Electronic Research Archive

ERA, 33(1): 231–254.
DOI: 10.3934/era.2025012
Received: 28 September 2024
Revised: 03 December 2024
Accepted: 10 December 2024
Published: 20 January 2025

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

Research article

An optimization-inspired intrusion detection model for software-defined networking

Hui Xu, Longtan Bai* and Wei Huang

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

*** Correspondence:** Email: 102301218@hbut.edu.cn.

Abstract: As an emerging network architecture, software-defined networking (SDN) has the core concept of separating the control plane from the network hardware and unifying its management by a central controller. Since the centralized control of SDN is such that an attack on the controller can lead to the paralysis of the entire network, intrusion detection has become particularly significant for SDN. Currently, more and more intrusion detection systems based on machine learning and deep learning are being applied to SDN, but most have drawbacks such as complex models and low detection accuracy. This paper proposes an enhanced spider wasp optimizer (ESWO) algorithm for feature dimensionality reduction of intrusion detection datasets and constructs a new intrusion detection model (IDM), namely ESWO-IDM, for SDN. The ESWO algorithm integrates multiple strategies, including tent chaotic map strategy and elite opposition learning strategy to improve the diversity of the population, Lévy flight strategy to prevent the algorithm from falling into local optimum in the early stage, and dynamic adjustment strategy of control parameters to balance exploration and exploitation of the algorithm. ESWO was empirically evaluated using eight benchmark test functions and four UCI datasets to comprehensively demonstrate its advantages. Binary and multiclassification experiments were conducted using the InSDN dataset to analyze the ESWO-IDM performance and compare it with other IDMs. The experimental results show that the ESWO-IDM achieves the best performance in all the metrics in both binary classification and multiclassification and has the most prominent effect on the detection of normal, denial of service (DoS), distributed DoS, and Brute Force Attack types, which effectively improves SDN intrusion detection from the viewpoint of optimization.

Keywords: software-defined networking; intrusion detection; optimization; spider wasp optimizer

1. Introduction

Software-defined networking (SDN) is a new type of network architecture that can be structured into three layers: forwarding plane, control plane, and application plane. SDN separates the data plane from the control plane and becomes more flexible and programmable, which improves the flexibility, real-time, and scalability of the network, optimizes communication and management in the network, and improves the efficiency of the entire industrial automation system [1]. Since SDN integrates traditional network functions and simplifies network configuration operations, it is more vulnerable to network attacks, and intrusion detection is an effective means to address SDN security issues [2]. With the development of machine learning, researchers have begun to use traditional machine learning algorithms (such as decision trees, support vector machines, etc.) to build a more complex intrusion detection system (IDS), which improves the accuracy and efficiency of intrusion detection. For intrusion detection systems in traditional networks, researchers have proposed many intrusion detection models (IDM) with good results, and nowadays many researchers apply traditional network intrusion detection methods to intrusion detection in SDNs. Ali et al. [3] designed a lightweight distributed denial of service (DDoS) attack detection and mitigation system based on an improved decision tree, which employs impurity culling and error pruning strategies to detect DDoS attack streams in the traffic and incorporates a dynamic whitelisting mechanism to block the attack traffic from entering the SDN. Madathi et al. [4] used a feature subset of the attack traffic to train the K-nearest neighbor (KNN) algorithm, which is then used to identify anomalous attacks by aggregating and classifying the attack traffic. Maheshwari et al. [5] proposed a novel optimized weighted voting integration model incorporating support vector machine (SVM), random forest (RF), and gradient boosting machine (GBM) for detecting DDoS attacks in SDN environments. Elsayed et al. [6] proposed a secured automatic two-level intrusion detection system (SATIDS) based on an improved long short-term memory (LSTM) network, which distinguishes between attacking and benign traffic and identifies the attack class. Al-Zewairi et al. [7] classified the attack types into type A and type B according to different characteristics and used shallow and deep artificial neural network (ANN) classifiers to detect these two types of unknown types; the results showed that it can effectively realize the problem of classifying the unknown attack types. Wang et al. [8] proposed an improved Naive Bayes (NB) model combined with the attribute addition algorithm, and the results show that the classification accuracy of the new model is significantly improved.

However, both independent machine learning detection methods and joint detection methods are often ineffective as the feature data used relies on manual labor. In order to improve the detection efficiency, researchers usually combine feature selection techniques to select the optimal features. In network intrusion detection systems, feature selection is one of the key steps in data preprocessing, which directly affects the efficacy and accuracy of the classifier. Feature selection aims to identify and select the most useful subset of features from the original dataset to improve model performance and reduce the risk of overfitting.

Cui et al. [9] proposed a feature selection method for high-dimensional data of SDN network data traffic to extract positive features that are effective for model decision-making and constructed a multiclass intrusion detection model based on multiple output nodes, which improved the accuracy of the model for emerging intrusion detection. Wang et al. [10] used RF information gain to select the optimal feature subset and a support vector machine for classification to construct a detection model with a high detection rate for DDoS. Sarica and Angin [11] proposed a solution for real-time intrusion

detection and mitigation in SDN based on automatic flow feature extraction and flow classification, while using a random forest classifier at the SDN application layer; experimental results showed that the proposed security method has broad prospects in the implementation of SDN-managed IoT networks. Zhang and Wang [12] proposed a network intrusion detection system (NIDS) based on feature selection and deep learning; the model uses methods such as filtering both non-IPv4-type data packets and consecutive identical packet-in messages and recovering missing information to remove a large number of redundant features and improve the detection accuracy. It is deployed in the SDN network for detecting abnormal traffic. However, the above-proposed methods have disadvantages such as large models, complex algorithms, low detection accuracy, and data that do not reflect the characteristics of the SDN environment.

Our prior work [13,14] involves the integration of classical metaheuristic optimization algorithms for feature selection. However, traditional meta-heuristic algorithms suffer from issues such as low search efficiency. Our prior work [15,16] also applies some new heuristic algorithms that integrate multiple strategies to intrusion detection and SDN networks, which effectively improves the performance of feature selection models and the efficiency of intrusion detection. Our prior work has focused on traditional network intrusion detection, and only preliminary experimental tests have been conducted for SDN intrusion detection.

Spider wasp optimizer (SWO) [17] is an emerging metaheuristic algorithm proposed by Abdel-Basset et al. in 2023. The SWO algorithm has several unique update strategies for a wide range of optimization problems with different exploration and exploitation needs. The SWO algorithm has been successfully applied to real-world optimization problems, one class being constrained engineering design problems such as WBD and pressure vessel design. The second class is to estimate the unknown parameters of the PV models including SDM, dual diode model (DDM), and triple diode model (TDM). In addition, Shtayat et al. [18] proposed an improved binary spider wasp optimization algorithm (IBSWO), which was experimentally verified to have a significant advantage over traditional heuristic algorithms in terms of performance and was applied to IIoT-IDS. The IBSWO primarily converts continuous SWO algorithms into binary SWO algorithms. IBSWO only improves the flat crossover in the SWO algorithm and does not optimize the SWO algorithm for its shortcomings in exploitation and exploration. Mohamed et al. [19] proposed an enhanced binary spider wasp algorithm (BSWO) and used it for high-dimensional feature selection, which effectively reduces the redundant structure of the data and improves the classification accuracy. Compared to SWO, the BSWO has higher time complexity and increases the time overhead. Although the SWO algorithm has strong global search and local exploitation capabilities, it suffers from the problems of unbalanced exploitation, slow convergence at later stages, and the possibility of getting stuck in local optimality. In order to overcome the problems of SWO mentioned above, to solve the challenges brought by facing high-dimensional data, and to improve the detection efficiency for anomalous traffic, in this paper, we propose a multi-strategy enhanced ESWO algorithm for feature selection and construct the intrusion detection model, which optimizes SDN intrusion detection. Tent chaotic map and elite opposition learning strategy increase the diversity and expand the search range of the population. The Levy flight strategy was introduced to avoid the algorithm falling into local optima early in the iteration. The dynamic parameter adjustment strategy balances the exploration and exploitation of the algorithm. The ESWO algorithm is applied to feature selection in intrusion detection to select the optimal subset, reduce data redundancy, and improve the detection efficiency of SDN intrusion detection.

The main contributions of this paper are as follows:

a) In order to improve the limitations of the SWO algorithm when applying intrusion detection for SDN, enhanced strategies are introduced for the ESWO algorithm.

b) An intrusion detection model based on ESWO for SDN is proposed, which uses the ESWO algorithm to solve the problem of data set feature redundancy in intrusion detection and improves the detection efficiency for abnormal attacks.

c) Since the ESWO algorithm is used to feature selection for intrusion detection, the binary version of ESWO is utilized to search for the optimal subset, and the intrusion detection model based on the ESWO algorithm is compared with other algorithms to verify the superiority of the proposed model in detecting abnormal attacks for SDN.

