Electronic Research Archive

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

ERA, 32(1): 418–444.
DOI: 10.3934/era.2024021
Received: 14 November 2023
Revised: 12 December 2023
Accepted: 19 December 2023
Published: 28 December 2023

Research article

Applying modified golden jackal optimization to intrusion detection for Software-Defined Networking

Feng Qiu, Hui Xu* and Fukui Li

School of Computer Science, Hubei University of Technology, Wuhan 430068, China

* **Correspondence:** Email: xuhui@hbut.edu.cn.

Abstract: As a meta-heuristic algorithm, the Golden Jackal Optimization (GJO) algorithm has been widely used in traditional network intrusion detection due to its ease of use and high efficiency. This paper aims to extend its application to the emerging field of Software-Defined Networking (SDN), which is a new network architecture. To adapt the GJO for SDN intrusion detection, a modified Golden Jackal Optimization (mGJO) is proposed to enhance its performance with the use of two strategies. First, an Elite Dynamic Opposite Learning strategy operates during each iteration to find solutions opposite to the current global optimal solutions, which increases population diversity. Second, an updating strategy based on the Golden Sine II Algorithm is utilized in the exploitation phase to update the position information of the golden jackal pairs, which accelerates the search for the best feature subset indexes. To validate the feasibility of the mGJO algorithm, this paper first assesses its optimization capability using benchmark test functions. Then, four UCI datasets and the NSL-KDD dataset are used to test the classification capability of the mGJO algorithm and its application in traditional network intrusion detection. Furthermore, the InSDN dataset is used to validate the feasibility of the mGJO algorithm for SDN intrusion detection. The experimental results show that, when the mGJO algorithm is applied to SDN for intrusion detection, the various indexes of classification and the selection of feature subsets achieve better results.

Keywords: Software-Defined Networking; intrusion detection; feature selection; Golden Jackal Optimization; Elite Dynamic Opposite Learning; Golden Sine II Algorithm

1. Introduction

Over the past few years, a new type of network architecture called as Software-Defined Networking (SDN) has been introduced. SDN centralizes control in a network controller and divides it from the data plane, which provides a fresh way of looking at network architecture, thus adding more programmability [1]. The Computer Numerical Control (CNC) separation can solve the limitations of traditional networks and provide more advanced control and services. Still, it also faces challenges of scalability, consistency, and reliability due to its physically centralized controller [2,3].

Because of the convenience that SDN provides to computer networks, its security is getting more and more attention. One of the most important solutions to SDN security issues is network intrusion detection [4]. Badotra et al. [5] analyzed the vulnerability of two types of SDN controllers that are ONOS and ODL, for DDoS attacks, arranged SDN controllers on a distributed cloud and performed DDoS attacks and analyzed the vulnerability, and finally concluded that the OPL-3 node clusters are the most effective. Nadeem et al. [6] conducted a comparative analysis of feature selection classifiers and machine learning classifiers by combining Random Forest with SDN controllers to detect SDN attacks and trained the model by feature elimination pattern to validate the excellence of the proposed method. Wang et al. [7] applied Convolutional Neural Networks (CNN) to intrusion detection in SDN, optimizing CNN using the firefly algorithm after increasing population diversity and proposed an SDN with loT (SD-loT) framework which ensured the accuracy of normal traffic because it detected DDoS attacks on SD-loT.

Network Intrusion Detection System (NIDS) involves the detection and mitigation of malicious network intrusions by monitoring and analyzing network traffic, system logs, and other data. As society progresses, there is a rising emphasis on network security, leading to the growing significance of NIDS as a research focal point within the area of network security [8]. Traditional NIDS and SDN Intrusion Detection Systems, also referred to as non-SDN-IDS and SDN-IDS, makeup NIDS.

For both non-SDN-IDS and SDN-IDS, there are many redundant or irrelevant features in high-dimensional data, which may affect the classification speed or even reduce the classification accuracy, so data dimension is a necessary part of NIDS [9]. As a result, feature selection emerges as a significant approach to enhance the performance of NIDS, which aims to decrease the dimension of the feature space, constructing a more comprehensible model. It usually leads to better learning performance by reducing the dimension [10].

Friha et al. [11] proposed a NIDS for both SDN and non-SDN with a federated learning strategy and tested the classification accuracy by detecting attacks on three different datasets such as CIC-IDS2018, MQTTest, and InSDN. Elsayed et al. [12] proposed an improved long and short-term based automatic two-stage NIDS for both SDN and no-SDN with ToN-loT and InSDN dataset to validate the accuracy and precision of the Intrusion Detection System. To propose the SDN-SlowRate-DDoS dataset and validate the generated dataset using the Long Short-Term Memory (LSTM) model of the attack flow, Yungaicela-Naula et al. [13] conducted 23 different experiments in which they set up the network topology and established parameters such as attacker, victim, and so on. They then restricted the connection either temporarily or permanently depending on the intensity of the assault flows. It is finally concluded that SDN-IDS works best when the attacker is 4 and the victim is 2. To detect DDoS attacks based on VANET for SDN, Anyanwu et al. [14] proposed a Radial Basis Function (RBF) kernel for Support Vector Machine (SVM) and an exhaustive parameter search method called Grid Search Cross Verification. Then the superiority of the RBF kernel is verified by comparing the RBF, Poly, Linear, and Sigmod kernel, followed by tuning the RBF kernel with C, V, testing the time complexity, and comparing it with other Machine Learning (ML) algorithms at CICIDS 2019 to validate the superiority of the model. Gu et al. [15] proposed a hybrid feature selection algorithm based on Hadoop to select the most effective set of features. Then, a semi-supervised learning K-means algorithm was incorporated to use DARPA DDoS, CAIDA "DoS attack 2007" and CICIDS "DoS attack 2017" to verify the superiority of the algorithm.

The Golden Jackal Optimization (GJO) algorithm is a newly proposed meta-heuristic algorithm developed by Chopra [16] in 2022. Due to its simplicity, ease of implementation, and robustness, the GJO has been widely used in various fields. For instance, the usage of the GJO was recommended by Ghandourah et al. [17] to improve the network model and utilize it for the prediction of the performance of aluminum and polycarbonate air cavity solar stills. Najjar et al. [18] combined the LSTM model with the GJO and applied it to predict the tribological properties of alumina-coated silver-reinforced copper nanocomposites. Devi et al. [19] proposed a binary LEO-GJO algorithm by integrating LEO escape factors into the GJO and validated it on the UCI dataset for feature selection. Das et al. [20] used the GJO for feature selection and validated it using the dataset from the PROMISE repository. They also compared different classifiers, such as K-Nearest Neighbors (KNN), Decision Tree (DT), Naive Bayes (NB), and Quadratic Discriminant Analysis (QDA). Arini et al. [21] combined the GJO with the Joint Opposite Strategy (GJO-JOS) and compared it with the GJO-JOS algorithm under CEC 2017 using multiple algorithms to verify its superiority. They also performed the Wilcoxon rank sum test for the GJO-JOS, illustrating the engineering usability of the improved algorithm. Lu et al. [22] proposed a Quantum Evolutionary Golden Jackal Optimization Algorithm (QEGJOA), which was able to obtain a deployment scheme for sensors by increasing the duty cycle. The complexity of the algorithm was reduced while the accuracy was improved.

Our prior works [23–26] include four parts as follows. First, we proposed an integrated optimization algorithm incorporating a weighted k-nearest neighbor feature selection combination strategy, the ability of the proposed algorithm for non-SDN-IDS was tested using the KDD Cup99 dataset, which was divided into four groups conditioned on weighted KNN with feature selection, and finally it was verified that the proposed method can elevate the value of the accuracy of non-SDN-IDS. Second, we proposed a jumping spider optimization algorithm that combines the Harris Hawk algorithm with small-hole imaging and tested it with the KDD Cup99 and UNSW-NB15 datasets, which verified that the proposed method can improve the accuracy as well as the convergence speed of non-SDN-IDS. Third, we proposed a Harris Hawk algorithm optimized by Sin chaotic mapping to adjust the initialization of the Control Placement Problems (CPP) scheme. After obtaining the CPP format, the diversity of CPP is increased by using the Cauchy variation to find the CPP with the Pareto front, and experiments show that the CPP after the proposed scheme is more robust. Fourth, we proposed an improved butterfly optimization algorithm combined with black widow optimization for the feature selection of NIDS. The UNSW-NB15 dataset is selected for binary classification and multi-classification simulation experiments. The experimental results show that the proposed algorithm can enhance the performance of the feature selection model in non-SDN-IDS and reduce feature dimensions.

Therefore, considering the above researches, a GJO integrating the Elite Dynamic Opposite Learning (EDOL) strategy and the Golden Sine II Algorithm (GoldenSAII) is applied to the feature selection problems of SDN-IDS. The EDOL during the initialization stage helps to enhance the quality of solutions while avoiding being trapped in local optimal solutions, and the GoldenSAII is brought in the exploitation phase to enhance the convergence capability in later phases of the GJO. This paper uses the non-SDN-IDS dataset to compare with existing algorithms to validate the feasibility of the modified algorithm on NIDS, and further conducts comparative experiments with the SDN dataset, comparing to current conventional and standard algorithms to assess the superiority of the modified algorithm for SDN-IDS. The main contributions of this paper are as follows.

1). The mGJO for the SDN-IDS is proposed. The method solves the drawback that the redundant data in the SDN dataset leads to slow convergence of the SDN-IDS algorithm and falls into local optimal solutions. The performance of the proposed method is compared with other methods using benchmark test functions, four UCI datasets, the NSL-KDD dataset, and the InSDN dataset.

2). The GJO is improved using two strategies. The EDOL strategy enhances the quality of solutions while avoiding being trapped in local optimal solutions. The GoldenSAII is brought into the exploitation phase to enhance the convergence capability in later phases of the GJO.

