Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.058081

ARTICLE

DDoS Attack Autonomous Detection Model Based on Multi-Strategy Integrate Zebra Optimization Algorithm

Chunhui Li[1,2], Xiaoying Wang[1,2,*], Qingjie Zhang[1,2], Jiaye Liang[1] and Aijing Zhang[1]

[1]School of Information Engineering, Institute of Disaster Prevention, Langfang, 065201, China

[2]Langfang Key Laboratory of Network Emergency Protection and Network Security, Langfang, 065201, China

*Corresponding Author: Xiaoying Wang. Email: wangxiaoying@cidp.edu.cn

Received: 04 September 2024 Accepted: 18 October 2024 Published: 03 January 2025

ABSTRACT

Previous studies have shown that deep learning is very effective in detecting known attacks. However, when facing unknown attacks, models such as Deep Neural Networks (DNN) combined with Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN) combined with LSTM, and so on are built by simple stacking, which has the problems of feature loss, low efficiency, and low accuracy. Therefore, this paper proposes an autonomous detection model for Distributed Denial of Service attacks, Multi-Scale Convolutional Neural Network-Bidirectional Gated Recurrent Units-Single Headed Attention (MSCNN-BiGRU-SHA), which is based on a Multi-strategy Integrated Zebra Optimization Algorithm (MI-ZOA). The model undergoes training and testing with the CICDDoS2019 dataset, and its performance is evaluated on a new GINKS2023 dataset. The hyperparameters for Conv_filter and GRU_unit are optimized using the Multi-strategy Integrated Zebra Optimization Algorithm (MI-ZOA). The experimental results show that the test accuracy of the MSCNN-BiGRU-SHA model based on the MI-ZOA proposed in this paper is as high as 0.9971 in the CICDDoS2019 dataset. The evaluation accuracy of the new dataset GINKS2023 created in this paper is 0.9386. Compared to the MSCNN-BiGRU-SHA model based on the Zebra Optimization Algorithm (ZOA), the detection accuracy on the GINKS2023 dataset has improved by 5.81%, precision has increased by 1.35%, the recall has improved by 9%, and the F1 score has increased by 5.55%. Compared to the MSCNN-BiGRU-SHA models developed using Grid Search, Random Search, and Bayesian Optimization, the MSCNN-BiGRU-SHA model optimized with the MI-ZOA exhibits better performance in terms of accuracy, precision, recall, and F1 score.

KEYWORDS

Distributed denial of service attack; intrusion detection; deep learning; zebra optimization algorithm; multi-strategy integrated zebra optimization algorithm

1 Introduction

Distributed Denial of Service (DDoS) attack [1] is one of the severe security threats. DDoS attack is a Denial of Service (DoS) attack caused by attacks from multiple independent nodes [2]. DoS attacks are a one-to-one approach. Whereas DDoS attacks are in the form of many-to-one and attack the attacked on a larger scale [3]. DDoS attacks are mainly divided into Application-Layer

Attacks, Volumetric Attacks, and Low and Slow Attacks [4]. Nowadays, many tools are available on the Internet to perform DDoS attacks on the target server. The attacker hides his identity by using legitimate third-party components such as LOIC, HOIC, and Hping3 [4]. Low attack costs and high traceability difficulty characterize DDoS attacks. Moreover, the frequency, scale, and complexity of DDoS attacks have been showing an increasing trend. According to the "2023 DDoS Threat Report" [5] written and published by NSFOCUS Information Technology, DDoS attacks are no longer limited to traditional network-layer attacks but tend to complex application-layer attacks and new reflection attacks. At the same time, with the gradual commercialization and service of attacks. Attack tools are easier to acquire and don't even require advanced technical skills. For example, in the well-known DDoS attack platform stress.ru, any user can launch custom attacks through paid services [5]. In recent years, DDoS attacks have not only been an individual action. In December 2023 [5], Brazil was hit by a massive blanket DDoS attack. Approximately 8.8 million IP addresses were targeted, representing 12 percent of the overall IP address count in Brazil. The attack involved a wide range of industries, covering government websites, communication operators, education sectors, financial institutions, and other infrastructure. In October, the Internet Archive, along and its founder, Brewster Kahle, issued a statement saying that the organization suffered a data breach and DDoS attack on Wednesday, leading to the website running slowly and going offline intermittently [6]. DDoS attacks are now gradually evolving as a precursor to Advanced Persistent Threats (APT) and ransomware attacks. Attackers are increasingly using DDoS attacks to distract response teams from more serious security incidents. Therefore, the purpose of DDoS attacks is no longer limited to network interference. DDoS attacks can be used to confuse the public and mislead the focus of the defenders. Under the cover of DDoS attacks, more covert APT attacks, data theft, malware injection, and other malicious behaviors are implemented.

Currently, DDoS detection methods are mainly divided into four categories: the detection method based on mathematical statistics, the detection method based on machine learning, detection methods that simple sequential stacking deep learning models, and detection methods based on Metaheuristic Algorithms. The detection method based on mathematical statistics [7] needs to formulate rules in advance and has poor adaptability in the rapidly changing network environment. The detection method based on machine learning mainly comprises feature selection and classifiers, such as References [8–10]. The prediction accuracy of the detection method is increased by improving the feature selection. However, DDoS attack detection methods based on machine learning rely too much on feature selection. As a new field of machine learning, deep learning has powerful feature extraction and learning capabilities. For example, References [11–13] avoid excessive reliance on feature extraction, while experiments show that it is also superior to traditional machine learning models in accuracy. However, there is still room for improvement. In terms of datasets, acquiring DDoS attack data is very difficult, as the publicly available datasets are obsolescent and imbalanced. In terms of models, existing models only perform simple sequential stacking, leading to the loss of fine-grained features. Moreover, selecting appropriate features and optimal hyperparameters for the model is challenging. In response to these challenges, researchers have explored various algorithms for feature selection and hyperparameter tuning to enhance the detection of DDoS attacks [14]. Therefore, in recent years, researchers have focused on metaheuristics, introducing metaheuristics to select appropriate features or hyperparameters. Metaheuristics are increasingly favored due to their proven ability to yield superior solutions compared to other approaches, such as iterative methods, optimization algorithms, or straightforward heuristics, all while requiring less computational effort [14].

In summary, this paper proposes a novel model for autonomous detection of DDoS attacks, termed MSCNN-BiGRU-SHA. It utilizes the ZOA to optimize the parameters of the MSCNN-BiGRU-SHA model. The model is trained and tested based on the CICDDoS2019 dataset, and the detection ability of the model for the untrained data set is evaluated on the newly generated DDoS attack dataset GINKS2023 with the traditional DNN [15], CNN [16], Gated Recurrent Unit (GRU) [17], and LSTM [18]. The MSCNN-BiGRU-SHA model uses an improved ZOA called MI-ZOA to optimize the parameters further. Compared with the current most advanced methods, the experimental results show that the detection ability of the MSCNN-BiGRU-SHA model based on the multi-strategy integration zebra optimization algorithm is better than that of other models.

The main contributions are as follows:

1. A new deep learning model, named MSCNN-BiGRU-SHA, has been proposed to enhance the detection accuracy of untrained DDoS attack datasets. The ZOA is employed to optimize the parameters of the MSCNN-BiGRU-SHA model, aiming to search for the optimal hyperparameters within continuous value ranges.

2. A new dataset, GINKS2023, has been created, and the CICDDoS2019 dataset and GINKS2023 dataset have preprocessing to enhance data quality. CICDDoS2019 is used for model training and testing. The GINKS2023 dataset is used as an evaluation set to evaluate the detection performance of the model on untrained DDoS attack datasets. Furthermore, the model is retrained using various combinations of the CICDDoS2019 and GINKS2023 datasets to evaluate its learning capabilities.

3. An MI-ZOA is proposed to further optimize the parameters of MSCNN-BiGRU-SHA. The models are trained using the CICDDoS2019 and GINKS2023 datasets, respectively, while GINKS2023 and CICDDoS2019 serve as evaluation datasets to assess the detection capability of the optimized MSCNN-BiGRU-SHA on previously unseen data.

This document is structured as follows:

Section 1 provides an introduction. Section 2 is a literature review. Section 3 describes the Zebra Optimization Algorithm and the strategies used in the Multi-strategy Integrated Zebra Optimization Algorithm; Section 4 explains our proposed model; Section 5 describes the dataset construction; Section 6 evaluates the proposed model; and finally, concludes the paper.

2 Literature Review

For example, see Table 1.

Mishra et al. [7] proposed that entropy variation can reduce the computational overhead of DDoS attacks. By comparing the entropy change of DDoS attack traffic and normal traffic, the occurrence of attacks can be detected, and corresponding mitigation measures can be taken. Gu et al. [8] proposed a semi-supervised weighted K-means Method using the SKM-HFS. This method combined the K-means algorithm with the HFS to sort the traffic features to obtain a feature subset, which improved the prediction accuracy. However, the algorithm has high complexity and poor real-time performance. As a mainstream ensemble learning algorithm, Random Forest (RF) has the characteristics of simplicity and strong generalization ability. Pande et al. [9] proposed a DDoS attack detection model based on the RF, which has high accuracy for DDoS attack detection. The feasibility and efficiency of the Random Forest Algorithm are proved. Xu et al. [10] proposed a random forest DDoS attack detection method with feature selection. While improving the accuracy, the F1 score is also better

than the traditional random forest detection scheme. However, machine learning-based methods for identifying DDoS attacks rely too much on feature selection.

