Information Sciences Letters
An International Journal

http://dx.doi.org/10.18576/isl/121231

Detection of DDoS Attacks using Enhanced FS with BRSA-based Deep Learning Model in IoT Environment

Adwan A. Alanazi[1], *Arwa D. Alzughaibi*[2], *Ghada Amoudi*[3], *Manahill I. A. Anja*[4], *Abdelgalal O. I. Abaker*[5], *Salem Alkhalaf*[6,*]

[1]Department of Computer Science and Information, College of Computer Science and Engineering, University of Hail, Hail, Saudi Arabia
[2]Applied college, Taibah University, Taibah, Saudi Arabia
[3]Information Systems Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
[4]Computer Sciences Program, Department of Mathematics, Turabah University College, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia
[5]Applied College, Khamis Mushait, King Khalid University, Abha, Saudi Arabia
[6]Department of Computer, College of Science and Arts in Ar Rass, Qassim University, Ar Rass, Saudi Arabia

Received: 27 Sep. 2023, Revised: 2 Nov. 2023, Accepted: 25 Nov. 2023.
Published online: 1 Dec. 2023.

Abstract: Network assaults and floods, are rising due to the increasing number of IoT devices, posing security and dependability concerns. These attacks cause a denial of service (DoS) and network interruption for IoT devices. Researchers have established multiple methods to track down assaults on weak IoT gadgets. This study provides a deep learning and swarm metaheuristic technique for detecting DDoS assaults in an Internet of Things (IoT) setting. The group search firefly method, a revolutionary improvement on the classic firefly algorithm, is used as a feature selection tool to zero in on the best candidates. In addition, the hyperparameters of the DarkNet are selected and optimized with the help of a suggested technique called the Boosted Reptile Search technique (BRSA) for effective botnet detection. The operatives of the red fox algorithm (RFO) and the triangular mutation operator (TMO) were used to effect this change. The TMO was utilized to progress the misuse phase of the RSA, whereas the RFO was used to improve the exploration phase. The suggested model is verified using the N-BaIoT dataset. Various cutting-edge methods were employed to evaluate and contrast the projected model's efficacy. The outcome proves that the recommended approach is superior to alternatives in identifying multiclass botnet assaults.

Keywords: Boosted Reptile Search Algorithm; Denial of Service; Group Search Firefly Algorithm; Internet of Things; Red Fox Procedure; Triangular Operator.

1 Introduction

The proliferation of IoT devices over the last several years has improved our quality of life. The Cisco study predicted that by 2021, monthly mobile traffic would reach 49 exabytes [1,2]. Symantec found that, on average, attacks on IoT devices occurred every two minutes. Kaspersky predicts that in 2018, researchers will be able to acquire almost four times as many malware samples for IoT devices as they did in 2017 [3]. Most of this malware is sophisticated and harmful to the Internet of Things gadgets. Two examples of common IoT malware, BASHLILE and Mirai, compromised many connected devices by exploiting previously discovered security holes and distributing compromised authentication credentials. It was stated in 2015 that BASHLILE, also known as Gafgyt, existed; in 2016, its successor, Mirai, lived [4].

*Corresponding author e-mail: s.alkhalaf@qu.edu.sa

Infected gadgets will conduct scans to find the next potential weak point and report to the server. In 2016, Mirai allegedly infected IoT devices.

There have been several attempts to secure the Internet of Things (IoT) against botnet assaults. However, more work must be done to perfect a reliable detection system. When dealing with threats, an intrusion detection system (IDS) might be helpful [5]. However, owing to the resource limitation issue of these devices, standard IDSs are typically unable to be implemented for IoT contexts. Similarly, due to their complexity, many IoT devices need to include cryptographic techniques [6]. IDSs may be roughly divided into anomaly detection and abuse finding. The misuse-based, or signature-based, strategy relies on attack signatures, which are available in most public IDSs (such as Snort and Suricata). Signature-based techniques are technically easy to bypass, and the processes behind them must provide more assurance that they will identify novel attacks or variations on existing ones [7]. Using ordinary data as a foundation, anomaly-based solutions aid in the detection of previously undiscovered threats. However, the diversity of IoT devices makes it challenging to gather standard deviation statistics [8].

Conventional IDSs rely on signatures or deep packet inspection (DPI) methods for network intrusion detection. These methods categorize data by using packet headers and content information [9]. When used in high-bandwidth and high-speed backbone networks, such approaches constitute a bottleneck and negatively impact performance [10]. Furthermore, when encrypted communication is sent via the web, these methods do not verify the contents of individual packets [11]. For IoT attack detection, various machine learning (ML) based solutions have been offered; however, the model, notably a (CNN), is significantly superior and more effective than the ML models [12]. The efficient performance of deep learning models, notably CNN models, in the image processing and computer vision sector has led to their widespread use [13]. On the other hand, similar CNN models are also used to identify malicious activity on networks.

1.1. Challenging Issues

The nature of IoT devices and the varied design of the underlying network approaches mean that typical threat detection systems cannot be effectively moved in IoT contexts [14]. Furthermore, the possible attacks may vary from those that target conventional network hardware. The resource limitations of these devices prevent the use of the more cumbersome encryption techniques. But when it comes to private applications like those found in small businesses and intelligent household appliances, IoT devices have become very low-cost to deploy [15]. After infecting the target devices with botnets, the attackers began conducting assaults against the victim nodes.

Additionally, they may avoid detection by formal rule-based systems. While the machine learning-based approach can identify variations between assaults of different types, completely novel attacks may still be conducted occasionally. Lightweight attack detection systems on resource-constrained devices are difficult to construct due to the extensive processing of ML classifiers [16].

1.2. Contribution of the Research Work

By introducing a novel improved features selection approach and an optimized pre-trained model of CNN classifier using a new optimization algorithm, a powerful model is built and put into use to identify IoT botnet assaults. To choose an essential set of features, a two-stage method based on an improved firefly-based selection strategy is presented. The suggested features selection method makes use of between-class dissimilarity and within-class differentiation to effectively reduce excessive and redundant characteristics. To fine-tune the hyperparameters of an evaluable DarkNet'19, a unique and significantly enhanced metaheuristic approach is suggested.

1.3. Organization of the Work

The remaining sections of the paper are as shadows: The literature review that may be utilized to detect a DDoS assault is obtainable in Section 2. In Section 3, we provide a concise summary of the projected FS and classifier. In Section 4, we give a discussion and analysis of the validation of many different DL models. Lastly, the study is summarised in Section 5.

2 Related works

Using optimization-based deep learning and accounting for the smart contract, Ilyas et al. [17] provides a powerful solution for DDoS detection and prevention. After receiving a user request, the traffic is inspected and the user's identity is confirmed with the help of the smart contract. After the user has been authenticated, a response is delivered back to them while a deep neural network optimized with the Poaching Raptor algorithm monitors for suspicious traffic in an effort to identify distributed denial of service (DDoS) assaults. The proposed technique combines the common habits of the raptor with those of the Lobo, which share aspects like as hunting style and poaching, to increase the reliability of the identification. In order to respond to the nonattacker while simultaneously blocking the attacker, it is necessary to log the IP/MAC address of the assailant. The projected technique was evaluated using accuracy; the results were 95.12%.

Using information from the Kaggle website, Al-Juboori et al. [18] have applied a number of machine learning algorithms to the problem of protecting devices against MTM and DoS attacks. The large number of missing statistics necessitated the employment of pre-processing techniques in this research, such as filling in the blanks after obtaining the information. Then, we used four different machine learning methods to spot these breaches: GB, GB, XGBoost, and DT (where GB stands for random forest, XGBoost for extreme boosting, and DT for a decision tree). The performance of these algorithms is measured by a variety of classification metrics. The following was found to be consistent between the two data sets: When it comes to identifying MTM attacks, all algorithms perform at or above 99% across the board, and when it comes to detecting DoS attacks, all algorithms perform at or above 97% across the board. The findings proved the algorithms could recognize MTM and DoS attacks, therefore we put them to action protecting devices against them.

Gebrye et al. [19] built a wide variety of data mining methods. The vast majority of these techniques have the same drawbacks, as is evident from the literature. To address the limitations of previously available PCAP to CSV converters, we provide a sophisticated raw network application. To create a DDOS fit for models, we used a dataset. To ensure the validity of the data, we conducted extensive model evaluations. In terms of spotting the DDOS attack, the random forest model fared best.