3). The mGJO is a SDN-IDS feature selection method. This paper uses the binary version of the mGJO to select the optimal feature subsets. The mGJO has a robust global search ability and discovery ability.

2. Original algorithm

The GJO algorithm emulates the hunting behavior of the golden jackal pairs and employs mathematical models to simulate their prey-capturing strategies in various scenarios. The golden jackal pair hunting is classified into the following phases.

2.1. Initialization phase

During the initialization phase, the search space is populated with randomly generated solutions according to Eq (1).

$$Y_0 = LB + rand \cdot (UB - LB) \tag{1}$$

where UB denotes the upper bound and LB denotes the lower bound, and "$rand$" denotes a random vector with values ranging from 0 to 1. This equation can be used to generate random vectors whose values range from UB to LB. The prey matrix initialization is created based on Eq (2). By multiplying the values with the difference between UB and LB, this paper ensures that the vector remains within the specified range, and adding UB to each element ensures that the values do not exceed the upper limit.

$$Pos = \begin{bmatrix} Y_{1,1} & \cdots & \cdots & Y_{1,n} \\ Y_{2,1} & \cdots & \cdots & Y_{2,n} \\ \vdots & \cdots & \cdots & \vdots \\ Y_{N,1} & \cdots & \cdots & Y_{N,n} \end{bmatrix} \tag{2}$$

where N denotes the number of prey population and n denotes the dimension, and $Y_{N,n}$ represents the n th dimension of the n th prey.

The generated prey matrix Pos is computed for each of its positional fitness values to obtain the fitness matrix of Eq (3), whereas the one with the smallest fitness value is assigned to the male jackal and the second smallest one to the female jackal.

$$Fitness = \begin{bmatrix} f(Y_{1,1}, Y_{1,2}, \cdots Y_{1,n}) \\ f(Y_{2,1}, Y_{2,2}, \cdots Y_{2,n}) \\ \vdots \\ f(Y_{N,1}, Y_{N,2}, \cdots Y_{N,n}) \end{bmatrix} \tag{3}$$

After completing the initialization phase, the particularly important exploration and exploitation phases follow, as shown in Figure 1.

2.2. Exploration phase

Due to the nature of jackals, they possess the ability to sense and stalk their prey, but occasionally the prey can be difficult to capture. Therefore, jackals wait and look for other prey. This hunting behavior, led by male jackals, is known as the exploration phase, where the female jackals follow the male jackals. The mathematical representation of the exploration phase is according to Eqs (4) and (5) (|E| > 1).

$$Y_1(t) = Y_M(t) - E \cdot |Y_M(t) - rl \cdot Pos(t)| \tag{4}$$

$$Y_2(t) = Y_{FM}(t) - E \cdot |Y_{FM}(t) - rl \cdot Pos(t)| \tag{5}$$

where t is the current number of iterations, $Y_M(t)$ indicates the position of the male jackal, $Y_{FM}(t)$ indicates the position of the female jackal, and $Pos(t)$ is the position vector of the prey. $Y_1(t)$ and $Y_2(t)$ are the latest positions of the male and female jackals.

E is the escape energy of the prey and is calculated as follows.

$$E = E_1 \cdot E_0 \tag{6}$$

$$E_1 = c_1 \cdot (1 - (t/T)) \tag{7}$$

where E_0 denotes a random variable between -1 and 1, denoting the initial energy of the prey, c_1 is a default value set to 1.5, E_1 denotes the decreasing energy of prey, and T denotes the maximum number of iterations.

In Eqs (4) and (5), $|Y(t) - rl \cdot Pos(t)|$ denotes the distance between the golden jackal and the prey, and rl is a vector of random numbers computed by the Levy flight function. The product of rl and $Pos(t)$ simulates the movement of the prey in a Levy fashion.

$$rl = 0.05 \cdot LF(y) \tag{8}$$

$$LF(y) = 0.01 \cdot (\mu \cdot \sigma)/(\left|v^{(1/\beta)}\right|) \tag{9}$$

$$\sigma = \left\{ \frac{\Gamma(1+\beta) \cdot sin(\pi\beta/2)}{\Gamma\left(\frac{1+\beta}{2}\right) \cdot \beta \cdot (2^{\beta-1})} \right\} \tag{10}$$

where μ and v are random values in (0, 1) and β is a default constant set to 1.5.

$$Y(t+1) = \frac{Y_1(t) + Y_2(t)}{2} \tag{11}$$

where $Y(t+1)$ is the updated position of the prey based on the golden jackal pairs.

2.3. Exploitation phase

When prey is harassed by golden jackals, the energy to hide is reduced, and then the jackal pairs will encircle the prey detected in the previous phase. After encircling, they pounce on the prey and eat it. This behavior of male and female jackals hunting together is called as the exploitation phase. The mathematical representation of the exploitation phase is according to Eqs (12) and (13) ($|E| \leq 1$).

$$Y_1(t) = Y_M(t) - E \cdot |rl \cdot Y_M(t) - Pos(t)| \tag{12}$$

$$Y_2(t) = Y_{FM}(t) - E \cdot |rl \cdot Y_{FM}(t) - Pos(t)| \tag{13}$$

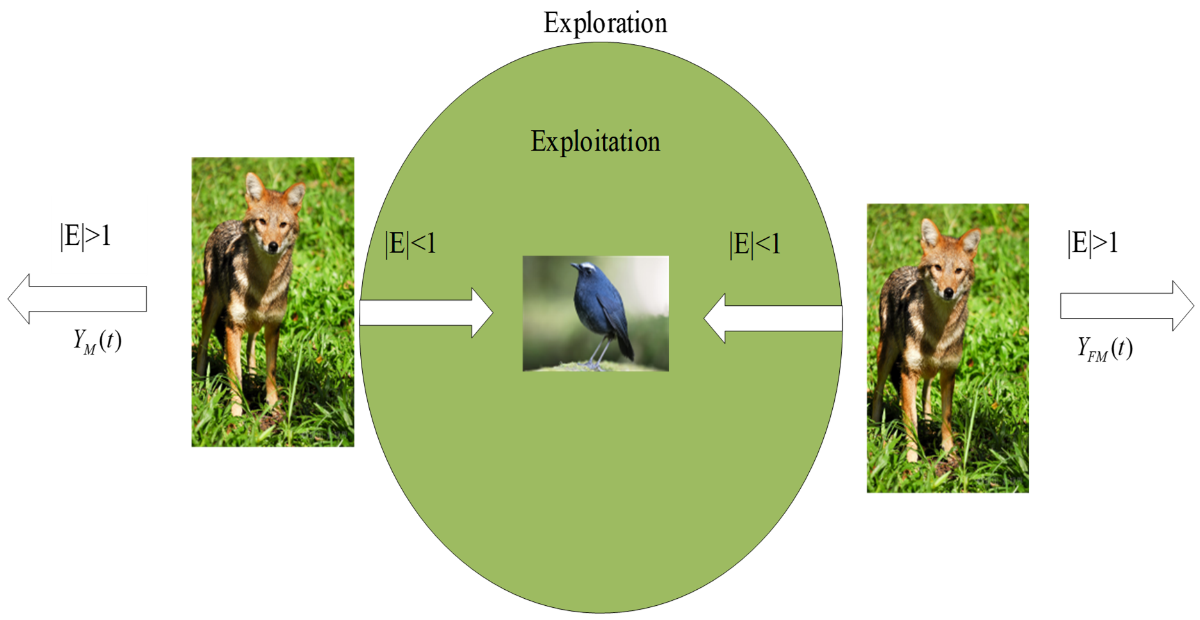

Figure 1. Golden jackal pairs and prey locations during the exploration and exploitation phases.

3. Modified strategies

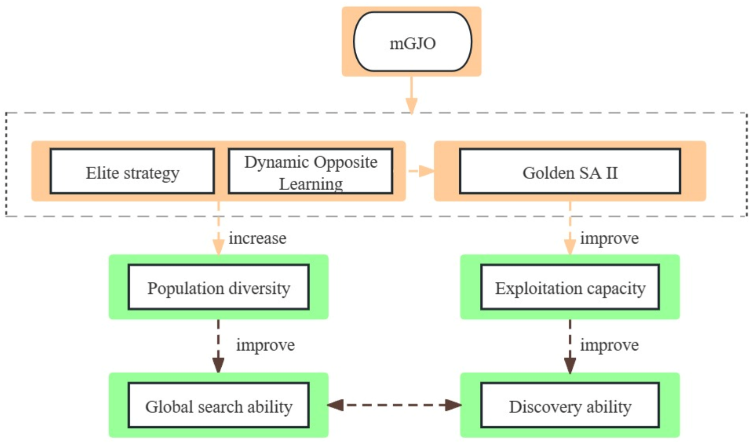

Figure 2. The proposed strategies to solve the deficiency of the GJO.

The GJO may face challenges in effectively addressing feature selection problems, including issues like iterative stagnation, slow convergence in later stages, limited exploration and exploitation capabilities, and quickly reaching local optimal solutions, especially in intricate feature spaces. In this part, this paper proposes the following two improvement strategies, EDOL and GoldenSAII, to figure out the above problems, which are used to solve the problems of insufficient exploitation capability and insufficient initialization diversity of the GJO, as shown in Figure 2.

3.1. Elite Dynamic Opposite Learning strategy

EDOL is an optimization strategy that employs two techniques, dynamic opposite learning (DOL) and elite strategy. In the context of feature selection, DOL introduces opposite individuals to enhance the diversity of the feature space. This variety reduces the risk of becoming trapped in local optimal solutions and allows for a more in-depth exploration of potentially useful features. Meanwhile, the elite strategy preserves the best individuals to make sure that the algorithm preserves its capability to explore global optimal solutions.

