |
| HHO | q | rand | - | - |
| BWO | Pp | 0.6 | *pm* | 0.4 |
| | | | *CR* | 0.44 |
| GWO | r1 | rand | *a*0 | 2 |
| | r2 | rand | *C* | [0,2] |

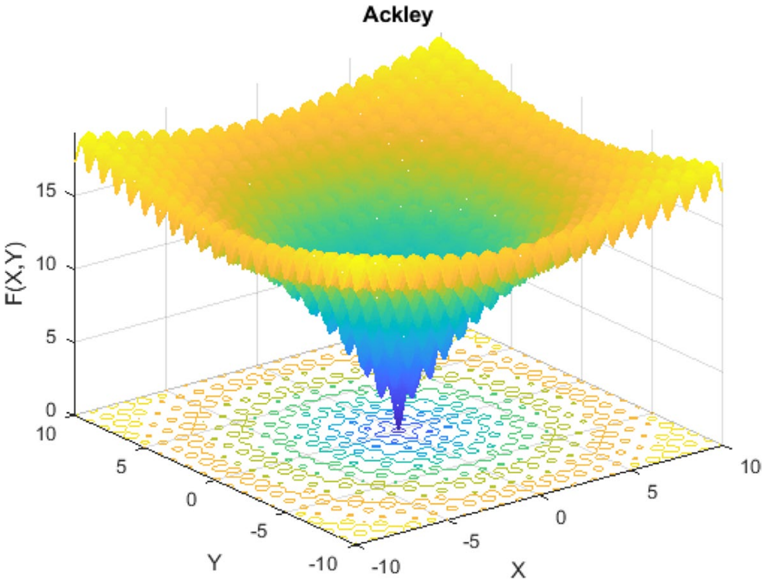


**Fig. 6** Ackley function in 3D dimensional space

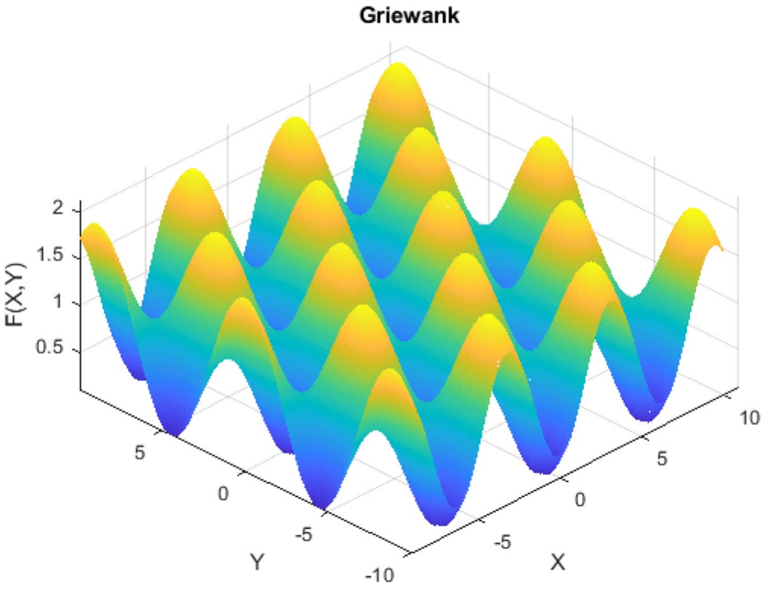


**Fig. 7** Griewank function in 3D dimensional space

cases, the proposed method performed better than others in obtaining the minimum global calculation error and getting a more optimal rank (lower number). The BOA has the worst performance in the mean error-index, and its error is higher than that of other methods. The WOA algorithm is a competitor to the proposed algorithm and, in most experiments, ranks second in error detection.