Using NSL-KDD datasets, Rajasekaran and Magudeeswaran [20] were able to recognize and classify many types of distributed denial of service attacks, flood, flood, and Slow Loris Brute Force attack. It is anticipated that a classifier based on a Gated (GRU-BWFA) would effectively detect DDoS attacks. Enhanced Salp Swarm Optimisation allows for the selection of the most effective aspects of intrusion detection. In order to evaluate how well the projected classifier detects DDoS attacks in comparison to existing methods, the UNSW-NB15 datasets are used. On the UNSW-NB 15 dataset, the recommended model had a value of 0.9936; dataset had a value of 0.9918. According to the data, machine learning algorithms are superior when recognizing and classifying potential dangers. The methods of machine learning, such as CNN, DT, and KNN, are used by Azizpour and Majma [21] to detect DDoS assaults. The intrusion detection systems of The Fog are now capable of identifying 99% of attempted DoS and DDoS attacks. Edge and fog nodes are both supported by NADA. Harmful communications may be identified via deep learning and a biological or genetic system. After that, a full vote-based validation is carried out, with the samples and CNN, DT, and KNN suspicious traffic. (NADA) performed calculations to determine precision, accuracy, recall, and error. The simulations performed by NADA were 7% more accurate than those performed by the technologies before it. Explainable artificial intelligence (XAI) is the basis for a novel DDoS detection method proposed by Kalutharage et al. [22]. An examination of the traffic at the network level could reveal unexpected tendencies. In addition, it chooses the most significant characteristics, gives a threshold value to each feature, and weights the significance of each attribute based on its relevance to each unique occurrence. (DoS) attacks by the use of feature threshold values, such as application layer, volume, and TCP state depletion, allows the implementation of security rules even while an assault is in progress. The data from the third layer enables the recommended technique to detect DDoS attacks on networks. The recommended method was put through a battery of tests utilizing the dataset provided by the USB-IDS component of the Intrusion Detection Scheme. In terms of detection accuracy and attack certainty, the approach that was projected beats the art.

In Pillai and Sharma's [23] study, deep learning was used for hybrid unsupervised detection. The (GAN) can enhance feature representation to identify online threats since it is fed the encoded outputs of the DAE and SAE. We developed a DBM-Bi LSTM classification model to categorize the various types of cyberattacks. DBM and Bi-LSTM are the types of assaults that are used. Classifier performance may be evaluated using several metrics, including accuracy. Our accuracy increased to 98% thanks to the approach. Also, some important quantum and classical techniques have been used for data classification [24-26].

3 The Proposed Model

The suggested perfect in this study employs a different features assortment strategy and a different detection strategy to boost the presentation of network replicas in an IoT context and create a more reliable and secure environment. Here, we dissect the model and lay out its parts for your review. Figure 1 depicts the major features and components of the projected perfect.

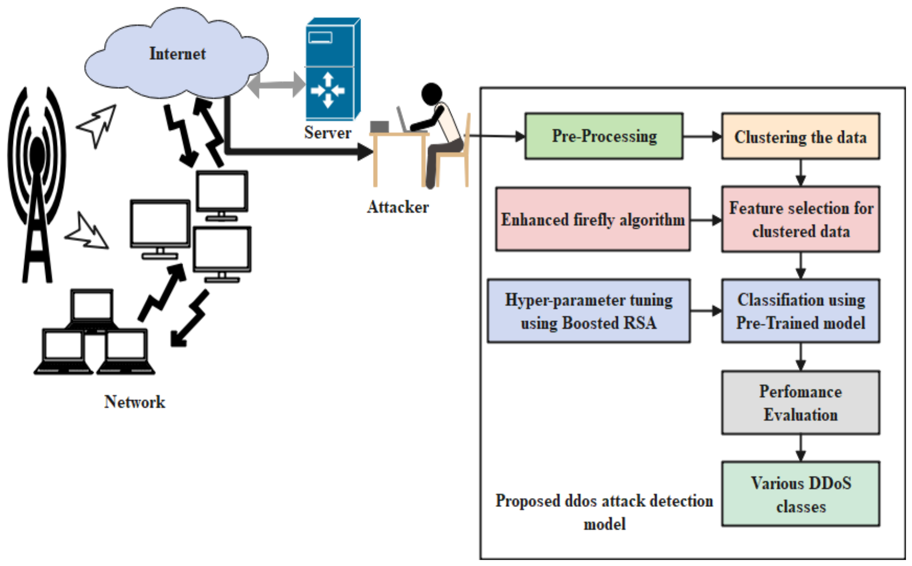
Internet
Server
Attacker
Network
Pre-Processing
Clustering the data
Enhanced firefly algorithm
Feature selection for clustered data
Hyper-parameter tuning using Boosted RSA
Classifiation using Pre-Trained model
Perfomance Evaluation
Various DDoS classes
Proposed ddos attack detection model

Fig.1: Working Flow of Projected Model

The dataset has been partitioned into sets after the preparation of the data, as seen in the figure. The following five phases of the suggested model are implemented:

- Data preprocessing,
- Dataset clustering,
- Features assortment for apiece cluster,
- Classification,
- Assessment.

The purpose of the data preparation phase is to prepare the dataset for analysis. The clustering process is meant to unearth underlying structures in the data.

3.1. Data Preprocessing

Successful data extraction relies on high-quality inputs, which may be achieved by data pretreatment. Data preparation in ML mostly involves cleaning and arranging raw data for use in training DL and ML models. In this research, we used a data cleaning and normalization procedure.

1) Data normalization

Data normalization is an essential data mining approach when working with a dataset, including characteristics of varying sizes. Its goal is to normalize their sizes such that all of the characteristics are equally weighted in the final model. The N-BaIoT dataset includes numerical characteristics. Therefore, the Min-Max normalization approach may be used to standardize all of them, as shown in Equation (1). Each feature in the N-BaIoT dataset has been normalized such that it lies on a scale from 0 to 1.

$$Anew_x = \frac{A_x - \min A_x}{\max A_x - \text{mi}\, A_x} \quad (1)$$

Let's take the xth sample of a dataset before normalization (denoted by A_x) and compare it to the xth sample of the same dataset after normalization (denoted by $Anew_x$). We find that the lowest and extreme values on the new-fangled scale variety from 0 to 1.

2) Data cleaning

Data removing inaccurate or misleading information. At first, we get rid of all the duplicates in the data set. Features with these duplicate values do not enhance the model's accuracy but do increase its complexity. The second step is to deal with missing and null data. All blanks or zeros have been replaced with the estimated median. Median imputation was used instead of mean innuendo because it is more robust against mistakes caused by extreme values.

3.2. Clustering the data

Clustering the data is an important preprocessing step that may boost the reliability of the classification. Clustering algorithms organize data by dividing it into groups whose members are increasingly similar to one another inside each group and decreasingly similar to one another across groups.

$$d_{xy}^2 = (x_1 - y_1)^2 + (x_2 - y_2)^2 \tag{2}$$

$$d_{x,y} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \tag{3}$$

The objective is to find a practical approach of retaining an important dataset while using fewer actual data points. Using a unique hybrid filter- for tackling the dimensionality issue. The huge number of features issue may be circumvented by the use of clustering to improve feature evaluation outcomes.

Clustering may be done in a number of ways, but the used is the K-means technique. The data structure is revealed, and the cluster is constructed using K-means clustering. Select k random characteristics from the original dataset to begin clustering. Based on how well their attributes align with the cluster mean, clusters are given their most comparable items. The average of each cluster is recalculated. We kept trying until no substantial feature redistribution occurred in any cluster. Kmeans clustering requires a user-specified number of clusters. We have divided these results into three groups. The Euclidean distance function, as shown in Equation (2), may be used to calculate the degree of similarity between two objects. Equation (3) represents the square of the distance between two vectors x2]. The distance between the x and y axes is represented by the d value in the coordinate system. After grouping similar characteristics together into distinct clusters, the next step is to remove any repetitions. Next, we'll go into the specifics of the feature selection process.

3.3. Features selection for each cluster

Following a brief overview of the foundational FA technique, this section delves into the rationale behind the suggested enhancements and the system's inner workings.

3.3.1. Original Firefly Procedure

The social behaviour of fireflies served as inspiration for the firefly procedure [24], a swarm intelligence model. The FA metaheuristic takes inspiration from the firefly's luminescence and attractiveness to represent its fitness functions. The scientists used numerous approximation principles to reduce the complicated structure of the insects' flashing behaviour. The value of the goal function is what determines the attractive force between the units. The algorithm for the minimization problem is described by the following Equation [4]:

$$I(x) = \begin{cases} 1/f(x) & ,if(x) > 0 \\ 1 + |f(x)| & if\ f(x) \leq 0 \end{cases} \tag{4}$$

where $I(x)$ stand for desirability and $f(x)$ is the worth of the impartial purpose at point x.