Table 1: DDoS attack detection studies, their methods and conclusions or results

Type	Method	Dataset	Conclusions or results	Year
Mathematical statistics	Entropy variation [7]	Mininet simulations generate experimental datasets	This paper proposes a DDoS attack detection and mitigation technique based on entropy variation, which has low computational overhead. Simulations were carried out in the Mininet simulator using POX controllers and open-flow switches.	2021
Machine learning	SKM-HFS [8]	DARPA DDoS CICIDS2017	This method combined the K-means algorithm with the Hybrid Feature Selection algorithm (HFS) to sort the traffic features to obtain a feature subset, which improved the prediction accuracy.	2019
	RF [9]	KDD Cup 99 NSL-KDD	The DDoS attack detection model, which utilizes a Random Forest (RF) algorithm, exhibits high accuracy in identifying DDoS attacks. The feasibility and efficiency of the Random Forest Algorithm are proved.	2021
	GBDT_RF [10]	CICIDS2017	Compared with the traditional RF detection scheme, GBDT_RF is superior to the traditional random forest detection scheme in recall rate and F1 score.	2023

(Continued)

Table 1 (continued)

Type	Method	Dataset	Conclusions or results	Year
	GB [19]	CICDDoS2019	Appropriate hyperparameters can enhance the model's performance. By reducing the feature space and tuning these hyperparameters, this paper has successfully improved the accuracy of the GB model.	2021
Deep learning	DNN [11]	CICIDS2017	A new DNN model detection mechanism that uses feedforward backpropagation to detect DDoS attacks achieves better accuracy on the CICIDS2017 dataset.	2020
	DNN [12]	CICDDoS2019	DNN model is used to detect data flooding or UDP flooding attacks in MANET. This model has a good detection effect on data flooding or UDP flooding attacks in CICDDoS2019 data.	2020
	DNN [20]	CICDDoS2019	The detection accuracy of DDoS attacks based on feedforward DNN is improved on the CICDDoS2019 dataset However, the accuracy of attack type is low and there is no comparison experiment with other models.	2021

(Continued)

Table 1 (continued)

Type	Method	Dataset	Conclusions or results	Year
	SDNN [21]	CICIDS2017 CICDDoS2019	This paper proposes an innovative method for detecting DDoS attacks called the encoder-stacked deep neural network model. This method utilizes stacked/packed Multilayer Perceptrons (MLP) to achieve accurate DDoS attack detection.	2024
	CNN-LSTM [22]	CICIDS2017	Compared with SVM, Bayesian, and RF, CNN-LSTM has an accuracy of 97.16% and a recall rate of 99.1%, which is better than other models.	2019
	CNN [13]	CICIDS2017	Compared with RNN, LSTM, and Reinforcement Learning (RL), the CNN framework has excellent detection accuracy and low computational cost.	2020
	LSTM [23]	CICDDoS2019	LSTM detects distributed reflection denial of service with high accuracy.	2020
	LSTM, GRU [24]	CICIDS2017 CICDDoS2019	In this paper, RNN, LSTM, and GRU models are used for training and validation on the latest CICDDoS2019 dataset and comparative analysis with the CICIDS2017 dataset.	2023

(Continued)

Table 1 (continued)

Type	Method	Dataset	Conclusions or results	Year
Swarm intelligence optimization	CSA-RBP [25]	NSL-KDD	This research presents a machine learning-based approach to detect application-layer distributed denial of service (App-DDoS) attacks. This approach is achieved by combining Radial Basis Function (RBF) neural network and Cuckoo Search Algorithm (CSA).	2022
	BIWSO3 [26]	CICIDS2017, Bot-IoT NSL-KDD, CICDDOS2019, CIRA-CICDosHBrw-2020U, UNSW-NB15, CIC-MalMem-2022, CSE-CICIDS2018 Intrusion Detection 2018, Phishing Legimate, LUFlow Network, KDDCup-99	The accuracy of BIWSO3 is higher than that of models KNN, NB, RF, and DT on multiple datasets.	2023
	ISHO-HBA and SE-ResNet152 [27]	UNSW-NB15 CSE-CICIDS2018 CICDDoS2019	This paper uses Improved Spotted Hyena Optimization algorithm (ISHO) and Honey Badger Algorithm (HBA), Squeeze-and-Excitation network (SE) and a SE-ResNet152 to improve the detection performance.	2024

In contrast, simple sequential stacking of deep learning models for detection methods can not only extract complex nonlinear relationships. The model itself can not only perform feature extraction but also classify the data [28]. Asad et al. [11] proposed a novel detection mechanism based on DNN. This mechanism uses feedforward backpropagation to accurately discover a variety of application-layer DDoS attacks. It is accurate accuracy on the CICIDS2017 dataset [29], but this method only targets

application-layer DDoS attacks. Sbai et al. [12] introduced a DNN model to detect data flooding or User Datagram Protocol (UDP) flooding attacks in MANETs, demonstrating effective detection performance on these types of attacks within the CICDDoS2019 dataset [30]. However, the authors have only worked on data flooding or UDP flooding attacks in the CICDDoS2019 dataset. There is a lack of comprehensiveness for other types of attacks. Haider et al. [13] proposed an integrated framework of deep CNN for efficiently detecting of DDoS attacks in SDN. Compared to existing detection methods, the accuracy of the CICIDS2017 dataset was improved. Shurman et al. [23] employed LSTM to detect Distributed Reflection Denial of Service attacks, achieving high accuracy. However, the evaluated performance of the model only uses the reflection-based CICDDoS2019 dataset, which is unclear for other types of DDoS attack detection. The types of DDoS attacks are variable, and it is not enough to detect only one type of DDoS attack. Roopak et al. [22] proposed a hybrid CNN-LSTM model, which achieved 97.16% accuracy and 99.1% recall compared to Supported Vector Machine (SVM), Bayesian, and RF. However, the model was trained on the CICIDS2017 dataset, containing incomplete attacks. Cil et al. [20] proposed a feedforward-based deep neural network to detect DDoS attacks, and the accuracy is improved on the CICDDoS2019 dataset. The classification of attack types is not very precise, and there is no comparison experiment with other models. Ramzan et al. [24] proposed a new LSTM and GRU model architecture with better accuracy and efficiency on the CICDDoS2019 dataset. Benmohamed et al. [21] proposed stacked/packed Multilayer Perceptron (MLP) to achieve accurate DDoS attack detection, but the algorithm is too complex, and the model training is time-consuming. Batchu et al. [19] found that suitable hyperparameters can improve the performance of models. Batchu et al. [19] improved the model performance by reducing the feature space and hyper-parameter tuning, resulting in a Gradient Boosting (GB) model accuracy of 99.97%. However, the effectiveness of this method has not been validated on deep learning models, as experiments were limited to machine learning models.

Metaheuristic algorithms are a method to realize optimization by simulating the behavior of social animals [31]. It usually possesses characteristics such as strong global optimization ability, strong adaptability, simple structure, strong parallelism, and strong robustness. In recent years, metaheuristic algorithms have become a reliable method to solve a variety of complex optimization problems [31]. Its applications involve almost all fields of science, engineering, and industry [31]. Beitollahi et al. [25] suggested a method for detecting application layer DDoS attacks utilizing a Cuckoo Search Algorithm-trained Radial Basis Function. Its accuracy is better than common machine learning models. However, it only aims at application-layer DDoS attacks. Alawad et al. [26] proposed a binary improved whale algorithm for an intrusion detection system. The improved whale algorithm was used to solve the feature selection problem. Saikam et al. [27] proposed an ISHO and a HBA to address the issues of data imbalance and overfitting. By utilizing SE-ResNet52 to eliminate less significant features and employing a list of decision trees at each iterative stage to monitor the classifier's performance, overfitting is prevented. This method demonstrates favorable performance on the UNSW-NB15, CSE-CICIDS2018, and CICIDS2019 datasets compared to traditional algorithms such as Recurrent Neural Network (KNN), RF, and SVM. Although the DDoS attack detection technology has innovated, the dataset quality and model performance still to be improved.

3 Zebra Optimization Algorithm and Multi-Strategy Integrated Zebra Optimization Algorithm

3.1 Zebra Optimization Algorithm

The ZOA [32] is a novel approach within the field of Meta-Heuristic Algorithms. Its implementation is relatively straightforward, requiring minimal complex parameter tuning [32]. ZOA is suitable for a wide range of optimization problems [32]. Each zebra symbolizes a possible solution to a problem,

and the domain in which the zebra is located outlines the search space for that problem. The Zebra Optimization Algorithm comprises two processes: exploration and exploitation.

3.1.1 Initialization

During the initialization phase, the specific numerical value of the decision variables is determined by the position of each zebra in the search space. Therefore, each zebra belonging to the ZOA member can be modeled as a vector, with each element of the vector representing the numerical value of the problem variable. A matrix can mathematically represent the total number of zebras. In the search space, the initial position of each zebra is randomly assigned. Each zebra corresponds to a potential solution to the optimization problem. Thus, the objective function is evaluated using the values proposed by each zebra for the problem's variables. The representation of the ZOA population matrix is shown in Eq. (1), and the outcomes from the objective function are depicted as an array in Eq. (2).

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,m} \end{bmatrix}_{N\times m} \tag{1}$$

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \tag{2}$$

In this context, X denotes the population of zebras, where X_i denotes the ith individual zebra, $x_{i,j}$ corresponds to the value of the jth problem variable associated with the ith zebra. The total count of members within the population (zebras) is N, while m signifies the number of decision variables involved. F is the objective function value vector, with F_i representing the objective function value of the ith zebra.

3.1.2 Phase 1: Foraging Behavior (Exploration)

The exploration phase of ZOA is inspired by zebras' behavior when searching for food. Zebras with better objective function values are considered as leaders and guiding the remaining members to their positions in the population. The positions of zebras during the foraging phase are represented mathematically through Eqs. (3) and (4).

$$x_{i,j}^{new,P1} = x_{i,j} + r \cdot \left(PZ_j - I \cdot x_{i,j}\right) \tag{3}$$

$$X_i = \begin{cases} X_i^{new,P1}, F_i^{new,P1} < F_i; \\ X_i, else, \end{cases} \tag{4}$$

where $X_i^{new,P1}$ denotes the updated state of the ith zebra in first phase, $x_{i,j}^{new,P1}$ represents its value in the jth dimension, $F_i^{new,P1}$ corresponds to its objective function value. PZ refers to the pioneer zebra, the best-performing member, with PZ_j indicating the jth dimension. The variable r is a random number

in interval [0, 1], $I = round\,(1 + rand)$, where *rand* is also a random value from [0, 1]. Consequently, I can take values in the set {1, 2}. If the parameter $I = 2$, it implies that there will be a greater degree of variation in the population's movement.

3.1.3 Phase 2: Predator-Targeted Defense (Exploitation)

In the second phase, the ZOA imitates the defense strategies of zebras against predators to update the distribution positions of ZOA population members within the search space. In the concept of the ZOA, zebras have two equal-probability reactions when facing an attack: one is to take evasive action when a lion launches an attack, and the other is to choose to counterattack when facing other predators.