The remainder of this paper is organized as follows: Section 2 introduces the basic algorithm. Section 3 presents enhanced strategies based on the SWO algorithm. Section 4 presents the intrusion detection model based on the ESWO algorithm. Section 5 provides experiments and results. Section 6 summarizes the research results and future development directions of this paper.

2. Basic algorithm

The SWO algorithm mainly simulates the behaviors of female spider wasps, such as hunting, nesting, and laying an egg on the abdomen of the spider for parasitism, and uses mathematical models to simulate this behavior in various scenarios.

2.1. Initialization phase

In the SWO algorithm, each spider wasp (female) represents a solution of the current generation, and the mathematical expression is described as follows:

$$\overrightarrow{SW} = [x_1, x_2, x_3, \ldots, x_D] \tag{1}$$

$$\overrightarrow{SW_i^t} = \vec{L} + \vec{r} \times \left(\vec{H} - \vec{L}\right) \tag{2}$$

where t denotes the generation index, i indicates the population index (i = 1, 2, …, N), $\vec{r}$ is a vector of D-dimension randomly initialized numbers between 0 and 1, $\vec{H}$ is the upper bound of the pre-set parameter, $\vec{L}$ is the lower bound of the pre-set parameter, and $\overrightarrow{SW_i^t}$ represents the ith spider wasp in the t-th round of iteration.

2.2. Hunting and nesting behavior

2.2.1. Searching stage

In the initial stage, the female spider wasp randomly explores the search space with a constant step length to search for spider prey suitable for their offspring. In the t + 1 iteration of this stage, the position $\overrightarrow{SW_i^t}$ of the ith spider wasp mathematical expression is described as follows:

$$\overrightarrow{SW_i^{t+1}} = \overrightarrow{SW_i^t} + \mu_1 * \left(\overrightarrow{SW_a^t} - \overrightarrow{SW_b^t}\right) \tag{3}$$

where *a* and *b* are two indices selected at random from the population to determine the exploration direction, and μ_1 is the coefficient vector using the following formula:

$$\mu_1 = |rn| * r_1 \tag{4}$$

where r_1 is a number randomly generated at the interval of zero and one, and rn is also a random number but generated using the normal distribution.

Spider wasps sometimes search the entire area around where the spider has fallen, with mathematical expressions as follows:

$$\overrightarrow{SW_i^{t+1}} = \overrightarrow{SW_c^t} + \mu_2 * \left(\vec{L} + \overrightarrow{r_2} * \left(\vec{H} - \vec{L}\right)\right) \tag{5}$$

$$\mu_2 = B * cos(2\pi l) \tag{6}$$

$$B = \frac{1}{1+e^l} \tag{7}$$

where *c* is an index randomly selected from the population, and *l* is a number randomly generated between 1 and –2.

The mathematical expression of the position update of the spider wasp at this stage is as follows:

$$\overrightarrow{SW_i^{t+1}} = \begin{cases} Eq\ (3) & r_3 < r_4 \\ Eq\ (5) & otherwise \end{cases} \tag{8}$$

where r_3 and r_4 are two random numbers in [0, 1].

2.2.2. Following and escaping stage

After the spider wasp catches the target spider, sometimes the spider will escape, and the spider wasp will hunt. At first, the distance between the spider wasp and the spider is very small, and then it will increase or decrease according to the speed of the two. The expression of this behavior is as follows:

$$\overrightarrow{SW_i^{t+1}} = \overrightarrow{SW_i^t} + C * \left|2 * \overrightarrow{r_5} * \overrightarrow{SW_a^t} - \overrightarrow{SW_i^t}\right| \tag{9}$$

$$C = \left(2 - 2 * \left(\frac{t}{t_{max}}\right)\right) * r_6 \tag{10}$$

where *a* is an index randomly selected from the population, *t* and t_{max} indicate the current and maximum evaluation, respectively, $\overrightarrow{r_5}$ is a vector that represents the values randomly generated in the interval [0,1], and r_6 is a random number in the interval [0,1].

When the spider escapes the hunt of the spider wasp, the distance between the spider and the spider wasp gradually increases. The expression of this behavior is as follows:

$$\overrightarrow{SW_i^{t+1}} = \overrightarrow{SW_i^t} * \overrightarrow{vc} \tag{11}$$

$$k = 1 - \left(\frac{t}{t_{max}}\right) \tag{12}$$

where $\overrightarrow{vc}$ is a vector generated between k and $-k$ according to the normal distribution.

The two behaviors above are performed randomly with the expression as follows:

$$\overrightarrow{SW_i}^{t+1} = \begin{cases} Eq\ (9) & r_3 < r_4 \\ Eq\ (11)\ otherwise \end{cases} \tag{13}$$

At the start of the optimization process, the spider wasp performs a global search for the optimization problem in the search phase, and the algorithm explores and exploits the areas around the current wasps during the subsequent iteration, The expression for adjusting the two stages is as follows:

$$\overrightarrow{SW_i}^{t+1} = \begin{cases} Eq\ (8) & p < k \\ Eq\ (13) otherwise \end{cases} \tag{14}$$

where p is a random number in [0,1].

2.2.3. Nesting behavior

After the spider wasp catches the target prey, it will drag the prey into the preprepared nest. The spider wasp has different nesting behaviors. The expressions are as follows:

$$\overrightarrow{SW_i}^{t+1} = \overrightarrow{SW}^* + cos(2\pi l) * \left(\overrightarrow{SW}^* - \overrightarrow{SW_i}^t\right) \tag{15}$$

$$\overrightarrow{SW_i}^{t+1} = \overrightarrow{SW_a}^t + r_3 * |\gamma| * \left(\overrightarrow{SW_a}^t - \overrightarrow{SW_i}^t\right) + (1 - r_3) * \vec{U} * \left(\overrightarrow{SW_b}^t - \overrightarrow{SW_c}^t\right) \tag{16}$$

$$\vec{U} = \begin{cases} 1 & \overrightarrow{r_4} > \overrightarrow{r_5} \\ 0\ otherwise \end{cases} \tag{17}$$

$$\overrightarrow{SW_i}^{t+1} = \begin{cases} Eq\ (15) & r_3 < r_4 \\ Eq\ (16)\ otherwise \end{cases} \tag{18}$$

where $\overrightarrow{SW}^*$ represents the best-so-far solution, r_3 and r_4 is a random number created in the interval [0,1], γ is a number generated according to the Lévy flight, a, b, and c are indices of three solutions randomly selected from the population, $\vec{U}$ is a binary vector used to determine when a step size is applied to avert building two nests at the same position, and $\overrightarrow{r_4}$ and $\overrightarrow{r_5}$ are two vectors that represent the random values in the interval [0,1]; if an element of $\overrightarrow{r_4}$ is larger than the corresponding element of $\overrightarrow{r_5}$, the value of the element corresponding to $\vec{U}$ is 1, otherwise the value is 0.

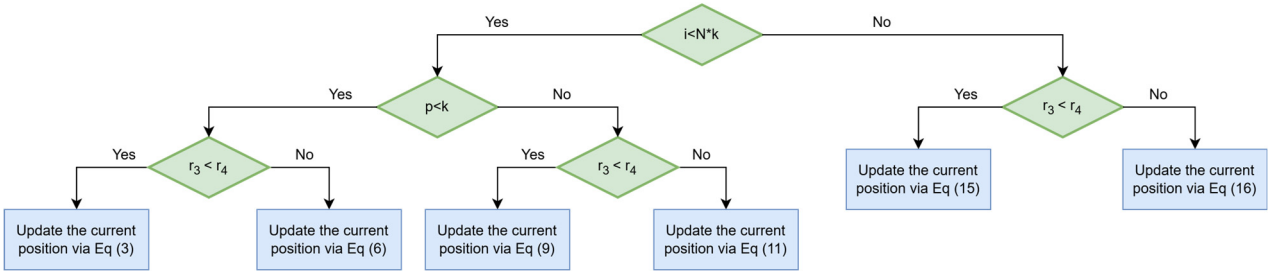

Figure 1. Flowchart of the hunting and nesting behaviors in the SWO algorithm.