3.1.1. Dynamic opposite learning

As for meta-heuristic algorithms, the initialization is usually done by creating random individuals in the search space. While this approach encourages diversity within the population, it does not provide any assurances regarding the quality of the initial solutions. However, studies have demonstrated that the initialization directly influences both the accuracy and convergence speed [27,28]. Xu et al. [29] present a novel solution to address this problem by introducing an innovative approach. They leverage an asymmetric dynamic search space to effectively enhance the quality of solutions. Additionally, they select the best individuals for initialization through the greedy strategy, thereby improving the initial solutions. Therefore, a dynamic opposite learning strategy which increases the quality of the initialization is calculated as follows.

$$X_{OP} = r1 \cdot (LB + UB - X_{pos}) \tag{14}$$

$$X_{DOL} = X_{pos} + r2 \cdot (X_{OP} - X_{pos}) \tag{15}$$

where X_{pos} represents the initialized population generated by random generation, $r1$ and $r2$ represent random numbers uniformly distributed between 0 and 1. Since dynamic boundaries are used, UB represents the max value in X_{pos}, LB represents the min value in X_{pos}. First, the GJO generates two populations, X_{pos} and X_{DOL}, whose fitness values are calculated separately. Second, Comparing fitness values of X_{pos} and X_{DOL}, the smaller one is assigned to the male jackal. In contrast, compared to directly assigning fitness values that are second to male jackals to female jackals, the DOL strategy assigns fitness values that are greater than that of male jackals and less than that of female jackals obtained in the previous cycle will be assigned to female jackals.

3.1.2. Elite strategy

In a population, elite individuals contain more valid information than others, and constructing elite individuals can increase population diversity [30]. To be included in the elite generation, an individual must have the best fitness value at the beginning of each iteration. The elite individual is introduced for the following iteration once the current generation of iterations is finished, and the worst fitness value is eliminated.

3.2. Golden sine II strategy

Since GoldenSAII possesses a robust global search capability and exhibits fast convergence, it enables to search for the optimal feature subset in feature selection. This, in turn, leads to enhanced classifier performance. In addition, it expedites convergence and improves exploration capability throughout the process.

GoldenSAII is an algorithm proposed by Tanyildizi [31] to improve Golden Sine Algorithm (GoldenSA) [32]. The optimization procedure is usually classified into exploration and exploitation. GoldenSAII considers these two phases more than GoldenSA, and the model is shown as follows.

$$\theta = r_4 \cdot x_1 \cdot D_p - x_2 \cdot X_i^t \tag{16}$$

$$X_i^{t+1} = \begin{cases} X_i^t - dr_t \cdot sin(\omega \cdot t \cdot r_3) \cdot \theta & r_5 < 0.5 \\ X_i^t + dr_t \cdot sin(\omega \cdot t \cdot r_3) \cdot \theta & r_5 > 0.5 \end{cases} \tag{17}$$

where X_i^t is the current position of the ith individual on the generation t. r_3, r_4 and r_5 represent random numbers uniformly distributed between 0 and 1. dr_t is the amplitude of the sinusoidal function in the t th iteration, which indicates the subsequent location range or the vector representing the direction of movement, and the subsequent location range can be both near and far from the target, and if $dr_t > 1$, it indicates the location range far away from the target position; otherwise, if $dr_t < 1$, indicates the opposite. ω is the angular frequency. D_p is the value that indicates where the ideal global target is located.

$$x_1 = a \cdot (1 - \tau) + b \cdot \tau \tag{18}$$

$$x_2 = b \cdot (1 - \tau) + a \cdot \tau \tag{19}$$

x_1 and x_2 are the coefficients derived by the golden section method, where a and b are intervals to be searched and τ is the golden ratio with a value of about 0.618033.

$$dr_t = 2 \cdot (1 - t/T) \tag{20}$$

where t is the current number of iterations and T is the maximum number of iterations. The fundamental concept of the GoldenSA class of algorithms is to continuously reduce the search area to identify the best solutions, this is accomplished by utilizing the golden ratio method and the simplified mode of the sinusoidal function, while GoldenSAII introduces the concept of dr_t, which can better narrow down the search scope compared to GoldenSA, and jump out of local optimal solutions.

$$\omega = 2 \cdot \pi \cdot Fc \tag{21}$$

where Fc is the frequency. Fc is the maximum number of iterations multiplied by the population size.

3.3. The proposed algorithm

To overcome the lack of the GJO in feature selection, this paper proposed an algorithm called the mGJO. The mGJO includes two significant improvements, the EDOL and the GoldenSAII.

In the context of feature selection, the traditional GJO typically initializes the golden jackal pairs randomly. However, this random initialization may result in a lack of diversity among the pairs, which can adversely affect the global optimization search, so the EDOL strategy is suggested. First, the EDOL strategy, employed in each iteration, plays a crucial role in generating opposite solutions that are distant from local extreme points using Eq (14). This strategy can make the GJO break free from local optimal solutions and significantly enhances its capability to identify the best solutions [33]. Furthermore, the tracking search pattern, which is based on the dynamic boundaries defined by Eq (15), helps locate individuals within a shrinking search region [34]. The elite strategy can retain individuals who contain more information, thus selecting a better subset of features. The incorporation of the EDOL strategy, which enhances the diversity of the feature subset and the initialization, can find the best solutions.

Second, the GoldenSAII improves the exploitation phase of the GJO, and the position updates are according to the following equations.

$$D_M = r_6 \cdot x_1 \cdot Y_M(t) - x_2 \cdot Pos(t) \tag{22}$$

$$D_{FM} = r_6 \cdot x_1 \cdot Y_{FM}(t) - x_2 \cdot Pos(t) \tag{23}$$

$$Y_1(t) = \begin{cases} Pos(t) - dr_t \cdot sin(\omega \cdot t \cdot r_8) \cdot D_M & r_7 < 0.5 \\ Pos(t) + dr_t \cdot sin(\omega \cdot t \cdot r_8) \cdot D_M & r_7 > 0.5 \end{cases} \tag{24}$$

$$Y_2(t) = \begin{cases} Pos(t) - dr_t \cdot sin(\omega \cdot t \cdot r_8) \cdot D_{FM} & r_7 < 0.5 \\ Pos(t) + dr_t \cdot sin(\omega \cdot t \cdot r_8) \cdot D_{FM} & r_7 > 0.5 \end{cases} \tag{25}$$

The GoldenSAII, due to its sine-based characteristics, provides advantages over the randomness of the Levy flight. It explores the search space evenly and comprehensively, enabling the discovery of potentially useful features. By strengthening the capacity to explore the solution space, this enhanced exploration capability raises the opportunity to discover a subset of extremely valuable features. By guarding against from trapped in local optimal solutions, the mGJO can enhance the overall capability. In addition, the search area is gradually reduced according to the golden segmentation coefficients x_1 and x_2 obtained in Eqs (18) and 19, and the updating distance and direction of the golden jackal pairs are adjusted according to r_6 in Eqs (22) and (23), and r_8 in Eqs (24) and (25), which guides the individuals to quickly approach the subset of more optimal features, improving the searching speed.

The flowchart of the mGJO, which combines EDOL and GoldenSAII, is shown in Figure 3. And the steps are as follows.

Step 1. Initialize the algorithm parameters.

Step 2. Initialize the positions of the golden jackal pairs population and calculate fitness values.

Step 3. Set the parameters of the golden sine, while deciding the elite individuals of the jackal pairs based on the fitness values.

Step 4. Obtain dynamic opposite solutions for the jackal pairs population based on Eqs (14) and (15), then modify the optimal solutions based on the fitness values.

Step 5. Perform exploitation when the escape energy E value is less than 1 and update the jackal pairs position according to Eqs (24) and (25).

Step 6. Perform exploration when the escape energy E value is greater than or equal to 1, and update the jackal pairs position according to Eqs (4) and (5).

Step 7. Update the position of the prey based on the jackal pairs according to Eq (11).

Step 8. Exclude individuals with the largest fitness value, and reserve the elite individuals obtained in Step 3 to the population.

Step 9. Verify if the max iterations have been reached. If it has, end, else return to Step 3.

3.4. The time complexity of the mGJO

The time complexity indirectly reflects the convergence rate of the algorithm. The time to initialize the parameters is t_1. The population size is n and the dimension is d. According to the equations of the update the position of the prey and the golden jackals, the time is t_2, and the time provision for calculating the fitness is T_f. The time complexity of GJO is as follow.

$$O_1(n) = O\left(t_1 + n \cdot \left(d \cdot t_2 + T_f\right)\right) = O\left(d + T_f\right) \tag{26}$$

In the mGJO, the time to initialize the parameters is t_3, and the time to perform the EDOL is t_4. According to the equations of the update the position of the prey and the golden jackals, the time is t_5. The time complexity of the mGJO is as follow.

$$O_2(n) = O\left(t_3 + n \cdot \left(t_4 + d \cdot t_5 + T_f\right)\right) = O\left(d + T_f\right) \tag{27}$$

The mGJO proposed in this paper has the same time complexity as the GJO.