And since light strength decreases with distance, the attraction value also decreases [24]:

$$I(r) = \frac{I_0}{1+\gamma \times r^2} \tag{5}$$

anywhere I(r) is the light strength at distance r, is the preoccupation coefficient limit, and I_0 is the light intensity at the light's origin. Most FA applications approach the following Gaussian form [24] by combining the belongings of the opposite square rule for distance and g.

$$I(r) = I_0 . e^{-\gamma \times r^2} \tag{6}$$

Each firefly has a distance-adjusted attraction that increases in direct proportion to how bright it is

$$\beta(r) = \beta_0 . e^{-\gamma \times r^2} \tag{7}$$

in which β_0 is pull at distance $r = 0$. The writers of the unique FA propose that Equation (7) is often exchanged for the subsequent reckoning:

$$\beta(r) = \beta_0/(1 + \gamma \times r^2) \tag{8}$$

The search reckoning for the chance separates i, who in repetition t+1 travels to a new position x_i in the way of extra firefly j with a higher fitness value, is [27] based on Equation (8):

$$x_i^{t+1} = x_i^t + \beta_0 . e^{-\gamma \times r_{i,j}{}^2}\left(x_j^t - x_i^t\right) + \alpha^t(k - 0.5) \tag{9}$$

where $r_{\mathrm{i,j}}$ is the separation among firefly i and j, k is a chance value a unchanging distribution, and an is the randomization parameter. The optimal values for β_0 and an are one and [0,1], respectively. Cartesian distance, denoted by the r_(i,j) parameter, is determined by the formula::

$$r_{\mathrm{i,j}} = \left\|x_i - x_j\right\| = \sqrt{\sum_{k=1}^{D}\left(x_{i,k} - x_{j,k}\right)^2} \tag{10}$$

where the parameter D signifies the sum of the limits of a specific problem.

3.3.2. Motivation and Predictable Improved Group Search Firefly Procedure

Based on previous research, we know that the fundamental FA is capable of reasonably effective exploitation but has room for improvement in its exploratory capabilities [28, 29]. Although the enhanced/hybridized form of FA has seen widespread use in recent years, there is always room for development in the field. Because the search equation of the FA, which performs efficient intensification, may be efficiently integrated with unique mechanisms and processes from different metaheuristics, the scope of possible enhancements is almost limitless.

This paper proposes an improved version of FA that uses the disputation operator from the new algorithm to address the application. Therefore, the term "group search" is utilized in this investigation in place of the more common "disputation" to describe the search process that this operator really performs inside a selected subset of the population's solutions.

When people on social networks are discussing and defending their perspectives on a topic to one another, this is known as the disputation phase. Users may also band together into themed discussion groups. Users are swayed by the availability of competing viewpoints on a certain subject. In this stage, Equation (11) is used to get fresh perspectives by observing a random sample of users as commenters or group members:

$$x_{i\,new} = x_i + rand(0,1) \times (M - AF \times x_i)$$
$$M = \frac{\sum_{T}^{N_R} x_t}{Nr} \tag{11}$$
$$AF = 1 + round(rand)$$

where x_i is a vector on behalf of the i-th user's opinion, rand (0, 1) is a vector drawn at random from the interval [0, 1], and M is the average opinion. The admittance factor (AF), which only accepts integer values (1 or 2), is used to show how firmly people believe their views while debating them. While rand is a random sum in the range [0, 1], the round () function is used to round the input to the closest integer. The number of commenters or the size of the group is indicated by the Nr parameter. Where N is the total sum of people on the network, it may take on any positive integer value among 1 and N.

The AF parameter determines how much emphasis is placed on either intensification or diversity throughout the search process. Adjusting the AF to a value of 2 prioritizes exploration, while setting it to a value of 1 facilitates more robust exploitation via the disputation technique. The exploration and exploitation procedures used by the disputation operator are described in great length in [60]. Since the disputation operator employed in the SNS metaheuristic differs somewhat from the one suggested in this research, the term "group search" this investigation.

With the SNS method, the AF can only have the values 1 and 2 since the first operand in the AF look, Equation (11), is always 1. Empirical tests undertaken for the purpose of this investigation, however, suggest that it is preferable to first establish a bigger step size, which emphasizes exploration, and then progressively lower it during the duration of a run, favouring diversification. Furthermore, it is preferable to allow AF to take incessant value in order to provide fine-tuned search.

To account for this, the suggested technique adapts a second, dynamic control parameter called the and uses the following Equation to determine the step size AF at each iteration:

$$AF = gsp + round(rand) \tag{12}$$

where the GSP decreases in size dynamically during the course of the run by an amount equal to T times the iteration number t:

$$gsp = gsp - \frac{t}{T} \tag{13}$$

The ratio of exploiting to exploring may be adjusted by the value of the gsp parameter. This parameter initially has a bigger value (in actual tests, the beginning 2, but it is dynamically lowered during the iterations) to ensure that exploration is successful in the early rounds. The suggested technique improves upon the SNS algorithm by using a finer-grained step size AF for the search.

The proposed approach also uses two different kinds of group search. In mode 1, solutions from the population are selected at random (N_r), whereas in mode 2, the group is defined as the N_b (mode 2). Random integers N_r and N_b are generated between 1 and N for each cycle. Equation (26) is used in both modes, with the step size determined by Equation (12).

In subsequent rounds, when the algorithm is expected to have converged to the best area of the search space, group search mode 2 is conducted to do fine-tuned searches around the present best answers. The parameter is dependent on the finish condition argument to decide when in the procedure's execution mode 1 is converted to mode 2.

Most improved implementations of prior FAs attempt to remedy the problems of insufficient exploration by including techniques that may promote diversity in preliminary trials. However, the methodology suggested in this research takes a different tack. Again based on empirical evidence, we know that an efficient FA's search technique may quickly converge to an optimal solution if the initial population formed by the FA is close to optimum areas. When the opposite occurs, however, sub-optimal regions of the search space will be settled upon by the whole population. Therefore, the approach described in this paper fires this operation after the gss (group search start) rounds, giving an accidental to the fundamental FA's search without considerably increasing the computational time complexity.

Whenever the prerequisites for initiating a group search are met, a new-fangled solution x_new is created, and the current worst solution x_worst is chosen through greedy iteration.

The suggested approach is called group search FA (GSFA), after the group search mechanism is established in the original FA. Three additional control limits, one of which is a dynamic one, are introduced by the GSFA. Both T and the extreme sum of fitness function evaluations (FFEs) affect the values of all three parameters. The experimentally established values for these parameters are shown in Table 1 and are utilized in subsequent simulations.

Table 1. Exact GSFA limits' settings.

Parameter	Expression	Description
GSP	$gsp = gsp - \frac{t}{T}$	dynamic group search limit, preliminary value 2
gss	$gss = \frac{T}{2}$	group search start
CMT	$cmt = gss + \frac{T}{3}$	change mode trigger

In addition to contradicting previous studies [29], the suggested GSFA does not use a dynamic randomization parameter a. In addition, testing with a dynamic randomization parameter revealed that the converge too rapidly towards the unpromising answer, resulting in poorer results towards the conclusion of a run. In [30], the computational complexity of the unique FA method is given in terms of FFEs. The proposed GSFA is more challenging than the normal FA only for (T - gss) FFEs due to the fact that after initiating the group search, the solution is produced in each iteration. In order to provide reliable results, this factor was taken into consideration in the comparative research.

3.4. Classification using DarkNet19

The CNN used here is called DarkNet19. We utilized a model called DarkNet19, which consists of 19 convolutional layers. DarkNet19 is made up of 25 max-pooling layers and 19 (CL), with multiple 11 CL to 33 minimum triangle parameters. By using a transfer learning approach to 75,000 photos that serve as the network's features, we are able to train the pre-trained network. At this point, the photos were classified using DarkNet19 into a total of 1000 classes; however, we only used the first 5 classes for analyzing the data. Additionally, DarkNet19's input layer accepts images up to 256 by 256 pixels in size. The maximum input size for this network is 244 by 244. The value of parameters and other specifics of the architectural neural network of DarkNet19 are provided in Table 2.

Table 2: Description of DarkNet19 construction.

