1-1-2023

Reliable Machine Learning Model for IIoT Botnet Detection

Fatma Taher
Zayed University

Mahmoud Abdel-salam
Faculty of Computer and Information

Mohamed Elhoseny
Faculty of Computer and Information

Ibrahim M. El-hasnony
Faculty of Computer and Information

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Reliable Machine Learning Model for IIoT Botnet Detection

Fatma Taher[1], Mahmoud Abdel-salam[2], Mohamed Elhoseny[2,3] (Senior member, IEEE), Ibrahim M. El-hasnony[2]

[1]College of Technological Innovation, Zayed University, Dubai, UAE
[2]Faculty of Computer and Information Science, Mansoura University, Egypt; mahmoud20@mans.edu.eg, ibrahimhesin2005@mans.edu.eg, melhoseny@ieee.org
[3]College of Computing and Informatics, University of Sharjah, United Arab Emirates.

Corresponding author: Mahmoud Abdel-salam (e-mail: mahmoud20@mans.edu.eg).

This paragraph of the first footnote will contain support information, including sponsor and financial support acknowledgment. For example, "This work was supported in part by the U.S. Department of Commerce under Grant BS123456."

Abstract Due to the growing number of Internet of Things (IoT) devices, network attacks like denial of service (DoS) and floods are rising for security and reliability issues. As a result of these attacks, IoT devices suffer from denial of service and network disruption. Researchers have implemented different techniques to identify attacks aimed at vulnerable Internet of Things (IoT) devices. In this study, we propose a novel features selection algorithm FGOA-kNN based on a hybrid filter and wrapper selection approaches to select the most relevant features. The novel approach integrated with clustering rank the features and then applies the Grasshopper algorithm (GOA) to minimize the top-ranked features. Moreover, a proposed algorithm, IHHO, selects and adapts the neural network's hyper parameters to detect botnets efficiently. The proposed Harris Hawks algorithm is enhanced with three improvements to improve the global search process for optimal solutions. To tackle the problem of population diversity, a chaotic map function is utilized for initialization. The escape energy of hawks is updated with a new nonlinear formula to avoid the local minima and better balance between exploration and exploitation. Furthermore, the exploitation phase of HHO is enhanced using a new elite operator ROBL. The proposed model combines unsupervised, clustering, and supervised approaches to detect intrusion behaviors. The N-BaIoT dataset is utilized to validate the proposed model. Many recent techniques were used to assess and compare the proposed model's performance. The result demonstrates that the proposed model is better than other variations at detecting multiclass botnet attacks.

INDEX TERMS BOTNET, FEATURES SELECTION, GRASSHOPPER ALGORITHM, HARRIS HAWKS ALGORITHM, SECURITY

I. INTRODUCTION

Due to its replacement of earlier networks, the Internet of Things (IoT) has fundamentally altered the world. Undoubtedly, the Industrial Internet of Things (IIoT) is expanding at an astounding rate, creating a massive digital environment and eventually becoming an integral part of our daily lives [1]. When designing reliability and security, consider risks. Non-malicious primary reliability risks include a faulty software update or device failure. Threats come from adversaries who exploit system vulnerabilities. When designing for reliability, you expect some problems. When planning for security, assume an adversary could cause issues at any point. As a result, different systems respond to failures differently. Without an adversary, systems fail safely. System privacy, integrity, and availability are essential to security and reliability, but they look at these things differently. It has been a critical part of fifth-generation (5G) and beyond networks. E-health, smart agriculture, smart cities, e-wearables, etc., benefit from the IIoT ecosystems [2]. Intelligent, linked, and location-aware smart devices create a bright user environment and produce massive amounts of IoT data for applications in decision-making, artificial intelligence, and behavioral analytics [3][4].

Nevertheless, A lack of network and data privacy will discourage stakeholders of IoT technology [5]. However, as the world becomes more interconnected through the Internet and the IoT ecosystem, the necessity to adequately provide more protection for the components of IoT is becoming more critical, which emerges in terms of scale, complexity, and connectivity. Since many IoT systems are created without focusing on security concerns, the IoT has become a

significant security risk. Security threats have increased due to the expanding attack surface and technologies being deployed, according to the latest reports of cybersecurity [6]. As a result, engineers in this field have paid attention to designing IoT systems with safe property [7].

When discussing IoT networks and devices, it is vital to remember that hackers could compromise IoT devices and add them to IoT botnets under their control. Reports stated that well-known IoT Botnet attacks like Mirai [8] and BALLSHITE have increased, which remain serious Distributed denial-of-service (DDoS) threats around the world [9]. The IoT's fundamental characteristics, scalability, heterogeneity, and limited resources make it difficult to mitigate such threats. Thus, it has become an essential concern in the IoT sector among academics to create systems that can detect abnormal behaviors in IoT environments.

Simply put, a botnet is a computer network that shares the same autonomous software. In this network, an infected computer, or "bot," is used to perform malicious tasks by the attacker. A botnet is a nefarious group's botnet that operates a bot on many internet connected devices [6]. Botnets concern network security because they make it feasible for numerous Internet crimes, including identity theft, DDoS threats, click fraud, and email spam. Botnet attacks aim for many threats; the victim network's bandwidth and resources are depleted, and services are disrupted. Currently, these attacks in the IoT are usually monitored using machine learning techniques [10].

DDoS is simple, but its attacks are very successful due to the sheer number of bots in a botnet and their total combined bandwidth. DDoS attacks can be launched against any Internet-connected machine, exhausting its available resources and making it unavailable to its intended users. Today, botnets are used in nearly all DDoS assaults.

An attacker can drastically slow down the system by launching numerous attacks or getting unauthorized access to the user's data. A computer network is vulnerable to various assaults, such as probing (Probe), user-to-root (U2R), brute force, etc. Any protocol, such as the User Datagram Protocol (UDP), the Transmission Control Protocol (TCP), and so on, can carry out these assaults. Intrusion detection systems should be proposed and used to stop these assaults by scanning the network and locating them [11].

Intrusion detection systems have been developed using a variety of machine learning approaches to preserve reliability and security, but these approaches struggle with high traffic volumes. Also, the lack of an optimization perspective in deep learning approaches results in limited model generalization. For static botnet data sets, researchers may now create high-detection detectors, including binary classification methods [12]. However, due to the dynamic nature of botnet attack detection strategies, a high detection rate for merely predefined data sets cannot guarantee excellent accuracy when handling traffic data. Therefore, classification methods based on neuro-evolution, which can aid in determining the optimal number of neurons and layers for detection, are required [13].

Features selection FS is a method for identifying unnecessary and redundant features, which can degrade the classifier's efficiency. Improved classification accuracy, dimensionality reduction, and even less time spent training are all possible results of FS. Exhaustive searches and other classic search methods have running times growing exponentially with complexity. But, if there are many features, this will be practically impossible. There has been a lot of research into new search strategies lately, but the problem of local minima has restricted most of them. The optimization of the FS has recently been efficient by applying metaheuristic algorithms, with encouraging results [14].

While a neural network is being trained and its weights are being learned, the hyperparameters should be tuned. For this reason, many optimizers have been presented in recent publications [15]. But, they ignore the data set dimension and problem nature; thus, numerous algorithms [16] have been used to solve this issue and maximize the hyperparameters.

Clustering can mitigate these unexpected threats in conjunction with supervised methods [17]. To this end, we present a novel feature selection method that can significantly minimize the feature set. Clustering has been included as a preliminary processing step to facilitate feature selection. A new approach for reducing the dimensionality called Fisher Grasshopper Optimization Algorithm (FGOA-kNN) is proposed. The FGOA-kNN is a two-step hybrid approach that incorporates two feature selection techniques. We first use a Fisher-score-based filter to narrow the search space to obtain the most relevant features from each cluster. Next, we apply a Grasshopper Optimization Algorithm (GOA) based wrapper method to pick the most miniature set of features that still provides the best prediction results. A new optimization approach is proposed to tune the hyperparameters of neural networks for effective botnet attack detection, called the Improved Harris Hawks Optimization Algorithm for Neural Networks (IHHO-NN). There are three main enhancements to IHHO-NN. The first is a more efficient population initialization strategy that uses Chaotic maps to provide population diversity. Second, the proposed approach uses a novel energy update formula to improve the HHO exploration phase. Lastly, Finally, to avoid falling into local minima, a new operator ROBL is applied to enhance the exploitation stage of HHO.