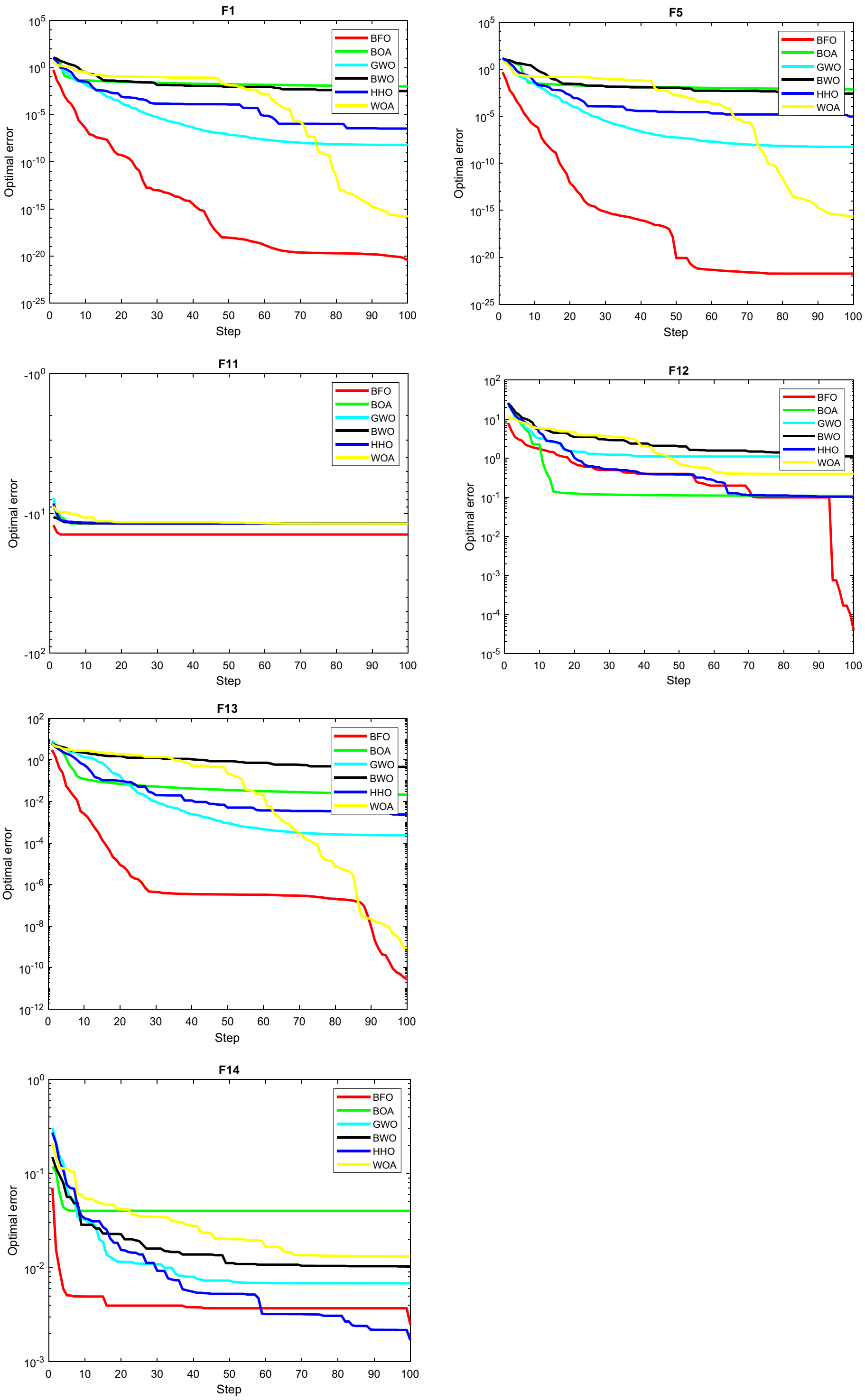


**Fig. 8** Comparison of the BFO error on several functions with other meta-heuristic algorithms

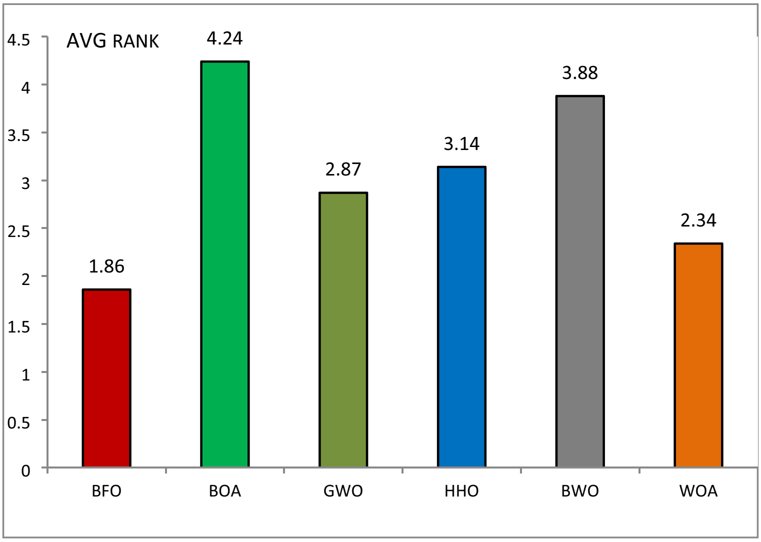


**Fig. 9** Comparison of the rank of algorithms in the error-index

### 4.1.5 Stability analysis

The stability index plays an essential role in analyzing meta-heuristic algorithms. Sustainability means how the algorithm is in finding the optimal solution. One way to measure the stability of algorithms is to use the standard deviation (STD) of experiments to find the optimal solution. Any meta-heuristic algorithm with a low standard deviation in finding the optimal solution is more efficient regarding stability. Figure 10 shows the rank of the proposed algorithm and other algorithms in the stability index or standard deviation.

According to the rank tests of BFO, GWO, WOA, BOA, HHO, and BWO algorithms on uni, complex, and combined evaluation functions in the standard deviation index are equal to 1.93, 3.87, 2.18, 3.56, 3.42, and 2.29. The BFO algorithms has the lowest standard deviation to find the optimal solution among the compared methods. The lower value of the standard deviation in the BFO algorithm compared to other meta-heuristic algorithms indicates the more excellent stability of the BFO algorithm in finding the optimal solution. The GWO and WOA algorithms are following in the SI algorithms regarding stability.

### 4.1.6 Optimal local convergence analysis

The convergence of local optimizations is a good indicator for analyzing meta-heuristic algorithms. Any algorithm less likely to converge in local optimizations shows a high intelligence in finding the optimal solution. Figure 11 shows the percentage of probability of being caught in the local optimal complex benchmark functions with local optimum.

Percentage convergence to local optimal in BFO, GWO, WOA, BOA, HHO, and BWO algorithms is 8.63%, 15.67%, 10.25%, 11.68%, 13.42%, and 9.87%, respectively. The analysis shows that the BFO algorithms converge the solutions to the local optimal. The worst performance to converge in local optimization solutions is also related to the BOA than the BWO.