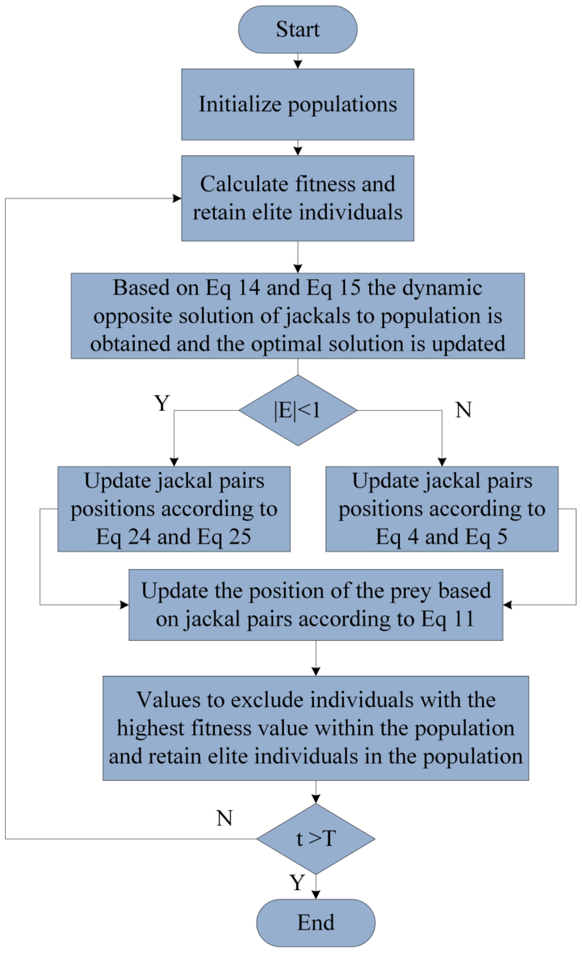
Start
Initialize populations
Calculate fitness and retain elite individuals
Based on Eq 14 and Eq 15 the dynamic opposite solution of jackals to population is obtained and the optimal solution is updated
|E|<1
Y
N
Update jackal pairs positions according to Eq 24 and Eq 25
Update jackal pairs positions according to Eq 4 and Eq 5
Update the position of the prey based on jackal pairs according to Eq 11
Values to exclude individuals with the highest fitness value within the population and retain elite individuals in the population
t >T
N
Y
End

Figure 3. Flow chart of the mGJO.

4. Steps and methods

This section describes applying the mGJO to feature selection problems for non-SDN-IDS and SDN-IDS. The following stages involve in the application of meta-heuristic algorithms for feature selection in NIDS.

4.1. Data preprocessing

After downloading the datasets and examining the samples, the first step involves cleaning the data. This contains tasks such as imputing missing data, denoising data, and so on. Once the above process is completed, the next is to convert the data using the label encoding method. This method is applied to transform string or categorical attributes into their corresponding numeric representations. Label encoding is used because the classifier applies to numeric types. The last step involves normalizing the data by performing a min-max normalization on all features. This process involves normalizing all features to the range between 0 and 1, which is convenient for the algorithm to process. The normalization method is shown in Eq (28).

$$Z_n = \frac{Z - Z_{min}}{Zmin_{max}} \tag{28}$$

where Z_{max} represents the maximum boundary of the target, Z_{min} represents the minimum boundary of the target, Z represents the input data, and Z_n represents the normalized data.

4.2. Feature selection

Feature selection can identify the best subset of features while reducing the dimension. First, the processed dataset is classified into a test set, used for selecting the feature subset, and a training set, used for initialization. Once the population has been initialized and the population fitness has been computed, the elite individual is the one with the lowest fitness value. Next, using the EDOL, the population opposite solution is obtained, and the current optimal solution is updated. When it comes to exploitation, the GoldenSAII is introduced to find more potential features, which facilitates the discovery of optimal feature subsets. During the exploration phase, the initial algorithm is employed. Before the elite strategy is implemented, it updates the location of prey based on jackal pairs. After all the iterations are completed, the feature indexes are output. At last, the feature subsets from the test set are selected according to the feature indexes.

4.3. Classifier evaluation

This paper selects KNN [35] as the classifier. The KNN classifier uses the training set to train and get the feature indexes, selection is made from the test set based on the indexes obtained. Then the confusion matrix is quoted as obtaining *TP*, *TN*, *FN*, and *FP*, the significance of these four values will be described in Section 5, and the remaining metrics *Pre*, *Rec*, *F1*, and *Acc* are calculated.

The flowchart for using the mGJO to feature selection is displayed in Figure 4.

The steps to use the mGJO for the feature selection are as follows.

Step 1. Preprocess and divide the data.

Step 2. The training set is applied to initialize the parameters of the mGJO.

Step 3. Initialize the positions of the jackal pairs population and calculate fitness values.

Step 4. Set the parameters of the golden sine, while determining the elite individuals of the jackal pairs based on the fitness values.

Step 5. Obtain dynamic opposite solutions for the jackal pairs population based on Eqs (14) and (15), then modify the optimal solutions based on the fitness values.

Step 6. Perform exploitation when the escape energy E value is less than 1 and update the jackal pairs position according to Eqs (24) and (25).

Step 7. Perform exploration when the escape energy E value is greater than or equal to 1, and update the jackal pairs position according to Eqs (4) and (5).

Step 8. Update the position of the prey based on jackal pairs according to Eq (11).

Step 9. Exclude individuals with the largest fitness value within the population based on the fitness value, and retain the elite individuals obtained in Step 4 to the population.

Step 10. Verify if the max iterations have been reached. If it has, go to Step 11, else return to Step 4.

Step 11. Compare the feature index of the optimal individual with the test set, get the test set after feature selection, and then use the classifier for evaluation.

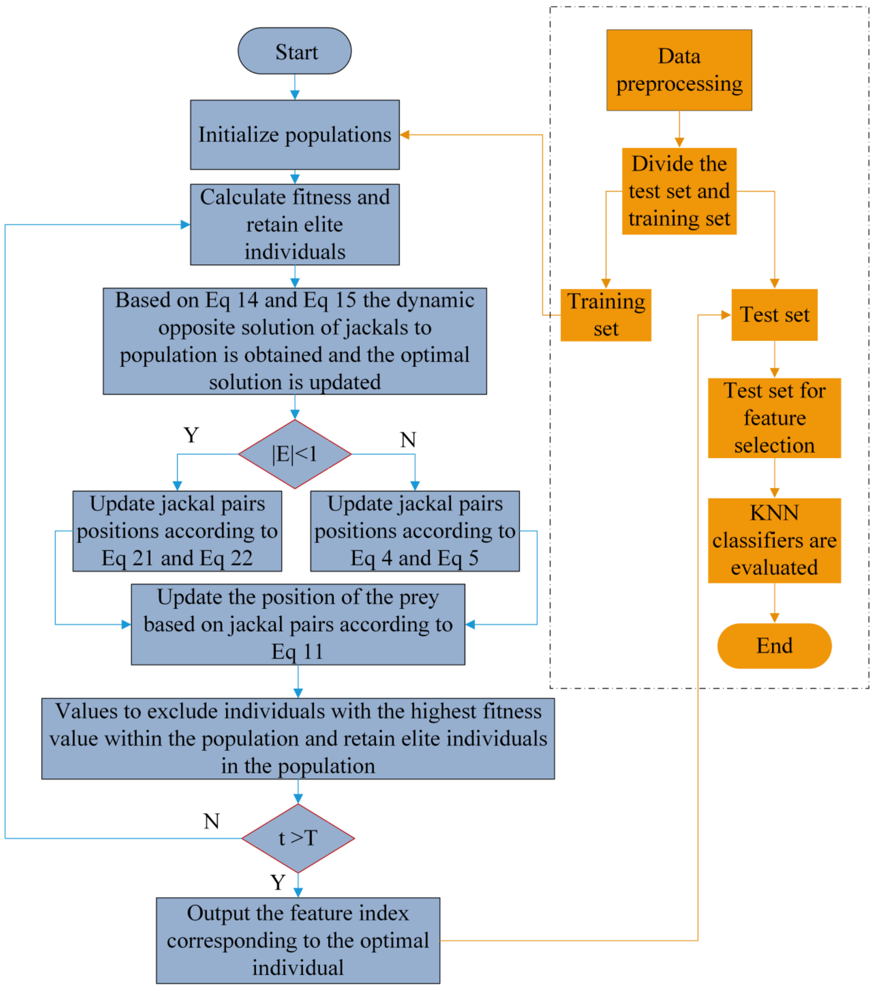
Start
Initialize populations
Calculate fitness and retain elite individuals
Based on Eq 14 and Eq 15 the dynamic opposite solution of jackals to population is obtained and the optimal solution is updated
Y
|E|<1
N
Update jackal pairs positions according to Eq 21 and Eq 22
Update jackal pairs positions according to Eq 4 and Eq 5
Update the position of the prey based on jackal pairs according to Eq 11
Values to exclude individuals with the highest fitness value within the population and retain elite individuals in the population
N
t >T
Y
Output the feature index corresponding to the optimal individual
Data preprocessing
Divide the test set and training set
Training set
Test set
Test set for feature selection
KNN classifiers are evaluated
End

Figure 4. Feature selection flow chart of the mGJO.

5. Experiments and results

5.1. Benchmark function testing

The simulation test environment is: operating system Windows 11, 64-bit, CPU Intel i5 12400F with Nvidia GeForce RTX 3060ti, 16 GB of RAM, main frequency 2.50 GHz, simulation software Matlab2020b.

This paper selects some swarm intelligence algorithms for tests, which are Grey Wolf Optimizer (GWO) [36], Whale Optimization Algorithm (WOA) [37], GJO, and the mGJO, which incorporates EDOL and GoldenSAII, for comparison. The parameters of all algorithms are displayed in Table 1.

To test the optimization effect of the mGJO, this paper selects four unimodal functions (f1–f4) and four multimodal functions (f5–f8), Table 2 describes the details of the test functions. Table 3 and Figure 5 present the outcomes obtained from running the four algorithms independently for 30 iterations on each test function.

Based on the selected test functions, results indicate that the mGJO possesses the strongest optimization-seeking performance. The mGJO performs significantly better than GJO, GWO, and WOA. The mGJO achieves a 100% optimization effect for the functions f1–f8, with an optimal value of 0 that can be directly searched for. Except for functions f6–f8, the standard deviation of the improved algorithm is 0, indicating strong stability of the mGJO. For functions f3 and f4, the optimization effect of the mGJO is notably higher than that of other novel algorithms, with optimal values exceeding 270 orders of magnitude. The optimization accuracy improves quite a lot compared to the GJO.