In the first strategy, when zebras are attacked by lions, they will choose to dodge nearby and avoid the lion's attack. Thus, this strategy can be mathematically represented using the model S_1 in Eq. (5). In the second strategy, when a zebra encounters an attack from other predators, other members of the zebra group will move closer to the attacked companion and attempt to form a defensive line to deter or confuse the opponent. This zebra strategy is mathematically represented by the model S_2 in Eq. (5). When updating the zebras' positions, a new position is accepted if the objective function produces a better value at that point. This criterion for updating positions is formulated by Eq. (6).

$$x_{i,j}^{new,P2} = \begin{cases} S_1 \colon x_{i,j} + R \cdot (2r - 1) \cdot \left(1 - \dfrac{t}{T}\right) \cdot x_{i,j}, P_s \leq 0.5; \\ \qquad S_2 \colon x_{i,j} + r \cdot \left(AZ_j - I - x_{i,j}\right), else, \end{cases} \tag{5}$$

$$X_i = \begin{cases} X_i^{new,P2}, F_i^{new,P2} < F_i; \\ \qquad X_i, else, \end{cases} \tag{6}$$

where $X_i^{new,P2}$ represents the new state of the ith zebra in the second phase, $x_{i,j}^{new,P2}$ denotes its value in the jth dimension, $F_i^{new,P2}$ corresponds to its objective function value, t is the current iteration, T is the maximum number of iterations, R is a constant set to 0.01. P_s denotes the probability, within the range of [0, 1], of opting for one of the two strategies that are randomly selected. AZ refers to the status of the zebra that has been targeted for an attack, and AZ_j signifies the value in the jth dimension of this attacked zebra.

3.2 Multi-Strategy Integrated Zebra Optimization Algorithm

The ZOA algorithm has the characteristics of simple structure, easy implementation, and strong optimization ability. It has achieved good test results in engineering design problems such as tension/-compression spring design optimization, welding design optimization, reducer design optimization, and pressure vessel design optimization [32]. However, the ZOA also has problems, such as poor convergence accuracy and being prone to falling into local optimums.

This paper proposes the Multi-strategy Integrated Zebra Optimization Algorithm (MI-ZOA) to solve the problem of the ZOA algorithm being prone to falling into local optimums, improve its global exploration ability, and enhance its performance in solving complex problems.

3.2.1 Tent Chaotic Map

The ZOA uses random numbers as the initial population information, which is large and random and makes it difficult to retain the diversity of the population. As a result, the population gathers in some areas of the search space and cannot fully explore the whole search space. The Tent Chaotic

Map is a simple one-dimensional nonlinear dynamic system that exhibits chaotic behavior. The x-axis represents the control parameter u of the Tent map, which determines the shape of the mapping function, and the y-axis represents the values of the state variable x produced by the iteration of the Tent map at each fixed u value. In the long term, the points on the y-axis show the steady state or periodic behavior that x may be achieved. When the control parameter u is large enough to exhibit chaotic behavior, it is called a Tent chaotic map. In this case, the points on the y-axis densely fill certain areas, demonstrating irregular behavior. As shown in Fig. 1, Sanliang et al. have demonstrated that the Tent Chaotic Map can generate relatively uniform initial values within the interval [0, 1] while also improving the optimization speed of algorithms. Therefore, this paper adds the Tent Chaotic Map to the ZOA population initialization stage to increase the population's diversity and improve the algorithm's exploration ability in the global search stage. The specific equations are (7) and (8).

$$TentMap\left(x_{n+1}\right) = \begin{cases} \dfrac{x_n}{a}, & 0 < x_n < a \\ (1 - x_n)/(1 - a), & a < x_n \leq 1 \end{cases} \tag{7}$$

$$X_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{in} \end{bmatrix}, x_{ij} = TentMap\left(x_{ij-1}\right), j = 1, 2, \ldots, n \tag{8}$$

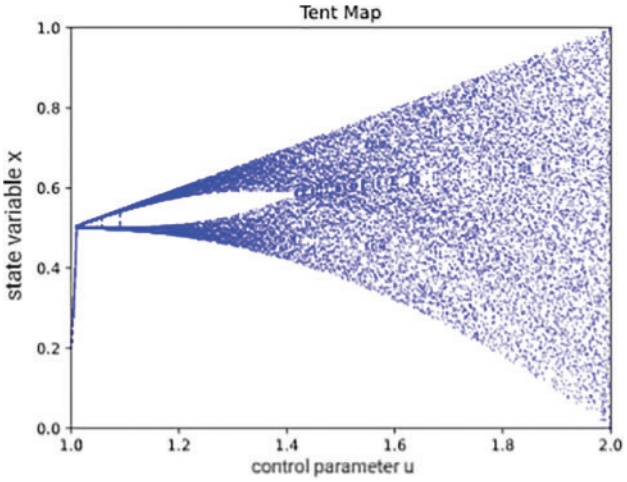

Figure 1: Tent map

X_i represents the position of the ith individual in the population, and x_{ij} represents the position of the jth dimension of the ith individual. $TentMap\left(x_{n+1}\right)$ is a mapping function. For each individual X_i in the population, by independently applying the Tent map to each dimension, an initial position for each individual in the population can be generated in each dimension. This allows for the creation of a diverse initial population with a good foundation for exploration.

3.2.2 Lévy Flight Strategy

The ZOA algorithm focuses on the defense phase against predators. As a result, the new variables identified by ZOA often converge to local values, making the algorithm more suitable for addressing smaller and less complex problems. Therefore, the Lévy Flight Strategy is added to the ZOA in

the foraging behavior (exploration) and predator defense (exploitation) phases. This can enhance the ability of the algorithm to optimize nonlinear functions and jump out of local optimal. Lévy flight, based on the Ryan distribution, is commonly employed to improve optimization algorithms. Its formula is shown in Eq. (9).

$$\sigma = \frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\left[\Gamma\left(\frac{1+\beta}{2}\right)\beta 2^{\frac{\beta-1}{2}}\right]^{\frac{1}{\beta}}} \tag{9}$$

Among them, σ is a proportional factor used to control the step size in Lévy flight. This value is determined based on the characteristics of the distribution, ensuring that the step size of the Lévy flight has an appropriate proportion. Γ is the Gamma function, a function that extends the concept of factorial to all complex and real numbers.

The position formula of the ZOA with the Lévy Flight Strategy introduced are as follows (Eqs. (10) and (11)):

$$x_{i,j}^{new,P1} = x_{i,j}^{new,P1} + Lévy\,(dim) \tag{10}$$

$$x_{i,j}^{new,P2} = x_{i,j}^{new,P2} + \text{Lévy}\,(\text{dim}) \tag{11}$$

where $x_{i,j}^{new,P2}$ represents its jth dimension value, and dim is the initialized dimension.

3.2.3 Random Walk Strategy

The Random Walk Strategy is a mathematical model. It characterizes the random trajectory of an object moving continuously over time without a specific direction or endpoint. For multidimensional spaces, each random walk step can be carried out independently in multiple directions. In order to enhance the ZOA's ability to escape local optima more efficiently and avoid premature convergence. The ZOA adds a Random Walk Strategy after executing the foraging behavior (exploration) and predator defense (exploitation) phases at each iteration. That is, the optimal position of each iteration is perturbed, which is convenient for the algorithm to jump out of the local optimal solution to explore new areas and enhance the global search ability. The pseudocode for the Random Walk Strategy is shown in Algorithm 1.

Algorithm 1: Pseudo-code for random walk initialization

Start Random Walk Initialization.

1.Input: Dimension (dim), Maximum Iterations ($iter_{max}$), Lower Bound (lb), Upper Bound (ub), Initial Position ($position$), Current Iteration ($iter_{current}$).

2. Set Coefficient I *to* 1.

3. Adjust I based on the progress of iterations:

a. If $iter_{current}$, set $I = 1 + 100 * \left(\frac{iter_{current}}{iter_{max}}\right)$.

b. If $iter_{current} > \frac{iter_{max}}{2}$, set $I = 1 + 1000 * \left(\frac{iter_{current}}{iter_{max}}\right)$.

(Continued)

Algorithm 1 (continued)

c. If $iter_{current} > 3 * \frac{iter_{max}}{4}$, set $I = 1 + 10000 * \left(\frac{iter_{current}}{iter_{max}}\right)$.

d. If $iter_{current} > 0.9 * iter_{max}$, set $I = 1 + 100000 * \left(\frac{iter_{current}}{iter_{max}}\right)$.

e. If $iter_{current} > 0.95 * iter_{max}$, set $I = 1 + 1000000 * \left(\frac{iter_{current}}{iter_{max}}\right)$.

4. Update search range based on I:
   a. Divide lb and ub by I.
   b. Transpose lb and ub.
5. Shift range towards the target:
   a. If random number < 0.5, add *position* to lb, otherwise, subtract lb from *position*.
   b. If random number $>= 0.5$, add *position* to ub, otherwise, subtract ub from *position*.
6. Initialize output array out to a zero matrix of size $[1, dim]$.
7. For each dimension i in dim:
   a. Create an array A of size $[iter_{max}, 1]$ with random binary values mapped to $[-1, 1]$.
   b. Calculate the cumulative sum of A transposed (X).
   c. Normalize X based on the bounds [a, b, c, d] which correspond to the min and max of A and the updated bounds lb and ub.
   d. Assign the value of X at $iter_{current}$ to the ith dimension of out.
8. Return the modified position(out).

End Random Walk Initialization.

3.2.4 The Detail of Our Proposed MI-ZOA

This paper proposes an improved MI-ZOA by combining the three strategies introduced in Sections 3.2.1 to 3.2.3 with ZOA. In the population initialization phase, the MI-ZOA algorithm adds the Tent Chaotic Map introduced in Section 3.2.1. In the foraging behavior (exploration) and the predator defense (exploitation) phases, the MI-ZOA algorithm adds the Lévy Flight Strategy of Section 3.2.2. During the algorithm iteration, the MI-ZOA adds the Random Walk Strategy of Section 3.2.3. The pseudocode is shown in Algorithm 2. The flowchart is for one example, see Fig. 2.