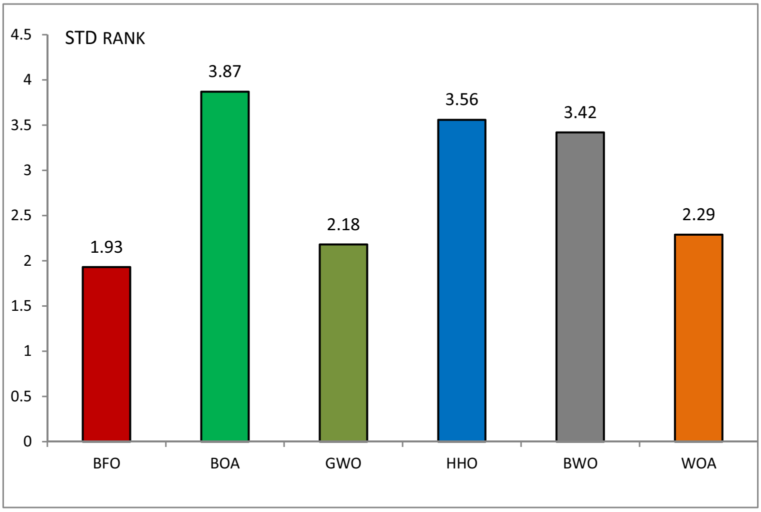


**Fig. 10** Comparison of the rank of algorithms in the standard deviation index

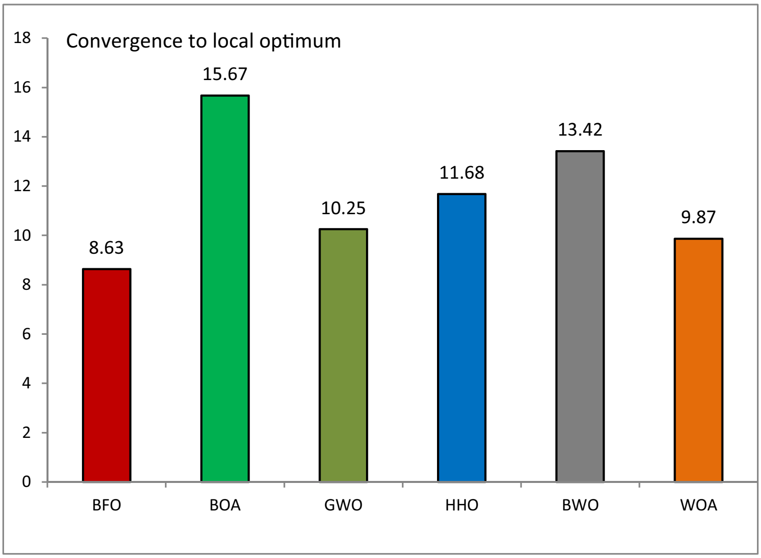


**Fig. 11** Comparison of local optimal convergence percentage

### 4.1.7 Runtime analysis

The execution time of the BFO algorithm with other algorithms is shown on the opioid benchmark functions of Fig. 12. The BFO algorithm is an algorithm with low complexity and can be used to solve optimization problems. One way to measure the efficiency of the BFO algorithm is to use the execution time of the meta-heuristic algorithms to find the optimal solution and compare them. Execution of the BFO algorithm and meta-heuristic

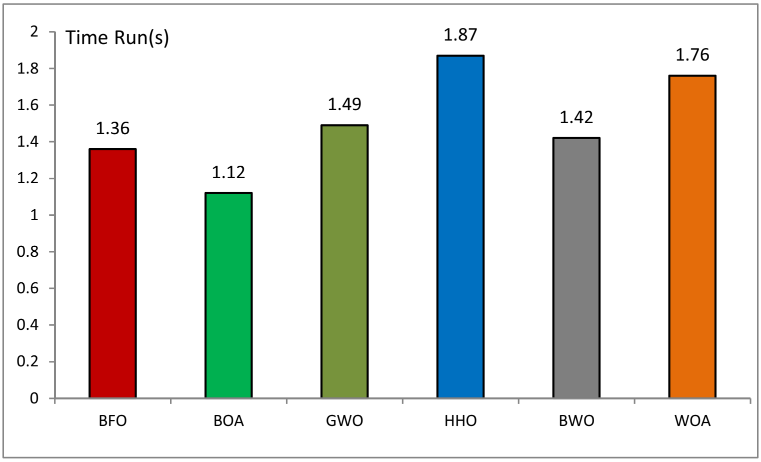


**Fig. 12** Comparison of execution time of the proposed algorithm and other swarm intelligence algorithms

algorithms on benchmark functions and calculation of execution time can be suitable criteria for predicting the time complexity of algorithms. The evaluations use a system with a 5-core Intel processor and 6 GB of memory. Execution times of BFO, GWO, WOA, BOA, HHO, and BWO algorithms are equal to 1.36, 1.12, 1.49, 1.87, 1.42, and 1.76, respectively.

Experiments show that the BOA algorithm has the shortest execution time because it has only two relationships. Although the BOA algorithm has less execution time, its error rate is higher than other methods. The BFO algorithm is faster in terms of execution time index to find the optimal solution, at least faster than the methods of the GWO, WOA, HHO, and BWO algorithms. The execution time complexity of the BFO algorithm is only higher than the BOA algorithm, but it has less time complexity than other meta-heuristic algorithms. The advantage of the BFO algorithms, besides the acceptable execution time, is less error in the optimal calculation than other algorithms.

## 4.2 Functional analysis

In the previous Section of Algorithm 4.1, the BFO algorithm was evaluated with benchmark functions. Experiments showed that the BFO algorithm has a lower error in global optimal calculation than premium meta-heuristic methods such as WOA, HHO, and GWO. In this section, instead of analyzing and testing some classical problems, the BFO algorithm is compared with similar methods on more practical problems, such as network attack detection, phishing attack detection, and disease detection.

### 4.2.1 Network intrusion detection

In this section, the BFO algorithm is used for feature selection in determining the type of traffic in terms of normal and abnormal. An MLP artificial neural network is used to classify and evaluate the feature vector. For implementation, the number of feature vectors is 15, the number of iterations is 50, and the number of experiments is 20. The multilayer artificial neural network has two hidden layers and an output that determines the type of traffic. Each hidden layer has 20 hidden neurons, and the type of function of the neurons is sigmoid. The NSL_KDD dataset is used to analyze and evaluate the proposed method for detecting network intrusion. Each column in this dataset represents a feature such as service type, protocol type, source address, destination address, number of packets sent, etc. Each data set record has 41 features. The 42 columns also specify the type of attack and the output. This dataset contains 24 attacks divided into five categories: standard, DOS, R2L, U2R, and Prob. Some columns' values are numerical, and others' values are non-numerical and symbolic, which in the preprocessing phaseconverted into a numerical format suitable for machine learning [31].

This section uses the BFO algorithm for feature selection in combination with a multilayer neural network to detect network attacks. Accuracy, sensitivity, and precision indicators are used to evaluate the proposed intrusion detection system. The formula of evaluation indicators for intrusion detection into the network, such as accuracy, sensitivity, and precision, are presented in Eq. (20), (21), and (22), respectively:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{20}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{21}$$

$$Precision = \frac{TP}{TP + FP} \tag{22}$$

TP, TN, FP, and FN indices are using in the evaluations. The concepts of each are as follows:

- TP: The traffic is in the attack category, and the proposed method correctly classifies it in the attack category..
- TN: The traffic is in the normal category, and the proposed method correctly classifies it in the normal category.
- FP: The traffic is in the normal category, and the proposed method misclassifies it in the attack category.
- FN: The traffic is in the attack category, and the proposed method misclassifies it in the normal category.

The proposed IDS is an efficient, accurate, intelligent intrusion detection system. The proposed method has an intelligent ability to reduce the dimensions and feature selection in intrusion detection. Combine this feature with neural network learning to penetrate the network more accurately.

Experiments show that in the proposed method, if the population of feature vectors is 15 and the number of iterations is 50, then the accuracy, sensitivity, and precision for penetration detection are 98.86%, 98.67%, and 98.45%. In the Fig. 13, the accuracy, precision, and sensitivity index of the proposed method in classifying normal to abnormal traffic compared with data mining methods such as NNIA, DT, RF XGBoot, and CNN.