The experiments show that the mGJO overcomes the problem that the GJO tends to fall into iterative stagnation and converge slowly at the later stage, and greatly improves optimization accuracy and convergence speed.

5.2. Dataset testing

In feature selection, the number of selected features and the classification error rate decide the capability of the feature subset. The evaluation function is according to Eq (29).

$$fitness = \alpha \cdot error + \beta \cdot \frac{|Nf|}{T} \tag{29}$$

where $fitness$ represents the best value of individual viable solutions of the population, $error$ indicates the classification error rate, Nf expresses the feature number, T denotes the max iterations, α and β are two parameters, α = 0.99, β = 0.01.

Table 1. Algorithms parameters.

Parameters	Symbol	Numbers
Population size	N	50
Dim	D	30
Max iterations	T	500

Table 2. Test function.

Names	Test functions	Dim	Range	Min
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-100, 100]	0
Schwefel's 2.22	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10, 10]	0
Schwefel's 1.2	$f_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_j)^2$	30	[-100, 100]	0
Schwefel's 2.21	$f_4(x) = max_i\{\lvert x_i \rvert, 1 \leq i \leq n\}$	30	[-100, 100]	0
Rastrigin	$f_5(x) = \sum_{i=1}^{n} [x_i^2 - 10\,cos(\,2\pi x_i) + 10]$	30	[-5.12, 5.12]	0
Ackley	$f_6(x) = -20\,exp(\,-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n}\sum_{i=1}^{n} cos(\,2\pi x_i)) +20 + e$	30	[-32, 32]	0
Griewank	$f_7(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600, 600]	0
Generalized Rastrigin	$f_8(x) = \frac{\pi}{n}\left\{10\,sin^2(\,\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2\,[1 + 10\,sin^2(\,\pi y_{i+1})] +(y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4}, u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	[-50, 50]	0

Table 3. Experimental results of test functions.

Functions	Algorithms	Ave	Std	Min
f_1	GWO	2.5e-34	0	2.5e-34
	WOA	1.15e-85	0	1.15e-85
	GJO	6.39e-65	0	6.39e-65
	mGJO	**0**	**0**	**0**
f_2	GWO	3.73e-20	5.28e-20	2.5e-34
	WOA	1.98e-56	2.8e-56	1.15e-85
	GJO	1.48e-37	2.1e-37	6.39e-65
	mGJO	**0**	**0**	**0**
f_3	GWO	4.78e-10	8.28e-10	2.5e-34
	WOA	1.06e+04	1.84e+04	1.15e-85
	GJO	1.77e-25	3.07e-25	6.39e-65
	mGJO	**0**	**0**	**0**
f_4	GWO	9.6e-09	1.92e-08	0
	WOA	1.95	3.9	0
	GJO	2.25e-20	4.5e-20	0
	mGJO	**0**	**0**	**0**
f_5	GWO	0	0	0
	WOA	0	0	0
	GJO	0	0	0
	mGJO	**0**	**0**	**0**
f_6	GWO	4e-15	1.26e-14	0
	WOA	4.44e-16	1.4e-15	0
	GJO	4.44e-16	1.4e-15	0
	mGJO	**8.88e-17**	**2.81e-16**	**0**
f_7	GWO	3.63e-15	1.21e-14	0
	WOA	4.04e-16	1.34e-15	0
	GJO	4.04e-16	1.34e-15	0
	mGJO	**8.07e-17**	**2.81e-16**	**0**
f_8	GWO	2.22e-03	7.7e-03	0
	WOA	1.65e-04	5.72e-04	0
	GJO	1.21e-02	4.19e-02	0
	mGJO	**2.34e-07**	**8.12e-07**	**0**

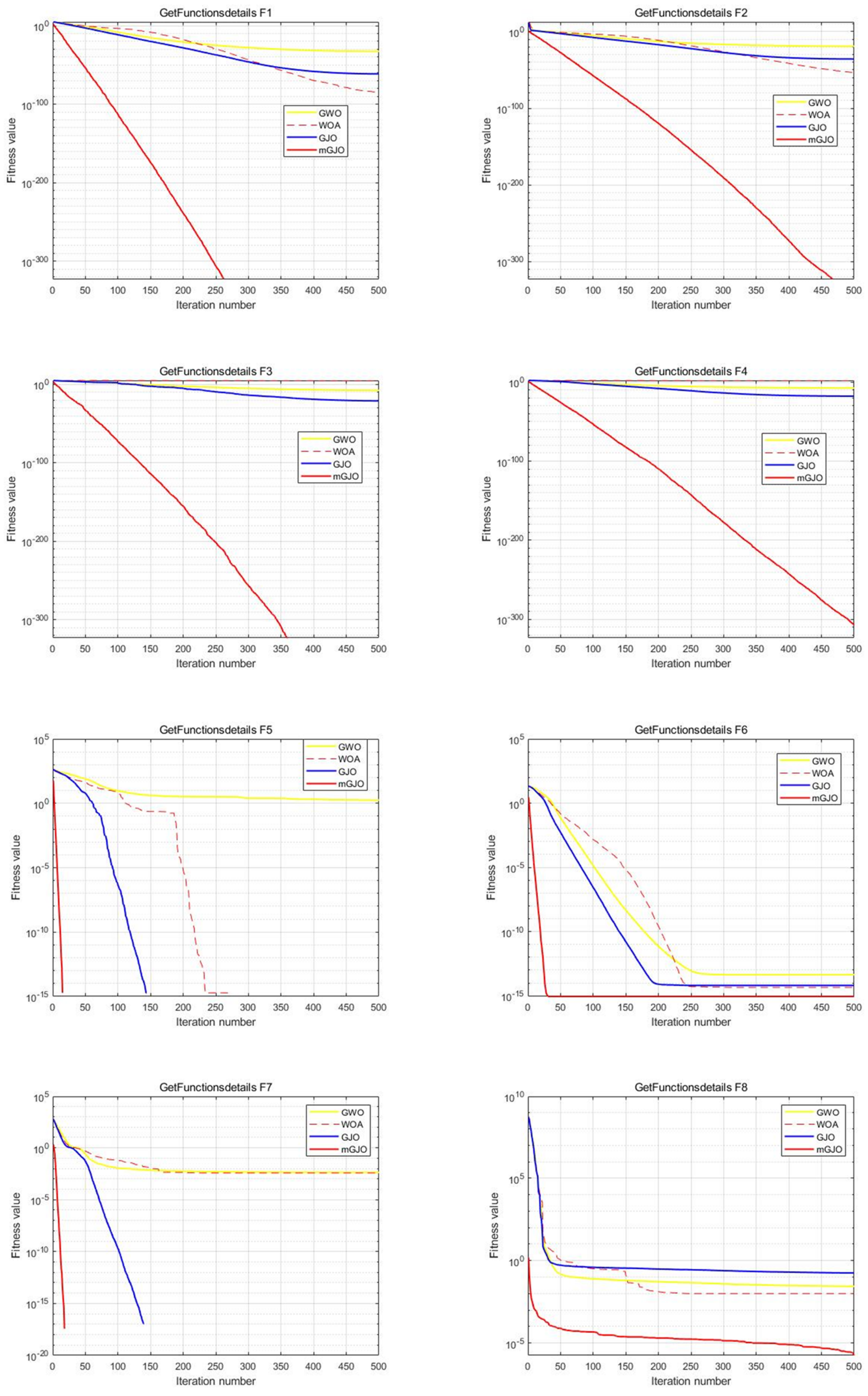

Figure 5. Test function experiment results.

The classifier is KNN with k = 10, and a few evaluation metrics commonly employed in classification detection experiments: Accuracy (*Acc*), Recall (*Rec*), F1 score (*F1*), and Precision (*Pre*) are used to evaluate the performance of the algorithm, unit in %. The calculation is according to the following equations.

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} \tag{30}$$

$$F1 = \frac{2\times TP}{2\times TP+FP+FN} \tag{31}$$

$$Rec = \frac{TP}{TP+FN} \tag{32}$$

$$Pre = \frac{TP}{TP+FP} \tag{33}$$

where TP (True Positive) and TN (True Negative) denote the values of positive class samples, while FP (False Positive) and FN (False Negative) denote the values of negative class samples. The comparison algorithms chosen for the experiments are still GWO, WOA, and GJO. When it comes to parameters, except N is the same as Table 1, T is set to 50 and Dim is the feature number of each dataset.

5.2.1. UCI dataset

In this section, four UCI datasets are selected, namely Ionosphere, WDBC, Heartstatlog, and Sonar. Table 4 displays the results of the four algorithms that are run on each dataset, along with Figure 6.

Based on the given four datasets, the test results indicate that mGJO performs best on the Ionosphere dataset, thus proving its efficiency on this dataset. On the WDBC and Sonar datasets, the remaining three metrics, except for *Rec*, are the best performers. This shows that the mGJO accurately recognizes both positive and dissimilar classes. Although *Rec* is not the highest, it ranks second and is only about 1% lower than the first place. For the Heartstatlog dataset, *Acc*, *Rec*, and *F1* are all the best performers. This means that the mGJO effectively detects positive samples and performs well for balanced classifier performance. However, the performance of *Pre* is not very good, about 9% lower than the first place. Nonetheless, it still ranks second. Furthermore, Figure 6 depicts the optimization search process of the mGJO on different datasets. According to Figure 6, the mGJO performs faster optimization searches on the WDBC and Ionosphere datasets. However, on the Heartstatlog and Sonar datasets, there are constant transitions and occasional escapes from local optimal solutions.

On the given UCI datasets, *F1*, *Acc* of the mGJO are consistently higher than those of the other comparative algorithms, which indicates that the mGJO can accurately classify most of the samples with strong classifiability.