Type	**Size/stride**	**Filters**	**Output**
Convolutional	3×3	256	28×28
Maxpool	2×2/2		14×14
Convolutional	3×3	34	224×2 24
Maxpool	2×2/2		112 ×112
Convolutional	3×3	64	112×112
“	3×3	128	56×56
“	1×1	64	56×56
“	3×3	128	56×56
Maxpool	2×2/2		28×28
Convolutional	3×3	256	28×28
“	3×3	512	14 ×14
“	1×1	256	14×14
“	3×3	512	14 ×14
“	1×1	256	14 ×14
“	3×3	512	14×14
Maxpool	2×2/2		7×7
Convolutional	3×3	1024	7×7
“	1×1	512	7×7
“	3×3	1024	7×7
“	3×3	1024	7×7
“	1×1	5	7×7
New_softmax	Global	Filters	5

3.4.1. Hyper-parameter Tuning using Boosted RSA

3.4.1.1. The Reptile Search Procedure

This part introduces the traditional RSA, which mimics the behaviour of real crocodiles. The process consisted of two phases: the first local search and the subsequent worldwide search.

Exploration Search

The RSA will transition from the examination stage to the exploitation phase when the maximum number of peers is alienated into four shares, as per [31]. In addition, in the RSA exploration phase, the search areas are explored using the two major search algorithms to find a more trustworthy agent. This part of the search process requires two things. Agents are changed by t 2 T4 and t > T4 if the extended mobile plan is not updated by t T4. The locations for the exploration phase may be updated using Equation (14).:

$$x_{ij}(t+1) = \begin{cases} Best_j(t) \times -\eta_{ij}(t) \times \beta - R_{ij}(t) \times rand, \\ t \leq \frac{T}{4} \\ Best_j(t) \times x_{(r_1,j)} \times ES(t) \times rand, \\ t \leq 2\frac{T}{4} \text{ and } t > \frac{T}{4} \end{cases} \tag{14}$$

where $Best_j$ denotes the best answer at j. $x_{(r_1,j)}$ is the value of a key chosen at random in dimension j, where rand is a random number and =0.1 is the same as in [31]. There have been T generations thus far. The sum of solutions, N, and the searching operator, _ij, are shown, respectively. The symbol _ij has the following definition:

$$\eta_{ij} = Best_j(t) \times P_{ij} \tag{15}$$

$$R_{ij} = \frac{Best_j(t) - x(r_2, j)}{Best_j(t) + \in} \tag{16}$$

$$ES(t) = 2 \times r_3 \times \left(1 - \frac{1}{T}\right) \tag{17}$$

In Equation (17), ∈ mentions to a minor value. $r_3 \in [-1, 1]$ is a random number value, while P_{ij} is definite as

$$P_{ij} = \alpha + \frac{x_{ij} - M(x_i)}{Best_j(t) \times \left(UB_{(j)} - LB_{(j)}\right) + \in\prime} \tag{18}$$

In Equation (18), $UB_{(j)}$ and $LB_{(j)}$ signify the limits of the search area, and $\alpha = 0.1$. $M(x_i)$ denotes the mean worth of X, and it is given as

$$M(x_i) = \frac{1}{n} \sum_{j=1}^{n} x_{ij} \tag{19}$$

Exploitation Search

Using the hunting organization plan, $t \leq 3\frac{T}{4}$ and $t > 2\frac{T}{4}$ adopt the go-betweens in this phase. In contrast, if $t \leq T$ and $t > 3\frac{T}{4}$, The agreed-upon cooperative hunting strategy is implemented. The revised value obtained by exploitation is given by Equation (20).

$$x_{ij}(t+1) = \begin{cases} Best_j(t) \times P_{ij}(t) \times rand, \\ t \leq 3\frac{T}{4} \text{ and } t > 2\frac{T}{4} \\ Best_j(t) - \eta_{ij}(t) \times \in - R_{ij}(t) \times rand, \\ t \leq T \text{ and } t > 3\frac{T}{4} \end{cases} \tag{20}$$

3.4.1.2. Red Fox Algorithm

The parameters are first determined, and then the populace of N foxes is generated using Equation (21).

$$X_{ij} = LB_j + r \times \left(UB_j - LB_j\right), i = 1, \ldots, N, j = 1, \ldots, D, r \in [0,1] \tag{21}$$

where LB_j and UB_j indicate the limits of the search area, and D is the dimension of the foxes' X_ij space. The optimal answer (X_b) is then assigned based on the objective value of X_i. The next step is to revise X_ir using the following Equation.:

$$X_{ir} = X_i + a\,sign(X_b - X_i) \tag{22}$$

In Equation (22), $a \in [0, d_{ib}]$ stands for a stochastically selected scaling hyperparameter, and d_ib means

$$d_{ib} = \sqrt{\|X_i - X_b\|} \tag{23}$$

In case X_{ir} has a If X_ir has a higher fitness value than X_i, then X_i is replaced with X_i. Otherwise, we held onto X_i as it was. In the event that the fox is not seen, the observation radius r is modified using the formula (24):

$$r = \begin{cases} a \frac{\sin(\phi_0)}{\phi_0} & if\ \phi_0 \neq 0 \\ \theta & otherwise \end{cases} \tag{24}$$

In Equation (24), $\theta \in [0, 1]$ is a a random number used to regulate inclement weather (fog, rain, etc.). After that, X is modified using the subsequent Equation:

$$\begin{cases} x_0^{\overset{ne2.}{35w}} = ar.\cos(\phi_1) + x_0^{act} \\ x_1^{new} = ar.\sin(\phi_1) + ar.\cos(\phi_2) + x_1^{act} \\ x_2^{new} = ar.\sin(\phi_1) + ar.\sin(\phi_2) \\ \quad +ar.\cos(\phi_3) + x_2^{act} \\ x_{n-1}^{new} = ar.\sin(\phi_1) + ar.\sin(\phi_2) \\ \quad + \cdots + ar.\sin(\phi_{n-1}) + x_{n-1}^{act} \end{cases} \tag{25}$$

where $\phi_i \in [0, 2\pi]$, $i = 1, 2, \ldots, n - 1$ and each bony value signifies a randomized one for apiece point.

This equations model the fox's assault strategy when identifying potential prey. The fitness value of X_i is then evaluated, and the values are ranked accordingly. Hunters may eliminate the poorest solution, and the X solution might be revised using Equation (26).

$$X = \begin{cases} nomadic\ agent \qquad if > 0.5 \\ Reproduction\ of\ the\ alpha\ couple\ \ o.w. \end{cases} \tag{26}$$

Consequently, in the first fork of Equation (26), the repopulating solutions act as migratory agents, venturing beyond the environment in search of fresh territory. The answer is picked at random and is always found outside of the habitat and inside the search zone. The formula for finding the habitat's geographic centre (C_{H}) is

$$C_{\mathrm{H}} = \frac{X_b + X_\beta}{2} \tag{27}$$

In Equation (27), X_b and X_β represent the first- and solutions, correspondingly.

$$X = \frac{X_b + X_\beta}{2} \tag{28}$$

In addition, in the additional case of Equation (26), the following formulation is used to update X.

3.4.1.3. Triangular Mutation Operator

An integration vector is used in the mutation method, and it is built from three different vectors that have been chosen during the competition (the best, the worst, and the best). The resulting hybrid vector is used to generate the mutant one.

$$V_i(t+1) = M_1 \times (X_b(t) - \bar{X}_{br}(t) + M_2 \times \left(X_b(t) - X_w(t)\right) \\ +M_3 \times \left(X_{br}(t) - X_w(t)\right) + \bar{X}_c(t)) \tag{29}$$

In Equation (29), M_1, M_2, and M_3 are formed according to a consistent distribution, and pertain to the xi-related mutation factors. Furthermore, three rivalries are listed as $X_w(t)$, $X_{br}(t)$, and $X_b(t)$. Furthermore, the triangular combination vector $X_c(t)$ is defined as:

$$\bar{X}_c(t) = w_1 \times X_b + w_2 + X_{br}(t) + w_3 \times X_w(t), w_i \geq 0, \sum_{i=1}^{3} w_i = 1 \tag{30}$$

where w_i is a real heaviness and w_i is expressed as w_i = pi/ $\sum_{i=1}^{3} P_i$.