This paper's contribution is summed up as follows:

- An efficient model is constructed and implemented to detect the IoT botnet attacks by presenting a new hybrid features selection method and an optimized neural network classifier with a new optimization algorithm;

- An approach of two-steps is proposed for picking the most relevant feature set by applying unsupervised learning with a new hybrid filter-wrapper-based selection approach;
- The proposed features selection approach efficiently minimizes unnecessary and redundant features by maximizing the between-class dissimilarity and minimizing the within-class distinction.
- A novel and effectively improved metaheuristic algorithm is proposed for adapting the hyperparameters of a neural network with model evaluation.

The paper is defined as follows. Discussions about the related state-of-art research is reviewed in section 2: section 3 reviews and presents an overview of the used material and methods. The details of the proposed model with its components are given in section 4. Section 5 offers the experimental setup with an analysis of the results and a comparison with others. Section 6 finishes the presented work and introduces a vision for future work.

II. A SYSTEMATIC REVIEW OF THE RECENT LITERATURE

Studies have shown that novel data mining technologies have piqued the interest of academics in the field. As a result, they have effectively used them for IoT security and reliability breaches and botnet attacks.

Tripathi et al. [18] applied the grasshopper algorithm (GOA) toward optimizing their models. Detection system was the suggested approach for identifying harmful from benign traffic. Four different models including multilayer perceptron, a decision tree, a support vector machine and a naive Bayes determine the method of assault. Tripathi et al. [18] used the grasshopper method (GOA) to optimize their models. They put their technique through its paces using the KDD Cup 99 and CIC-IDS 2017 datasets.

To (very) efficiently detect the different kinds of attacks, Amiri et al. [19] suggested a new feature selection approach based on improving the mutual information implemented using the SVM. They proved that even data with a large dimensionality might be improved using the features selection method.

To illustrate their experiences with two chosen ML algorithms, Sung et al. [20] gradually eliminated features (SVM and neural network). Finally, they tested their approach on the DARPA-ID-1998 intrusion detection dataset. Experimental outcomes with this dataset's five-class classification (target variables) revealed that using only a few most relevant attributes, 34 attributes, instead of the complete set of features led to a statistically negligible gain in performance for intrusion detection.

To choose first-order incremental features, Peng et al. [21] introduced the minimal-redundancy-maximal-relevance measure. This specification employs a practical way of picking features at a low price. They've compared their proposed method to maximal relevance standards using three distinct classifiers. The results of their studies demonstrated that the classification performance might be drastically enhanced. Both discrete and continuous data sets are suitable for the method.

An advanced multilayer classification intrusion detection strategy was developed in [22]. Detecting an intrusion and determining its nature took place in two phases. Moreover, an oversampling method was used to improve the categorization outcomes further. The trials revealed that 150 neurons for both LSTM and the Single-Layer Feed-Forward Network (SLFN) were the most influential parameters for the proposed technique. The results proved that the suggested method achieved a higher G-mean value than the other well-known methods, 78% versus 75% for KNN and less than 50% for the other methods.

As part of an intrusion detection system, [23] proposes a GA-based features selection method. Using the KDD Cup 99 datasets, The fuzzy support vector machine was used in conjunction with the genetic algorithm, and the results were impressive, and the results were impressive. It was suggested to use the Tabu Search method for new feature selection by Nazir and Khan [24] for use in constructing a reliable intrusion detection system by training the Random Forest (RF) classifier. The UNSW-NB15 dataset was used to test the proposed technology, TS-RF. The evaluation's findings showed that Tabu Search performed better than several feature selection techniques and that the suggested feature selection approach increased classification performance.

A new framework for developing NIDS was given in a recent paper [25] that uses DL and ML techniques. They hoped their in-depth understanding of AI-based NIDS would prove helpful to researchers, and they looked for potential roadblocks to the suggested approach. Mayuranathan et al. [26] proposed a more effective detection method for intrusions, using the Random Harmony Search algorithm for feature selection. To identify DDoS attacks, a classifier based on Restricted Boltzmann Machines was used (DDoS). When tested on the KDD'99 datasets, the proposed system performed admirably.

A features selection and intrusion detection system was proposed by Javadpour et al. [27] using the KDD99 dataset to automatically detect suspicious patterns based on linear correlation and mutual information with random forests, CART, and decision trees algorithms. They claimed that this system enhanced the accuracy of the intrusion detection rate.

The authors of [28] proposed a cutting-edge technique for intrusion detection. In order to complete the assignment, they integrated the clustering, oversampling, and classification procedures. The proposed approach makes use of an SLFN classifier. Results from the SVM, SLFN, and Synthetic Minority Oversampling combination outperformed other classification techniques.

The functional trees provide the highest performance compared to other classifiers, according to Firdaus et al. [29], who employed the genetic search (GS) method and five

machine learning models, including the J48, Naive Bayes, Random Forest, Multilayer Perceptron, and Functional Trees.

In [30], the authors describe another method for using DL and autoencoders to look for deviations in IoT networks. To identify botnet assaults, the authors recommend observing behavioral "snapshots" of the regular traffic. Their practical usefulness is measured by replicating the actions of well-known botnets like Mirai and Bashlite across a testbed network. Each IoT device has its autoencoder that is used to learn the properties of regular traffic and alert users when the autoencoder cannot reconstruct the snapshots of harmless communication. It's possible, though, that this technique won't work for more extensive networks. Network security can be challenging to develop and manage when an autoencoder is employed because each IoT device needs to update and maintain a distinct model.

The creators of GTO optimization algorithm [31] claim their algorithm is head and shoulders above other popular metaheuristics. Local optimum problems and premature convergence plague GTO, as they do different meta-heuristic algorithms. As a result, many authors have attempted to improve GTO's effectiveness by combining it with other strategies.

In [32], semi-supervised support vector machines (SVMs) are presented as a way to identify variations from different quantities of labeled data utilizing a quasi-Newton and k-means techniques, both of which produce good classification performance and reduce labeling effort in the presence of vast volumes of unlabeled data.

Kasun et al. [33] looked at the Extreme Learning Auto-Encoder on the USPS dataset for handwritten digit recognition and the CIFAR-10 and NORB datasets for their dimensionality reduction study.

For instance, in [34], a DNN is utilized in conjunction with XGBoost to pick features for intrusion detection. The NSL-KDD data set was utilized for testing, and the model included in the proposal deals with classification, feature selection, and normalization.

III. Preliminaries

A. Deep Artificial Neural Networks (DNN)

A collection of connected processing units that maps inputs to desired outputs and performs processing defines the mechanism of an artificial neuron. The mathematical representation of the artificial neuron's output is shown in Equation (1). The DNN model has three levels: input, output, and hidden layers.

$$\mathrm{y_i} = f_i\left(\sum_{i=1}^{n} w_{i,j}\cdot \mathrm{x_i} + \mathrm{B_i}\right) \tag{1}$$

An environment can send a signal to any neuron. Each x_i is the input signal associated with a weight $w_{i,j}$. The environment is responsible for calculating the output y_i. In

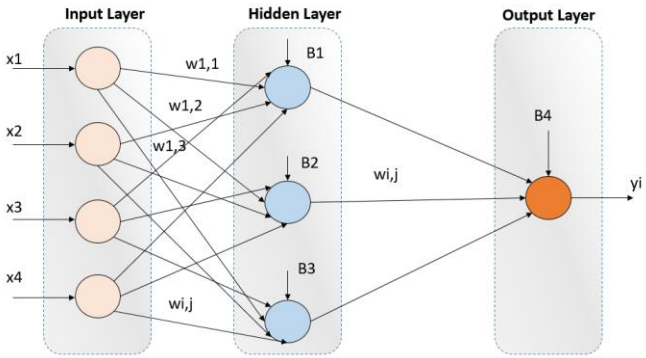

FIGURE 1 Simple DNN architecture

Equation (1), y_i represents the node's output, x_i represents the node's ith input, $w_{i,j}$ represents the weight between the input and node, f_i represents the node's activation function and B_i represents the node's bias. The activation function of a node is typically a nonlinear function like a sigmoid, Gaussian, etc. A single neuron in the output layer represents each class. The simple architecture for DNN components is shown in Figure 1.