Algorithm 2: Pseudo-code for MI-ZOA

Start ZOA.
1. Input: Provide the necessary information related to the optimization problem.
2. Set Parameters: Define the number of iterations (T) and the population size (N) of Zebras.
3. Initialization: Use the Tent map to initialize the positions of the zebra population.
4. Loop: For each iteration $t = 1$ to T:
5. Update the pioneer zebra (PZ).
6. For each Zebra $i = 1$ *to* N:
7. Phase 1: Foraging Behavior:
8. Calculate the new state of the ith zebra using Eq. (3).
9. Update the position of the ith zebra using Eq. (4).
10. Phase 2: Defense Strategies Against Predators:

(Continued)

Algorithm 2 (continued)

If $P_s < 0.5$, *assign a random value to* $P_s = rand$.
Strategy 1: Defending Against Lion:
Calculate new status of the ith zebra using mode S_1 in (5).
Apply Lévy flight strategy.
Else:
Strategy 2: Defending Against Other Predators:
Calculate new status of the ith zebra using mode S_2 in (5).
Apply Lévy flight strategy.
End if.
Update the ith zebra using (6).
End for each zebra i.
Apply Random Walk strategy.
Record the best solution found so far.
End for each iteration t.
Output: Provide the optimal solution achieved by the ZOA for the specified optimization problem
End ZOA.

4 Deep Learning Model Framework

4.1 Multi-Scale Convolutional Neural Network

The multi-scale CNN performs feature extraction on the previous input in parallel through convolutional layers with different convolutional scales, extracting visual information in different scales. The multi-scale convolutional neural network has achieved excellent results in fields such as medicine, remote sensing, and chemistry.

However, it has been less applied in the field of network security. Therefore, a Multi-Scale Convolutional Neural Network (MSCNN) is designed in this paper to extract features from input data at multiple scales. Using convolution scales of 7, 5 and 3, MSCNN performs convolution operations on multiple scales and can capture feature information from coarse-grained to fine-grained. This allows the model to detect both macro-level patterns of DDoS attacks and identify subtle abnormal patterns, thereby enhancing the robustness of the model.

4.2 Bidirectional Gated Recurrent Units

GRU is a variant of Recurrent Neural Networks. They aim to solve the vanishing gradient problem of traditional RNNs on long sequences. GRUs introduce two types of gate mechanisms: the Update Gate and the Reset Gate. These two gates control the flow of state information, allowing the model to better capture long-distance dependencies in the sequence. The computation formulas for each part are as follows:

$$z_t = \sigma\left(W_z \times [h_{t-1}, x_t] + b_z\right) \tag{12}$$

$$r_t = \sigma\left(W_r \times [h_{t-1}, x_t] + b_r\right) \tag{13}$$

$$\widehat{h_t} = tanh\left(W \times [r_t \odot h_{t-1}, x_t] + b\right) \tag{14}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h} \tag{15}$$

where x_t represents the input data at time step t; h_{t-1} denotes the hidden state from the previous time step $t-1$; z_t is the update gate's activation value; r_t is the reset gate's activation value; $\widehat{h_t}$ corresponds

to the candidate hidden state; h_t refers to the final hidden state at time step t; W_z, W_r, W are the corresponding weight matrices; b_z, b_r, b are the corresponding bias vectors; $\odot$ represents element-wise multiplication (Hadamard product).

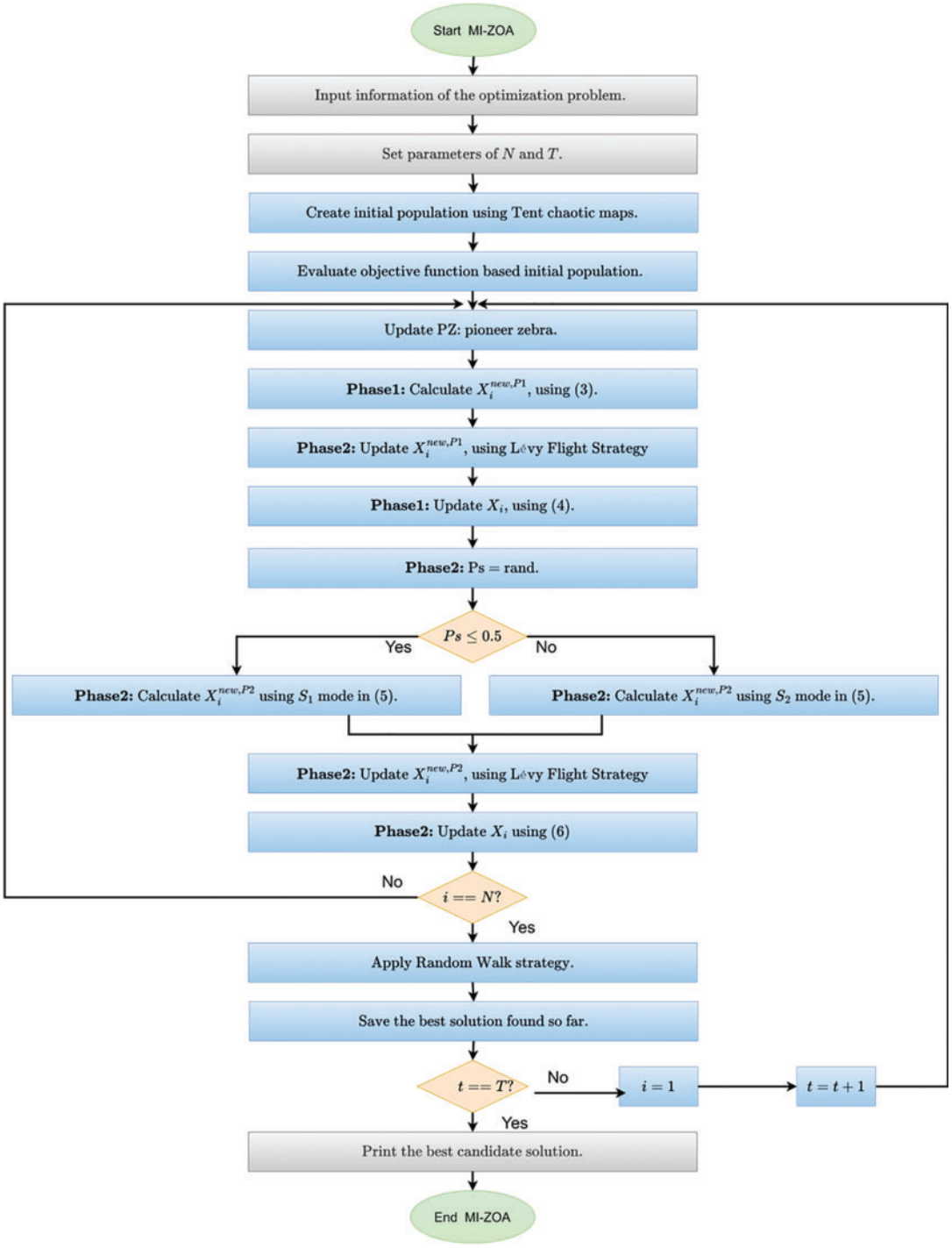

Figure 2: The flowchart of MI-ZOA

Bidirectional Gated Recurrent Units (BiGRU) is a combination of bidirectional RNNs and GRUs. The idea behind using a bidirectional structure is to capture both forward and backward information in the input sequence.

In this paper, a BiGRU is added after the MSCNN. BiGRU uses two GRU layers to extract the features of the time dimension of the input data from the forward and backward respectively. At the same time, BiGRU learns long-distance dependence information through structures such as the update gate and reset gate to avoid the information loss problem of unidirectional GRU. The output bidirectional sequential feature vector is $[H_0, H_1, \ldots, H_n]$.

4.3 Single Headed Attention

A language model architecture called Single Headed Attention-RNN (SHA-RNN) was proposed, which outperforms models like Transformers in byte-level compression tasks. It uses a Single Attention Head, reducing memory usage, avoiding large matrix multiplications within each time step, and speeding up training and inference time.

Therefore, this paper adds a Single Headed Attention after the BiGRU to enhance its ability to focus on key features in long input sequences. It also reduces memory pressure. For one example, see Fig. 3 below, the Single Headed Attention receives the bidirectional temporal features $[H_0, H_1, \ldots, H_n]$ outputted by the BiGRU. These bidirectional temporal features are assigned to Q (query), K (key), and V (value). Then, the attention scores QK^T are computed. These scores are converted to weights multiplied by V by the softmax function to input the Boom layer. In the Boom layer, the input tensor's dimensions are first expanded, then split and summed. The calculation formula for the attention mechanism is as follows:

$$Attention\ Score_{ij} = Q_i K_j^T \tag{16}$$

$$W_{A_{ij}} = \frac{exp\left(Attention\ Score_{ij}\right)}{\sum_k exp\left(Attention\ Score_{ik}\right)} \tag{17}$$

$$Output_i = \sum_j W_{A_{ij}} \times V_j \tag{18}$$

$$Output = [Output_1, Output_2, \cdots, Output_n] \tag{19}$$

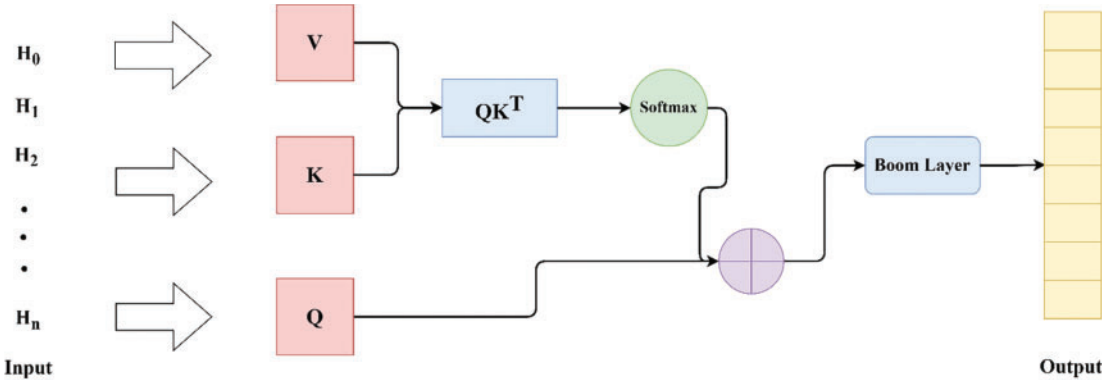

Figure 3: Single headed attention structure diagram

For one example, see Fig. 4, the model consists of four modules, as shown in Fig. 3, the MSCNN module, the BiGRU module, the Single Headed Attention (SHA) module, and the output module. The data is input into the MSCNN module, where parallel multi-scale convolutional layers and pooling layers are utilized to extract spatial features from the input data. Simultaneously, weight sharing is utilized to decrease the total number of parameters and reduce the model's complexity. The BiGRU module utilizes two GRU layers to extract temporal dimension features from the data in forward and backward directions. Additionally, the update and reset gates enable the model to learn long-distance dependencies more effectively. In this way, the unidirectional GRU information loss problem is avoided. The DDoS attack has a long time span, and the recurrent neural network will lose information and have a low ability to distinguish important features when dealing with information that has a long time to transfer. Therefore, adding a Single Headed Attention module not only improves the model's attention to the key features but also reduces the influence of low correlation features on the detection results. For one example, see Table 2, the model framework details are shown in Table 2.