The mutation is also employed to strike a balance between the inclination towards exploitation and the ability to explore, with the main mutation leaning towards the latter. Therefore, employing the proposed mutation has a 50% better chance of success than using the basic principles alone. The advanced mutation methodology is based on the discrepancy evolution (DE) method. As a result, the following permutation is produced using the foundational mutation strategy DE/rand/1/bin:

If $rand \leq \left(\frac{2}{3}\right), then$

$$V_i(t+1) = M_1 \times \left(X_b(t) - X_{br}(t)\right) + M_2 \times \left(X_b(t) - X_w(t)\right) \\ +M_3 \times (X_{br}(t) - X_w(t)) + \bar{X}_c(t) \tag{31}$$

Else:

$$V_i(t+1) = X_{r1}(t) + F\left(X_{r1}(t) - X_{r3}(t)\right) \tag{32}$$

In Equation (32), F is a chance value at ∪ $(0,1)]$ and rand 2 [0, 1] a random value.

3.4.1.4. Proposed RSRFT Technique

Here, we lay out the steps involved in the created method, which involves adjusting the RSA's performance by means of TMO and RFO. RFO's primary goal is to speed up the process of finding a workable solution inside the workable region, whereas TMO is used to foster cooperation between the exploitation and exploration phases. RSRFT builds its seed population entirely inside the bounds of the search space. After calculating each solution's fitness, the best candidate is picked from the pool of candidates. The RSA, RFO, and TMO operators are used to update the solutions. A group of people who have come to a local halt will then be brought up to date using the RFO method. Repeated population updates are performed until the final criterion is met. In the subsequent sections, we will label the RSRFT method in more depth.

Initial Phase

The RSRFT makes the N agent starting inhabitants (X_{i}) as

$$X_{\mathrm{i}} = rand(1, D) \times (UB - LB) + LB, i = 1, \dots, N \tag{33}$$

In Equation (33), $rand \in [0, 1]$ attitudes for a chance vector with measurement D. N is the sum of solutions.

Updating Phase

The RSA, RFO, or TMO operators are used to modify the current value of X. There is a method to do this. The fitness value for X_{i} must be determined first.

We then find the solution that corresponds to the least possible fitness value. Then, X is updated using either the RSA's operators or RFO and TMO, depending on the fitness value's likelihood. To calculate this likelihood, we have

$$Pr_i = \frac{Fit_i}{\sum_{i=1}^{N} Fit_i} \tag{34}$$

Then, the informing is achieved using the subsequent formula:

$$X_i(t+1) = \begin{cases} X_i^{RSA} & if\ Pr_i > r_{p_r} \\ X_i^{RT} & otherwise \end{cases} \tag{35}$$

where r_{p_r} is computed as

$$r_{p_r} \min(Pr) + (\max(Pr) - \min(Pr)) \times rand \tag{36}$$

In addition, the value of X_i^{RT} is computed as

$$X_i(t+1) = \begin{cases} X_i^{TMO} & if\ rand > 0.5 \\ X_i^{RFO} & otherwise \end{cases} \tag{37}$$

$$X_i^{TMO} = \begin{cases} Use\ equation\ (31)\ to\ update\ X_i & if\ rand > 0.5 \\ Use\ equation\ (32)\ to\ update\ X_i & otherwise \end{cases} \tag{38}$$

while the standards of X_i^{RFO} and X_i^{RSA} refer to the efficient value of Xi using the workers of RFO and the RSA, correspondingly.

Terminal Phase

During this stage, we check if the halting requirements have been met, and if they have, we stop the updating procedures and return X_b. If not, the upgrading process starts again.

4 Results and Discussion

In this article, we'll examine the efficacy of several optimal neural network models in identifying botnets in IoT systems and compare their findings. In this case, we make use of multiclass classification, a technique that learns to categorize threats into more specific subcategories based on their characteristics. Using the suggested model, we carefully choose the characteristics needed to build a reliable neural network IoT botnet detection system.

4.1. Data Set Description

The 115-feature N-BaIoT dataset [32] is utilized for training. Numerous studies have used and implemented the botnet identification in an IoT/IIoT setting [33]. Many relevant and prior publications have utilized the same dataset, hence it was deemed appropriate for use in this study. The used dataset was designed to facilitate not only binary classification but also multiclassification. By emulating the connections of IoT gadgets, researchers were able to collect the data. The harmless information was gathered as soon as the network was properly setup.

Table 3. The perfect and device category of the N-BaIoT dataset.

Name of perfect	**Kind of device**
SimpleHome XCS7-1003-WHT	22
Ecobee	Thermostat
Samsung SNH 1011 N	Webcam
Ennio Danmini	Doorbell
Philips B120N/10	Baby monitor

Every statistical feature, including the timestamps of each packet's arrival, was extracted together with the number of packets, their sizes, and their jitter. Our model makes use of all 115 characteristics; the ones with the highest weight were determined using a features selection technique. Injections of Mirai and Bashlite assaults were employed to capture datasets, and the Internet of Things devices utilized to collect them are detailed in Table 3. Lizard Squad created Bashlite, also known as gafgyt, in C. This botnet infects devices and uses them to perform distributed service assaults. Several protocols, including UDP and TCP, may be attacked via a flooding technique. In extensive assaults against IoT, cybercriminals like Paras use their Mirai virus. In order to recreate all 10 of the attack samples, we choose N-BaIoT devices including a webcam. Figure 2 shows the (CM) for the dataset.

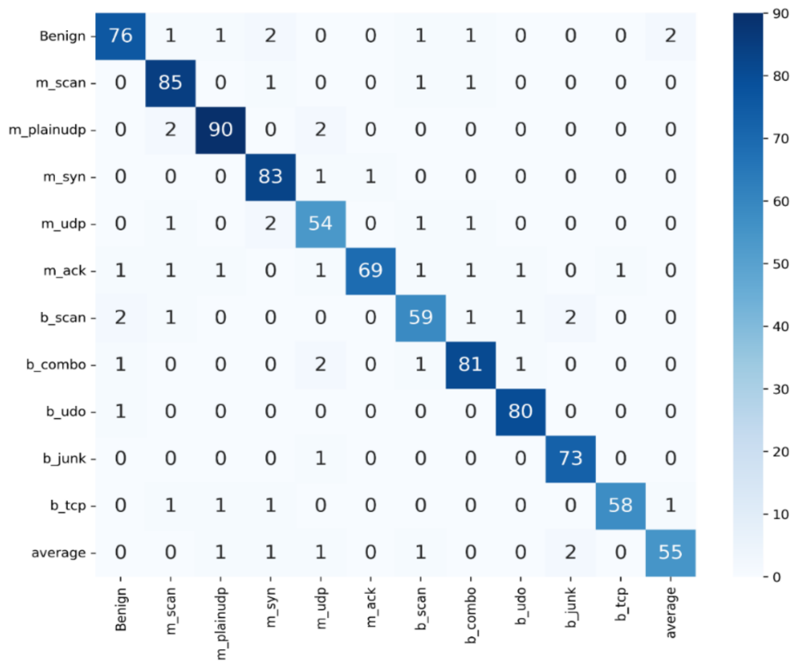

Fig. 2: CM of the proposed research work.

4.2. Data splitting

Afterward the data has been preprocessed, it is split into a training set and a testing set so that the effectiveness of the projected model can be evaluated. The samples in the N-BaIoT dataset were divided at random for this study. For data partitioning, we turn to the sklearn package's train_test_split method. The dataset was split into a set (consisting of 75%) and a testing set (consisting of 25%). The training set was used to construct and fine-tune the model, while the test set was used to assess its efficacy. In addition, a validation set composed of 20% of the training data was used to prevent overfitting. To avoid the validation loss rising as the training loss falls, we calculate it in the training phase.

4.3. Experimental Setup

Using the TensorFlow framework and the scikit-learn and pandas components, our team built the suggested model from the ground up in Python. We used a machine with a 64-bit form of Windows 10 and 16 GB of RAM, along with an Intel Core i7 CPU.

The research used four separate measures to evaluate the presentation of the suggested model. The matrix is used to calculate measures of precision, F1-score, accuracy, and recall. The confusion matrix is a data table that includes columns for both accurate and incorrect classifications. Keep in mind that the true negative, false negative, positive, and false positive counts all originate from the confusion matrix. Figure 3 and 4 shows the precision-recall curve and ROC for the proposed model.