A meta-heuristic algorithm can be used to train neural networks in three different ways. First, techniques are utilized to minimize Mean Squared Error to obtain an appropriate combination of bias and weight.

Second, algorithms are utilized to choose the ideal structure for a problem. Lastly, the parameters of the learning algorithm are adjusted using a meta-heuristic approach.

B. Harris Hawks Optimization algorithm

The Harris Hawks Optimization (HHO) algorithm was developed by Heidari et al. [35], who aimed to replicate the behavior of Harris Hawks using an algorithm with a similar theoretical foundation. The Harris Hawks have unusual social behavior to follow and attack their prey. The exploration and exploitative phases of the algorithm entail seeking out the prey, making a swift move, and employing various attack techniques. Using two different exploring techniques, Harris Hawks are randomly assigned to areas where they wait for the prey. They are regarded as potential solutions, and the ideal one is the one that satisfies the goal or comes close to it. The first strategy includes Harris Hawks perching on a location and analyzing where other family members and the prey are. The second strategy is the Hawks on tall trees waiting for prey. Here is how to show that each of the two plans has an equal chance of getting q:

$$x(t+1) = \begin{cases} x_r(t) - r_1\left|x_r(t) - 2r_2 x(t)\right| & q \geq 0.5 \\ x_{\text{rabbit}}(t) - x_{\text{mean}}(t) - r_3\left(Lb + r_4\left(Ub - Lb\right)\right) & q < 0.5 \end{cases} \tag{2}$$

In the upcoming and recent iteration, the Hawks position vectors are individually x(t+1) and x(t).$x_r(t)$ is a hawk selected at random from the population. $x_{\text{rabbit}}(t)$ is the rabbit location. q, r_1, r_2, r_3 and r_4 are numbers that are created randomly. Lower and Upper bounds are denoted by Lb and

Ub, which are used to generate random places inside the home of Hawks. The initial location of Hawks in the population is referred by $x_{\text{mean}}(t)$, which is determined as:

$$x_{\text{mean}}(t) = \frac{1}{h}\sum_{i=1}^{h} x_i(t) \tag{3}$$

At iteration t and $i = 1, \ldots, h$, each hawk has an ith position vector of $x_i(t)$. The entire number of Hawks in the population is h. The algorithm can transition from the exploring to the exploiting phase depending on the rabbit's escape energy E as follow:

$$E = 2E_0\left(1 - \frac{t}{\text{Max\_iter}}\right) \tag{4}$$

A randomly generated value between -1 and 1 is assigned for E_0 at the initial energy. Max_iter refers to the maximum iterations to be reached. Hawks search for additional areas to discover the rabbit's position when $|E| \geq 1$; else, the exploitation phase happens. With a similar chance p, ($p \geq 0.5$) is the success; otherwise, failure ($p < 0.5$) of the escape of rabbit created in the algorithm. As well, the Hawks will conduct a soft hard when $|E| < 0.5$ or soft besiege $|E| \geq 0.5$. The following equation expresses the soft besiege:

$$x(t+1) = \Delta x(t) - E|\ J \times x_{\text{rabbit}}(t) - x(t) \tag{5}$$

$$\Delta x(t) = x_{\text{rabbit}}(t) - x(t) \tag{6}$$

$$J = 2(1 - \text{rand}) \tag{7}$$

The difference between the hawk and rabbit locations will be presented as $\Delta x(t)$. The rabbit's jump strength, J, is chosen randomly using rand. On the opposite side, the following equation expresses the hard besiege:

$$x(t+1) = x(t) - E\left|\Delta x(t)\right| \tag{8}$$

when ($p < 0.5$) and ($|E| \geq 0.5$), a soft besiege consisting of progressive quick dives is conducted. The Hawks are capable of choosing the optimal dive. To mimic the leapfrog behavior of the prey, levy flight is used. To determine whether the dive is good, the Hawks' next step is estimated as follow:

$$k = x_{\text{rabbit}}(t) - E\left|J \times x_{\text{rabbit}}(t) - x(t)\right| \tag{9}$$

The Hawks will dive utilizing the lévy flying L pattern as follows if the earlier dive was not suitable:

$$z = k + s \times L(d) \tag{10}$$

Where s is a random vector whose size is d where d is the problem dimension. Lévy can be computed by [36]:

$$\text{Levy} = \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}} \tag{11}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2\left(\frac{\beta-1}{2}\right)}\right)^{\frac{1}{\beta}} \tag{12}$$

Random values in a range between 0 and 1 are generated for v and u, and the value 1.5 is set as a value for parameter β. To update the soft besiege progressive rapid dives are using:

$$x(t+1) = \begin{cases} k & \text{if } f(k) < f(x(t)) \\ z & \text{if } f(z) < f(x(t)) \end{cases} \tag{13}$$

Where Equations 9 and 10 are used to compute values of z and k. When ($|E| \geq 0.5$) and ($p < 0.5$), a hard besiege with progressive rapid dives is occurred using Equation (13), where the value of k is determined as follow:

$$k = x_{rabbit}(t) - E\left|J \times x_{rabbit}(t) - x_{mean}(t)\right| \tag{14}$$

C. The Grasshopper Optimization Algorithm

In 2016, Saremi et al. [37] proposed the GOA algorithm. The swarming strategy of real-world grasshoppers inspires this method. Three factors affect a grasshopper's flight direction while in a swarm: social interaction (S_i), gravity (G_i), and wind advection (A_i). The GOA algorithm defines the primary search mechanism in the following ways:

$$S_i = \sum_{j=1, j\neq i}^{N} s\left(d_{\text{ij}}\right)\hat{d}_{\text{ij}} \tag{15}$$

where d_{ij} is the distance between ith and jth grasshopper, and it is determined as $\text{d}_{\text{ij}} = |x_j - x_i|$, s is used to gauge the power of social forces, and the unit vector $\hat{d}_{\text{ij}} = \frac{x_j - x_i}{d_{\text{ij}}}$ represents the ith grasshopper and the jth grasshopper. This equation shows that the function s is the primary driver of social interaction. The following equation is a description of the function that determined how a grasshopper would migrate within the swarm:

$$s(r) = fe^{\frac{-r}{T}} - e^{-r} \tag{16}$$

where r denotes the attractiveness's length scale and f denotes the attractiveness's intensity.

The grasshoppers experience attractive and repulsive forces due to this function. Grasshoppers avoid colliding by repelling one another when their distance from one another is in the range [0,2.079]. To keep the swarm together, the attractive force rises when the distance is between [2.079,4]. The Comfort zone refers to the region where the distance between two points is precisely 2.079 when no force is exerted. At a distance of 2.079 astronomical units, both attractive and repulsive forces cancel out. At a distance of

2.079 units, the attractive force is at its maximum and then begins to fall to below four progressively. The swarming behavior is highly sensitive to the parameters in the Equation for the function s(r). If you want an accurate simulation of grasshopper social interactions, use the swarm model. An optimization algorithm, however, requires some tweaking. While grasshoppers interact, the following model is presented by the authors in [37] for search process. This mathematical model can be expressed as an equation, which is as follows:

$$X_i^d = c\left(\sum_{j=1, j\neq i}^{N} c\frac{ub_d - lb_d}{s} s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right) + \hat{T}_d \qquad (17)$$

Where $\hat{T}_d$ is the value of the dth dimension's best solution, ub_d is the dth dimension's upper bound, lb_d is the dth dimension's lower bound, and c is a coefficient to reduce the attraction area and repulsion and comfort area. This equation shows how the swarm adjusts its position about a target $(\hat{T}_d)$. The swarm can be guided in the direction of the target by adjusting the parameter c.