The flowchart of the hunting and nesting behavior is shown in Figure 1. In summary, the expression of the whole behavior is as follows:

$$\overrightarrow{SW_i^{t+1}} = \begin{cases} Eq\ (14) & i < N * k \\ Eq\ (18) & otherwise \end{cases} \tag{19}$$

2.3. Mating behavior

One of the main characteristics of spider wasps is their ability to determine gender. Gender is determined based on the size of the host in which an egg is laid. In this algorithm, each spider wasp is the solution of the current generation, and the wasp egg represents the potential solution. The expressions for generating the wasp egg are as follows.

$$\overrightarrow{SW_i^{t+1}} = Crossover\left(\overrightarrow{SW_i^t}, \overrightarrow{SW_m^t}, CR\right) \tag{20}$$

$$\overrightarrow{SW_m^{t+1}} = \overrightarrow{SW_i^t} + e^l * |\beta| * \overrightarrow{v_1} + (1 - e^l) * |\beta_1| * \overrightarrow{v_2} \tag{21}$$

$$\overrightarrow{v_1} = \begin{cases} \overrightarrow{x_a} - \overrightarrow{x_i} & f(\overrightarrow{x_a}) < f(\overrightarrow{x_i}) \\ \overrightarrow{x_i} - \overrightarrow{x_a} & otherwise \end{cases} \tag{22}$$

$$\overrightarrow{v_2} = \begin{cases} \overrightarrow{x_b} - \overrightarrow{x_c} & f(\overrightarrow{x_b}) < f(\overrightarrow{x_c}) \\ \overrightarrow{x_c} - \overrightarrow{x_b} & otherwise \end{cases} \tag{23}$$

where *Crossover* indicates the uniform crossover operator that is applied between solutions $\overrightarrow{SW_i^t}$ and $\overrightarrow{SW_m^t}$ with a probability known as crossover rate (*CR*), and $\overrightarrow{SW_m^t}$ and $\overrightarrow{SW_i^t}$ are two vectors that represent the male and female spider wasps, respectively. β and β_1 are two numbers randomly generated according to the normal distribution, *e* is the exponential constant, and *a*, *b*, and *c* are indices of three solutions randomly selected from the population; *a*, *b*, *c* and *i* are all different.

2.4. Population reduction and memory saving

When the spider wasp lays eggs in the host, the nest will be closed, which also indicates that the role of the spider wasp in the optimization process has been completed. In the remaining optimization process, the evaluation of their functions will be handed over to other spider wasps, which is conducive to better results. The number of new populations is updated by the following formula.

$$N = N_{min} + (N - N_{min}) \times k \tag{24}$$

where N_{min} indicates the minimum number of the population employed to avoid being stuck into local minima within the different stages of the optimization process.

3. Enhanced strategies

The SWO algorithm has multiple unique exploration strategies and is characterized by fast search speed and high solution accuracy. However, due to its multiple random exploration strategies, the late convergence of the algorithm is slow and can lead to falling into the local optimum. Therefore, in this paper, a tent chaotic map is introduced in the initialization stage, which makes the initialized population more widely distributed and expands the search range. At the beginning of each round of iteration, an elite reverse learning strategy is introduced, which allows the quality of the population to be enhanced throughout the iteration by retaining the better individuals. In the capture and escape phase of the algorithm, the Lévy flight strategy is strengthened to prevent the algorithm from falling into a local optimum in the early stages. The control parameters in the algorithm are also adjusted to enhance the overall optimization ability of the algorithm.

3.1. Tent chaotic map strategy

Chaos has randomness and ergodicity, which can accelerate the convergence of the algorithm [20]. The chaotic sequence is generated by a tent map, so that the initial solution is evenly distributed in the search space. The expression of the tent map is shown as follows:

$$X_{i+1} = \begin{cases} \frac{X_i}{a} & X_i < a \\ (1 - X_i)/(1 - a) & X_i \geq a \end{cases} \tag{25}$$

where i is the number of iterations, $a \in (0,1)$.

3.2. Elite opposition learning strategy

In the population, elite individuals have more effective information than ordinary individuals, and the generation of elite individuals can increase the diversity of the population. The strategy first performs reverse learning on each individual X_i in the population to obtain the reverse solution OP_i [21]. The formula for generating the reverse solution is as follows:

$$OP_i = k * (ub - lb) - X_i \tag{26}$$

where $k \in (0,1)$ and lb and ub are the lower and upper bounds of search space, respectively. By comparing the fitness value of X_i and its reverse solution OP_i, the elite individuals are obtained by retaining the smaller fitness value. The formula is as follows:

$$X_i = \begin{cases} X_i & fitness(X_i) < fitness(OP_i) \\ OP_i & otherwise \end{cases} \tag{27}$$

3.3. Strengthened Lévy flight strategy

In the early stage of the pursuit and escape phase of the SWO algorithm, due to the large control factor C of the speed, the algorithm will perform local exploration. As the number of iterations increases, the control factor C gradually decreases, and the algorithm will be transformed into a global exploration at this stage. Therefore, the Lévy flight strategy is also introduced at this stage. The Lévy

flight strategy [22] will enhance the spatial search ability of the algorithm and prevent it from falling into a local optimum in the early stages. The mathematical expressions after the introduction of Lévy flight are as follows:

$$\overrightarrow{SW_i^{t+1}} = \overrightarrow{SW_i^t} + C * \left|2 * levy(\alpha) * \overrightarrow{SW_a^t} - \overrightarrow{SW_i^t}\right| \tag{28}$$

$$levy(\alpha) = 0.05 * \frac{u}{|v|^{1/\beta}} \tag{29}$$

$$u \sim N(0, \sigma_u^2) \tag{30}$$

$$v \sim N(0, \sigma_v^2) \tag{31}$$

$$\sigma_u = \left\{\frac{\Gamma(1+\beta)sin\left(\frac{\pi\times\beta}{2}\right)}{\Gamma(1+\beta)\times\beta\times 2^{\frac{\beta-1}{2}}}\right\}^{\frac{1}{\beta}}, \sigma_v = 1 \tag{32}$$

where $levy(\alpha)$ is the step length, u and v follow the normal distribution with mean of 0 and variance of σ_u^2 and σ_v^2, and the value of β is generally 1.5.

3.4. Dynamic adjustment strategy of control parameters

The basic algorithm has two important control parameters *TR* and *CR*. *TR* controls the occurrence of hunting and nesting behavior and mating behavior. According to the sensitivity measurement results of *TR*, it can be seen that increasing *TR* can improve the performance of the single-peak test function, and the optimal range of parameters is 0.3–0.5. Reducing *TR* can improve the performance of multi-modal test functions, and the optimal range of parameters is 0.1–0.3, so the *TR* value of the basic algorithm is 0.3. In order to improve the comprehensive performance of the algorithm, *TR* is changed to a dynamic parameter that changes with the number of iterations [23]. The expression is as follows:

$$TR = 0.3 * \left(1 - \frac{t}{t_{max}}\right) + 0.2 \tag{33}$$

3.5. ESWO algorithm pseudocode

The ESWO algorithm is constructed based on the SWO algorithm and consists of four main sections. In the initiation phase of the algorithm, the tent chaotic map strategy is introduced to increase the diversity of the population. The algorithm's global search capability is improved by utilizing an elite opposition learning strategy at the beginning of each iteration of the algorithm. The inclusion of the Lévy flight strategy in the pursuit and escape phase prevents the algorithm from converging at an early stage. Finally, the dynamic adjustment strategy of control parameters balances the exploration and exploitation of algorithms. The pseudo-code of the ESWO algorithm is shown in Algorithm 1.

Algorithm 1. Pseudo-code of the ESWO

Input: N, N_{min}, t_{max}

Output: $\overrightarrow{SW}^*$

Initialize *N* female wasp $\overrightarrow{SW_i^t}(i = 1,2, \dots, N)$,using Eq (2)
Evaluate each $\overrightarrow{SW_i^t}$ and finding the onewith the best fitness in $\overrightarrow{SW}^*$
t=1
while$(t < t_{max})$
    Updating population using Eqs (26) and (27)
    Updating *TR* using Eq (33)
    r_6:generating a random number between 0 and 1
    If$(r_6 < TR)$
      for *i=1:N* **do**
        if *i* < *N***k* **then**
          if $p < k$ **then**
            if $r_3 < r_4$ **then**
              Applying Eq (3)
            else
              Applying Eq (5)
            end if
          else
            if $r_3 < r_4$ **then**
              Applying Eq (28)
            else
              Applying Eq (11)
            end if
          else
            if $r_3 < r_4$ **then**
              Applying Eq (15)
            else
              Applying Eq (16)
            end if
          end if
        Compute $f(\overrightarrow{SW_i})$
        t=t+1;
      End for
    Else
      for *i=1:N*
        Applying Eq (20)
        t=t+1;
      End for
    End if
    Applying Memory Saving and updating ***N*** using Eq (24)
End while