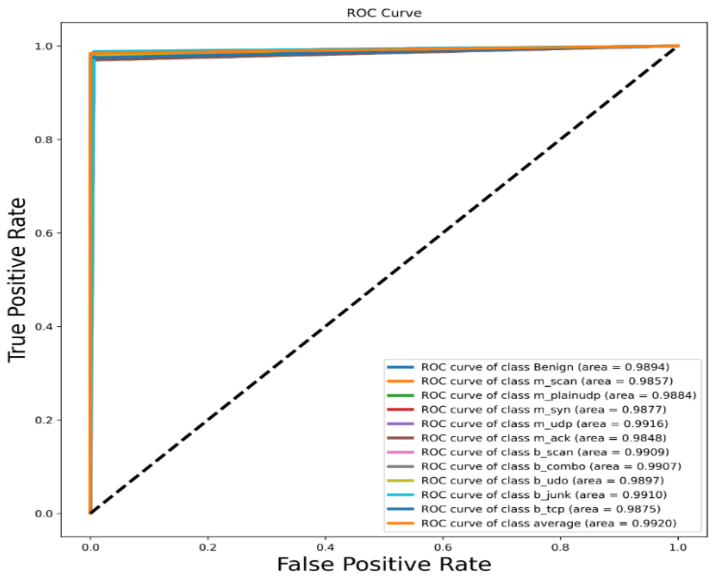

Fig.3: ROC Curve for the proposed model.

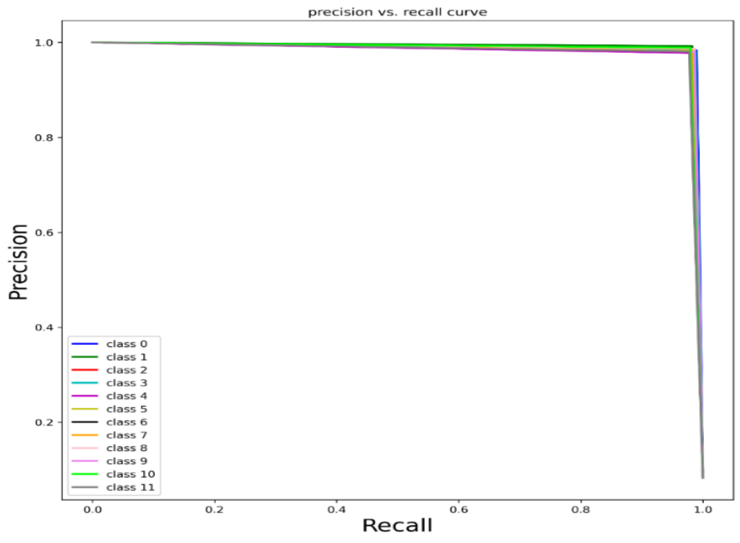

Fig.4: Precision-Recall Curve for multi-class attack.

4.4. Validation Analysis of Proposed Model

Table 4. Optimizing the testing the proposed network's presentation using 10 types of assaults against a single kind of benign.

Category of attack	Accuracy	Precision	F1-Score	Recall
m_ack	0.9492	0.9130	0.9360	0.9602
b_scan	0.9592	0.9502	0.9567	0.9633
b_combo	0.9248	0.9417	0.9170	0.8936
Benign	0.9831	0.9858	0.9845	0.9831
m_scan	0.9831	0.9721	0.9831	0.9943
m_plainudp	0.9336	0.9385	0.9481	0.9589
m_syn	0.9492	0.8974	0.9377	0.9817
m_udp	0.9392	0.9313	0.9468	0.9607
b_udo	0.9696	0.9611	0.9666	0.9722
b_junk	0.9564	0.9438	0.9620	0.9809
b_tcp	0.9340	0.9190	0.9344	0.9502
average	0.9529	0.9414	0.9521	0.9636

Table 4 above displays the results of evaluating the performance of ten various types of attacks in contradiction of a single type of benign data while optimizing neural network hyperparameters without taking feature selection into account. In Benign, the recall value was 0.9831, the precision value was 0.9859, the F1-score was 0.984, and the accuracy value was 9831.The recall value for the A nd m_scan was 0.9943, the precision value was 0.9722, the accuracy value was 0.9831, and the F1- score was 0.9831. Another m_plainudp achieved recall values of 0.9589, 0.9375, 0.9481, and 0.9336, as well as precision and accuracy values. Additionally, m_syn achieved the following values: 0.9817 for recall, 0.8974 for precision, 0.9377 for F1-score, and 0.9492 for accuracy. m_udp reached the recall value of 0.9607, the precision value of 0.9333, the F1- score value of 0.9468 and the accuracy value of 0.9392. A recall value of 0.9602, a precision value of 0.9130, a score value of 0.9360, and an accuracy value of 0.9492 were then obtained using m_ack. Additionally, b_scan's recall value was 0.9633, precision was 0.9502, F1-score was 0.9567, and accuracy was 0.9592. And subsequently, b_combo reached the recall value of 0.893, 0.9417, the F1- score of 0.9170 and the accuracy value of 0.9248. b_udo reached the recall value of 0.9722, the precision value of 0.9611, the F1- score value of 0.9666 and the accuracy value of 0.9696. b_junk gained a recall value of 0.9809 and got an F1- score value of 0.9438, went an F1- score value of 0.9620, and earned an accuracy value of 0.9564. b_tcp reached the recall value of 0.9502 and the precision value of 0.9190. An average scheme came with a recall value of 0.9636, went an F1- score value of 0.9521 and an accuracy of 0.9529 accordingly.

Table 5. The suggested model is evaluated using Feature Selection on ten attacks and one benign class.

Type of attack	Precision	F1-Score	Recall	Accuracy
Benign	0.9890	0.9931	0.9972	0.9954
m_plain udp	0.9783	0.9863	0.9945	0.9896
m_scan	0.9677	0.9796	0.9917	0.98478
m_udp	0.9574	0.9677	0.9783	0.9697

m_syn	0.9756	0.9839	0.9923	0.9839
b_scan	0.9534	0.9698	0.9868	0.9684
m_ack	0.9626	0.9756	0.9890	0.9894
b_udo	0.9600	0.9751	0.9906	0.9724
b_tcp	0.9651	0.9646	0.9641	0.9761
b_junk	0.9809	0.9823	0.9836	0.9773
b_combo	0.9847	0.9882	0.9917	0.9815
average	0.9704	0.9787	0.9873	0.9807

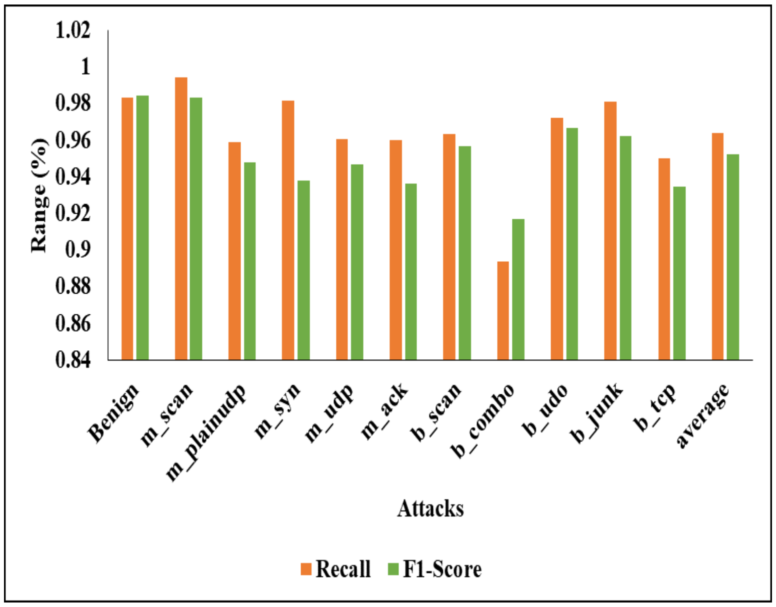

Fig. 5: Analysis of the Proposed model on multiclass attacks without FS.