The GOA algorithm operates under the hypothesis that the final goal is the optimal solution found so far. If a better solution is found while grasshoppers engage and pursue the objective, the best solution will be modified to reflect this. Updates to the GOA algorithm's primary control parameter, c, are calculated with the following formula:

$$c = c_{\max} - l\frac{c_{\max} - c_{\min}}{L} \qquad (18)$$

where L shows the total number of iterations, the current iteration is defined by l, $c_{\max} = 1$, and $c_{\min} = 0.00001$.

D. Fisher Score

To pick relevant features, the Fisher score uses a filtering mechanism. As a powerful supervised approach, it has seen extensive use for fixing many real-world issues involving feature selection [38]. Using a filter-based method, we first assess the importance of each feature by assigning it a score; next, we use that score to determine the feature's extent to the model [39]. A subset of features is formed by selecting the corresponding number of features. Each feature's score and the entire features in the subset are then ranked in descending order to determine the final subset. Evaluative criteria based on the Fisher score [40] might be stated as follows:

$$S_F(f_i) = \frac{\sum_{j=1}^{C} n_j\left(\mu_{i,j} - \mu_i\right)^2}{\sum_{j=1}^{C} n_j \sigma_{i,j}^2} \qquad (19)$$

where the features' mean is defined by μ_i ; n_j expresses the total sample count in the j th class; $\sigma_{i,j}$ and $\mu_{i,j}$ are the variance and mean of the feature f_i resides in the j class, respectively.

IV. Proposed framework

In order to improve the performance of neural network models in an IoT context and provide a more dependable and secure setting, the proposed model in this research uses a novel features selection approach and a new detection approach. The elements of the model are broken down here for your consideration. The main aspects of the proposed model and its components are shown in Figure 2. After data preprocessing process, the dataset has been split into testing and training sets, as shown in the figure. The proposed model's next five steps are applied:

- Data preprocessing,

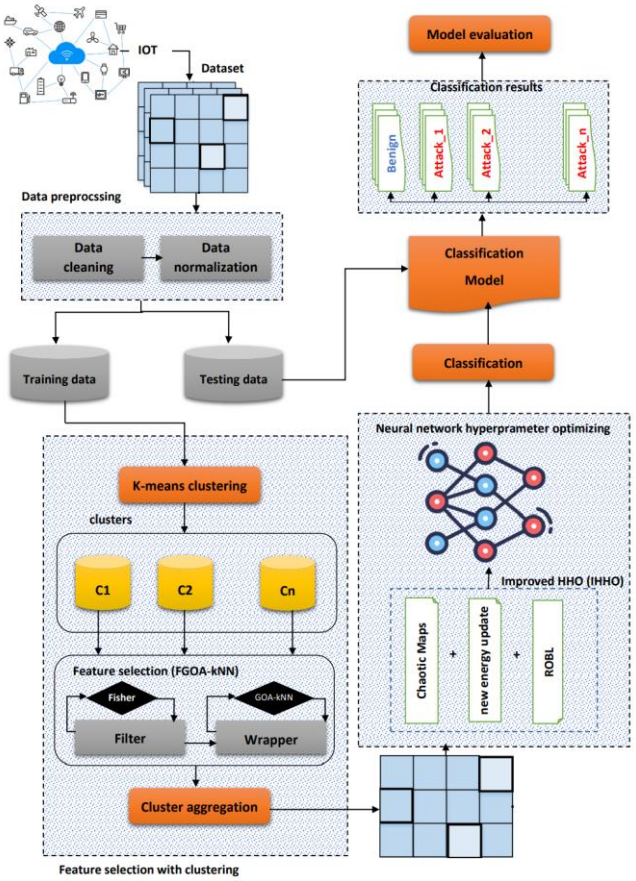

FIGURE 2 **The proposed model.**

Algorithm 1: FGOA-kNN-IHHO-NN

Input: dataset, k

Output: ten attacks classes, benign class, PREC, F1-score, ACC, REC

```
processed-dataset=DataProc(dataset)
Test, train = divide(processed-dataset)
clusters = k-means(k, training set)
new-clusters = feature_select_by_FGOA-kNN(clusters)
new-dataset-reduced = aggregate(new-clusters)
optimized-hyperparams = IHHO(neural-network-params)
Model-new = IHHO-NN(new-dataset-reduced)
classes-predicted = predict(IHHO-NN model, test)
PREC, F1-score, ACC, REC = evaluate(classes-predicted)
```

- Dataset clustering,
- Features selection for each cluster,
- Classification,
- Evaluation.

In the following parts, we'll discuss each of these steps in further depth. Algorithm 1 is a schematic of the proposed model that will be used. The goal of data preprocessing step is to normalize and clean the dataset for further processing. The purpose of the clustering step is to reveal previously hidden patterns in the dataset. In the features selection step, we try to narrow down the features that are in each cluster. The fisher score is employed during the first step of the features selection stage because it may be used to reduce the gap between features within classes and increase the distance between features within classes. A new wrapper-based approach is used with the fisher score to ignore the irrelevant feature and select the most important ones. Classification using an optimized neural network with a new, improved metaheuristic algorithm is applied to detect botnet attacks. Evaluation metrics are employed to rate the effectiveness of the suggested model. The algorithm accepts the dataset and number of clusters as input. The dataset is split into testing and training sets. The clustering process with feature selection is presented in lines 3-5. The training data is clustered using K-means. FGOA-kNN features selection
approach is applied at each cluster to select the essential features in line 4. The new reduced clusters are aggregated to form a new reduced dataset. The neural network is optimized using an improved HHO in line 6. The optimized neural network classifies the testing data set to predict the class label in lines 7 and 8. The model's performance is evaluated using different evaluation metrics in line 9.

A. Data preprocessing

Data preprocessing is an essential step that raises the data's quality to support fruitful data extraction. It mostly relates to organising and cleaning raw data in ML so that DL and ML models can be trained on it. Data cleansing and normalization were employed in this study.

1) Data normalization

While utilizing a dataset that has features at different scales, data normalization is a crucial technique in data mining. It aims to harmonize their scales such that all features contribute equally to the model. Numerical features are present in the N-BaIoT dataset. As a result, all of them can be normalized using the Min-Max normalization method, as stated in Equation (20). The N-BaIoT dataset's features have all been scaled to fall between 0 and 1.

$$Anew_x = \frac{A_x - \min A_x}{\max A_x - \min A_x} \tag{20}$$

Where A_x is the xth sample before applying normalization in a dataset, $Anew_x$ is is the xth sample after applying normalization in a dataset, minimum and maximum values refer to the new scale range of 0 and 1.

2) Data cleaning

The act of editing or deleting incomplete, duplicate, or improperly formatted data is known as data cleansing. First, all of the duplicate values are eliminated from the dataset. The model's accuracy is not improved by features that have these redundant values; rather, they make the model more complex. The second stage involves handling null and missing values in the dataset. With the projected median value, all missing or null values have been filled. The choice of median was made because, in contrast to mean imputation, it is less prone to errors due to outliers [41].

B. Dataset clustering

Data clustering is a crucial preprocessing step that helps improve classification accuracy. Algorithms that cluster data split objects into categories with increasing similarity within each category and decreasing the similarity between categories [42].

$$d_{xy}^2 = \left(x_1 - y_1\right)^2 + \left(x_2 - y_2\right)^2 \tag{21}$$

$$d_{x,y} = \sqrt{\left(x_1 - y_1\right)^2 + \left(x_2 - y_2\right)^2} \tag{22}$$

The goal is to afford a way to preserve a valuable dataset but with fewer instances. This paper proposes a new features selection strategy for dealing with the dimensionality problem by combining clustering with a novel hybrid filter-wrapper-based features selection approach. Better results can be achieved when evaluating features in combination with clustering because this approach overcomes the large number of features problem.