3.6. Time complexity

Time complexity is a critical performance metric that measures the operation efficiency of an algorithm. Time complexity is crucial for the timely detection of attacks in intrusion detection [24]. The computing speed of the algorithm affects the detection efficiency of the whole intrusion detection system. Therefore, enhancing the performance of the algorithms is also required to minimize the increase in time complexity. The time complexity of the ESWO algorithm is determined by the population size *N*, the maximum function evaluation t_{max}, and the number of dimensions *D*. The time complex of the SWO is as follows:

$$T(SWO) = O(Hunting\ and\ nesting\ behaviors) + O(Mating\ behaviors) \tag{34}$$

$$T(SWO) = O(t_{max}DN) + O(t_{max}DN) = O(t_{max}DN) \tag{35}$$

Compared to SWO, the extra time overhead of the ESWO algorithm is concentrated in the elite opposition learning strategy. The time complex of the ESWO is as follows:

$$T(SWO) = O(Elite\ opposition\ learning\ strategy) + O(Hunting\ and\ nesting\ behaviors) + O(Mating\ behaviors) \tag{36}$$

$$T(SWO) = O(t_{max}DN) + O(t_{max}DN) + O(t_{max}DN) = O(t_{max}DN) \tag{37}$$

As mentioned above, the proposed ESWO algorithm has the same time complexity as the SWO algorithm. The time complexity of the PSO algorithm used in the subsequent experimental section is $O(t_{max}DN)$ [25]. Similarly, the time complexity of the WOA algorithm is $O(t_{max}DN)$. The population size N_{PSO} and N_{WOA} is constant during the iteration process, whereas the population size N_{SWO} and N_{ESWO} becomes progressively smaller during the iteration process due to the population reduction and memory saving. So, the overall time complexity of ESWO and SWO is less than PSO and WOA.

4. Proposed intrusion detection model

This section discusses the proposed IDM to detect the attack in SDN. In this paper, the ESWO algorithm is used to perform feature selection on the data to select the optimal subset of features and construct the intrusion detection model namely ESWO-IDM for SDN. The construction of the intrusion detection model can be divided into four phases: data preprocessing phase, feature selection phase, training phase, and classification phase. The intrusion detection model construction is shown in Figure 2.

1) Data preprocessing phase

Generally, datasets contain a large number of redundant features, and each data entry contains attributes that represent all the fields of the complete data package. These attributes are available in various forms, such as character types, numeric types, etc. Therefore, it is necessary to transform the unprocessed data into useful information. The data preprocessing stage mainly includes data cleaning, label encoding, and normalization. After the data is preprocessed, the processed data is divided into a training set and a test set in a ratio of 7:3.

Data cleaning: Data cleaning is the process of eliminating or correcting inaccurate, duplicate, or incomplete records and filling in missing values in the data set provided, which is used to improve the accuracy of predictions.

Label encoding: The dataset usually contains categorical variables, and such variations cannot be recognized by machine learning algorithms, so categorical variables need to be transformed into numerical variables.

Normalization: In the original dataset, attributes differed in scale, which resulted in significant differences in the magnitude of attribute values. Normalization scales all attribute values in the dataset to a range of 0 to 1 and mitigate the adverse effects of value differences on model training.

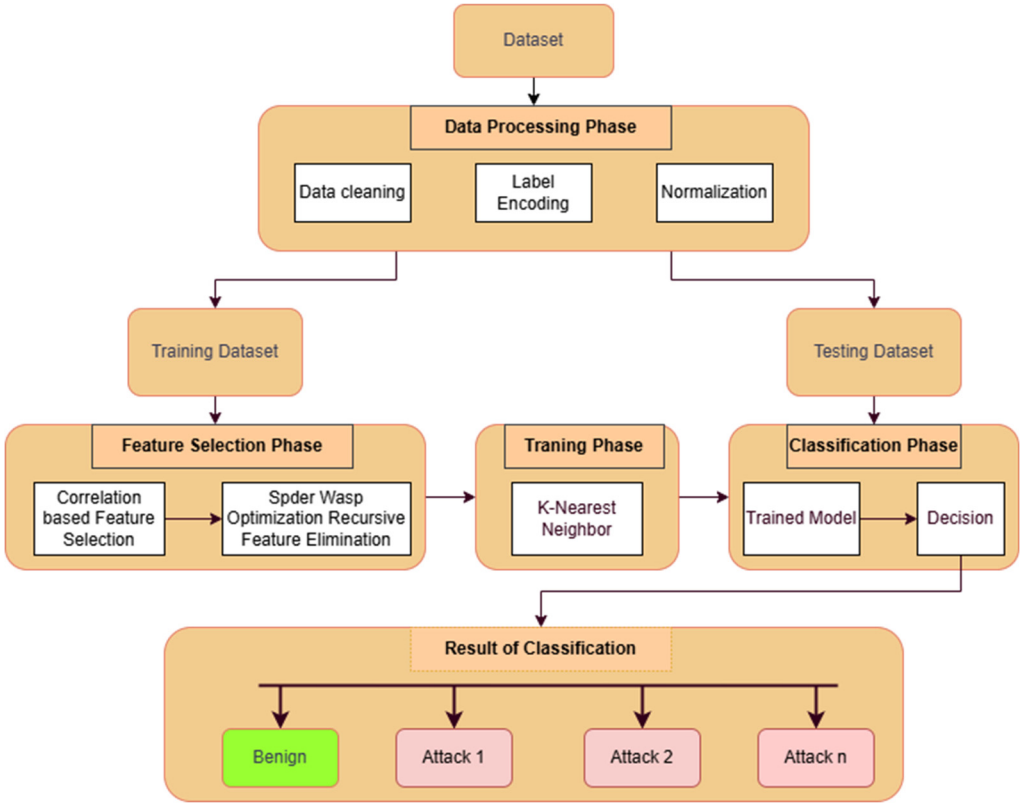

Figure 2. Proposed ESWO-IDM for SDN.

2) Feature selection phase

After grabbing datasets from the web and cleaning the data, simple data filtering has taken shape to some extent. However, since network data is usually very large, there are still many redundant features that are not visible to the naked eye. Feature selection allows further processing of the dataset before intrusion detection. This can effectively reduce redundant features and the amount of data and improve detection accuracy.

In this phase, feature selection is performed on the preprocessed dataset using an intelligent optimization algorithm. Spider wasps are constantly engaged in hunting, nesting, and mating behaviors. When the iteration ends, an optimal subset is finally obtained. The specific flowchart of the ESWO algorithm for feature selection is shown in Figure 3.

3) Training phase

In the training phase, a suitable classifier needs to be chosen. In [26], authors measured the performance of many classifiers, such as SVM, KNN, XGBoost, and RF. The KNN algorithm has the advantages of high accuracy, insensitivity to outliers, and a fast computation time. So, in this paper,

KNN is selected to establish the classification model. The training set is used as input to the model that trains the optimized KNN model.

4) Classification phase

Based on the results of the training phase, the final trained model is used to predict the test set and identify the benign types and various attack types. Finally, the results related to SDN intrusion detection are outputted.