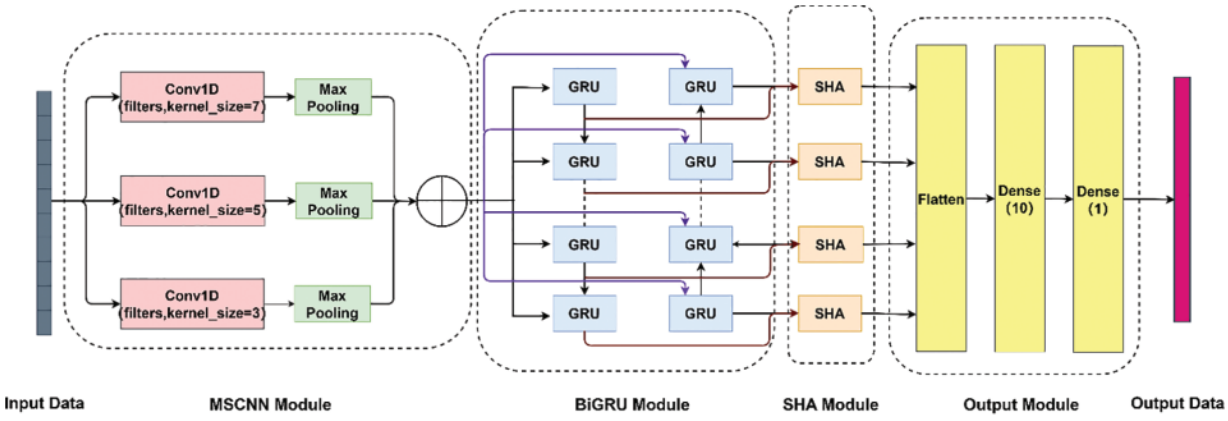

Figure 4: MSCNN-BiGRU-SHA model structure

Table 2: MSCNN-BiGRU-SHA model framework

Layer	Type	Parameter description
Inputs	Input	input_shape
conv1, conv2, conv3	Conv1D	filters = Conv_filter, activation = 'relu', padding = 'same'
pool1, pool2, pool3	MaxPooling1D	pool_size = 2
merged	Add	(pool1, pool2, pool3)
context_vector	SHA_BiGRU	units = GRU_filter
flattened	Flatten	
drop	Dropout	rate = 0.5
dense	Dense	units = 10, activation = 'relu'
outputs	Dense	units = 1, activation = 'sigmoid'

4.4 Parameter Setting

Deep learning models contain hyperparameters that need to be adjusted, and these hyperparameters affect the model's performance. Sometimes, the hyperparameter selection is even more important than the model itself. Common hyperparameters selection methods include random search, grid search, and Bayesian optimization. However, the hyperparameter range searched by these methods is discrete, and the problem is that the optimal parameters are missing. The metaheuristic optimization algorithm can find optimal parameter values on continuous spaces, so it has been widely used in hyperparameter optimization of models in recent years. The ZOA is a new approach within the field of Metaheuristic Algorithms. Its implementation is relatively straightforward, requiring minimal complex parameter tuning. Therefore, this paper utilizes the ZOA to select Conv_filter and GRU_unit. Where Conv_filter refers to the convolutional filters, GRU_unit refers to the number of GRU units in the BiGRU layer.

For one example, see Fig. 5. The flowchart for hyperparameter selection of the MSCNN-BiGRU-SHA model based on the ZOA is shown in Fig. 5. After initializing the ZOA parameters, the Loss value of the MSCNN-BiGRU-SHA model is used as the fitness of the ZOA. It is convenient for ZOA to search the optimal hyper-parameters Conv_filter and GRU_unit of the MSCNN-BiGRU-SHA model. After generating the initial optimal fitness, Conv_filter and GRU_unit, each population individual performs foraging behavior and defense behavior against predators to explore the best fitness in the given space. Then, update the fitness, Conv_filter and GRU_unit. The updated individual fitness is reordered to compare the optimal fitness with the initial optimal fitness, and the two are selected as the best. The Conv_filter and GRU_unit corresponding to the optimal fitness are the optimal hyperparameters.

5 Dataset Construction

When applying existing classification models, the dataset is seriously unbalanced. In this paper, the CICDDoSS2019 dataset [21] is oversampled with attack data to obtain the CICDDoS2019 dataset which consists of CICDDoS2019, benign, DoS, and DDoS flows. The GINKS2023 dataset was collected using independent methods and tools. Therefore, to assess the model's detection capabilities on untrained DDoS attack datasets, this paper generates the GINKS2023 dataset for model evaluation.

5.1 GINKS2023 Dataset

For one example, Fig. 6 illustrates a simple attack diagram. While capturing new data, a DDoS attack script is written using the Python Scapy module. Host H8 runs Wireshark, while H3 runs Cobalt Strike to control H4–H7 to execute the Python script. Multiple IP addresses are spoofed to carry out TCP, UDP and HTTP flood attacks against H8. However, the script does not set the range of the number of packets, packet size, window size, and port number. This holds in a really life environment where the attacker can change the attack mode frequently. At the same time, hosts H1 and H2 send legitimate requests to host H8. The generated '.pcap' raw data is processed using the CICFlowMeter [33] traffic analyzer and converter, converting the captured data into '.csv' files.

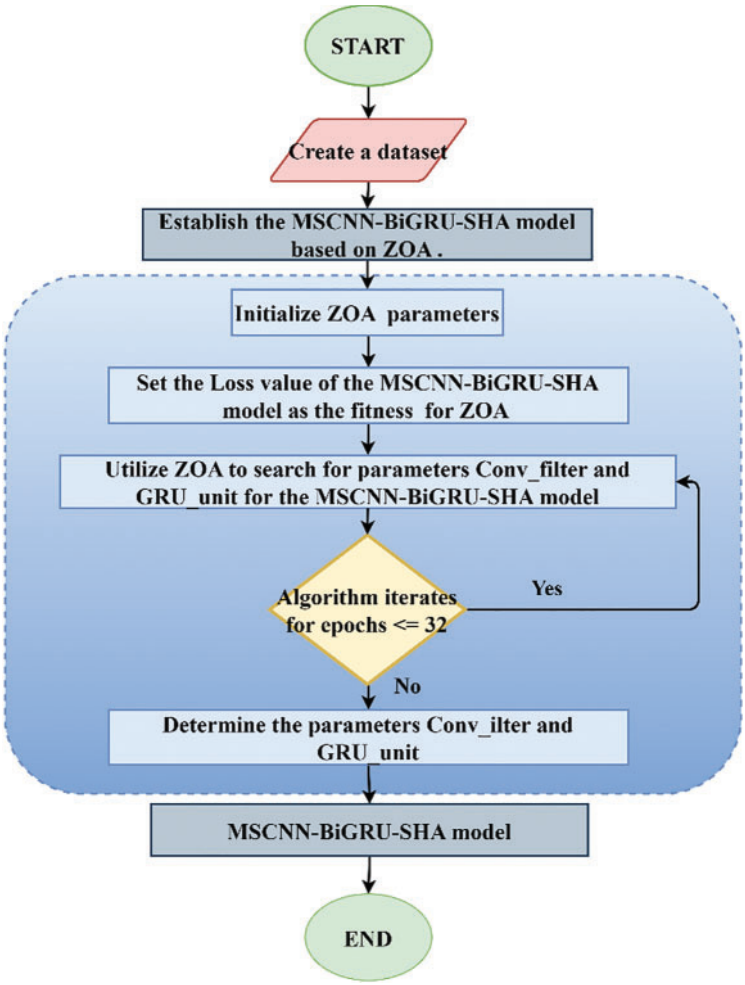

Figure 5: Flowchart of parameter selection for the MSCNN-BiGRU-SHA model based on ZOA

5.2 Dataset Preprocessing

5.2.1 Removing Unnecessary Features

Due to the differences in the performance of these feature values in different networks and the high similarity between attackers' and normal users' IP addresses, the model may rely too much on these specific features during training. The CICDDoS2019 dataset and the GINKS2023 dataset contain 88 features. Training a model using socket-related features can lead to overfitting [15]. In this paper, we remove all unnecessary socket-related features such as 'Unnamed: 0', 'Flow ID', 'Destination IP', 'Destination Port', 'SimillarHTTP', 'Source IP', 'Source Port', 'Timestamp', 'Inbound', 'Fwd Header Length.1' [15]. To make the model training and performance evaluation more representative and robust, we have ensured 78 features. The feature set can not only improve the model's generalization ability but also help it better adapt to the data differences in different network environments, which helps improve the practicability and transferability of the model.