Table 4. Experimental results of UCI datasets.

Datasets	Algorithms	*Acc* (%)	*Pre* (%)	*Rec* (%)	*F1* (%)	Total rank
Ionosphere	GWO	96.1905	95.8333	98.5714	97.1831	2
	WOA	93.3333	92	98.5714	95.1724	4
	GJO	94.2857	93.2432	98.5714	95.8333	3
	mGJO	**97.1429**	**95.8904**	**100**	**97.9021**	**1**
WDBC	GWO	92.9412	91.8182	97.1154	94.3925	4
	WOA	93.5294	92.6606	97.1154	94.8357	2
	GJO	93.5294	92.6606	97.1154	94.8357	2
	mGJO	**94.7059**	**94.3925**	97.1154	**95.7346**	**1**
Heartstatlog	GWO	90.1235	87.5	87.5	87.5	2
	WOA	87.6543	86.6667	81.25	83.871	4
	GJO	90.1235	96.1538	78.125	86.2069	3
	mGJO	**91.358**	87.8788	**90.625**	**89.2308**	**1**
Sonar	GWO	93.5484	87.8788	88.6667	93.5484	2
	WOA	87.0968	83.871	90.6667	86.6667	4
	GJO	88.7097	80.5556	87.6667	89.2308	3
	mGJO	**95.1613**	**90.625**	89.6667	**95.082**	**1**

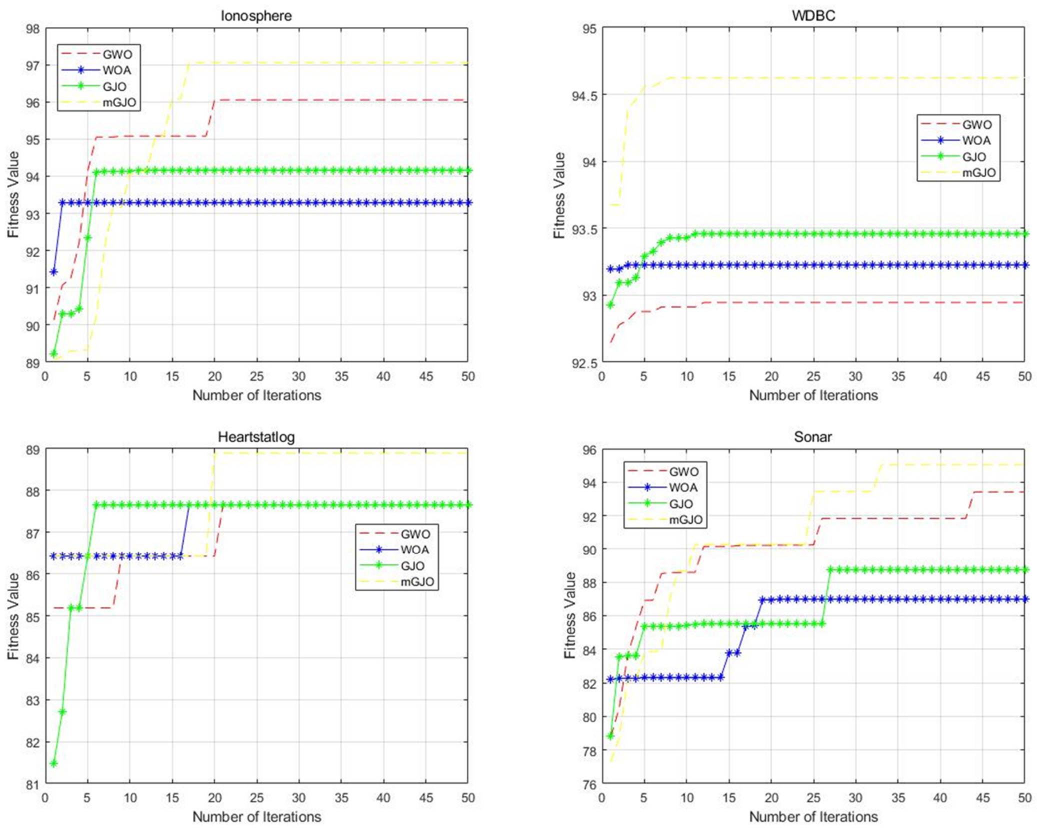

Figure 6. UCI datasets test results.

5.2.2. NSL-KDD dataset

This section uses KDDTrain+_20Percent.txt with KDDTest-21.txt for the test, which is part of the NSL-KDD [38] dataset.

First, the attack stream type column is used as the label after the string type in the dataset transforms into a data type. The DoS, Probe, U2R, and R2L classes are all set to 1, representing the Abnormal class, while the Normal class is set to 0. Second, the dimension of the dataset is decreased. The features of columns 10 through 22 of the network connection vector are eliminated, leaving the remaining columns as features. At last, the normalization is carried out, this paper uses the mapminmax function so that the data maps to the interval from 0 to 1. This paper takes 5% of the data in KDDTrain and KDDTest, KDDTrain is used for training while KDDTest is used for prediction. Table 5 displays the results of the four algorithms that are run, along with Figure 7.

Based on the test results of the NSL-KDD dataset, the mGJO has been ranked as the best algorithm overall, with a 3% to 5% increase in *Acc*, a 4% to 8% increase in *Pre*, and a 2% to 4% increase in *F1* in comparison to other algorithms. The highest *Acc* signifies that the mGJO is highly accurate in classifying the entire dataset and has a minimal tendency to misclassify the Normal class samples as Abnormal. The highest *Pre* indicates that the mGJO is precise in predicting samples to the Abnormal class, which helps to avoid incorrect predictions of Normal class samples as Abnormal. The highest *F1* signifies that the mGJO finds the best balance between *Pre* and *Rec*, accurately predicting Abnormal class samples and efficiently identifying the most relevant features that can be selected for all Abnormal class samples. As shown in Figure 7, Fitness values of the mGJO continue to rise with fluctuating during the updating, indicating that the mGJO keeps overcoming the local optimal solution and has a strong capability to select the best features. Furthermore, the mGJO reaches the optimal fitness value in the 19th generation, and the convergence speed is far ahead of other algorithms.

Compared to tests on the UCI datasets, the results of the NSL-KDD experiment can further illustrate the classification capability of the mGJO. The results for *F1*, *Acc*, and *Pre* metrics are indicative of a commendable performance. The mGJO can find the optimal subset of features in a better way, precisely classify the samples, and strike an optimal balance between the prediction and identification of Abnormal class samples.

Table 5. Experimental results of classification of NSL-KDD dataset.

Algorithms	*Acc* (%)	*Rec* (%)	*Pre* (%)	*F1* (%)	Number of features	Total rank
GWO	82.2695	97.3251	71.6667	82.5480	10	3
WOA	79.3440	97.1193	68.3068	80.2040	11	4
GJO	82.5355	97.1193	72.0611	82.7344	10	2
mGJO	**85.4610**	95.4733	**76.5677**	**84.9817**	**6**	**1**

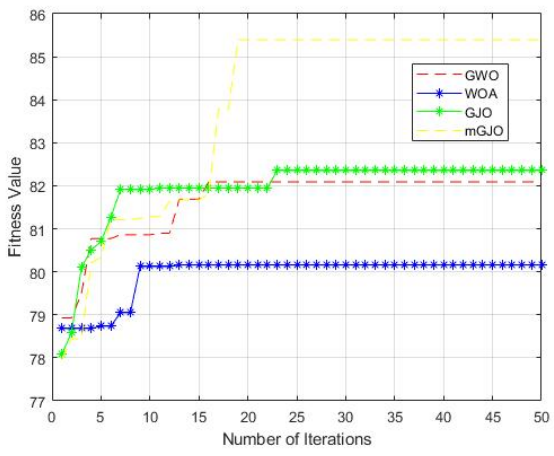

Figure 7. NSL-KDD dataset test results.

In addition, this paper also tests the ROC curve with AUC values shown in Table 6 and Figure 8. Based on Table 6 and Figure 8, it can be concluded that the mGJO performs exceptionally well on the NSL-KDD dataset. The ROC curve of the mGJO is near the left top corner which implies it reduces the possibility of false positives, which suggests that the mGJO performs very close to the optimal performance under different thresholds. The AUC value of the mGJO achieves closest to 1, exhibiting an enhancement of 0.4% to 4% compared to other algorithms, which indicates the mGJO performs well on a high true positive rate and a low false positive rate for sample recognition. The mGJO performs better under different thresholds and can classify efficiently.

Table 6. AUC results of classification of NSL-KDD dataset.

Algorithms	AUC
GWO	0.9088
WOA	0.8711
GJO	0.9107
mGJO	**0.9146**

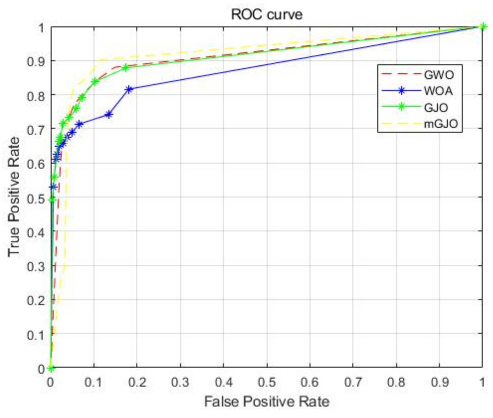

Figure 8. ROC curves.

5.2.3. InSDN dataset

Today, SDN enables a wide range of applications, which are accompanied by threats and vulnerabilities, making intrusion detection for SDN essential. Elsayed et al. [39] published an attack-specific SDN dataset, InSDN, which includes several categories of attacks against SDN components, including attacks on the data plane, attacks on control plane communications, attacks on SDN controllers, and attacks on the application plane. InSDN contains three datasets in CSV format, Normal, OVS, and metasploitable-2, which contains a total of 343,889 pieces of data, 1 label tag, 6 socket information, and 77 different features [40]. Table 7 displays that the Normal Group contains information about all normal flows, while the OVS Group with Metasploitable-2 Group represents the attack flows.