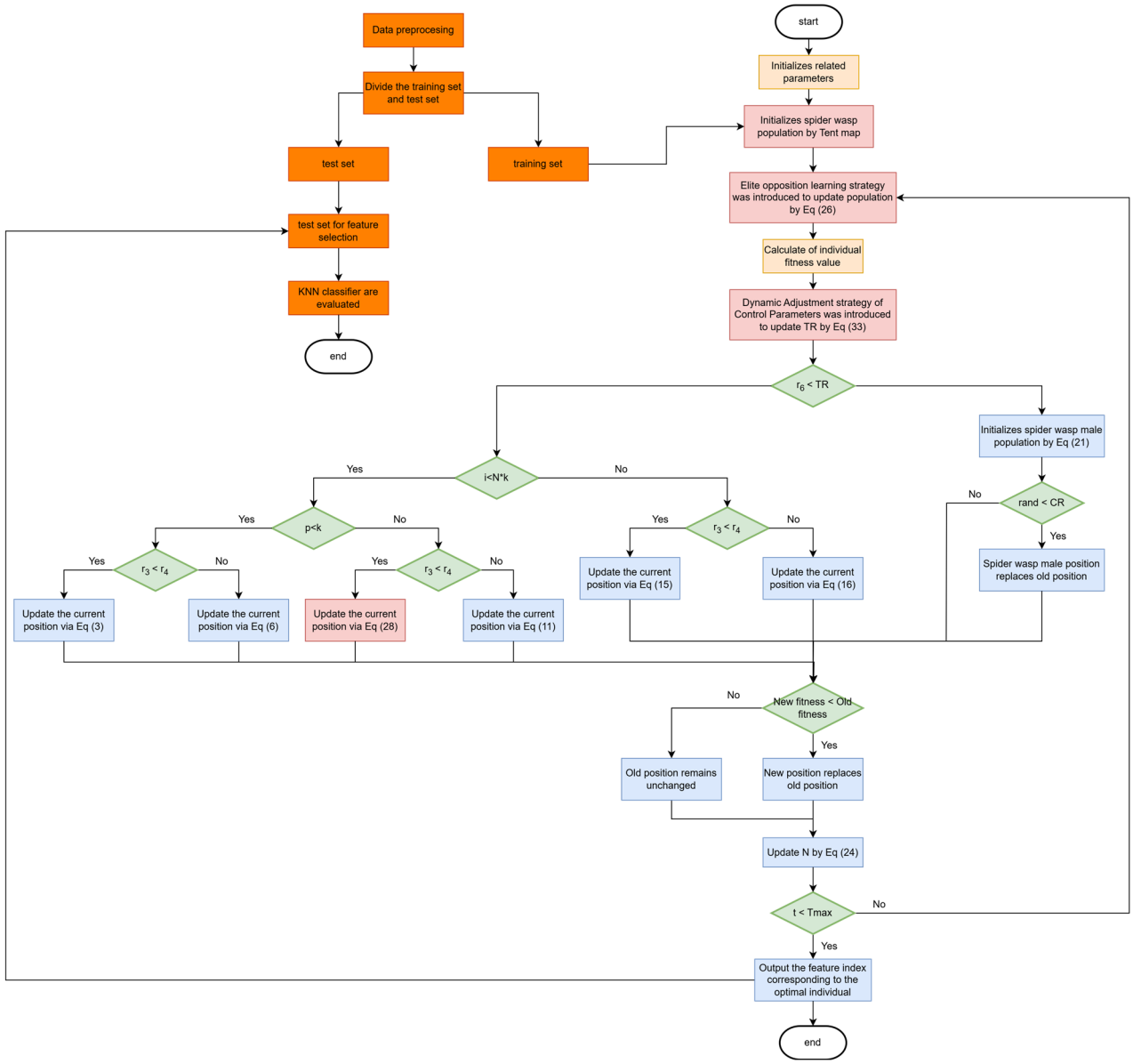

Figure 3. Feature selection flowchart of ESWO algorithm.

5. Experiments and results

5.1. Benchmark function testing

The experimental test is carried out in a single environment to ensure the objectivity and fairness of

the experiment. An Intel Core i7-13650HX CPU @ 3.00 GHz processor type, Windows 11 operating system, and MATLAB 2022b programming language were used in the experimental settings. In this experiment, eight common benchmark functions were selected, including four unimodal functions (F1–F4) and four multimodal functions (F5–F8). The experiment involves a certain degree of randomness. The test function in the experiment runs independently 30 times. The benchmark test function is shown in Table 1.

Table 1. Test functions.

Function expressions	Dimension	Range	Min
$F_1(X) = \sum_{i=1}^{n} (\sum_{j=1}^{i} X_j)^2$	30	[100,100]	0
$F_2(X) = \max\{X_i\|, 1 \ll i \ll n\}$	30	[−100,100]	0
$F_3(X) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	30	[−30,30]	0
$F_4(X) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[−100,100]	0
$F_5(X) = \left[x_i^2 - 10cos(2\pi x_i) + 10\right]$	30	[−5.12,5.12]	0
$F_6(X) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e$	30	[−32,32]	0
$F_7(X) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600,600]	0
$F_8(X) = \frac{\pi}{n}\{10sin^2(\pi y_i) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(X_i, 10, 100, 4),$ $y_i = 1 + \frac{1}{4}(X_i + 1)$ $u(X_i, a, k, m) = \begin{cases} k(X_i - a)^m, X_i > a \\ 0, -a \le X_i \le a \\ k(-X_i - a)^m, X_i < -a \end{cases}$	30	[−50,50]	0

In this paper, the ESWO algorithm is compared with the basic SWO algorithm, WOA, and PSO. In order to compare different algorithms, the original iteration number in SWO and ESWO is changed from individual iteration to population iteration. The population number is set to 30, and the number of iterations is 1000 times, which are tested in single-peak and multi-peak functions. The final test results are shown in Figure 4 and Table 2.

The experimental results show that the adaptation change curve of ESWO is progressively optimized, which avoids falling into local optimum as occurs with PSO and WOA. According to the detailed data in Table 2, the standard deviation of ESWO in the functions F5–F7 is 0, which indicates that ESWO has strong stability. The optimization effect of ESWO is also significantly higher than that of other algorithms in the other functions. By combining all eight functions, ESWO achieves comprehensive optimization in performance. Compared with the original algorithm, the optimization accuracy improves a lot.

In summary, experiments of benchmark function testing show that ESWO optimizes the exploration ability of the algorithm compared to SWO and overcomes the disadvantage of slow convergence at the later stages of SWO iterations, as well as further improving the optimization

seeking ability of the algorithm.

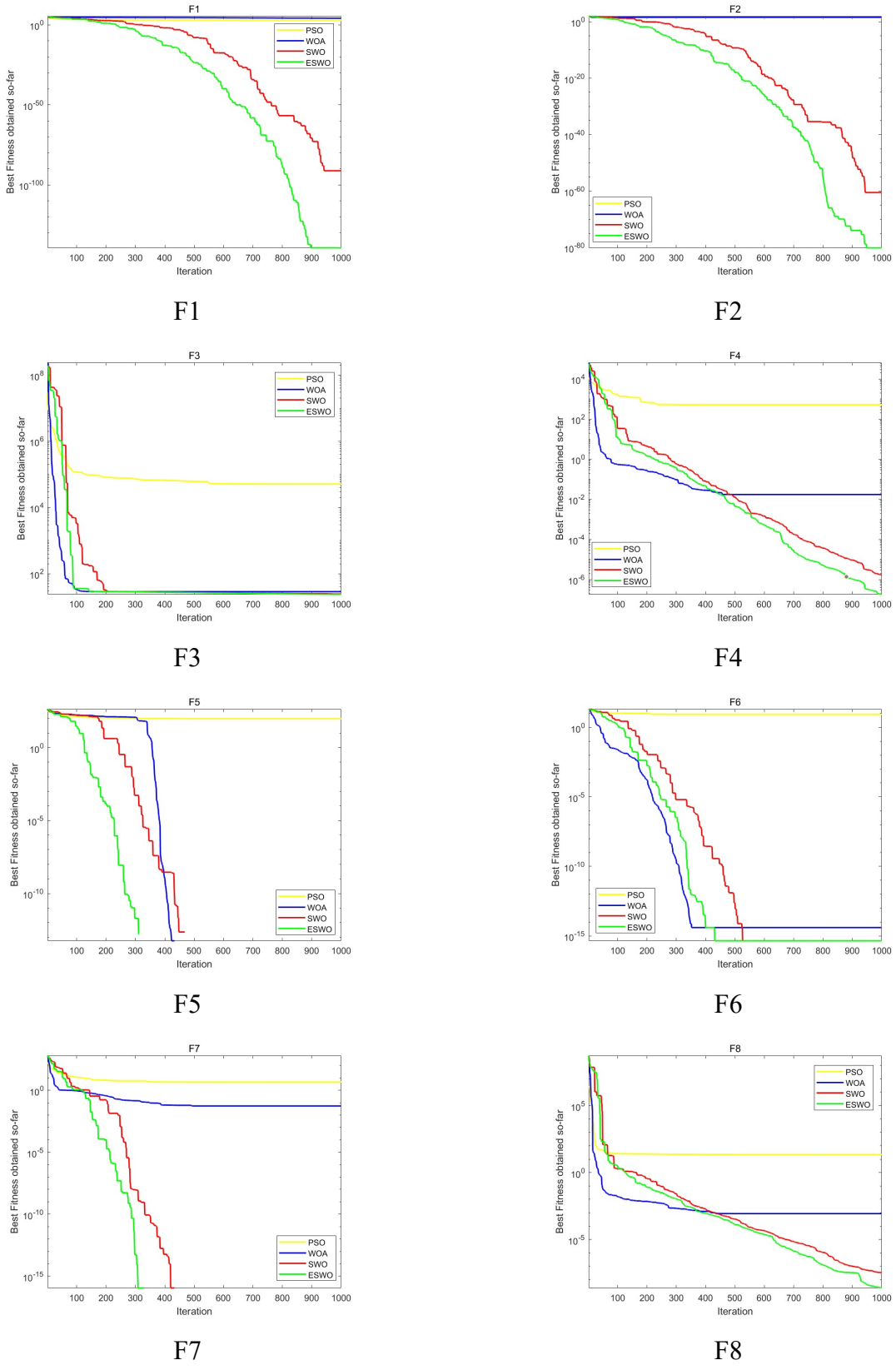

Figure 4. Convergence curve of fitness.