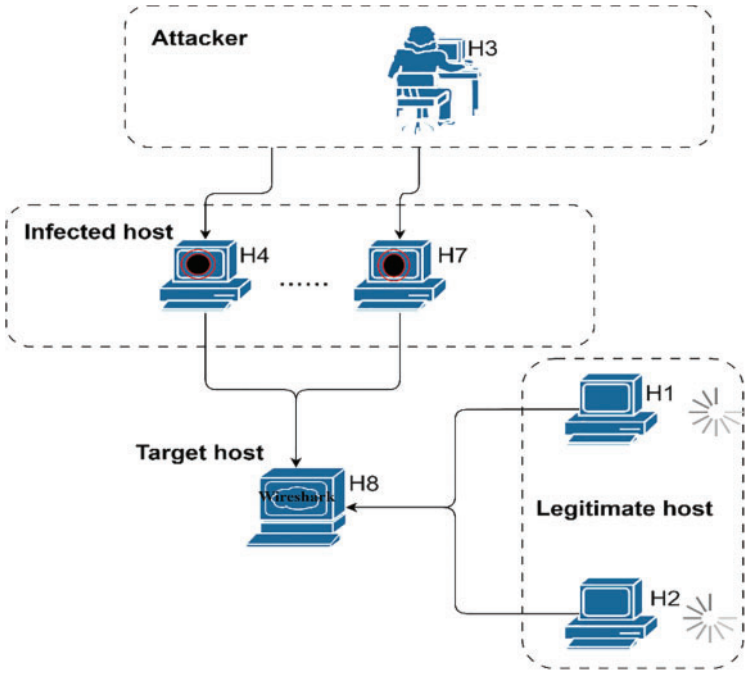

Figure 6: Schematic diagram of DDoS attack on new data

5.2.2 Data Clearing

Missing data or noise impacts the accuracy of the model. So, we replace "Infinity" in our data with 0. Type conversion for columns such as "Flow Packets/s". Part of the data is filled with data, data label encoding as well as temporal hashing processing. Such cleaning operations will help improve the data's accuracy and usability, making it more suitable for model training and testing. At the same time, in order to speed up the model reading and convergence speed, "max-min" normalization is used to map the data to [−1, 1], X_{std} refers to the normalized eigenvalue, X refers to the original eigenvalue, X_{min} refers to the minimum value of the eigenvalue, X_{max} refers to the maximum value of the eigenvalue, and the mapping function is shown in Eq. (20).

$$X_{std} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{20}$$

For example, see Tables 3 and 4. The tables provide a detailed analysis of the number and percentage of benign data and attack data instances in the CICDDoS2019 dataset and GINKS2023 dataset in this paper.

Table 3: Data statistics of CICDDOS2019

Class	Number of instances	Percentage (%)
BENIGN	220,717	55.59
DrDoS_DNS	17,010	4.28
DrDoS_LDAP	8060	2.03

(Continued)

Table 3 (continued)

Class	Number of instances	Percentage (%)
DrDoS_MSSQL	10,030	2.53
DrDoS_NTP	71,825	18.09
DrDoS_NetBIOS	8535	2.15
DrDoS_SNMP	7535	1.90
DrDoS_SSDP	3815	0.96
DrDoS_UDP	10,785	2.72
Syn	1960	0.49
TFTP	18,260	4.60
UDP-lag	18,501	4.66
WebDDoS	24	0.01
Total	397,057	

Table 4: Data statistics of GINKS2023

Class	Number of instances	Percentage (%)
BENIGN	50761	17.89
DDoS attack	233,022	82.11
Total	283,783	

6 Experimental Setup and Model Evaluation

We use TensorFlow for model building, and the model is set with three parameters: Conv_filter, GRU_unit and input_shape. Where Conv_filter refers to the number of convolutional filters, GRU_unit refers to the number of GRU units in the BiGRU layer, and input_shape refers to the shape of the input data. Through the ZOA algorithm, Conv_filter is selected as 7 and GRU_unit as 57. Through the MI-ZOA algorithm, Conv_filter is selected as 4 and GRU_unit as 15. The training batch is set to 256 and the learning rate is set to 0.001 for all models. To further prevent overfitting, we stop training when the error between adjacent epochs is less than 0.01 or when the upper limit on the number of epochs is reached. For one example, see Table 5.

Table 5: MSCNN-BiGRU-SHA parameters

Parameters	MSCNN-BiGRU-SHA
Activator	ReLu, Sigmoid
Optimizer	Adam
Learning rate	0.001
Loss	Binary cross entropy
Conv_filter	7
GRU_unit	57

(Continued)

Table 5 (continued)

Parameters	MSCNN-BiGRU-SHA
Batch size	256
Epochs	50

Experimental hardware: a server with RTX 3080 Ti (12 GB) GPU, 12 vCPU Intel(R) Xeon(R) Silver 4214R CPU @ 2.40 GHz. A computer with 16 GB RAM, 12th Gen Intel(R) Core (TM) i7-12700 2.10 GHz running 64-bit Microsoft Windows® 11 operating system.

Software Environment: Python 3.8 Ubuntu20.04, CICFlowmeter-V4.0, TensorFlow 2.9.0, Cuda 11.2.

6.1 Evaluation Metrics

In this paper, accuracy, precision, recall, and F1 score are used to evaluate the model classification performance. The calculation formula is shown in Eqs. (21)–(24).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{21}$$

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

$$Recall = \frac{TP}{TP + FN} \tag{23}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{24}$$

TP (True Positives) represents the number of instances accurately identified as positive, while *TN* (True Negatives) refers to those correctly classified as negative. *FP* (False Positives) counts the instances mistakenly identified as positive, and *FN* (False Negatives) refers to those incorrectly classified as negative. A higher value for the evaluation metric indicates better classification performance.

6.2 Training Models

In order to test whether the model is effective on the untrained DDoS attack dataset, the DNN [15], CNN [16], GRU [17], LSTM [18], and MSCNN-BiGRU-SHA (MBS) models are trained and tested on the CICDDoS2019 dataset. After each training session, the model's performance is evaluated on the GINKS2023 dataset. Data 1 is the data listed in Table 2. Data 2, for the GINKS2023 dataset in Table 3, random oversampling is performed with the ratio set to balance the benign data and attack data to 1:1. Data 3, the data collection of the CICDDoS2019 dataset where the ratio of benign data to attack data is close to 1:1, and the GINKS2023 dataset where the ratio of benign data to attack data is close to 1:1. Data 4, the data listed in Table 4. Data 5, the data collection of the CICDDoS2019 dataset, where the ratio of benign data to attack data is close to 1:1, and the GINKS2023 dataset, where the ratio of benign data to attack data is close to 2:8. For one example, see Table 6. Detailed data descriptions are provided in Table 6.

Table 6: Description of the evaluation model performance dataset

Data	Description	Total	Benign data (%)	Attack data (%)
Data 1	CICDDoS2019	397,057	55.59	44.41
Data 2	Part of GINKS2023	121,793	41.62	58.38
Data 3	Data 1 + Data 2	518,850	52.31	47.69
Data 4	Part of GINKS2023	283,783	17.89	82.11
Data 5	Data 1 + Data 4	680,840	39.87	60.13

6.3 Experimental Results and Analysis

The datasets are divided into 80% training data and 20% test data using the 'train_test_split' function from sklearn. 'Test' data and 'evaluation' data refer to two distinct datasets used for testing and assessing the models' performance. For one example, see Tables 7 and 8.

Table 7: Test results of five models

Experiment	Train and test dataset	Evaluation dataset	Test dataset accuracy (%)				
			DNN	CNN	LSTM	GRU	MBS
Case 1	Data 1	Data 2	97.83	98.32	98.91	99.31	**99.71**
Case 2	Data 2	Data 1	95.17	94.40	99.30	99.16	**99.59**
Case 3	Data 3	Data 1	96.65	97.91	99.70	99.39	**99.78**
		Data 2					
Case 4	Data 5	Data 1	95.27	98.02	99.61	99.58	**99.70**
		Data 2					

Table 8: Evaluation results of five models

Experiment	Train and test dataset	Evaluation dataset	Evaluation dataset accuracy (%)				
			DNN	CNN	LSTM	GRU	MBS
Case 1	Data 1	Data 2	64.36	63.03	69.82	83.33	**88.05**
Case 2	Data 2	Data 1	58.84	67.70	76.64	77.42	**84.50**
Case 3	Data 3	Data 1	97.82	92.56	99.67	99.30	**99.74**
		Data 2	93.12	94.51	99.54	99.36	**99.55**
Case 4	Data 5	Data 1	94.74	84.55	99.59	99.39	**99.67**
		Data 2	93.89	95.75	99.55	98.95	**99.68**

For Case 1, training tests with the Data 1 dataset and evaluating model detection capabilities on the Data 2 dataset. The MBS test accuracy is 99.87%, the evaluation accuracy is 88.05%, the GRU accuracy is 99.31%, and the evaluation accuracy is 83.33%. The LSTM test accuracy is 98.91%, and the evaluation accuracy is 69.82%. The CNN test accuracy is 98.32%, the evaluation accuracy is 63.03%. The DNN test accuracy is only 97.83%, and the assessment accuracy is 64.36%. The accuracy of all

five models on the evaluation dataset is lower than the test accuracy, indicating that the models are not as accurate as the known dataset for the untrained DDoS attack dataset. However, MBS achieves 88.05% accuracy for the untrained DDoS attack dataset compared to the other models.

For Case 2, training tests with the Data 2 dataset and evaluating model detection capabilities on the Data 1 dataset. The MBS test accuracy is 99.58%, and the evaluation accuracy is 84.50%. The GRU test accuracy is 99.16%, and the evaluation accuracy is 77.42%. The LSTM test accuracy is 99.30%, and the evaluation accuracy is 76.64%. The CNN test accuracy is only 94.40%, the evaluation accuracy is 67.70%. The DNN test accuracy is 95.17%, the evaluation accuracy is 58.84%. The accuracy of all five models on the evaluation dataset is lower than the test accuracy, indicating that the models are not as accurate as the known dataset for the untrained DDoS attack dataset. However, MBS achieves 84.50% accuracy for the untrained DDoS attack dataset compared to the other models.

For Case 3, when a partial evaluation set is added to the training set, the DNN Data 1 evaluation accuracy is 97.82%, and the DNN Data 2 evaluation accuracy is 93.12%. The CNN Data 1 evaluation accuracy is 92.56%, and the CNN Data 2 evaluation accuracy is 94.51%. LSTM Data1 evaluation accuracy is 99.67%, and LSTM Data 2 evaluation accuracy is 99.54%. Moreover, the GRU Data 1 evaluation accuracy is 99.30%, the GRU Data 2 evaluation accuracy is 99.36%. The MBS Data 1 evaluation accuracy is 99.74%, and the MBS Data 2 evaluation accuracy is 99.55% Compared to the Case 1 and Case 2 experimental results, five models have better accuracy than Case 1 and Case 2 for the same evaluation dataset. When the evaluation data is added to the training set to train the models, the models can recognize the attack patterns in the evaluation dataset, which results in high evaluation accuracy.