This paper uses the InSDN dataset for SDN-IDS, which focuses on feature selection for the subset features of the SDN environment. This paper combines labels that are from the original dataset and features that are shown in Table 8 together as a new SDN dataset.

Table 7. InSDN dataset composition.

Data group	Traffic distribution	Total number of instances	Total%	PCAP size
Normal group	Skype, Facebook, File Transfer Youtube, Email, DNS, Chat, Browsing	68,424	19.90%	3.58 GB
Metasploitable-2 group	DDoS	73,529	39.76%	667 MB
	Probe	61,757		
	DoS	1145		
	BFA	295		
	U2R	17		
OVS group	DoS	52,471	40.34%	1.21 GB
	DDoS	48,413		
	Probe	36,372		
	BFA	1110		
	Web_attack	192		
	Botnet	164		

Since the labels of the InSDN dataset are divided into 8 classes, which are Normal, DoS, DDoS, Probe, BFA, Web-Attack, Botnet, and U2R, these label classes are converted to data types corresponding to 0, 1, 2, 3, 4, 5, 6, and 7 respectively when performing multiclassification, as the label classes of the new dataset, and the remaining 48 columns are normalized using the mapminmax function in Matlab to get the processed SDN dataset. Due to the large amount of data in the InSDN dataset, all the data in the BFA, Web-Attrack, Botnet, and U2R categories, which have a small amount of data, are taken out for the experiment here. The Normal, DoS, DDoS, and Probe categories based on the percentage of the original dataset are then randomized to a total of 10,000 instances for testing. The proportion of the training set to the test set is 7 to 3. Tables 9 and 10 present the outcomes obtained from running the four algorithms independently for 30 iterations.

Table 8. The subset features for SDN environment.

No.	Attribute name	No.	Attribute name
1	Protocol	25	Fwd-IAT-Min
2	Flow-duration	26	Bwd-IAT-Tot
3	Tot-Fwd-Pkts	27	Bwd-IAT-Mean
4	Tot-Bwd-Pkts	28	Bwd-IAT-Std
5	TotLen-Fwd-Pkts	29	Bwd-IAT-Max
6	TotLen-Bwd-Pkts	30	Bwd-IAT-Min
7	Fwd-Pkt-Len-Max	31	Fwd-Header-Len
8	Fwd-Pkt-Len-Min	32	Bwd-Header-Len
9	Fwd-Pkt-Len-Mean	33	Fwd-Pkts/s
10	Fwd-Pkt-Len-Std	34	Bwd-Pkts/s
11	Bwd-Pkt-Len-Max	35	Pkt-Len-Min
12	Bwd-Pkt-Len-Min	36	Pkt-Len-Max
13	Bwd-Pkt-Len-Mean	37	Pkt-Len-Mean
14	Bwd-Pkt-Len-Std	38	Pkt-Len-Std
15	Flow-Byts/s	39	Pkt-Len-Var
16	Flow-Pkts/s	40	Pkt-Size-Avg
17	Flow-IAT-Mean	41	Active-Mean
18	Flow-IAT-Std	42	Active-Std
19	Flow-IAT-Max	43	Active-Max
20	Flow-IAT-Min	44	Active-Min
21	Fwd-IAT-Tot	45	Idle-Mean
22	Fwd-IAT-Mean	46	Idle-Std
23	Fwd-IAT-Std	47	Idle-Max
24	Fwd-IAT-Max	48	Idle-Min

Based on the results, the mGJO outperforms the other algorithms and claims the top rank. Among the four categories, namely DoS, DDoS, BFA, and Web-Attack, the mGJO has the highest *Pre*. On DoS and DDoS, the mGJO improves by about 0.1% compared to the other three algorithms. On BFA and Web-Attack, the *Pre* of the mGJO enhances about 1% than WOA. In terms of overall performance, the mGJO ranks second on Normal, Probe, and Botnet. Regarding *Rec*, Normal performs the best, with an improvement of around 0.2% to 0.6% compared to the other three algorithms. However, on DDoS, Probe, BFA, and Web-Attack, Normal ranks second. When it comes to *F1*, Normal, DoS, DDoS, and Web-Attack are the four categories with the highest performance, with an improvement of about 0.5% compared to other algorithms. In other categories, the mGJO ranks second. When it comes to *Acc*, the mGJO also performs the best. Overall it shows that the mGJO performs excellently in *Acc*, *F1*, and *Pre*. *Rec*, conversely, is slightly worse than *F1* and *Pre*. However, compared to the other algorithms, the mGJO ranks first in overall *Rec*.

From the above analysis, it is easy to see that the mGJO has an improvement in *Acc*, *F1*, and *Pre* compared to the remaining three algorithms, which indicates that the mGJO can also perform at a better level in multi-classification problems, with less misclassification of these types of data in the

InSDN dataset, and can make the correct predictions. For Normal, DoS, DDoS, and Probe, which are classes with high data volume in the original dataset and a lot of attack behaviors that require high *Acc* and *Pre*, the mGJO is the best performer in these classes combined. As for BFA, Web-Attack, and BOTNET, which are classes with fewer attack flows in the InSDN dataset itself, high *Rec* and high *F1* are often required. The mGJO in these classes also achieves the combined optimum in these two metrics. It is shown that the mGJO is suitable for the InSDN dataset and can also perform feature selection on the dataset. The experimental data also further illustrates that the mGJO can show strong *Acc*, *F1*, and *Pre* when performing feature selection.

Table 9. Experimental results of classification of InSDN dataset.

		Normal	DoS	DDoS	Probe	BFA	Web-Attack	BOTNET	U2R
	GWO	**97.2468**	95.1207	99.8043	94.9998	91.3698	81.1554	93.7705	NaN
Pre	WOA	96.4762	95.4179	99.8039	94.7440	90.7353	80.4253	94.3478	NaN
(%)	GJO	96.6281	95.4217	99.9076	**95.3107**	91.6636	81.7507	**96.4385**	NaN
	mGJO	96.7177	**95.5942**	**99.9078**	95.2272	**91.6947**	**82.1369**	95.5201	NaN
	GWO	94.8096	**97.5758**	**99.6663**	95.4955	93.7767	**69.1525**	100	0
Rec	WOA	94.8898	96.8595	99.5857	95.2102	93.2067	68.4746	100	0
(%)	GJO	95.2505	97.4931	99.5397	**95.7958**	**94.0143**	68.1356	100	0
	mGJO	**95.4509**	97.3553	99.6087	95.7808	93.8717	68.4746	100	0
	GWO	96.0120	96.3293	99.7351	95.2459	92.5565	74.6601	96.7798	NaN
F1	WOA	95.6748	96.1317	99.6945	94.9752	91.9525	73.9526	97.0720	NaN
(%)	GJO	95.9319	96.4447	99.7232	**95.5520**	**92.8237**	74.3071	**98.1793**	NaN
	mGJO	**96.0760**	**96.4651**	**99.7580**	95.5014	92.7699	**74.6669**	97.7051	NaN
	GWO				99.0027				
Acc	WOA				98.9397				
(%)	GJO				99.0317				
	mGJO				**99.0368**				

Table 10. Ranks and total ranks of each indicator.

	Pre	*Rec*	*F1*	*Acc*	Total rank
GWO	3	**1**	3	3	3
WOA	4	4	4	4	4
GJO	2	3	2	2	2
mGJO	**1**	**1**	**1**	**1**	**1**

6. Conclusions and future works

To promote the application of the GJO in SDN-IDS, this paper proposes the mGJO that incorporates EDOL and GoldenSAII strategies. First, the global search capability of the mGJO is verified using the benchmark test function; second, the UCI datasets are used to verify the classification ability of mGJO in feature selection; subsequently, the NSL-KDD dataset is used to verify the feasibility of mGJO in non-SDN-IDS, which further verifies the ability of mGJO for feature subset selection; in the end, the experiments with the InSDN dataset conclude that mGJO can be used for feature selection on InSDN dataset, and it also has a very good effect for SDN-IDS, and it can be further extended for use in SDN.

The mGJO combines the advantages of the EDOL strategy with the GoldenSAII strategy compared to the GJO. From the experimental results above, it can be seen that the mGJO performs well in the three indicators of *F1*, *Acc*, and *Pre*, but it is difficult to achieve a high value of *Rec* and at the same time, *F1* balances *Pre* and *Rec*. Moreover, there are fewer datasets for SDN-IDS, which lacks the generalizability to be applied in real SDN scenarios. All of these will be further studied and improved. In the next step, our work intends to conduct simulation experiments using the SDN environment to obtain the flow data for further testing.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work has been supported by the National Natural Science Foundation of China under Grant 61602162 and Hubei Provincial Science and Technology Plan Project under Grant 2023BCB041.

Conflict of interest

The authors declare no conflict of interest.