Various methods can be employed in the clustering process; the K-means approach is generally adopted due to its popularity and ease of use. The data structure is uncovered; furthermore, the K-means clustering method is used to build the cluster [43]. To begin clustering, choose k features at random from the original dataset. Clusters are assigned their most similar objects based on the similarity between their features and the cluster mean. Each cluster's mean is re-estimated. We iterated until no cluster experienced any significant feature redistribution. The number of clusters selected by the user is required by the k-means clustering method. We've used three clusters to organize these data. As indicated in Equation (21), the similarity between two items can be determined using the Euclidean distance function. Squaring the distance between two vectors, y = [y1, y2] and x = [x1, x2], is represented by the equation (22). The coordinate system's d value corresponds to the separation between the x and y vectors [44].

When the clusters have been established, the next step is eliminating duplicate features from each cluster. The details of the features selection step are discussed in the next section.

C. Features selection for each cluster

This study employs the Fisher-score-based filter approach, the Grasshopper Optimization Algorithm, and the k-Nearest Neighbor GOA-kNN-based wrapper method for feature selection. Fisher Grasshopper Optimization Algorithm with k-Nearest Neighbor (FGOA- kNN) is the name of the feature-selection method proposed. After applying the hybrid filter-wrapper features selection FGOA-kNN method to each cluster to generate an updated set of clusters, the resulting clusters are aggregated to provide a smaller but more accurate dataset. After the dataset has been updated, it is passed on to the classification stage, where the attacks will be classified.

This is helpful since it reduces the amount of data processing required while still allowing for a comprehensive dataset to be kept and managed. Specifically, the wrapper approach of GOA-kNN is used to implement the feature selection for highly ranked features via the fisher score.

At the same time, the kNN algorithm is employed as the fitness function. Filters-based feature selection approaches are recommended when working with a dataset of high dimensionality because they are quicker than wrapper methods and do not rely on machine learning [45]. Wrapper approaches are slower and computationally more costly since they rely on the learning algorithm to judge the significance of feature subsets, as opposed to filter methods which perform this evaluation irrespective of any classification process [46]. Wrappers, on the other hand, almost always ensure superior FS outcomes compared to filters.

This paper utilized the fisher score to identify the essential features. The Fisher score highlights a group of features where the most significant gaps between data points with various labels are present. The shortest distances between data points with the same labels, however, do exist [40]. Therefore, two steps determine the feature subset:

- Firstly, for each feature G^i, the fisher score is determined using;

$$F\left(G^i\right)=\frac{\sum_{k=1}^{c}\eta_k\left(\mu_k^i-\mu^i\right)^2}{\sum_{k=1}^{c}\eta_k\left(\sigma_k^i\right)^2} \tag{23}$$

Where the standard deviation and the mean of the k-th class are denoted by σ_k^i, μ_k^i for the i-feature. The mean of the overall ith feature is indicated by μ^i.

- Secondly, after scoring each feature in the dataset, the topRanked with the highest scores are chosen as the most informative features for each cluster.

The three major parts of a wrapper-based features selection system are a search technique, an inductive algorithm, and an evaluation measurement [47]. Our method employs GOA as a search algorithm for locating an appropriate subset of features, kNN as an inductive approach, and classification accuracy as an assessment metric. In the work proposed, we use GOA as a wrapper method to search engines to investigate potential avenues for further study. Our rationale for selecting this is GOA's speed and accuracy in searching for the best possible features. Two formulation concerns must be addressed before any metaheuristic algorithm may be used to optimize a problem in a wrapper-based setting. The first involves representing the solution, while the second consists of choosing a fitness function to evaluate the solution's efficacy. Both concerns are addressed in our proposed method as follows:

Population/Individual representation: We must determine which choice variables will be represented in the individual. In this scenario, these factors serve as input features. As seen in Equation (24), a single solution is implemented in our system as a one-dimensional vector of absolute values. The vector's elements show a collection of flags, each representing a different feature. The feature is picked if the $(flag's\ value \geq 0.5)$; otherwise, it is discarded.

$$I=[X1\ \ X2\ \ X3\ \ \ldots\ \ Xn] \tag{24}$$

Fitness evaluation: Each created solution is evaluated based on a set of standards. A fitness function needs to be established for this goal. The average classification accuracy of the kNN model is used as the fitness function in the proposed model. The fitness function used to evaluate the performance of individual i is shown in Equation (25)

$$\mathrm{Fit}_i=\lambda\times\gamma_i+\left(1-\lambda\right)\times\left(\frac{L}{D}\right) \tag{25}$$

Where the classification error computed during the kNN model is indicated by γ_i, The overall set size of features is denoted by D, and the selected features' number is denoted by L. $\lambda \in [0,1]$ is a random weighting value for minimizing classification error.

Once these two design decisions have been made, the FGOA-kNN procedure can be conducted as stated below:

- The FGOA-kNN method, as stated in Equation (24), generates a random population of candidate solutions. Each solution consists of a collection of flags that map to features in the data set.
- In this step, the GOA's reproduction operators are applied to each individual to generate a new set of potential solutions.
- The fitness function specified in Equation (25) is employed to assess each generated solution.
- The search for the optimal solution stops when a predetermined number of iterations is reached. Then, using the FGOA-kNN approach, the set of features with the highest classification accuracy is reported. A high-level overview of the proposed FGOA-kNN feature selection method is shown in Figure 3.

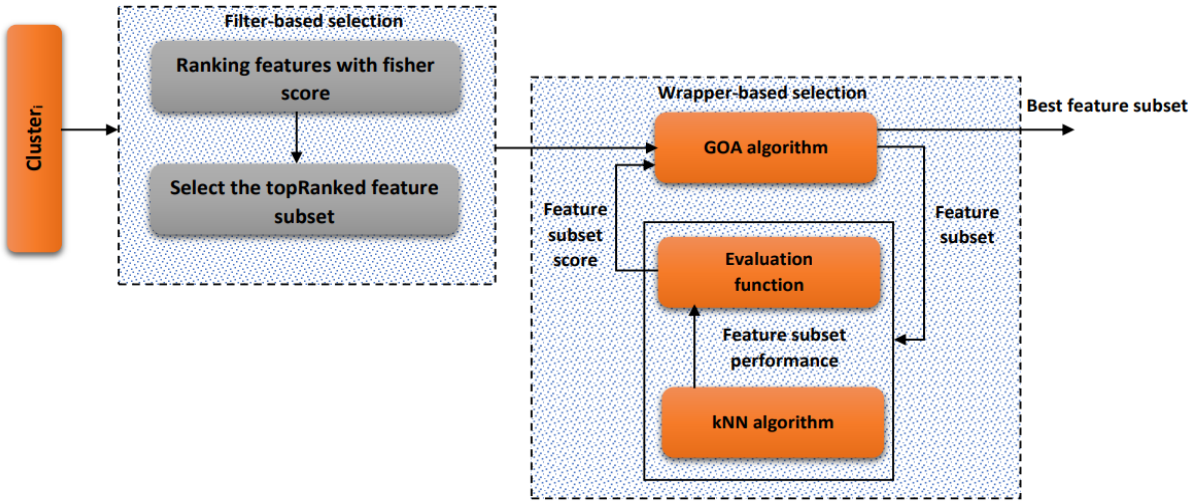

FIGURE 3 **The proposed features selection approach**

D. Classification

In this step, we propose an improved version of Harris Hawks Optimization (IHHO) to determine optimal hyper parameters for Neural Networks, allowing them to detect botnet attacks more effectively. Compared to the original HHO, the IHHO provides faster convergence and higher quality solutions. Neural network weights and biases were optimized with the improved HHO. Next, the optimized Neural Network classifier uses the features chosen in the previous stage as input to determine the type of botnet attack it has encountered.

In the HHO algorithm, the prey's energy attenuation triggers the switching from a global to a local search. HHO is simple and with a simple structure. Nevertheless, when applied to complex issues like the features selection problem, the traditional HHO algorithm has some critical flaws. The poor convergence speed, the problem of local optima, and the problem of solution variety are only a few examples. This paper proposes three significant improvements to the HHO to address its concerns.

The paper presents three ways to improve the HHO. 1) The chaotic map technique is used in place of the original HHO algorithm's random initialization strategy, which utilizes the properties of the chaotic system to generate populations with significant diversity. 2) A new formula is used to update the energy component better to balance global exploration and local exploitation. 3) the ROBL technique is implemented during exploitation to improve further the capability to escape the local optimum.