The accuracy index of the proposed method, NNIA, DT, RF, XGboot, and CNN is 98.86%, 94.60%, 89.80%, 90.30%, 90.10%, and 79.20%, respectively. The accuracy index of the proposed method in detecting network intrusion is more accurate than NNIA, DT, RF, Xgboot, and CNN algorithms. The precision index of the proposed method, NNIA, DT, RF, Xgboot, and CNN for classification standard to abnormal traffic is 98.67%, 95.11%,

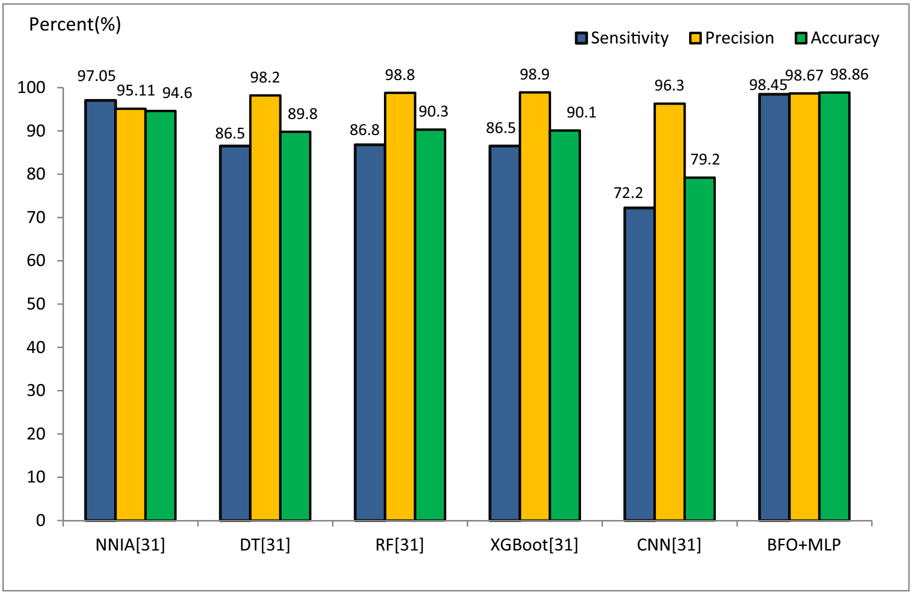


**Fig. 13** Comparison of accuracy, sensitivity, and precision of the proposed intrusion detection system in the NSL-KDD dataset

98.20%, 98.80%, 98.90%, and 96.30%, respectively. The proposed method in the index shows only 0.23% less than the Xgboot, but it has a higher precision index in diagnosis than other methods. Experiments show that the sensitivity index of the proposed method, NNIA, DT, RF, Xgboot, and CNN for classifying normal to abnormal traffic is 98.45%, 97.05%, 86.50%, 86.80%, 86.50%, and 72.20%, respectively. The proposed method has a higher sensitivity index than the compared methods in detecting network intrusion.

Datasets play a essential role in evaluating feature selection and machine learning methods. Another useful dataset that is widely used like the NSL-KDD dataset is the UNSW-NB15 dataset. The UNSW-NB15 dataset was compiled by researchers at the Australian Center for Cyber Security (ACCS) with the IXIA tool. The complete UNSW-NB15 dataset contains 2.5 million traffic related to computer networks. The UNSW-NB15 dataset provides one normal class and nine attack classes including Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode and Worms. The UNSW-NB15 dataset has 49 features. Figures 14 and 15 compare the order of the accuracy index and the number of features selected in the UNSW-NB15 data set.

Experiments show that the accuracy of the proposed method(BFO + MLP) in this data set is 96.72%. From the combined feature selection methods DNN + Statistical [39], GA [40], DT + XGBoost [41], PSO-CO-GA [42], RF DNN [43] and DT-REECV [44] is more accurate in detecting penetration. The UNSW-NB15 dataset has 49 features, and experiments show that the proposed method selects 16 features on average. According to the comparison of Fig. 15, the proposed method reduces more dimensions than the feature selection methods DNN + Statistical, GA, DT + XGBoost, PSO-CO-GA, and RF + DNN have. The proposed method only selects more features than the DT-REECV method, but the accuracy of the proposed method in detecting attacks is about 1.42% higher than the DT-REECV method.

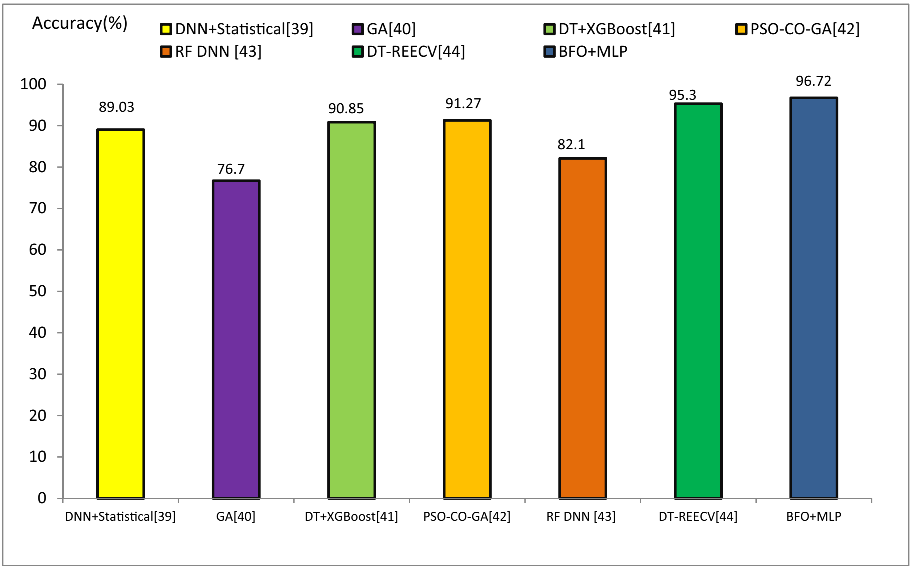


**Fig. 14** Comparing the accuracy of the proposed method (BFO + MLP) in intrusion detection with similar methods in the UNSW-NB15 dataset