Table 2. Benchmark function results.

Function	Type	PSO	WOA	SWO	ESWO
F1	Min	$3.39 \times 10^{+02}$	$3.87 \times 10^{+03}$	8.47×10^{-140}	3.77×10^{-160}
	Ave	$3.13 \times 10^{+03}$	$1.97 \times 10^{+04}$	1.52×10^{-76}	1.87×10^{-106}
	Std	$5.53 \times 10^{+03}$	$9.55 \times 10^{+03}$	8.30×10^{-76}	1.02×10^{-105}
F2	Min	$6.45 \times 10^{+00}$	$1.37 \times 10^{+00}$	8.41×10^{-70}	2.70×10^{-88}
	Ave	$1.49 \times 10^{+01}$	$3.71 \times 10^{+01}$	1.62×10^{-42}	2.55×10^{-58}
	Std	$4.38 \times 10^{+00}$	$2.47 \times 10^{+01}$	8.86×10^{-42}	1.40×10^{-57}
F3	Min	$2.13 \times 10^{+03}$	$2.64 \times 10^{+01}$	$2.31 \times 10^{+01}$	$2.27 \times 10^{+01}$
	Ave	$3.67 \times 10^{+04}$	$2.71 \times 10^{+01}$	$2.39 \times 10^{+01}$	$2.35 \times 10^{+01}$
	Std	$3.80 \times 10^{+04}$	5.91×10^{-01}	4.33×10^{-01}	4.14×10^{-01}
F4	Min	$1.46 \times 10^{+02}$	5.17×10^{-03}	3.98×10^{-08}	5.50×10^{-08}
	Ave	$5.84 \times 10^{+02}$	6.08×10^{-02}	8.32×10^{-07}	1.70×10^{-06}
	Std	$2.90 \times 10^{+02}$	5.85×10^{-02}	7.91×10^{-07}	2.17×10^{-06}
F5	Min	$6.89 \times 10^{+01}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	Ave	$1.14 \times 10^{+02}$	7.58×10^{-15}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	Std	$2.40 \times 10^{+01}$	2.47×10^{-14}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
F6	Min	$5.60 \times 10^{+00}$	4.44×10^{-16}	4.44×10^{-16}	4.44×10^{-16}
	Ave	$8.49 \times 10^{+00}$	3.52×10^{-15}	4.44×10^{-16}	4.44×10^{-16}
	Std	$1.42 \times 10^{+00}$	2.42×10^{-15}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
F7	Min	$2.35 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	Ave	$5.80 \times 10^{+00}$	6.64×10^{-03}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	Std	$2.83 \times 10^{+00}$	2.05×10^{-02}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
F8	Min	$3.06 \times 10^{+00}$	5.40×10^{-04}	2.08×10^{-09}	1.21×10^{-09}
	Ave	$1.26 \times 10^{+01}$	7.15×10^{-03}	4.07×10^{-08}	2.18×10^{-08}
	Std	$7.37 \times 10^{+00}$	6.45×10^{-03}	4.14×10^{-07}	1.06×10^{-08}

5.2. Feature selection fitness function

The number of features selected in feature selection and the classification error rate determine the ability of the feature subset. Therefore, the fitness function expression is as follows:

$$fitness = \alpha \cdot error + \beta \cdot \frac{|Nf|}{T} \tag{38}$$

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, and α and β are two parameters in which α = 0.99 and β = 0.01.

The SWO algorithm is supposed to address continuous optimization problems. In the space of feature selection, the feature is binary in nature, and it will only appear in two cases: selected and unselected. So, we use the sigmoid function to convert the original solution into a binary solution. Each individual spider wasp is denoted as $SW_i = \{x_{i1}, x_{i2}, \ldots, x_{iD}\}$, where D is the feature dimension and d is a random number in [0,D]. When $x_{id} = 1$, the feature is selected. Otherwise, the feature is not selected. The sigmoid formula is as follows:

$$sigmoid(x_{id}) = \frac{1}{1+e^{-x_{id}}} \tag{39}$$

$$x_{id} = \begin{cases} 1, & rand > sigmoid(x_{id}) \\ 0, & rand < sigmoid(x_{id}) \end{cases} \tag{40}$$

The classifier is KNN. The evaluation indexes of classification detection are accuracy, recall, precision, and F1-score, with unit %.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{41}$$

$$Recall = \frac{TP}{TP+FN} \tag{42}$$

$$Precision = \frac{TP}{TP+FP} \tag{43}$$

$$F1-score = \frac{2*TP}{2*TP+FP+FN} \tag{44}$$

5.3. UCI datasets

In this paper, UCI datasets are used to evaluate the effectiveness of ESWO in dimensionality reduction. In this section, four datasets are selected, namely Ionosphere, WDBC, Heartstatlog, and Sonar. The measurement results of the four datasets are shown in Table 3.

Table 3. Test result of the UCI datasets.

Algorithms	Metrics	Ionosphere	WDBC	Heartstatlog	Sonar
PSO	Accuracy (%)	92.381	97.647	83.951	88.710
	Recall (%)	96.807	95.238	86.667	93.103
	Precision (%)	90.411	98.361	84.783	84.375
	F1-score (%)	94.286	96.774	85.714	88.525
	Computational time (s)	9.124	10.934	12.013	10.730
WOA	Accuracy (%)	90.476	97.647	85.185	88.710
	Recall (%)	95.522	96.825	86.667	82.759
	Precision (%)	90.141	96.825	84.783	92.308
	F1-score (%)	92.754	96.825	85.714	87.273
	Computational time (s)	4.278	5.343	5.892	4.942
SWO	Accuracy (%)	92.381	97.647	83.951	90.323
	Recall (%)	95.522	95.238	88.889	**96.552**
	Precision (%)	**92.754**	98.361	83.333	84.848
	F1-score (%)	94.118	96.774	86.022	90.323
	Computational time (s)	3.318	3.959	4.149	3.612
ESWO	Accuracy (%)	**93.333**	**99.412**	**87.654**	**96.774**
	Recall (%)	**98.507**	**98.413**	**91.111**	93.103
	Precision (%)	91.667	**1**	**87.234**	**1**
	F1-score (%)	**94.964**	**99.20**	**89.130**	**96.429**
	Computational time (s)	3.342	4.307	4.543	3.967

Based on the measurement results of four datasets, ESWO is superior to the other algorithms in four evaluation indicators in the WDBC and Heartstatlog datasets. In the Ionosphere dataset, accuracy, recall, and F1-score of the ESWO algorithm are higher than those of the other algorithms. Although precision is lower than for SWO, it is still 1% higher than that of PSO and WOA. In a comprehensive comparison, the ESWO is still the best of the four algorithms. For the Sonar dataset, the recall of ESWO is 3% lower than that of SWO, but other indicators are much higher, and the effect is more than 5% higher. In addition, according to the computation time results, the SWO algorithm is faster than PSO and WOA on the four datasets. The computation time of the ESWO algorithm increased but is still faster than PSO and WOA.

In summary, the ESWO algorithm has the best comprehensive performance in the four datasets, and the accuracy and F1-score values are higher than other algorithms, indicating that ESWO can accurately classify the samples and has better classification performance.

5.4. InSDN dataset

The InSDN dataset is a public dataset specifically for SDN intrusion detection, which aims to improve the performance evaluation and research of IDS in the SDN environment [27,28]. The InSDN dataset not only contains data on different attack types but also covers various key parts of the SDN platform, making the dataset a comprehensive assessment of IDS performance.

This section uses the intrusion detection model based on the ESWO algorithm to perform binary and multiclassification experiments on InSDN. The InSDN dataset contains three files, one of which contains normal data and one other contains attack-type data. In [27], the author selected a subset of 48 features from InSDN for intrusion detection, which are important for detecting attacks. So, we selected the same 48 features for feature selection. The binary classification experiment was performed first. Initially, some redundant feature columns are deleted, the label "normal" is assigned to 0, and the label of other attack-type data is assigned to 1. Then, the data is normalized, and more than 10,000 data entries are randomly selected from the dataset for experimental testing.