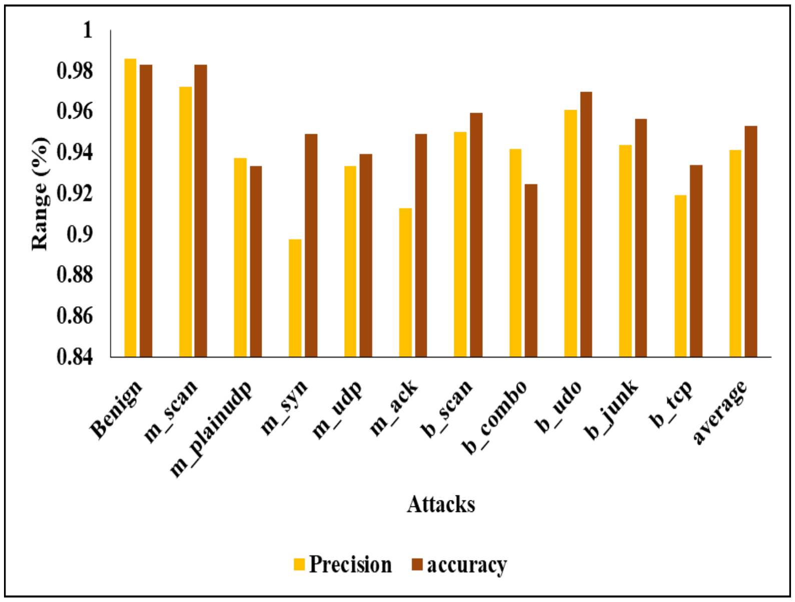

Fig.6: Graphical Comparison of various attacks.

The effectiveness of the projected perfect is evaluated against 10 distinct attack categories and a single benign class using Feature Selection in Table 5. In this analysis, the recall value for Benign was 0.9972, the precision value was 0.9890, the F1-score value was 0.9931, and the accuracy value was 0.9953. A different method of m_plainudp achieved recall values of 0.9945, precision values of 0.9783, and F1-score values of 0.9863. In addition, m_scan achieved recall values of 0.9917, 0.9677, an F1- score of 0.9796, and an accuracy value of 0.9847. m_udp attained recall values of 0.9783, 0.9574, an F1-score of 0.9677, and an accuracy value of 0.9696.In addition, m_syn achieved a recall value of 0.9923, a precision value of 0.9756, an F1-score value of 0.9839, and an accuracy value of 0.9839.Following this, _scan achieved a recall value of 0.9868, a precision value of 0.9534, an F1- score value of 0.9698, and an accuracy value of 0.9680. Then, m_ack 0.9890, the achieved F1- score of 0.9626, the achieved F1- score of 0.9756, and the achieved accuracy value of 0.9890. In addition, b_udo reached recall values of 0.9906 and precision values of 0.9600, as well as F1- score values of 0.9751 and accuracy values of 0.9724. A b_tcp attained recall values of 0.9641 and precision values of 0.9651, as well as F1- score values of 0.9646 and accuracy values of 0.9761. Then b_junk attained a recall value of 0.9836, a precision value of 0.9823, and F1-score values of 0.9809, 0.9823, as well as an accuracy value of 0.9773. The recall value for Aldo's b_combo was 0.9917, the precision value was 0.9847, the F1- score was 0.9882, and the accuracy value was 0.9815. The average recall value was then 0.9873, the precision value was 0.9704, the F1- score was 0.9787, and the accuracy value was 0.9807, respectively.

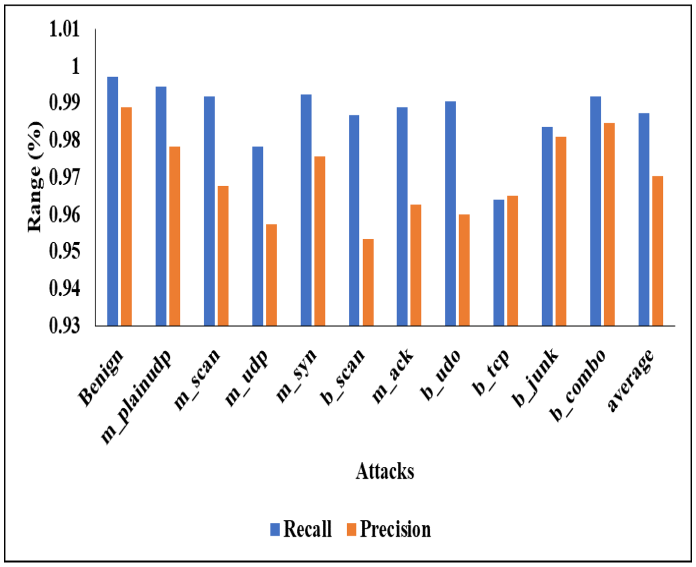

Fig. 7: Analysis of Classifier with FS.

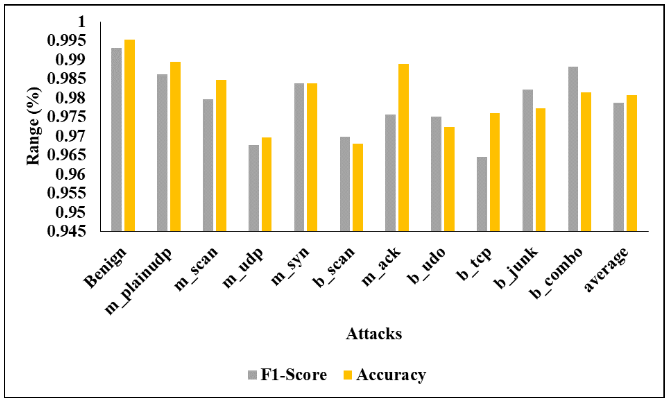

Fig. 8: Comparative analysis of multiclass attacks.

Table 6. Relative analysis of Proposed CNN with various Deep Learning Procedures.

Model	Recall	Precision	F-Measure	Accuracy
ResNet	85.00	75.00	80.95	80.00

VGGNet	80.00	70.00	76.19	75.00
AlexNet	90.00	80.00	85.71	85.00
DenseNet	90.00	85.00	87.80	87.50
DarkNet	92.41	90.00	95.24	95.00
DarkNet-BRSA	98.73	97.04	97.87	98.07

Table 6 above represents the Comparative investigation of Projected CNN with various Deep Learning Techniques. In this analysis, the ResNet model reached a recall value of 85.00, a precision value of 75.00 and an F-Measure of 80.95 and attained an accuracy of 80.00. Another model, as VGGNet model, reached a recall value of 80.00, a precision value of 70.00 and an F-Measure of 76.19. and attained an accuracy of 75.00. Also, the AlexNet model reached a recall value of 90.00, a precision value of 80.00 and an F-Measure of 85.71 and achieved an accuracy of 85.00. Also, the DenseNet model came with a recall value of 90.00 and the precision value of 85.00. and also the F-Measure as 87.80 and attained an accuracy of 87.50. Also, the DarkNet model reached a recall value of 92.41, a precision value of 90.00 and an F-Measure of 95.24, achieving an accuracy of 95.00. Also, the DarkNet-BRSA model called a recall value of 98.73, a precision value of 97.04 and an F-Measure of 97.87 and attained an accuracy of 98.07, respectively.

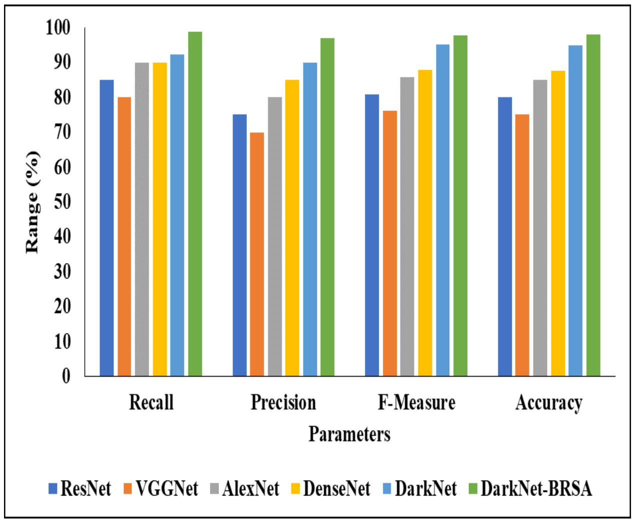

Fig.9: Analysis of Various Pre-trained CNN Models

5 Conclusions

This study presents a potential approach for monitoring IoT networks for botnet assaults. A unique feature selection technique (improved firefly algorithm) and an optimized pre-trained CNN network are given in the suggested model for identifying botnets. This study presents research that proposes a new form of the well-known FA algorithm—the GSFA metaheuristic—in an effort to fix its most glaring flaws. The presented GSFA strategy utilizes the SNS metaheuristic's suggested disputation operator. To optimize the tuning process of the suggested network and hence facilitate classification, a unique metaheuristic enhancement (BRSA) to the traditional RSA is presented. The results of the assessment indicated that the innovative selection strategy used with the new-fangled model achieved higher levels of accuracy and recall than any of the competing models. In addition, several classifiers are tested on the N-BaIoT dataset using the determined feature fraction. Future improvements to the detecting procedure will make use of additional, newly developed optimization techniques.