Algorithm 2 provides the IHHO pseudo-code. As a result, the improved version of the algorithm has preserved HHO's global search capabilities, ability to faster convergence speed and avoid local optima.

Therefore, the resultant solution is of higher quality. In the following paragraphs, we'll discuss the specifics of each of the three improvements made to HHO. Firstly, the solution diversity is increased by employing chaotic maps during the initialization step. The optimization field offers a plethora of chaotic maps [48]. Randomness and ergodicity are characteristics of chaotic systems. Using these features to generate more diverse populations can boost performance

Algorithm 2 IHHO algorithm

```
Input: Maximum number of iterations T, The size of population N
Output: The location of best rabbit (solution)
Initialize max number of iterations T and size of population N
Perform logistic chaotic strategy using Eq. 24
Population initialization   Xi(i=1,2, …, N)
The current iteration number is set t=0
    while t < T
        The fitness value of hawks is calculated
        The best position is set for the position of the prey Xrabbit
        foreach Xi individual
            The   energy E is updated using Eq. (26)
                if(|E|>=1)
              Update location by using Eq. (2)
         end if
         if (|E|<1)
           if(r>= 0.5 & |E| >=0.5)
                Update location by using Eq. (4)
           else if (r>= 0.5 & |E| <0.5)
                Update location by using Eq. (7)
           else if   (r< 0.5 & |E| >=0.5)
                Update location by using Eq. (12)
           else if
                Update location by using Eqs. 12, 13
            end if
         end for
        Carry out ROBL using Eq. 25
        t=t+1
    end while
```

and quicken the algorithm's convergence time. By creating the initialization population using the chaotic map instead of a randomly generated population, Kaur et al. [49] increased the whale algorithm's optimization performance. They made it simple to escape the local optimum. Positions of the Harris Hawks population were previously initialized using randomly generated values. However, this step has since been replaced by using the chaotic map value. Chaotic maps are used to produce chaotic values. The paper's initial population is derived using the Logistic map. Here is how we characterize the Logistic map:

$$x_{i+1} = \mu x_i \left(1 - x_i\right),\ i = 1, 2, \ldots, N-1, \qquad (26)$$

Where N is the total number of individuals in the population, x1 is a random value generated between 0 and 1, and μ is 4.

Secondly, Random Opposition-based learning (ROBL) is used to increase both the speed of convergence and the search mechanism's potential to be exploited.

Tizhoosh's sophisticated optimization method, opposition-based learning (OBL) [50], considers both an estimate's fitness and its matching opposite assessment to find a more promising set of candidates. The OBL idea has been implemented in several meta-heuristics algorithms with positive results [51]. In this paper, rather than using the traditional OBL method, a refined version of OBL called Random Opposition-Based Learning (ROBL) [52] is applied, which can be summarized as follows:

$$\hat{x\_new}_j = low_j + upper_j - \mathrm{r} \times \hat{x\_new}_j, j = 1,2, \ldots, n \qquad (27)$$

Where the lower and upper bounds of the problem are denoted by $upper_j$ and low_j respectively, the opposite solution is represented by $\hat{x}_j$And rand is a random value generated between 0 and 1. Compared to the original OBL, the solution provided by Equation (27) is more spontaneous and can assist the population in avoiding the local optimum. In this paper, we use an improved OBL technique to improve the exploitation phase of HHO in order to increase performance and preventing the original HHO from becoming stuck in local minima. ROBL combined with HHO is an effective means of avoiding the trap of local minima.

Lastly, the exploration and exploitation phases must be kept in balance with the help of the energy factor E. When |E| is significant, the HHO algorithm is more likely to explore. The opposite is true; the greater the tendency for exploitation. Because the energy component E in the HHO method drops linearly from large to small, it is not well suited to depict the actual circumstance in which Harris hawks round up prey in nature. Because of the Harris hawks' nature and their prey's multistage battle, the prey's vitality cannot be accurately conveyed by linear changes alone. If the hawks are going to catch their prey, the energy of the prey needs to fluctuate at regular intervals and eventually equal zero. The prey will obtain a rest to regain energy between rounds of rounding up, but this rest period will diminish with time until the prey has no energy left to regain. This study employs the cosine function to characterize the repeating increase and decrease in prey energy. Here is the formula:

$$\mathrm{E} = 2 \times \cos\left(\frac{5\pi \mathrm{t}}{\mathrm{T}}\right) \times \left(1 - \frac{\mathrm{t}}{\mathrm{T}}\right) \times \left(2\ \mathrm{rand} - 1\right). \qquad (28)$$

To classify ten distinct kinds of botnet attacks and one kind of benign target, the neural network uses IHHO for weight optimization and hyperparameter tuning. For NN training, the IHHO is used as a back-propagation technique. The classifier receives the signal after each input has been given a weight to produce a prediction. This paper's primary goal is to improve ANN training by replacing standard back-propagation methods with a more robust approach, hoping to achieve superior outcomes in terms of classification accuracy and convergence.

V. Experiments and discussion

This section presents a comparison of the results of several models of optimal neural networks to determine which is most suited for detecting botnets in IIoT systems. Here, we employ a method of multiclass classification, which learns to distinguish between different types of threats down to the granularity of individual attacks and more generalized categories. Utilizing this proposed model, we select relevant features to create an efficient neural network IoT botnet detection model.

A. Data Set Description

N-BaIoT dataset [53] is used, which includes data samples with 115 features. The N-BaIoT dataset was utilized and applied for many previous research for botnet detection in IoT/IIoT environment [54]–[63]. The used dataset proved high performance in the field of botnet detection according to many related and previous works therefore it was used in this research. The applied dataset created to serve multiclassification besides binary classification. The information was obtained by mirroring the ports of Internet of Things devices. After the network was configured correctly, the benign data was collected immediately.

TABLE 1.
The name of the model and device type of the N-BaIoT dataset.

Name of model	Type of device
Ecobee	Thermostat
Ennio Danmini	Doorbell
Samsung SNH 1011 N	Webcam
Philips B120N/10	Baby monitor
SimpleHome XCS7-1003-WHT Provision PT-838 SimpleHome XCS7-1002-WHT Provision PT-737E	Security camera

TABLE 2.
THE NUMBER OF USED SAMPLES IN THIS PAPER.

Botnet	Attack type	Baby Monitor	Doorbell	Webcam	Security Camera	Total
Benign		40,034	10,098	11,704	22,888	**84,724**
Mirai	**m_scan**	25,905	26,921	10,970	24,274	**88,070**
	m_plainudp	20,202	20,495	20,902	13,446	**75,045**
	m_syn	29,532	30,643	30,422	15,462	**106,059**
	m_udp	54,258	59,416	39,394	39,652	**192,720**
	m_ack	22,780	25,548	27,087	14,499	**89,914**
Bashlite	**b_scan**	6,964	7,462	6,924	7,099	**28,449**
	b_combo	14,538	14,929	14,667	14,382	**58,516**
	b_udo	26,445	26,468	27,654	26,164	**106,731**
	b_junk	7,087	7,267	7,076	7,267	**28,697**
	b_tcp	23,145	23,035	24,445	22,346	**92,971**
Total		**270,890**	**252,282**	**221,245**	**207,479**	

TABLE 3
EVALUATION ACCURACY OF THE PROPOSED APPROACH
OF FEATURES SELECTION IN COMPARISON TO EXISTING METHODS.