Table 4 provides the binary classification results for the InSDN dataset. Results reveal that the ESWO-IDM outperforms other IDMs with 98.833% accuracy, 98.976% precision, and 98.946% F1-score. Accuracy and precision are optimized by 0.2–0.3% over other algorithms, and F1 values are improved by 0.1–0.25%. The four indicators of the ESWO show that the four algorithms are optimal, and the ESWO has the least number of features, which effectively reduces redundant features and workload and improves detection efficiency. Table 4 shows the computational time (CT) of binary classification. Due to the multiple strategies incorporated into the ESWO algorithm, the computation time is slightly improved compared to SWO, 70 seconds faster than PSO, and 13 seconds faster than WOA. Therefore, ESWO-IDM has a faster operation speed. Table 5 shows the construction and prediction time of different models with different data volumes after the feature selection phase. Due to feature selection, which eliminates many redundant features, ESWO-IDM has the fastest prediction time. As the amount of data increases, the overall time savings also increases. The experimental results show the advantages of the ESWO algorithm for feature selection, while the ESWO-IDM improves detection efficiency of intrusion detection.

Table 4. Binary classification results of the InSDN dataset.

Metrics	PSO	WOA	SWO	ESWO
Accuracy (%)	98.667	98.567	98.683	**98.833**
Recall (%)	**99.037**	98.916	98.856	98.916
Precision (%)	98.562	98.501	98.767	**98.976**
F1-score (%)	98.799	98.708	98.811	**98.946**
Number of features	11	10	8	**6**
Computational time (sec)	106.687	49.561	**34.579**	36.296

Table 5. Construction and prediction times of different models (s).

Data volumes	PSO	WOA	SWO	ESWO
3000	0.03269	0.01511	0.01302	**0.01079**
6000	0.05890	0.27954	0.02319	**0.01976**
10,000	0.15484	0.03281	0.03551	**0.03002**

Figures 5–7 show experimental results in the InSDN dataset. Although the ROC curve and the AUC result of the four algorithms are extremely similar and all close to excellent, the ESWO algorithms are still slightly better than the others, reflecting the authenticity of the detection method. The fitness curve is also gradually increasing, showing excellent ability to find the best value. Even in the later stage of iteration, the new optimal value will still be found, and the optimal value is the best of the four algorithms. WOA, PSO, and SWO algorithms fall into the local optimum in the later stage of iteration, which indicates a better optimization ability of the ESWO algorithm.

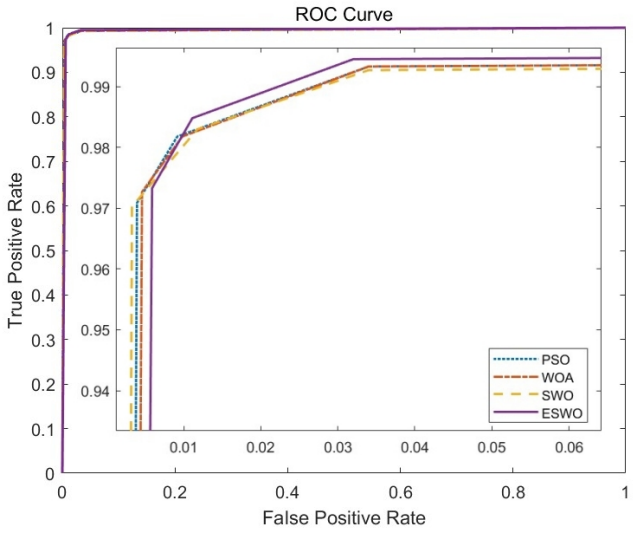

Figure 5. ROC curve of the InSDN dataset.

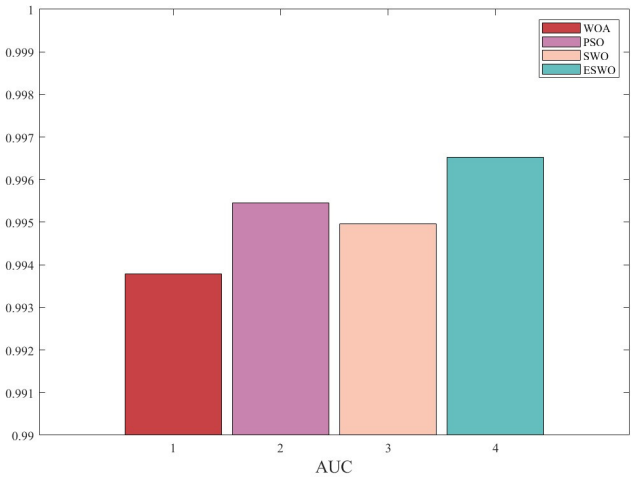

Figure 6. AUC result of the InSDN dataset.

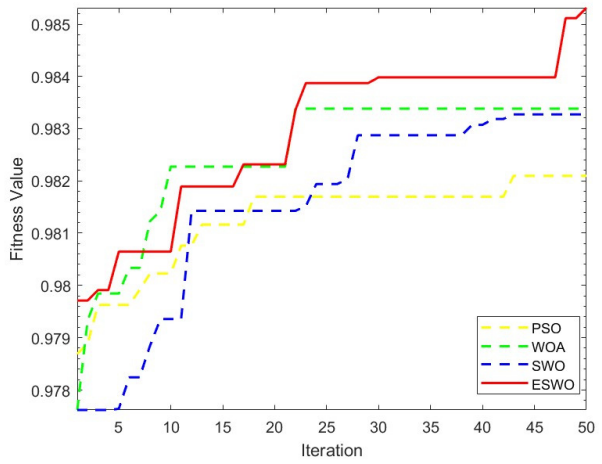

Figure 7. Fitness curve of the InSDN dataset.

For multiclassification experiments on the InSDN dataset, the labels normal, DoS, DDoS, probe, Brute Force Attack (BFA), web-attack, Botnet, and U2R are replaced with 0, 1, 2, 3, 4, 5, 6, and 7, respectively. The InSDN multiclassification results are shown in Table 6. Since there are fewer data labels of BFA, web-attack, Botnet, U2R in the dataset, all are selected for experimental testing. Other labels are randomly selected by percentage, and 10,000 instances are selected for the experiment. Table 6 shows the results of the InSDN multiclassification experiments. Among the four categories (Normal, DoS, DDoS, and BFA), ESWO-IDM shows optimal results in recall. Compared with the other models, the results are optimized by 0.2–0.5%. ESWO-IDM ranks first in performance in recall metrics and has the best results in DoS, DDoS, and BFA categories in precision. The results are optimized by 0.3–0.8%. SWO-IDS is superior in other categories; in precision metrics, ESWO-IDM as a whole is close to SWO-IDM, both outperforming the other models. ESWO-IDM has the best effect in the categories normal, DoS, DDoS, and BFA in F1-score; the effect is optimized by 0.3–0.4%. In the F1-score metric, the overall results are also optimal for the four models. For the accuracy metric, the ESWO-IDM result is 96.759%, which is 0.15% higher than the second-ranked SWO-IDM. Table 7 shows the CT of the four IDMs.

SWO-IDM has the shortest CT, followed by ESWO-IDM, with a close running time. The CT is much higher for PSO-IDM and WOA-IDM. The ESWO-IDM improves overall performance at the cost of a slight increase in CT.

In summary, the IDM constructed by the ESWO algorithm is optimized in all four metrics, with significant optimization in accuracy, recall, and F1-score metrics, which shows that ESWO has better performance in multiclassification problems, with fewer classification errors and more accurate predictions. For the normal, DoS, and DDoS categories, the original dataset has a large amount of data, while ESWO-IDM shows optimal accuracy, recall, and precision, demonstrating its advantages in detecting abnormal attacks. For BFA, web-attack, and Botnet categories, the original dataset has less data volume, which requires a higher F1-score and recall, and ESWO-IDM presents higher values than the other algorithmic models in these two metrics. ESWO-IDM improves overall accuracy. For DoS, DDoS, and probe attacks with more instances, ESWO-IDM further improves detection accuracy. ESWO-IDM can also accurately detect BFA and Botnet attacks with fewer instances and more complex features. Compared with other models, the optimization effect is obvious, which effectively improves the efficiency of intrusion detection in SDN.

Table 6. Multiclassification results of the InSDN dataset.