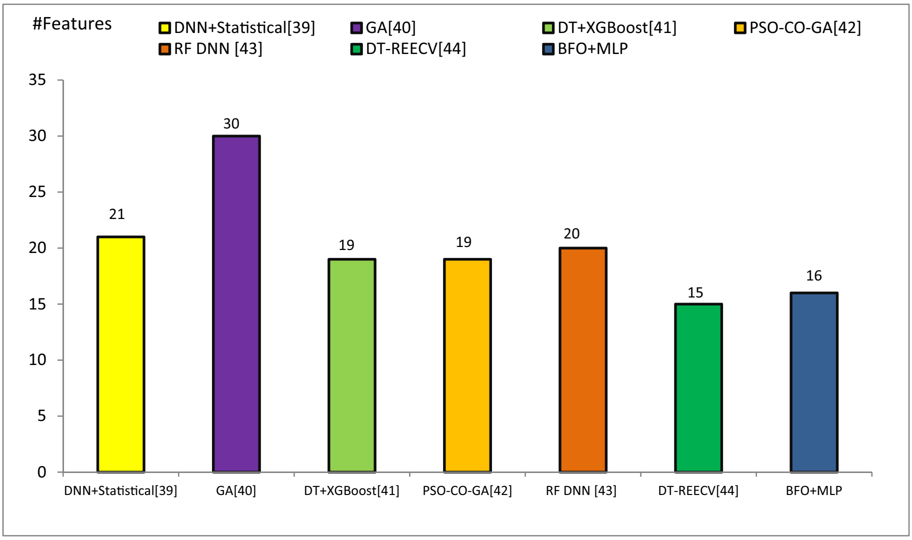


**Fig. 15** Comparison of the number of selected features in the UNSW-NB15 dataset

### 4.2.2 Detection of phishing attacks

Phishing attacks are one of the challenges of computer networks. In phishing attacks, fake pages are loaded instead of legitimate pages, and users' information is stolen through fake pages. This section evaluates the proposed method or BFO + MLP for detecting phishing attacks. The UCI dataset [48] is used to detect phishing attacks. The phishing dataset has 31 features, 30 of which are input, and the 31-th feature is output. The output attribute has two values: legitimate and phishing. This dataset has more than 11 thousand records. In the tests, 70% of the samples are educational, and the other 30% are of the test type. The type of validation used is cross-validation, and K = 10 is considered in the validation. In this section of the optimization problem, the selection of features is essential to recognize web pages with a small error. The proposed method reduces the detection error of phishing attacks by optimizing neural network inputs. Classification indices such as accuracy, sensitivity, and precision are used to evaluate the proposed method in this problem. In Fig. 16, the accuracy of the proposed method for detecting phishing attacks is calculated as 98.24%. The accuracy of methods such as Ensemble bagging [45], ForestPA-PWDM xxxxxxx[46], DNN + Adam [47], and Random Forest [48] are 95.4%, 97.4%, 96%, and 97.3%, respectively. The proposed method is more accurate in detecting phishing attacks due to optimizing MLP neural network inputs with the BFO algorithm. The proposed method provides fewer and more optimal features for neural network learning due to the use of swarm intelligence of the BFO algorithm. Neural network learning on important features increases the accuracy of attack detection.

### 4.2.3 Clustering

Diagnosing diseases such as heart disease, COVID-19, and kidney disease and their clustering based on patient data is a practical issue. One method for analyzing the information on patients with Covid-19 is data clustering. In the proposed method, the BFO algorithm is used in combination with the K-means(KM) algorithm for optimal clustering of cluster centers. The data set [36] is used in these experiments to analyze the proposed method. This dataset contains information on patients with COVID-19, liver disorders, diabetes, heart disease, and kidney disease. The role of the BFO algorithm in this section, unlike Sections 4.2.1 and 4.2.2, is not feature selection but finding cluster centers for clustering optimization. Table 5 shows the memory consumption (MC)

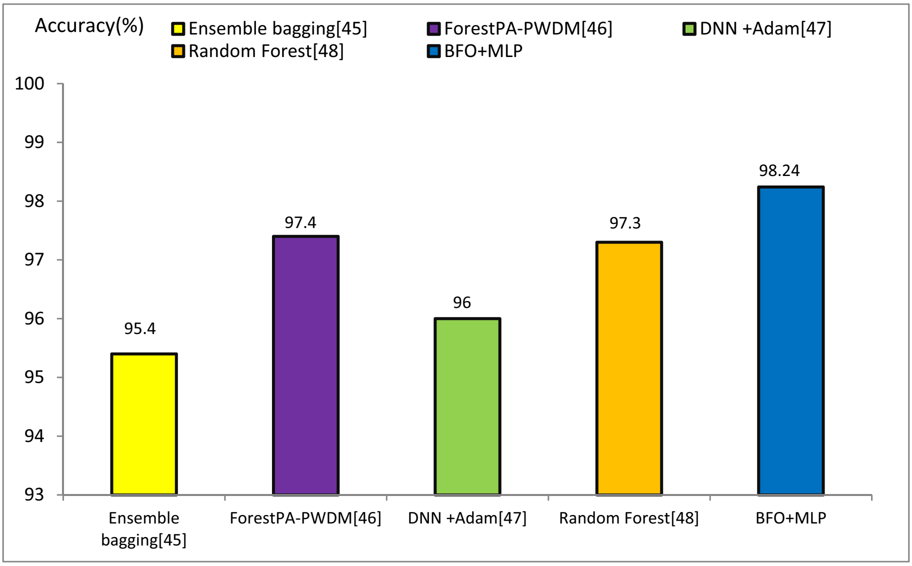


**Fig. 16** Comparison of the proposed method in detecting phishing attacks with similar methods

**Table 5** Comparison of memory consumption of the proposed method with similar methods in disease diagnosis

| Datasets | iECA* | ECA* | G+ + | Deep KNN | LVQ | SVM | ANN | KNN | BFO | Winner |
|---|---|---|---|---|---|---|---|---|---|---|
| COVID-19 | 86.134 | 96.283 | 142.928 | 104.918 | 151.9282 | 98.283 | 102.817 | 114.273 | 82.538 | BFO |
| Liver disorder | 19.481 | 23.168 | 31.384 | 26.184 | 29.474 | 26.833 | 29.857 | 28.437 | 21.86 | iECA* |
| Diabetes | 36.244 | 39.172 | 51.761 | 44.183 | 50.819 | 49.128 | 41.347 | 47.127 | 32.531 | BFO |
| Kidney disease | 18.177 | 21.661 | 29.384 | 17.987 | 23.817 | 24.981 | 19.384 | 18.341 | 16.529 | BFO |
| Heart disease | 42.387 | 46.126 | 56.128 | 48.128 | 54.128 | 53.383 | 45.651 | 50.841 | 45.823 | iECA* |