Conflicts of Interest Statement

The authors declare no conflict of interest.

Acknowledgment:

The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through large group Research Project under grant number RGP2/462/44

References

[1] Al-Hadhrami, Y. & Hussain, F.K. (2021). DDoS attacks in IoT networks: a comprehensive systematic literature review. World Wide Web., **(24)**, 971-1001 (2021)

[2] Snehi, M. & Bhandari, A. (2021). Vulnerability retrospection of security solutions for software-defined Cyber–Physical System against DDoS and IoT-DDoS attacks. Computer Science Review., **(40)**, 100371 (2021).

[3] Shah, Z., Ullah, I., Li, H., Levula, A. & Khurshid, K. (2022). Blockchain based solutions to mitigate distributed denial of service (ddos) attacks in the internet of things (iot): A survey. Sensors., **(22)**, 1094 (2021).

[4] Doshi, K., Yilmaz, Y. & Uludag, S. (2021). Timely detection and mitigation of stealthy DDoS attacks via IoT networks. IEEE Transactions on Dependable and Secure Computing., **(18)**, 2164-2176 (2021).

[5] Papalkar, R.R. & Alvi, A.S. (2022). Analysis of defense techniques for DDos attacks in IoT–A review. ECS Transactions, (107), 3061 (2022).

[6] Gaur, V. and Kumar, R., (2022). Analysis of machine learning classifiers for early detection of DDoS attacks on IoT devices. Arabian Journal for Science and Engineering., **(47)**, 1353-1374 (2022).

[7] Kumar, R., Kumar, P., Tripathi, R., Gupta, G.P., Garg, S. & Hassan, M. M. (2022). A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network. Journal of Parallel and Distributed Computing., **(164)**, 55-68 (2022).

[8] Bhayo, J., Jafaq, R., Ahmed, A., Hameed, S. & Shah, S.A. (2021). A time-efficient approach toward DDoS attack detection in IoT network using SDN. IEEE Internet of Things Journal., **(9)**, 3612-3630 (2021).

[9] Kumar, P., Kumar, R., Gupta, G.P. & Tripathi, R. (2021). A Distributed framework for detecting DDoS attacks in smart contract-based Blockchain-IoT Systems by leveraging Fog computing. Transactions on Emerging Telecommunications Technologies., **(32)**, e4112 (2021).

[10] Gopi, R., Sathiyamoorthi, V., Selvakumar, S., Manikandan, R., Chatterjee, P., Jhanjhi, N.Z. & Luhach, A.K (2021). Enhanced method of ANN based model for detection of DDoS attacks on multimedia internet of things. Multimedia Tools and Applications., 1-19 (2021).

[11] Mihoub, A., Fredj, O.B., Cheikhrouhou, O., Derhab, A. & Krichen, M. (2022). Denial of service attack detection and mitigation for the internet of things using looking-back-enabled machine learning techniques. Computers & Electrical Engineering, **(98)**, 107716 (2022).

[12] Sharma, D.K., Dhankhar, T., Agrawal, G., Singh, S.K., Gupta, D., Nebhen, J. & Razzak, I. (2021). Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks. Ad Hoc Networks., **(121)**, 102603 (2021).

[13] Almaraz-Rivera, J.G., Perez-Diaz, J.A. & Cantoral-Ceballos, J.A. (2022). Transport and application layer DDoS attacks detection to IoT devices by using machine learning and deep learning models. Sensors., (22), 3367 (2022).

[14] Khempetch, T. and Wuttidittachotti, P. (2021). DDoS attack detection using deep learning. IAES International Journal of Artificial Intelligence, **(10)**, 382 (2021).

[15] Lawal, M.A., Shaikh, R.A. & Hassan, S.R. (2021). A DDoS attack mitigation framework for IoT networks using fog computing. Procedia Computer Science., **(182)**, 13-20 (2021).

[16] Zeeshan, M., Riaz, Q., Bilal, M.A., Shahzad, M.K., Jabeen, H., Haider, S.A. & Rahim, A. (2021). Protocol-based deep intrusion detection for dos and ddos attacks using unsw-nb15 and bot-iot datasets. IEEE Access., **(10)**, 2269-2283 (2021).

[17] Ilyas, B., Kumar, A., Setitra, M.A., Bensalem, Z.A. & Lei, H. (2023). Prevention of DDoS attacks using an optimized deep learning approach in blockchain technology. Transactions on Emerging Telecommunications Technologies, p.e4729 (2023).

[18] Al-Juboori, S.A.M., Hazzaa, F., Jabbar, Z.S., Salih, S. & Gheni, H.M. (2023). Man-in-the-middle and denial of service attacks detection using machine learning algorithms. Bulletin of Electrical Engineering and Informatics., **(12)**, 418-426 (2023).

[19] Gebrye, H., Wang, Y. &Li, F. (2023). Traffic data extraction and labeling for machine learning based attack detection in IoT networks. International Journal of Machine Learning and Cybernetics., 1-16 (2023).

[20] Rajasekaran, P. & Magudeeswaran, V. (2023). Malicious attacks detection using GRU-BWFA classifier in pervasive computing. Biomedical Signal Processing and Control., **(79)**, 104219 (2023).

[21] Azizpour, S. & Majma, M. (2023). Nada: new architecture for detecting dos and ddos attacks in fog computing. Journal of Computer Virology and Hacking Techniques., **(19)**, 51-64 (2023).

[22] Kalutharage, C.S., Liu, X., Chrysoulas, C., Pitropakis, N. and Papadopoulos, P. (2023). Explainable AI-Based DDOS Attack Identification Method for IoT Networks. Computers., (12), 32 (2023).

[23] Pillai, S. & Sharma, A. (2023). Hybrid Unsupervised Web-Attack Detection and Classification–A Deep Learning Approach. Computer Standards & Interfaces., 103738 (2023).

[24] Alsolami, F., Alqurashi, F. A., Hasan, M. K., Saeed, R. A. & Abdel-Khalek, S. (2021). Machine learning techniques in internet of UAVs for smart cities applications, IEEE Access., **(9)**, 48010-48022, (2021).

[25] Alqurashi, F. A., Alsolami, F. & Abdel-Khalek, S., Sayed Ali E., Saeed, R. A. (2022). A novel enhanced quantum PSO for optimal network configuration in heterogeneous industrial IoT, Journal of Intelligent & Fuzzy Systems., **(42)**, 3203-3226 (2022).

[26] Abukhodair, F., Alsaggaf, W., Jamal, A. T. & Abdel-Khalek, S., Mansour, R. F. (2021), An intelligent metaheuristic binary pigeon optimization-based feature selection and big data classification in a MapReduce environment, Mathematics ., **(9)**, 2627 (2021).

[27] Yang, X.S. (2009) Firefly Algorithms for Multimodal Optimization. In Stochastic Algorithms: Foundations and Applications; Watanabe, O., Zeugmann, T., Eds.; Springer: Berlin/Heidelberg, Germany, 169–178 (2009).

[28] Bezdan, T., Cvetnic, D.; Gajic, L., Zivkovic, M.; Strumberger, I.; Bacanin, N. (2021) Feature Selection by Firefly Algorithm with Improved Initialization Strate gy. In Proceedings of the 7th Conference on the Engineering of Computer Based Systems (ECBS 2021), Novi Sad, Serbia, 26–27 May 2021; Association for Computing Machinery: New York, NY, USA .,(2021).

[29] Bacanin, N., Bezdan, T., Venkatachalam, K. & Al-Turjman, F. (2021). Optimized convolutional neural network by firefly algorithm for magnetic resonance image classification of glioma brain tumor grade. J. Real-Time Image Process., **(18)**, 1085–1098 (2021).

[30] Yang, X.S. & Xingshi, H. (2013). Firefly Algorithm: Recent Advances and Applications. Int. J. Swarm Intell., **(1)**, 36–50 (2013).

[31] Abualigah, L. Abd Elaziz, M. Sumari, P. Geem, Z.W. Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. Expert Syst. Appl., **(191)**, 116158 (2022).

[32] Meidan, Y. et al., N-baiot—network-based detection of iot botnet attacks using deep autoencoders. (2018) IEEE Pervasive Comput., **(17)**, 12–22 (2018).

[33] Shahin, M. (2022). Evaluating the Fidelity and Efficiency of Network Intrusion Detection Systems Via Deep Learning, Machine Learning, and Deep Hybrid Learning in Industrial IoT Devices," The University of Texas at San Antonio., (2022).