		Normal	DoS	DDoS	Probe	BFA	Web-	BOTNET	U2R
Recall (%)	PSO	98.241	96.013	99.344	93.352	91.395	76.744	96.0	0
	WOA	97.918	96.013	99.558	93.37	91.608	76.744	96.0	0
	SWO	97.99	97.651	99.559	**93.407**	92.217	**78.571**	96.0	0
	ESWO	**98.308**	**97.987**	**99.78**	93.132	**92.272**	76.744	96.0	0
Precision	PSO	98.562	95.695	99.342	91.576	95.157	60.0	100	NaN
	WOA	98.366	95.695	98.684	91.848	95.157	60.0	100	NaN
	SWO	98.758	96.358	99.123	**92.391**	94.673	60.0	100	NaN
	ESWO	98.758	**96.689**	**99.561**	92.12	**95.4**	60.0	100	NaN
F1-score (%)	PSO	98.401	95.854	99.670	92.455	93.238	67.347	97.959	NaN
	WOA	98.142	95.854	99.119	92.603	93.349	67.347	97.959	NaN
	SWO	98.372	97.0	99.341	**92.896**	93.429	**68.041**	97.959	NaN
	ESWO	**98.533**	**97.333**	**99.671**	92.623	**93.810**	67.347	97.959	NaN
Accuracy	PSO				96.476				
	WOA				96.287				
	SWO				96.602				
	ESWO				**96.759**				

Table 7. CT outcome of multiclassification.

Methods	PSO	WOA	SWO	ESWO
Computational time (s)	87.980	46.033	**26.758**	30.253

6. Conclusions

In order to improve the detection efficiency of intrusion detection in SDN networks, this paper proposes an ESWO algorithm that incorporates multiple strategies for feature selection to pick the optimal subset of features and constructs the ESWO-IDM for SDN intrusion detection. First, the global search ability of ESWO is verified using the benchmark test functions. Subsequently, the superior

classification ability of the ESWO algorithm in feature selection is verified using the UCI datasets. Finally, the constructed ESWO-IDM is subjected to binary and multiclassification experiments using the InSDN dataset, in which it is compared with other intrusion detection models based on SWO, PSO, and WOA. In the binary classification experiments, the proposed ESWO-IDM has 98.833, 98.976, and 98.946% accuracy, precision, and F1-score metrics, respectively, which are higher than those of the other compared IDMs. In the multiclassification experiments, the results of the ESWO-IDM are higher than those of other IDMs in normal, DoS, DDoS, and BFA categories; the comprehensive detection ability of the ESWO-IDM is optimal. The experimental results indicate that the proposed ESWO-IDM optimizes SDN intrusion detection. As the InSDN dataset is relatively small in quantity, further research and improvement are needed in subsequent work. Future work will conduct experimental simulations using the SDN environment to further test and improve the detection capability of the proposed ESWO-IDM for SDN.

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 the Hubei Provincial Science and Technology Plan Project under Grant 2023BCB041.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. B. Alhijawi, S. Almajali, H. Elgala, H. B. Salameh, M. Ayyash, A survey on DoS/DDoS mitigation techniques in SDNs: Classification, comparison, solutions, testing tools and datasets, *Comput. Electr. Eng.*, **99** (2022), 107706. https://doi.org/10.1016/j.compeleceng.2022.107706
2. R. U. Rasool, H. Wang, U. Ashraf, K. Ahmed, Z. Anwar, W. Rafique, A survey of link flooding attacks in software defined network ecosystems, *J. Network Comput. Appl.*, **172** (2020), 102803. https://doi.org/10.1016/j.jnca.2020.102803
3. J. Ali, G. Shan, N. Gul, B. Roh, An intelligent blockchain-based secure link failure recovery framework for software-defined internet-of-things, *J. Grid Comput.*, **21** (2023), 57. https://doi.org/10.1007/s10723-023-09693-8
4. M. Madathi, R. Harini, R. Monikaa, N. Gowthami, Detection of DDoS attack in SDN environment using KNN algorithm, *Int. J. Res. Anal. Rev.*, **9** (2022), 252–257.
5. A. Maheshwari, B. Mehraj, M. S. Khan, M. S. Idrisi, An optimized weighted voting based ensemble model for DDoS attack detection and mitigation in SDN environment , *Microprocessors Microsyst.*, **89** (2022), 104412. https://doi.org/10.1016/j.micpro.2021.104412

6. 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
7. M. Al-Zewairi, S. Almajali, M. Ayyash, Unknown security attack detection using shallow and deep ANN classifiers, *Electronics*, **9** (2020), 2006. https://doi.org/10.3390/electronics9122006
8. H. Wang, H. Chen, S. Liu, Intrusion detection system based on improved Naive Bayes algorithm, *Comput. Sci.*, **41** (2014), 111–115.
9. M. Cui, J. Chen, X. Qiu, W. Lv, H. Qin, X. Zhang, Multi-class intrusion detection system in SDN based on hybrid BiLSTM model, *Cluster Comput.*, **27** (2024), 9937–9956. https://doi.org/10.1007/s10586-024-04477-5
10. P. Wang, Z. Wang, F. Ye, X. Chen, ByteSGAN: a semi-supervised generative adversarial network for encrypted traffic classification in SDN Edge Gateway, *Comput. Networks*, **200** (2021), 108535. https://doi.org/10.1016/j.comnet.2021.108535
11. A. K. Sarica, P. Angin, Explainable security in SDN-Based IoT Networks, *Sensors*, **20** (2020), 7326. https://doi.org/10.3390/s20247326
12. L. Zhang, J. Wang, A hybrid method of entropy and SSAE-SVM based DDoS detection and mitigation mechanism in SDN, *Comput. Secur.*, **115** (2022), 102604. https://doi.org/10.1016/j.cose.2022.102604
13. 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
14. H. Xu, Y. Hu, W. Cao, L. Han, An improved jump spider optimization for network traffic identification feature selection, *Comput. Mater. Continua*, **76** (2023), 3239–3255. https://doi.org/10.32604/cmc.2023.039227
15. F. Li, H. Xu, F. Qiu, Modified artificial rabbits optimization combined with bottlenose dolphin optimizer in feature selection of network intrusion detection, *Electron. Res. Ach.*, **32** (2024), 1770–1800. https://doi.org/10.3934/era.2024081
16. F. Qiu, H. Xu, F. Li, Applying modified golden jackal optimization to intrusion detection for software-defined networking, *Electron. Res. Ach.*, **32** (2024), 418–444. https://doi.org/10.3934/era.2024021
17. M. Abdel-Basset, R. Mohamed, M. Jameel, M. Abouhawwash, Spider wasp optimizer: A novel meta-heuristic optimization algorithm, *Artif. Intell. Rev.*, **56** (2023), 11675–11738. https://doi.org/10.1007/s10462-023-10446-y
18. M. Shtayat, M. K. Hasan, A. K. Budhati, R. Solaiman, S. Islam, B. Pandey, et al., An improved binary spider wasp optimization algorithm for intrusion detection for industrial internet of things, *IEEE Open J. Commun. Soc.*, (2024), 1. https://doi.org/10.1109/OJCOMS.2024.3421647
19. E. A. Mohamed, M. S. Braik, M. A. Al-Betar, M. A. Awadallah, Boosted spider wasp optimizer for high-dimensional feature selection, *J. Bionic Eng.*, **21** (2024), 2424–2459. https://doi.org/10.1007/s42235-024-00558-8
20. E. Ott, K Wiesenfeld, Chaos in dynamical systems, *Phys. Today*, **47** (1994), 45. https://doi.org/10.1063/1.2808369
21. 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
22. H. Haklı, H. Uğuz, A novel particle swarm optimization algorithm with Levy flight, *Appl. Soft Comput.*, **23** (2014), 333–345. https://doi.org/10.1016/j.asoc.2014.06.034

23. Q. Liu, M. Li, N. Cao, Z. Zhang, G. Yang, Improved harris combined with clustering algorithm for data traffic classification, *IEEE Access*, **10** (2022), 72815–72824. https://doi.org/10.1109/ACCESS.2022.3188866
24. M. Injadat, A. Moubayed, A. B. Nassif, A. Shami, Multi-stage optimized machine learning framework for network intrusion detection, *IEEE Trans. Network Serv. Manage.*, **18** (2021), 1803–1816. https://doi.org/10.1109/TNSM.2020.3014929
25. R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, *Inf. Sci.*, **291** (2015), 43–60. https://doi.org/10.1016/j.ins.2014.08.039
26. I. H. Hassan, M. Abdullahi, M. M. Aliyu, S. A. Yusuf, A. Abdulrahim, An improved binary manta ray foraging optimization algorithm based feature selection and random forest classifier for network intrusion detection, *Intell. Syst. Appl.*, **16** (2022), 200114. https://doi.org/10.1016/j.iswa.2022.200114
27. 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
28. M. Abdallah, N. A. L. Khac, H. Jahromi, A. D. Jurcut, A hybrid CNN-LSTM based approach for anomaly detection systems in SDNs, in *ARES'21: Proceedings of the 16th International Conference on Availability, Reliability and Security*, (2021), 1–7. https://doi.org/10.1145/3465481.3469190

AIMS Press