of the proposed algorithm with methods: iECA*, ECA*, GENCLUST + + (G + +), Deep KNN, LVQ, SVM, ANN, and KNN. The conditions of the tests are also considered similar to [36]. It is not logical to compare the execution time of the proposed method with the research results [36] due to the use of different hardware in this manuscript and study [36]. Experiments show that the proposed method(BFO + Kmeans) on the dataset of COVID-19, diabetes, and kidney disease has less memory consumption than other methods such as iECA*, ECA*, GENCLUST + + (G + +), Deep KNN, LVQ, SVM, ANN, KNN. In the liver disorders and heart disease dataset, the iECA method is more successful than the proposed method. In general, among the compared methods, the iECA* method and BFO require less memory than other methods in clustering patients. Less memory consumption indicates the speed of execution of these algorithms in medical data clustering.

### 4.2.4 Classic problems

This section compares the proposed method for the welding beam problem, Spring Design, as two classical optimization problems. Two classic problems of welding beam and spring design problems are formulated in [34]. In Figs. 17 and 18, the cost function of the proposed method in these two problems is compared with the CSCF, FF, SCA, PSO, and ABC methods. Experiments showed that in the welding beam problem, the value of the cost function in the proposed method is 1.712, and it offers a lower cost than the FF, SCA, PSO, and ABC methods, but it costs a little more than the CSCF method. In the Spring Design problem, the proposed method costs 0.020284 and offers a lower cost structure than the CSCF, FF, SCA, PSO, and ABC methods.

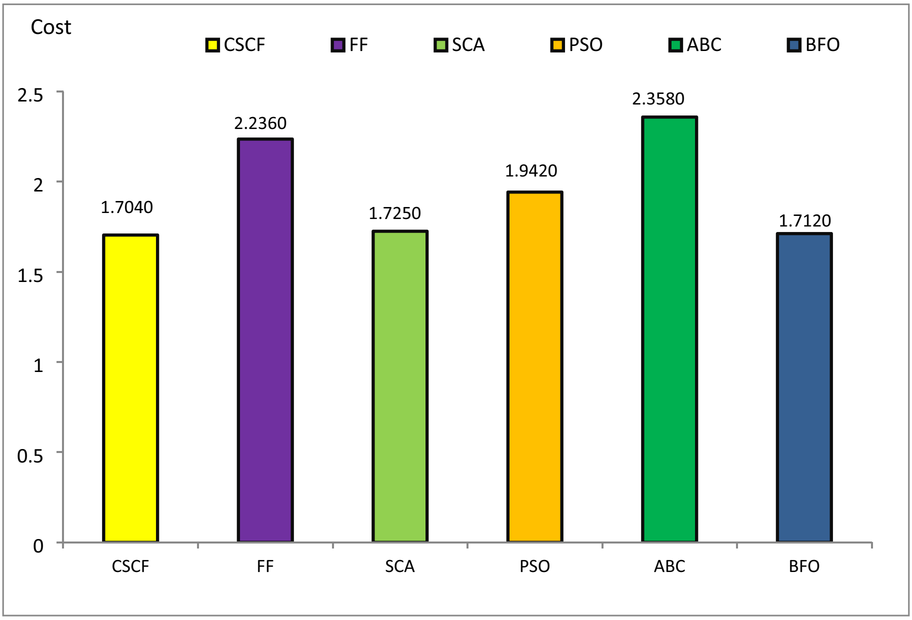


**Fig. 17** Comparing the cost of the proposed method with similar optimization methods in the welding beam problem

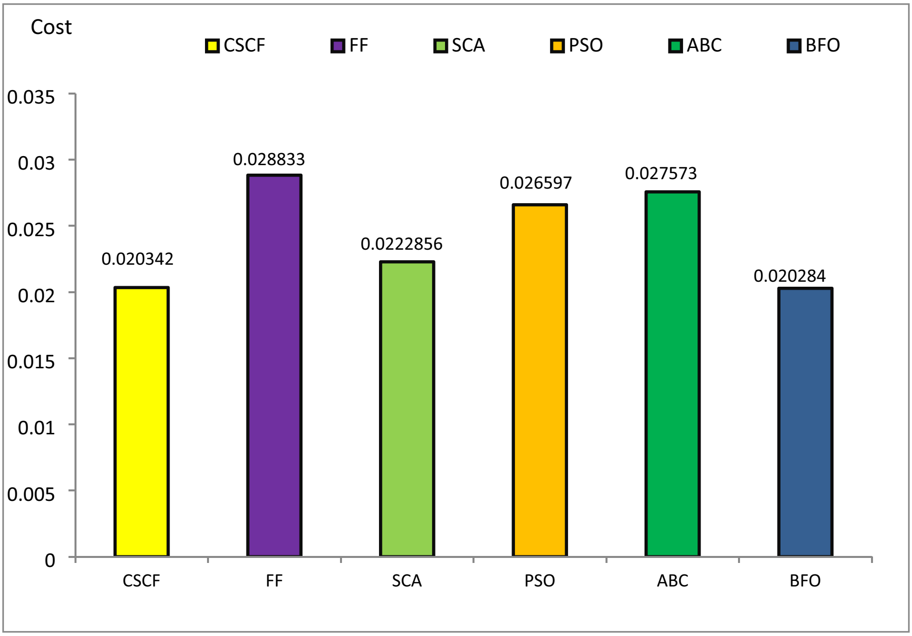


**Fig. 18** Comparing the cost of the proposed method with similar optimization methods in the Spring Design problem

## 5 Conclusion

In this manuscript, a new meta-heuristic algorithm based on the behavior of bitterling fish is introduced and formulated. Several search strategies, such as oyster finding, oyster competition, mating, and escape modeling, are used in the bitterling fish optimization algorithm. An advantage of the BFO algorithm is that it uses the most optimal solution and, at the same time, uses the position of qualified fish in the population. The proposed method focuses on searching for optimal solutions and other suitable solutions and is, therefore, less prone to converging to local optimizations. Experiments on several cec functions show that the BFO algorithm is at least one of the gray wolf optimization algorithms, the whale optimization algorithm, the butterfly optimization algorithm, the Harris hawks optimization algorithm, and the black widow has less error in finding the optimal solution. Experiments have shown that the stability of the BFO algorithm is at least higher than that of the GWO, WOA, BOA, HHO, and BWO algorithms. The runtime index only has more runtime than the BOA algorithm. The proposed method has the lowest probability of locally optimal convergence among the compared methods, and this probability is equal to 9.87%.