For Case 4, when the evaluation dataset without data balancing processing is added to the training set, the accuracy of the MBS model on the Data 1 evaluation set is 99.67%. The accuracy of the Data 2 evaluation set is 99.68%, which is 0.07% lower than that of the MBS model Data 1 evaluation set in Case 3, and 0.13% higher than that of the Data 2 evaluation set. The accuracy of the GRU model on the Data 1 evaluation set is 99.39%, and the accuracy of the Data 2 evaluation set is 98.95%, which is 0.09% higher than that of the GRU model Data 1 evaluation in Case 3, and 0.41% lower than that of the Data 2 evaluation set. The accuracy of the LSTM Data 1 evaluation set is 99.59%, and the accuracy of the Data 2 evaluation set is 99.55%, which is 0.08% lower than that of the Case 3 Data 1 evaluation set and 0.01% higher than that of the Case 3 Data 2 evaluation set. The accuracy of the CNN Data 1 evaluation set is 84.55%, and the accuracy of the Data 2 evaluation set is 95.75%. Compared with Case 3, the accuracy of the Data 1 evaluation set is reduced by 8.01%. The accuracy of the DNN Data1 evaluation set is 94.74%, which is 3.08% lower than that of the Case 3 Data 1 evaluation set. The accuracy of the DNN Data 2 evaluation set is 93.89%, which is 0.77% higher than that of Case 3 Data 2. Due to the unbalanced proportion of benign data and attack data in the added evaluation data set. The model learns during training, and cannot correctly learn the characteristics of the evaluation dataset, resulting in the accuracy of the evaluation dataset being lower than the accuracy of the test dataset. However, MBS is still more accurate than the other five models.

6.4 Parameter Optimization

The filters of the convolutional layer of the model refer to the filter used to extract features, and the number of filters affects the capacity and expression ability of the network. The number of units in each GRU layer of the BiGRU layer affects the model's transmission and memory of sequence information. Increasing the number of convolution kernels and GRU units helps to learn more complex features. It makes the model better able to capture more complex sequential relationships, but it also increases the

computational and training costs of the model. The Conv_filter and GRU_unit parameters selected based on experience often cannot make the model achieve optimal performance. Therefore, this paper utilizes the ZOA algorithm and MI-ZOA to optimize Conv_filter and GRU_unit. For one example, see Table 9. In this paper, Data 1 is used as the training set, and Data 2 is used as the evaluation set. Grid Search, Random Search, Bayesian Optimization, ZOA, and MI-ZOA are used respectively to select the Conv_filter and GRU_unit of the MBS model. The MBS model is optimal regarding accuracy, precision, recall, and F1 score.

Table 9: Performance comparison of Conv_filter and GRU_unit parameters of MBS model selected by different algorithms

Algorithm	Parameters	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
Grid search	(128, 64)	58.62	95.83	30.45	46.22
Random search	(8, 32)	57.87	90.45	30.58	45.71
Bayesian optimization	(32, 64)	58.04	94.12	29.99	45.49
ZOA	(7, 57)	88.05	95.81	83.17	89.05
MI-ZOA	(4, 15)	**93.86**	**97.16**	**92.17**	**94.60**

This paper compares the MBS model's floating-point operations per second (FLOPs) and the total number of parameters to be trained in the model under different algorithms for selecting Conv_filter and GRU_unit parameters.

The number of parameters and FLOPs are two important metrics to measure the algorithm's performance. In general, more parameters and a more complex model may achieve better performance, but at the same time, it is more computationally expensive. FLOPs is directly related to the computational efficiency of the algorithm, the higher the FLOPs, the longer the time it may take for the algorithm to execute. For one example, see Fig. 7, MI-ZOA performs relatively well in terms of the number of parameters and computational complexity and is a more efficient algorithm.

In order to further prove that the hyperparameters selected by the improved algorithm can improve the detection ability of the MSCNN-BiGRU-SHA model for untrained DDoS attack datasets. In this paper, the MI-ZOA_MBS model is compared with the most advanced detection models LSTM [25], GRU [25], SDNN [26], and CNN-LSTM [34]. For one example, see Table 10.

For Case 5, Case 5.1 training tests with Data 1 dataset and evaluating model detection capabilities on Data 2 dataset. Case 5.2 training tests with Data 2 dataset and evaluating model detection capabilities on Data 1 dataset. The evaluation accuracy of SDNN model on Data 1 is 42.30%, and the evaluation accuracy on Data 2 is 44.89%. The evaluation accuracy of CNN-LSTM model on Data1 is 58.84%, and the evaluation accuracy on Data 2 is 47.98%, which is significantly improved compared with the SDNN model. The evaluation accuracy of LSTM model on Data 1 is 41.66%, and the evaluation accuracy on Data 2 is 44.75%. Compared with other models, the detection rate is the lowest. The evaluation detection rate of the GRU model on Data 1 is 59.89%, and the evaluation accuracy rate on Data 2 is 44.75. The evaluation accuracy of MI-ZOA_MBS model on Data 1 is 93.86%, and the evaluation accuracy on Data 2 is 89.37%. Compared with SDNN, LSTM, GRU, and CNN-LSTM models, MI-ZOA_MBS has better detectability for the untrained training set than other models. Compared with the ZOA-MBS model, the detection accuracy of the untrained DDoS attack dataset is increased by 5.81% and 4.87%. It is proved that the hyperparameters selected by the

improved algorithm can improve the detection ability of the MBS model for untrained DDoS attack datasets.

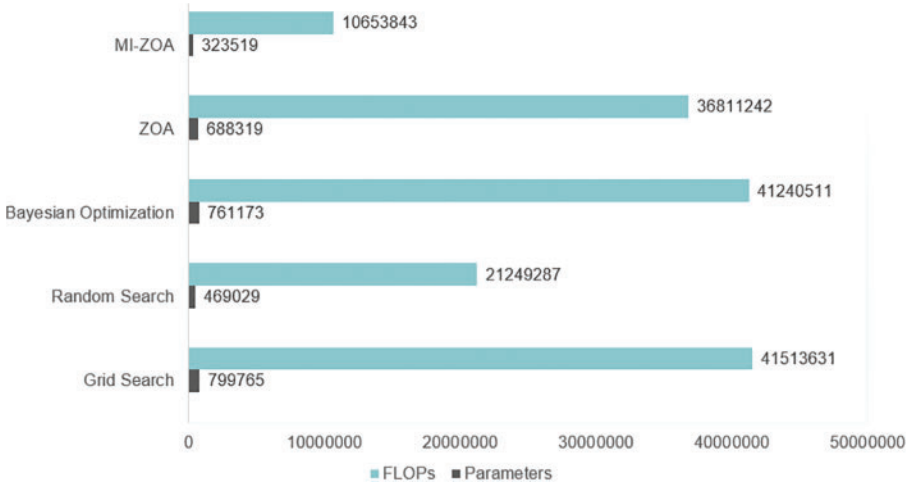

Figure 7: Comparison of parameter counts and FLOPs for different algorithms

Table 10: The evaluation results of the models SDNN, LSTM, GRU, CNN-LSTM, and MI-ZOA_MBS

Experiment		Train dataset	Evaluation dataset	Evaluation dataset accuracy (%)				
				SDNN	CNN-LSTM	LSTM	GRU	MI-ZOA_MBS
Case 5	Case 5.1	Data 1	Data 2	42.30	58.84	41.66	59.89	**93.86**
	Case 5.2	Data 2	Data 1	44.89	47.94	44.75	56.51	**89.37**
Case 6		Data 3	Data 1	55.51	96.98	87.48	91.44	**99.64**
			Data 2	42.40	99.40	86.09	81.41	**99.57**
Case 7		Data 5	Data 1	45.13	96.46	75.34	88.90	**99.66**
			Data 2	58.69	98.77	84.49	88.75	**99.60**

For Case 6 and Case 7, Data 3 and Data 4 are used for the model training data sets, respectively. The evaluation accuracy of the trained models SDNN, CNN-LSTM, LSTM, and GRU on Data 1 and Data 2 is lower than that of the MI-ZOA_MBS model.

7 Conclusion

This paper proposes a new automatic DDoS attack detection model called MSCNN-BiGRU-SHA. We did hyperparameter optimization through the Multi-strategy Integrated Zebra Optimization Algorithm to improve the model's accuracy in detecting untrained DDoS attack datasets. Firstly, the CICDDoS2019 dataset is subjected to a data balancing process. Avoid models that focus too much on features of attack data and ignore the features of benign data when there is a significant difference between the amount of benign and attack data. Secondly, the GINKS2023 dataset is generated as an evaluation set to evaluate the model's detection performance for untrained DDoS

attack datasets. Thirdly, the optimal hyperparameters Conv_filter and GRU_unit values are selected by the Zebra Optimization Algorithm to improve the model MSCNN-BiGRU-SHA performance. On this basis, it carried out comparative experiments with traditional models such as DNN, CNN, LSTM, and GRU. Three sets of experiments, Case 1, Case 2, Case 3, and Case 4, demonstrate that the MSCNN-BiGRU-SHA model based on Zebra Optimization Algorithm optimization has a higher accuracy than other models for untrained DDoS attack datasets. Adding different proportions of evaluation sets to the model training set, proves that the detection accuracy of the MSCNN-BiGRU-SHA model in the evaluation set can reach up to 99.78%. Finally, the MSCNN-BiGRU-SHA hyperparameters Conv_filter and GRU_unit are further optimized using the Multi-strategy Integrated Zebra Optimization Algorithm. The comparison experiment with the most advanced method is carried out. Compared to the MBS models based on Random Search, Grid Search, Bayesian Optimization, and ZOA, the MBS model based on MI-ZOA is optimal in terms of accuracy, precision, recall, F1 score, number of parameters, and FLOPs. Case 5, Case 6, and Case 7 prove that the MSCNN-BIRGRA-SHA model based on the Multi-strategy Integrated Zebra Optimization Algorithm further improves the accuracy of the model for untrained DDoS attack datasets. At the same time, relying on the school platform, the method is deployed in the network outlet. In HW action, it plays an early warning function for DDoS attacks, and the average detection time is 23.44 s.

This study acknowledges several limitations that should be addressed. Firstly, the dataset used for training the model is limited in size, which may affect the generalizability of the results. Secondly, the model is currently limited to handling binary classification tasks, which restricts its potential application in more complex classification problems. Thirdly, the hyperparameter tuning process is based on heuristic methods and is dependent on the dataset used for model training, making the optimal configuration of hyperparameters not unique. Fourthly, although the model shows excellent performance in attack detection, the current research is limited to the detection phase and lacks a defense mechanism against attacks.

To address these limitations, we propose several directions for future work. Firstly, larger and more diverse datasets should be used to train and evaluate the model, ensuring its robustness and generalizability. Secondly, more efficient algorithms and optimization strategies should be explored to reduce the model training time, enabling the model to handle more complex classification tasks and improving its performance and efficiency. Additionally, models that integrate detection and defense functions should be developed to achieve rapid identification and effective defense against attacks, enhancing the overall effectiveness of network security protection.