Classifier model	ReliefF	FGOA-kNN	IG
k-nearest neighbor (k-NN)	68.44	**96.03**	67.25
Deep Autoencoder	89.23	**98.13**	89.23
Support vector machine (SVM)	77.12	**96.12**	77.14
Naïve Bayes (NB)	78.94	**89.45**	78.93
Random forest	82.16	**92.24**	83.19
Decision Tree	79.77	**97.23**	78.23

Two packet sizes, a count of packets, and a jitter measurement were extracted together with every statistical characteristic, including the intervals between packet arrivals. All 115 features in our model are used, and a features selection method is employed to determine the most important ones. Datasets were gathered by injecting Mirai and Bashlite attacks, and the IoT devices used to collect them are listed in Table 1. Bashlite, or gafgyt as it's more often known, was developed in C by Lizard Squad. DDoS attacks are launched using this botnet, which infects Linux-based IoT devices. UDP and TCP attacks are just two examples of flooding attacks. Paras' Mirai malware is deployed in widespread attacks against the Internet of Things. We choose N-BaIoT devices such as a webcam (Samsung SNH 1011 N), security camera (Provision PT-838), baby monitor (Philips B120N/10) and doorbell (Ennio) so that we may simulate all 10 of the attack samples. Table 2 displays the total number of samples used from each device.

B. Data splitting

Data is divided into training and testing sets after preprocessing to increase the performance of the suggested model. We randomly split the N-BaIoT dataset's samples in this experiment. The function *train_test_split* in sklearn package is used for data splitting. The dataset was divided into a training set, which made up 75%, and a testing set, which made up 25%. The model was built and trained using the training set, and the test set was used to evaluate the model. Also, 20% of the training set was utilized as a validation set to avoid overfitting. We compute it during training to prevent the validation loss from increasing as the training loss decreases.

C. Experimental Setup

The proposed model was entirely created by our team in Python using the framework of TensorFlow [64] along with the scikit-learn and pandas modules. We worked on a computer running an Intel Core i7 processor, 64-bit Windows 10 and 16 GB of RAM.

D. Evaluation Metrics

We employed four distinct metrics to assess the proposed model's effectiveness. Precision, F1-score, Accuracy and Recall are used in this analysis obtained from the confusion matrix. The confusion matrix is a table that tallies the number of correctly identified cases and the number of misclassified occurrences for each class. Consider that the confusion matrix is the source of the numbers of true negative (T_{Neg}) examples, false negative (F_{Neg}) examples, true positive (T_{Pos}) examples, and false positive (F_{Pos}) examples. Here are some equations for defining the performance indicators:

$$\text{Recall} = \frac{\text{T}_{\text{Pos}}}{\text{T}_{\text{Pos}} + \text{F}_{\text{Neg}}} \tag{29}$$

$$\text{Precision} = \frac{\text{T}_{\text{Pos}}}{\text{T}_{\text{Pos}} + \text{F}_{\text{Pos}}} \tag{30}$$

$$\text{Accuracy} = \frac{\text{T}_{\text{Pos}} + \text{T}_{\text{Neg}}}{\text{T}_{\text{Pos}} + \text{T}_{\text{Neg}} + \text{F}_{\text{Pos}} + \text{F}_{\text{Neg}}} \tag{31}$$

$$\text{F1} - \text{score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{32}$$

E. Results and Discussion

To test the new features selection approach FGOA-kNN and the proposed model for identifying botnet assaults, the results and discussions are split into two sections. The first section considers the features selection approach FGOA-kNN using different classical classifier models. Various experiments are applied to evaluate the accuracy under other models. The impact of the percentage of selected subset features is also assessed. The second section considers the overall model of both FGOA-kNN and IHHO-NN. The effect of the new optimized neural network is evaluated.

1) Features selection approach FGOA-kNN evaluation

In this section, the performance of the proposed FGOA-kNN features selection method integrated with clustering is assessed and compared to the Information Gain (IG) and ReliefF approaches as the common feature selection approaches. Relief is a strategy for choosing features based on an attribute ranking approach that takes into account instances [65]. Each attribute is given a weight according to its importance, and instances are sampled randomly from the data before their nearest neighbors in the same and opposite classes are found. IG utilizes the concept of information gain [66]. IG is the most straightforward attribute ranking approach that accounts for the amount of knowledge acquired by knowing the attribute's value. It determines how much entropy was lost in the split by subtracting the initial entropy from the new entropy. After the clusters have been generated, FGOA-kNN is used to pick out the best features from each cluster. The results demonstrate that the proposed FGOA-kNN outperforms comparable techniques. After each instance in the cluster has been normalized, the Fisher score method is used to rank features in decreasing score order. This new set is then passed to a GOA-kNN-based wrapper method. Filter-based and wrapper-based processes are applied to each cluster individually. A new dataset is created by selecting the highest-scoring feature subset from each cluster. To test the performance of the proposed FGOA-kNN, many classical classification models are used on the N-BaIoT dataset. The proposed FGOA-kNN is evaluated later with the proposed optimized neural network to test its accuracy. Table 3 displays the results of a comparison of the accuracy of the proposed approach with two alternative methods.

The output of the filter phase is shown in Table 4 which lists the feature with its Fisher score. Table 5 displays the selected most relevant features after applying the hybrid FGOA-kNN approach, which are then used in the classification phase to detect the botnet class. The selected 15 features obtained the highest accuracy compared with other feature selection approaches and when using the entire feature set. The results of the proposed FGOA-kNN, which uses the top 20% of features, are superior to those of the related works (as shown in Table 6).

Consequently, the suggested method exhibits several benefits that raise the efficiency and efficacy of the detection of botnets in an IoT environment. Table 3 displays the results of an accuracy test of the proposed

FGOA-kNN method using various feature selections. Figure 4 illustrates the proposed FGOA-kNN method's performance compared to existing approaches. The accuracy of the proposed approach for various percentages amounts of features is displayed in Figure 5.

2) Proposed model evaluation

After aggregating the newly updated clusters, the classification stage is used to classify the attacks based on selected features. For the multi-classification of attacks into ten distinct attack classes and one more benign class, the neural network's hyperparameters are adapted with the help of the novel proposed approach IHHO-NN. Various evaluation measures are used to assess the model.

On the other hand, IHHO-NN has been taught to spot out-of-the-ordinary actions. Due to the ever-changing nature of botnet variations, it is crucial to have a technology like IHHO-NN that can identify previously undiscovered botnet behaviors.

The Internet of Things domain is overly complex compared to more conventional computing environments. By comparing each system to other botnet attack targets, IHHO-NN takes on the increasing complexity of Smart nodes. All associated hosts' traffic statistics are expected to be monitored in an enterprise setting.

However, there is too much-controlled traffic to be practical to store and use in any meaningful way for in-depth neural network training. The IHHO-NN can be trained remotely. Therefore, there is no worry about keeping space constraints of practical value. Because it is a network-based approach, IHHO-NN also requires no computing RAM on frequently limited IoT devices. So, IHHO-NN has no destructive effects on its functioning.

We compare the state-of-the-art approaches somewhat to our newly proposed optimized neural network. Our experimental setup for the proposed model consists of four distinct phases.

Firstly, without utilizing the new feature selection approach or hyperparameter optimization, a neural network was added to the experiments. Table 7 evaluates the neural network's performance without the novel hyperparameter optimization approach and features selection with ten different attacks against a single kind of benign target class.

Secondly, we evaluated the optimized neural network classifier using the novel optimization approach for tuning the parameters. Table 8 shows an assessment of the neural network's performance after optimization of the hyperparameter was applied with ten attacks on a single class of benign targets.