References

1. A. Savaliya, R. H. Jhaveri, Q. Xin, S. Alqithami, S. Ramani, T. A. Ahanger, Securing industrial communication with software-defined networking, *Math. Biosci. Eng.*, **18** (2021), 8298–8314. https://doi.org/10.3934/mbe.2021411
2. N. T. Hoang, H. N. Nguyen, H. A. Tran, S. Souihi, A novel adaptive east–west interface for a heterogeneous and distributed SDN network, *Electronics*, **11** (2022), 975. https://doi.org/10.3390/electronics11070975
3. P. Wu, Y. Shang, S. Bai, L. Cheng, H. Tang, A lightweight path consistency verification based on INT in SDN, *Math. Biosci. Eng.*, **20** (2023), 19468–19484. https://doi.org/10.3934/mbe.2023862
4. A. Yazdinejadna, R. M. Parizi, A. Dehghantanha, M. S. Khan, A kangaroo-based intrusion detection system on software-defined networks, *Comput. Networks*, **184** (2021), 107688. https://doi.org/10.1016/j.comnet.2020.107688
5. S. Badotra, S. Tanwar, S. Bharany, A. U. Rehman, E. T. Eldin, N. A. Ghamry, et al., A DDoS vulnerability analysis system against distributed SDN controllers in a cloud computing environment, *Electronics*, **11** (2022), 3120. https://doi.org/10.3390/electronics11193120
6. M. W. Nadeem, H. G. Goh, V. Ponnusamy, Y. Aun, DDoS detection in SDN using machine learning techniques, *Comput. Mater. Continua*, **71** (2022), 771–789. https://doi.org/10.32604/cmc.2022.021669

7. J. Wang, Y. Liu, H. Feng, IFACNN: Efficient DDoS attack detection based on improved firefly algorithm to optimize Convolutional Neural Networks, *Math. Biosci. Eng.*, **19** (2022), 1280–1303. https://doi.org/10.3934/mbe.2022059
8. F. Zhang, Z. Gao, K. Niu, Network intrusion detection model based on BiGRU system (in Chinese), *Comput. Technol. Dev.*, **33** (2023), 144–149. https://doi.org/10.3969/j.issn.1673-629X.2023.01.022
9. J. Liu, Y. Yan, Artificial fish feature selection network intrusion detection system (in Chinese), *J. Xidian Univ.*, **50** (2023), 132–138. https://doi.org/10.19665/j.issn1001-2400.2023.04.013
10. J . Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, et al., Feature selection: A data perspective, *ACM Comput. Surv.*, **50** (2017), 1–45. https://doi.org/10.1145/3136625
11. O. Friha, M. Ferrag, S. Lei, M. Leandros, C. Kim-Kwang, M. Nafaa, FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things, *J. Parallel Distrib. Comput.*, **165** (2022), 17–31. https://doi.org/10.1016/j.jpdc.2022.03.003
12. R. A. Elsayed, R. A. Hamada, M. I. Abdalla, S. A. Elsaid, Securing IoT and SDN systems using deep-learning based automatic intrusion detection, *Ain Shams Eng. J.*, **14** (2023), 102211. https://doi.org/10.1016/j.asej.2023.102211
13. N. M. Yungaicela-Naula, C. V. Rosales, J. A. Perez, E. Jacob, C. M. Cagnazzo, Physical assessment of an SDN-based security framework for DDoS attack mitigation: Introducing the SDN-SlowRate-DDoS dataset, *IEEE Access*, **11** (2023), 46820–46831. https://doi.org/10.1109/ACCESS.2023.3274577
14. G. O. Anyanwu, C. I. Nwakanma, J. M. Lee, D. S. Kim, Optimization of RBF-SVM kernel using grid search algorithm for DDoS attack detection in SDN-based VANET, *IEEE Internet Things J.*, **10** (2022), 8477–8490. https://doi.org/10.1109/JIOT.2022.3199712
15. Y. Gu, K. Li, Z. Guo, Y. Wang, Semi-supervised k-means DDoS detection method using hybrid feature selection algorithm, *IEEE Access*, **7** (2019), 64351–64365. https://doi.org/10.1109/ACCESS.2019.2917532
16. N. Chopra, M. Mohsin Ansari, Golden jackal optimization: A novel nature-inspired optimizer for engineering applications, *Expert Syst. Appl.*, **198** (2022), 116924. https://doi.org/10.1016/j.eswa.2022.116924
17. E. Ghandourah, Y. S. Prasanna, A. H. Elsheikh, E. B. Moustafa, M. Fujii, S. S. Deshmukh, Performance prediction of aluminum and polycarbonate solar stills with air cavity using an optimized neural network model by golden jackal optimizer, *Case Stud. Therm. Eng.*, **47** (2023), 103055. https://doi.org/10.1016/j.csite.2023.103055
18. I. R. Najjar, A. M. Sadoun, A. Fathy, A. W. Abdallah, M. A. Elaziz, M. Elmahdy, Prediction of tribological properties of alumina-coated, silver-reinforced copper nanocomposites using long short-term model combined with golden jackal optimization, *Lubricants*, **10** (2022), 277. https://doi.org/10.3390/lubricants10110277
19. R. M. Devi, M. Premkumar, G. Kiruthiga, R. Sowmya, IGJO: An improved golden jackel optimization algorithm using local escaping operator for feature selection problems, *Neural Process. Lett.*, **14** (2023), 1–89. https://doi.org/10.1007/s11063-023-11146-y
20. H. Das, S. Prajapati, M. K. Gourisaria, R. M. Pattanayak, A. Alameen, M. Kolhar, Feature selection using golden jackal optimization for software fault prediction, *Mathematics*, **11** (2023), 2438. https://doi.org/10.3390/math11112438

21. F. Y. Arini, K. Sunat, C. Soomlek, Golden jackal optimization with joint opposite selection: An enhanced nature-inspired optimization algorithm for solving optimization problems, *IEEE Access,* **10** (2022), 128800–128823. https://doi.org/10.1109/ACCESS.2022.3227510
22. Z. Lu, M. Tian, J. Zhou, X. Liu, Enhancing sensor duty cycle in environmental wireless sensor networks using Quantum Evolutionary Golden Jackal Optimization Algorithm, *Math. Biosci. Eng.*, **20** (2023), 12298–12319. https://doi.org/10.3934/mbe.2023547
23. H. Xu, K. Przystupa, C. Fang, A. Marciniak, O. Kochan, M. Beshley, A combination strategy of feature selection based on an integrated optimization algorithm and weighted k-nearest neighbor to improve the performance of network intrusion detection, *Electronics*, **9** (2020), 1206. https://doi.org/10.3390/electronics9081206
24. H. Xu, Y. Hu, W. Cao, L. Han, An improved jump spider optimization for network traffic identification feature selection, *Comput. Mater. Continua*, **16** (2023), 3239–3255. https://doi.org/10.32604/cmc.2023.039227
25. H. Xu, X. Chai, H. Liu, A multi-controller placement strategy for hierarchical management of software-defined networking, *Symmetry*, **15** (2023), 1520. https://doi.org/10.3390/sym15081520
26. H. Xu, Y. Lu, Q. Guo, Application of improved butterfly optimization algorithm combined with black widow optimization in feature selection of network intrusion detection, *Electronics*, **11** (2022), 3531. https://doi.org/10.3390/electronics11213531
27. P. Sun, H. Liu, Y. Zhang, Q. Meng, L. Tu, J. Zhao, An improved atom search optimization with dynamic opposite learning and heterogeneous comprehensive learning, *Appl. Soft Comput.*, **103** (2021), 107140. https://doi.org/10.1016/j.asoc.2021.107140
28. H. Jia, Q. Liu, Y. Liu, S. Wang, D. Wu, Hybrid aquila and harris hawks optimization algorithm with dynamic opposition-based learning (in Chinese), *J. Intell. Syst.*, **18** (2023), 104–116. https://doi.org/10.11992/tis.202108031
29. Y. Xu, Z. Yang, X. Li, H. Kang, X. Yang, Dynamic opposite learning enhanced teaching–learning-based optimization, *Knowl. Based Syst.*, **188** (2020), 104966. https://doi.org/10.1016/j.knosys.2019.104966
30. Y. Lai, H. Chen, F. Gu, A multitask optimization algorithm based on elite individual transfer, *Math. Biosci. Eng.*, **20** (2023), 8261–8278. https://doi.org/10.3934/mbe.2023360
31. E. Tanyildizi, A novel optimization method for solving constrained and unconstrained problems: Modified Golden Sine Algorithm, *Turk. J. Electr. Eng. Comput. Sci.*, **26** (2018), 3287–3304. https://doi.org/10.3906/elk-1802-232
32. E. Tanyildizi, G. Demir, Golden Sine Algorithm: A novel math-inspired algorithm, *Adv. Electr. Comput. Eng.*, **17** (2017), 71–78. https://doi.org/10.4316/AECE.2017.02010
33. Y. Guo, S. Liu, W. Gao, L. Zhang, Elite opposition-based learning golden-sine harris hawks optimization (in Chinese), *Comput. Eng. Appl.*, **58** (2022), 153–161. http://doi.org/10.3778/j.issn.1002-8331.2011-0321
34. P. Yuan, T. Zhang, L. Yao, Y. Lu, W. Zhuang, A hybrid golden jackal optimization and Golden Sine Algorithm with dynamic lens-imaging learning for global optimization problems, *Appl. Sci.*, **12** (2022), 9709. https://doi.org/10.3390/app12199709
35. J. M. Keller, M. R. Gray, J. A. Givens, A fuzzy k-nearest neighbor algorithm, *IEEE Trans. Syst. Man Cybernet.*, **SMC-15** (1985), 580–585. https://doi.org/10.1109/TSMC.1985.6313426
36. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey Wolf Optimizer, *Adv. Eng. Software*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

37. S. Mirjalili, A. Lewis, The Whale Optimization Algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
38. Canadian Institute for Cybersecurity, NSL-KDD dataset, 2009. Available from: https://www.unb.ca/cic/datasets/nsl.html.
39. M. S. Elsayed, N. A. Le-Khac, A. D. Jurcut, InSDN: A novel SDN intrusion dataset, *IEEE Access,* **8** (2020), 165263–165284. https://doi.org/10.1109/ACCESS.2020.3022633
40. L. Kou, S. Ding, T. Wu, W. Dong, Y. Yin, An intrusion detection model for drone communication network in SDN environment, *Drones*, **6** (2022), 342. https://doi.org/10.3390/drones6110342