This paper also presents a binary version of the BFO algorithm. The BFO algorithm's binary version is used to select the feature for detecting network intrusion. Experiments confirm that the proposed algorithm's network intrusion detection accuracy is higher than intrusion detection methods such as NNIA, DT, RF, XGBoot, and CNN. The most challenging meta-heuristic approach is the time overhead to classify network traffic.

BFO algorithm for attack detection is evaluated with the UNSW-NB15 dataset. The role of the BFO algorithm is feature selection in the UNSW-NB15 dataset and attack detection using an MLP neural network. In the UNSW-NB15 dataset, the BFO algorithm, in combination with MLP, managed to get more accuracy for detecting attacks than DNN + Statistical, GA, DT + XGBoost, PSO-CO-GA, RF DNN, DT-REECV methods. The BFO algorithm evaluated the phishing dataset. The tests showed that the accuracy of the proposed method in detecting phishing attacks is 98.24%, and it is more accurate than ensemble bagging, ForestPA-PWDM DNN + Adam, and Random Forest methods. The comparisons showed that, in most cases, the proposed method offers a lower cost than the CSCF algorithms of FF, SCA, PSO, and ABC in solving optimization problems. The advantages of the BFO algorithms are as follows:

- Accurate modeling of Bitterling fish behavior
- More accuracy in finding the optimal solution in NP-Hard than standard meta-heuristic methods.
- Reduction of convergence to local optimal solutions.
- Success in practical use in intrusion detection systems, phishing attack detection, clustering, and accurate disease diagnosis.

Like any meta-heuristic method, the proposed method has a time overhead in finding the optimal solution. Considering that the BFO algorithm is a meta-heuristic algorithm based on population, it can be presented parallel to increase its acceleration and speed. In the future work, a version of network traffic analysis with CUDA architecture and GPUs will be implemented. In future work, a parallel BFO algorithm will be developed to provide an accelerated version using the GPU architecture.

**Funding** Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK).

**Data availability** Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

## Declarations

**Conflict of interests** The authors declare that there is no conflict of interest regarding the publication of this paper.

## References

1. Chang GW, Chinh NC (2020) Coyote optimization algorithm-based approach for strategic planning of photovoltaic distributed generation. IEEE Access 8:36180–36190
2. Narmatha C, Eljack SM, Tuka AARM, Manimurugan S, Mustafa M (2020) A hybrid fuzzy brainstorm optimization algorithm for the classification of brain tumor MRI images. J Ambient Intell Human Comput 1–9. https://link.springer.com/article/10.1007/s12652-020-02470-5?wt_mc=Internal.Event.1.SEM.ArticleAuthorOnlineFirst&utm_source=ArticleAuthorOnlineFirst&utm_medium=email&utm_content=AA_en_06082018&ArticleAuthorOnlineFirst_20200815&error=cookies_not_1312-4a37-8a3c-8773851fee1e

3. Hemeida A et al (2020) Implementation of nature-inspired optimization algorithms in some data mining tasks. Ain Shams Eng J 11(2):309–318
4. Sharma P et al (2020) Artificial plant optimization algorithm to detect heart rate & presence of heart disease using machine learning. Artif Intell Med 102:101752
5. Xu J et al (2020) Biomechanical performance design of joint prosthesis for medical rehabilitation via generative structure optimization. Comput Methods Biomech Biomed Engin 23(15):1163–1179
6. Shabani A et al (2020) Search and rescue optimization algorithm: A new optimization method for solving constrained engineering optimization problems. Expert Syst Appl 161:113698
7. Abou Khamis R, Shafiq MO, Matrawy A (2020) Investigating resistance of deep learning-based ids against adversaries using min-max optimization. In: ICC 2020–2020 IEEE International Conference on Communications (ICC). IEEE
8. Ezugwu AE et al (2020) A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems. Neural Comput Appl 32(10):6207–6251
9. Družeta S et al (2020) Introducing languid particle dynamics to a selection of PSO variants. Egyptian Inform J 21(2):119–129
10. Ebadinezhad S (2020) DEACO: Adopting dynamic evaporation strategy to enhance ACO algorithm for the traveling salesman problem. Eng Appl Artif Intell 92:103649
11. Kumar V, Kumar D (2021) A systematic review on firefly algorithm: past, present, and future. Arch Comput Methods Eng 28(4):3269–3291
12. Nadimi-Shahraki MH, Taghian S, Mirjalili S (2021) An improved grey wolf optimizer for solving engineering problems. Expert Syst Appl 166:113917
13. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
14. Heidari AA et al (2019) Harris hawks optimization: Algorithm and applications. Futur Gener Comput Syst 97:849–872
15. Rezaei H, Bozorg-Haddad O, Chu X (2018) Grey wolf optimization (GWO) algorithm. Advanced Optimization by Nature-Inspired Algorithms. Springer, pp 81–91
16. Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70
17. Hayyolalam V, Kazem AAP (2020) Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. Eng Appl Artif Intell 87:103249
18. Too J, Rahim Abdullah A (2020) Binary atom search optimisation approaches for feature selection. Connect Sci 32(4):406–430
19. Rbouh I, El Imrani AA (2014) Hurricane-based optimization algorithm. AASRI Procedia 6:26–33
20. Das B, Mukherjee V, Das D (2020) Student psychology based optimization algorithm: A new population based optimization algorithm for solving optimization problems. Adv Eng Softw 146:102804
21. Morales-Castañeda B et al (2019) An improved Simulated Annealing algorithm based on ancient metallurgy techniques. Appl Soft Comput 84:105761
22. Falcón-Cardona JG, Coello CAC (2020) Indicator-based multi-objective evolutionary algorithms: A comprehensive survey. ACM Comput Surv (CSUR) 53(2):1–35
23. Cuevas E, Fausto F, González A (2020) New advancements in swarm algorithms: operators and applications. Springer
24. Alirezanejad M et al (2021) GSA-LA: gravitational search algorithm based on learning automata. J Exp Theor Artif Intell 33(1):109–125
25. Sajedi H, Mohammadipanah F (2020) Developed optimization algorithms based on natural taxis behavior of bacteria. Cogn Comput 12(6):1187–1204
26. Al-Sorori W, Mohsen AM (2020) New Caledonian crow learning algorithm: A new metaheuristic algorithm for solving continuous optimization problems. Appl Soft Comput 92:106325
27. Kaveh A, Eslamlou AD (2020) Water strider algorithm: A new metaheuristic and applications. in Structures. Elsevier
28. Zhang X et al (2020) Novel biogeography-based optimization algorithm with hybrid migration and global-best Gaussian mutation. Appl Math Model 86:74–91
29. Rawat PS et al (2020) Optimize task allocation in cloud environment based on big-bang big-crunch. Wireless Pers Commun 115(2):1711–1754
30. Xu Y et al (2020) An Adaptive Water Wave Optimization Algorithm with Enhanced Wave Interaction. In: 2020 IEEE Congress on Evolutionary Computation (CEC). IEEE
31. Wei W et al (2020) A multi-objective immune algorithm for intrusion feature selection. Appl Soft Comput 95:106522
32. Moghadam P, Ahmadi A (2023) A novel two-stage bio-inspired method using red deer algorithm for data clustering. Evol Intell 1–18. https://link.springer.com/article/10.1007/s12065-023-00864-w
33. Hassan BA, Rashid TA, Mirjalili S (2021) Performance evaluation results of evolutionary clustering algorithm star for clustering heterogeneous datasets. Data Brief 36:107044