Acknowledgement: None.

Funding Statement: This work was supported by Science and Technology Innovation Program for Postgraduate Students in IDP Subsidized by Fundamental Research Funds for the Central Universities (Project No. ZY20240335). At the same time, thanks to the strong support of the Research Project of the Key Technology of Malicious Code Detection Based on Data Mining in APT Attack (Project No. 2022IT173) and the Research Project of the Big Data Sensitive Information Supervision Technology Based on Convolutional Neural Network (Project No. 2022011033).

Author Contributions: Study conception and design: Chunhui Li, Xiaoying Wang; data collection: Chunhui Li, Qingjie Zhang; analysis and interpretation of results: Chunhui Li, Jiaye Liang, Aijing Zhang; draft manuscript preparation: Chunhui Li, Xiaoying Wang. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The datasets generated during or analyzed during the current study are available from the corresponding author on reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using artificial neural networks," *Neurocomputing*, vol. 172, no. 7, pp. 385–393, Jan. 2016. doi: 10.1016/j.neucom.2015.04.101.

[2] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Gener. Comput. Syst.*, vol. 28, no. 3, pp. 583–592, 2012. doi: 10.1016/j.future.2010.12.006.

[3] X. Wang, N. Guo, F. Gao, and J. Feng, "Distributed denial of service attack defence simulation based on honeynet technology," *J. Ambient Intell. Humaniz. Comput.*, vol. 15, no. 5, pp. 1–16, 2019. doi: 10.1007/s12652-019-01396-x.

[4] A. Singh and B. B. Gupta, "Distributed denial-of-service (DDoS) attacks and defense mechanisms in various web-enabled computing platforms: Issues, challenges, and future research directions," *Int. J. Semantic Web Inf. Syst.*, vol. 18, no. 1, pp. 1–43, 2022. doi: 10.4018/IJSWIS.297143.

[5] NSFOCUS, "2023 DDoS Threat Report," (in Chinese), 2024. Accessed: May 28, 2024. [Online]. Available: https://rsac.nsfocus.com.cn/html/2024/92_0528/219.html

[6] D. Winder, "Internet History Hacked, Wayback Machine Down—31Million Passwords Stolen," Accessed: Oct. 10, 2024. [Online]. Available: https://www.forbes.com/sites/daveywinder/2024/10/10/internet-hacked-wayback-machine-down-31-million-passwords-stolen/

[7] A. Mishra, N. Gupta, and B. Gupta, "Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller," *Telecommun. Syst.*, vol. 77, no. 1, pp. 47–62, 2021. doi: 10.1007/s11235-020-00747-w.

[8] Y. Gu, K. Li, Z. Guo, and Y. Wang, "Semi-supervised K-means DDoS detection method using hybrid feature selection algorithm," *IEEE Access*, vol. 7, pp. 64351–64365, 2019. doi: 10.1109/ACCESS.2019.2917532.

[9] S. Pande, A. Khamparia, D. Gupta, and D. N. Thanh, "DDOS detection using machine learning technique," in *Recent Studies Comput. Intell.: Doctoral Symp. Comput. Intell. (DoSCI 2020)*, Singapore, Springer, 2021, pp. 59–68. doi: 10.1007/978-981-15-8469-5_5.

[10] J. Xu, X. Chen, Y. Dong, and J. Yang, "DDoS attack detection by random forest fused with feature selection," *J. Comput. Appl.*, vol. 43, pp. 3497–3503, 2023. doi: 10.11772/j.issn.1001-9081.2022111792.

[11] M. Asad, M. Asim, T. Javed, M. O. Beg, H. Mujtaba and S. Abbas, "DeepDetect: Detection of distributed denial of service attacks using deep learning," *Comput. J.*, vol. 63, no. 7, pp. 983–994, 2020. doi: 10.1093/comjnl/bxz064.

[12] O. Sbai and M. El boukhari, "Data flooding intrusion detection system for MANETs using deep learning approach," in *Proc. 13th Int. Conf. Intell. Syst.: Theor. Appl.*, Rabat, Morocco, 2020, vol. 580, pp. 1–5. doi: 10.1145/3419604.3419777.

[13] S. Haider *et al.*, "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks," *IEEE Access*, vol. 8, pp. 53972–53983, 2020. doi: 10.1109/ACCESS.2020.2976908.

[14] S. Dwivedi, M. Vardhan, and S. Tripathi, "Defense against distributed DoS attack detection by using intelligent evolutionary algorithm," *Int. J. Comput. Appl.*, vol. 44, no. 3, pp. 219–229, 2022. doi: 10.1080/1206212X.2020.1720951.

[15] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, "Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic," *IEEE Sens. Lett.*, vol. 3, no. 1, pp. 1–4, 2019. doi: 10.1109/LSENS.2018.2879990.

[16] D. Kwon, K. Natarajan, S. C. Suh, H. Kim, and J. Kim, "An empirical study on network anomaly detection using convolutional neural networks," in *2018 IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Vienna, Austria, IEEE, 2018, pp. 1595–1598. doi: 10.1109/ICDCS.2018.00178.

[17] M. V. Assis, L. F. Carvalho, J. Lloret, and M. L. Proença Jr, "A GRU deep learning system against attacks in software defined networks," *J. Netw. Comput. Appl.*, vol. 177, 2021, Art. no. 102942. doi: 10.1016/j.jnca.2020.102942.

[18] G. Qin, Y. Chen, and Y. -X. Lin, "Anomaly detection using LSTM in IP networks," in *2018 Sixth Int. Conf. Adv. Cloud Big Data (CBD)*, Lanzhou, China, IEEE, 2018, pp. 334–337. doi: 10.1109/CBD.2018.00066.

[19] R. K. Batchu and H. Seetha, "A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning," *Comput. Netw.*, vol. 200, no. 5, 2021, Art. no. 108498. doi: 10.1016/j.comnet.2021.108498.

[20] A. E. Cil, K. Yildiz, and A. Buldu, "Detection of DDoS attacks with feed forward based deep neural network model," *Expert Syst. Appl.*, vol. 169, no. 4, 2021, Art. no. 114520. doi: 10.1016/j.eswa.2020.114520.

[21] E. Benmohamed, A. Thaljaoui, S. Elkhediri, S. Aladhadh, and M. Alohali, "E-SDNN: Encoder-stacked deep neural networks for DDOS attack detection," *Neural Comput. Appl.*, vol. 36, no. 18, pp. 1–13, 2024. doi: 10.1007/s00521-024-09622-0.

[22] M. Roopak, G. Y. Tian, and J. Chambers, "Deep learning models for cyber security in IoT networks," in *2019 IEEE 9th Annual Comput. Commun. Workshop Conf. (CCWC)*, Las Vegas, NV, USA, IEEE, 2019, pp. 452–457. doi: 10.1109/CCWC.2019.8666588.

[23] M. Shurman, R. Khrais, and A. Yateem, "DoS and DDoS attack detection using deep learning and IDS," *Int. Arab J. Inf. Techn.*, vol. 17, no. 4A, pp. 655–661, 2020. doi: 10.34028/iajit/17/4A/10.

[24] M. Ramzan *et al.*, "Distributed denial of service attack detection in network traffic using deep learning algorithm," *Sensors*, vol. 23, no. 20, 2023, Art. no. 8642. doi: 10.3390/s23208642.

[25] H. Beitollahi, D. M. Sharif, and M. Fazeli, "Application layer DDoS attack detection using cuckoo search algorithm-trained radial basis function," *IEEE Access*, vol. 10, pp. 63844–63854, 2022. doi: 10.1109/ACCESS.2022.3182818.

[26] N. A. Alawad, B. H. Abed-alguni, M. A. Al-Betar, and A. Jaradat, "Binary improved white shark algorithm for intrusion detection systems," *Neural Comput. Appl.*, vol. 35, no. 26, pp. 19427–19451, 2023. doi: 10.1007/s00521-023-08772-x.

[27] J. Saikam and K. Ch, "An ensemble approach-based intrusion detection system utilizing ISHO-HBA and SE-ResNet152," *Int. J. Inf. Secur.*, vol. 23, no. 2, pp. 1037–1054, 2024. doi: 10.1007/s10207-023-00777-w.

[28] M. Mittal, K. Kumar, and S. Behal, "Deep learning approaches for detecting DDoS attacks: A systematic review," *Soft Comput.*, vol. 27, no. 18, pp. 13039–13075, 2023. doi: 10.1007/s00500-021-06608-1.

[29] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Comput. Secur.*, vol. 86, no. 1, pp. 147–167, 2019. doi: 10.1016/j.cose.2019.06.005.

[30] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *2019 Int. Carnahan Conf. Secur. Technol. (ICCST)*, Chennai, India, IEEE, 2019, pp. 1–8. doi: 10.1109/CCST.2019.8888419.

[31] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Comput. Ind. Eng.*, vol. 137, no. 5, 2019, Art. no. 106040. doi: 10.1016/j.cie.2019.106040.

[32] E. Trojovska, M. Dehghani, and P. Trojovsky, "Zebra optimization algorithm: A new bio-inspired optimization algorithm for solving optimization algorithm," *IEEE Access*, vol. 10, pp. 49445–49473, 2022. doi: 10.1109/ACCESS.2022.3172789.

[33] Behavior-Centric Cybersecurity Center (BCCC), "GitHub-ahlashkari/CICFlowMeter: CICFlowmeter-V4.0 (formerly known as ISCXFlowMeter) is an Ethernet traffic Bi-flow generator and analyzer for anomaly detection that has been used in many Cybersecurity datsets such as Android Adware-General Malware dataset (CICAAGM2017), IPS/IDS dataset (CICIDS2017), Android Malware dataset (CICAndMal2017) and Distributed Denial of Service (CICDDoS2019)," Accessed: Jan. 5, 2022. [Online]. Available: https://github.com/ahlashkari/CICFlowMeter

[34] J. Du, K. Yang, Y. Hu, and L. Jiang, "NIDS-CNNLSTM: Network intrusion detection classification model based on deep learning," *IEEE Access Pract. Innov. Open Solut.*, vol. 11, pp. 24808–24821, 2023. doi: 10.1109/ACCESS.2023.3254915.