34. Hassan BA (2021) CSCF: a chaotic sine cosine firefly algorithm for practical application problems. Neural Comput Appl 33(12):7011–7030
35. Qader SM, Hassan BA, Rashid TA (2022) An improved deep convolutional neural network by using hybrid optimization algorithms to detect and classify brain tumor using augmented MRI images. Multimedia Tools and Applications 81(30):44059–44086
36. Hassan BA, Rashid TA, Hamarashid HK (2021) A novel cluster detection of COVID-19 patients and medical disease conditions using improved evolutionary clustering algorithm star. Comput Biol Med 138:104866
37. Hassan BA, Rashid TA, Mirjalili S (2021) Formal context reduction in deriving concept hierarchies from corpora using adaptive evolutionary clustering algorithm star. Complex Intelligent Syst 7(5):2383–2398
38. Maaroof BB, Rashid TA, Abdulla JM, Hassan BA, Alsadoon A, Mohamadi M, Mirjalili S (2022) Current studies and applications of shuffled frog leaping algorithm: a review. Arch Comput Methods Eng 2(1):1–16
39. Thakkar A, Lohiya R (2023) Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. Inform Fusion 90:353–363
40. Liu Z, Shi Y (2022) A hybrid IDS using GA-based feature selection method and random forest. Int J Mach Learn Comput 12(02):43–50
41. Kasongo SM, Sun Y (2020) Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. J Big Data 7:1–20
42. Tama BA, Comuzzi M, Rhee KH (2019) TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. IEEE Access 7:94497–94507
43. Eunice AD, Gao Q, Zhu MY, Chen Z, Na LV (2021) Network anomaly detection technology based on deep learning. In: 2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC) (pp 6–9). IEEE
44. Awad M, Fraihat S (2023) recursive feature elimination with cross-validation with decision tree: feature selection method for machine learning-based intrusion detection systems. J Sens Actuator Netw 12(5):67
45. Ubing AA, Jasmi SKB, Abdullah A, Jhanjhi NZ, Supramaniam M (2019) Phishing website detection: An improved accuracy through feature selection and ensemble learning. Int J Adv Comput Sci Appl 10(1). https://www.semanticscholar.org/paper/Phishing-Website-Detection%3A-An-Improved-Accuracy-Ubing-Jasmi/8b97ae0ba551083056536445d8c2507bb94b959f?p2df
46. Alsariera YA, Elijah AV, Balogun AO (2020) Phishing website detection: forest by penalizing attributes algorithm and its enhanced variations. Arab J Sci Eng 45:10459–10470
47. Lakshmi L, Reddy MP, Santhaiah C, Reddy UJ (2021) Smart phishing detection in web pages using supervised deep learning classification and optimization technique adam. Wireless Pers Commun 118(4):3549–3564
48. Alnemari S, Alshammari M (2023) Detecting phishing domains using machine learning. Appl Sci 13(8):4649
49. Trojovský P, Dehghani M, Hanuš P (2022) Siberian tiger optimization: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems. Ieee Access 10:132396–132431
50. Abdollahzadeh B, Gharehchopogh FS, Mirjalili S (2021) African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. Comput Ind Eng 158:107408
51. Braik M, Hammouri A, Atwan J, Al-Betar MA, Awadallah MA (2022) White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowl-Based Syst 243:108457
52. Dehghani M, Montazeri Z, Trojovská E, Trojovský P (2023) Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. Knowl-Based Syst 259:110011
53. Ye Z, Luo J, Zhou W, Wang M, He Q (2024) An ensemble framework with improved hybrid breeding optimization-based feature selection for intrusion detection. Future Gener Comput Syst 151:124–136. https://www.sciencedirect.com/science/article/pii/S0167739X23003667?casa_token=RsCYFYaSbOIAAAAA:QBWc3yrgv9Wjasq9Oq0r6ddhxjtZq94j5gSjDRw9QkAtDvQyJ6inYjUflQLDfTzbgosljtXT1Cez
54. Mohiuddin G, Lin Z, Zheng J, Wu J, Li W, Fang Y, ... Zeng X (2023) Intrusion detection using hybridized meta-heuristic techniques with Weighted XGBoost Classifier. Exp Syst Appl 120596
55. Alsenani TR, Ayon SI, Yousuf SM, Anik FBK, Chowdhury MES (2023) Intelligent feature selection model based on particle swarm optimization to detect phishing websites. Multimed Tools Appl 1–33. https://link.springer.com/article/10.1007/s11042-023-15399-6
56. Amutha T, Geetha S (2023) Automated Spam Detection Using Sandpiper Optimization Algorithm-Based Feature Selection with the Machine Learning Model. IETE J Res 1–8

57. Qu Z, Li Y, Jiang X, Niu C (2023) An innovative ensemble model based on multiple neural networks and a novel heuristic optimization algorithm for COVID-19 forecasting. Expert Syst Appl 212:118746
58. Yuan L, Yu S, Yang Z, Duan M, Li K (2023) A data balancing approach based on generative adversarial network. Futur Gener Comput Syst 141:768–776

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.