TABLE 4
THE FEATURES WITH THEIR FISHER SCORE AFTER FIRST PHASE OF FGOA-KNN APPROACH

Name	# F	F-Score	Name	# F	F-Score	Name	# F	F-Score
H _L0.01_ weight	28	1.24	HH_L0.01_std	61	0.31	HH_jit_L3_weight	69	0.05
H _L0.01_ mean	29	0.93	HpHp_L0.1_magnitude	105	0.29	HH_jit_L5_weight	66	0.05
MI _dir_L0.01_ mean	14	0.93	HpHp_L0.01_magnitude	112	0.29	HH_L5_weight	31	0.05
H_L0.1_mean	26	0.88	HpHp_L3_magnitude	91	0.29	HpHp_L1_weight	95	0.05
MI_dir_L0.1_mean	11	0.88	HpHp_L5_magnitude	84	0.29	HpHp_L0.1_weight	102	0.05
H_L0.01_variance	30	0.75	HH_L0.01_magnitude	62	0.29	HpHp_L0.01_weight	109	0.05
MI_dir_L0.01_variance	15	0.75	HpHp_L1_magnitude	98	0.29	HpHp_L3_weight	88	0.05
MI_dir_L0.1_variance	12	0.70	HH_L5_magnitude	34	0.27	HpHp_L1_radius	99	0.04
H_L1_mean	23	0.73	HH_L1_magnitude	48	0.27	HpHp_L0.01_radiu	113	0.05
MI_dir_L1_mean	8	0.73	HH_L0.1_magnitude	55	0.27	HpHp_L5_weight	81	0.05
H_L0.1_variance	27	0.73	HH_L3_magnitude	41	0.27	HH_jit_L0.01_variance	80	0.04
MI_dir_L0.01_weight	13	1.24	HpHp_L1_mean	96	0.31	HH_L3_weight	38	0.05
I_L1_variance	24	0.69	HH_jit_L0.01_mean	79	0.22	HH_jit_L0.1_variance	77	0.04
MI_dir_L1_variance	9	0.69	HH_L0.1_std	54	0.22	HpHp_L0.1_radius	106	0.04
H_L3_mean	20	0.69	HH_jit_L1_mean	73	0.22	HpHp_L3_radius	92	0.04
MI_dir_L3_mean	5	0.69	HH_jit_L0.1_mean	76	0.19	HpHp_L5_radius	85	0.04
H_L3_variance	21	0.64	HH_jit_L5_mean	67	0.19	HpHp_L1_covariance	100	0.04
MI_dir_L3_variance	6	0.64	HH_jit_L3_mean	70	0.16	HpHp_L5_covariance	86	0.04
H_L0.1_weight	25	0.64	HH_L0.01_weight	59	0.16	HpHp_L3_covariance	93	0.04
MI_dir_L0.1_weight	10	0.64	HpHp_L0.01_std	111	0.13	HH_L0.1_pcc	58	0.03
H_L5_mean	17	0.64	HpHp_L0.1_std	104	0.13	HH_L1_covariance	50	0.03
MI_dir_L5_mean	2	0.55	HH_jit_L0.01_weight	78	0.12	HpHp_L5_pcc	87	0.03
H_L5_variance	18	0.55	HH_L3_std	40	0.12	HpHp_L3_pcc	94	0.03
MI_dir_L5_variance	3	0.55	HH_L1_std	47	0.12	HpHp_L0.1_pcc	108	0.03
H_L5_weight	16	0.51	HH_L0.01_pcc	65	0.10	HH_L3_covariance	43	0.03
MI_dir_L5_weight	1	0.51	HH_L5_std	33	0.10	HH_L0.1_covariance	57	0.03
H_L3_weight	19	0.48	HpHp_L3_std	90	0.10	HH_L5_covariance	36	0.03
MI_dir_L3_weight	4	0.48	HpHp_L1_std	97	0.09	HpHp_L0.1_covariance	107	0.03
H_L1_weight	22	0.48	HH_L0.01_radius	63	0.09	HpHp_L1_pcc	101	0.03
MI_dir_L1_weight	7	0.48	HpHp_L5_std	83	0.09	HpHp_L0.01_covariance	114	0.02
HH_L0.1_mean	53	0.45	HH_L1_radius	49	0.07	HH_jit_L1_variance	74	0.02
HH_L0.01_mean	60	0.45	HH_L0.1_radius	56	0.07	HH_L3_pcc	44	0.01
HH_L1_mean	46	0.33	HH_jit_L0.1_weight	75	0.07	HH_L1_pcc	51	0.01
HpHp_L0.01_mean	110	0.33	HH_L0.1_weight	52	0.07	HH_L0.01_covariance	64	0.01
HpHp_L5_mean	82	0.33	HH_L5_radius	35	0.07	HH_L5_pcc	37	0.00
HH_L3_mean	39	0.33	HH_L3_radius	42	0.07	HH_jit_L5_variance	68	0.00
HpHp_L0.1_mean	103	0.33	HH_L1_weight	45	0.07	HH_jit_L3_weight	69	0.00
HH_L5_mean	32	0.33	HpHp_L0.01_pcc	115	0.05	HH_L3_weight	38	0.00
HpHp_L3_mean	89	0.33	HH_jit_L1_weight	72	0.05	HH_jit_L3_variance	71	0.00

TABLE 5
THE SELECTED FEATURES AFTER APPLYING FGOA-KNN

H_L0.01_weight	MI_dir_L0.01_variance	H_L0.01_variance
H_L0.01_mean	H_L0.1_variance	MI_dir_L1_variance
MI_dir_L0.01_mean	MI_dir_L0.1_variance	MI_dir_L3_variance
H_L0.1_mean	MI_dir_L0.01_weight	H_L3_variance
MI_dir_L0.1_mean	MI_dir_L1_mean	H_L5_mean

TABLE 6
COMPARISON OF THE PROPOSED METHOD'S CLASSIFICATION ACCURACY USING A DIFFERENT PERCENTAGE OF FEATURE SELECTIONS.

Classifier model	50%	20%	100%
k-nearest neighbor (k-NN)	96.01	**96.03**	96.03
Deep Autoencoder	95.21	98.13	**98.27**
Support vector machine (SVM)	91.33	**96.12**	95.88
Naïve Bayes (NB)	88.15	**89.45**	89.45
Random forest	91.16	**92.24**	92.19
Decision Tree	96.17	**97.23**	97.23

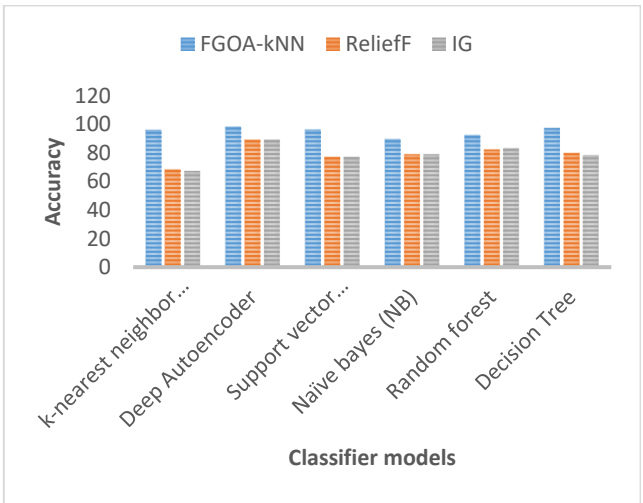

FIGURE 4. Comparison of proposed FGOA-kNN with other classical methods for features selection

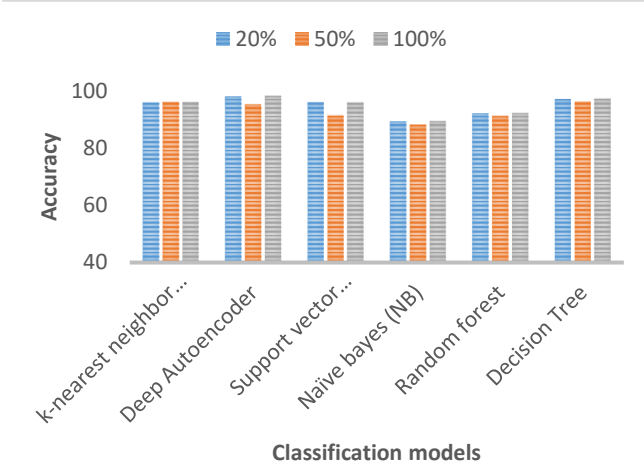

FIGURE 5. The proposed method's classification accuracy is in light of the number of features to be considered.

TABLE 7.
EVALUATING THE NEURAL NETWORK'S PERFORMANCE WITH TEN CLASSES OF ATTACKS AGAINST A SINGLE CLASS OF BENIGN WITHOUT OPTIMIZING THE HYPERPARAMETERS AND WITHOUT FEATURES SELECTION.

Type of attack	Recall	Precision	F1-Score	Accuracy
Benign	0.9836	0.9917	0.9877	0.9918
m_plainudp	0.8917	0.8537	0.8723	0.9152
m_scan	0.8235	0.7368	0.7777	0.8018
m_udp	0.9722	0.8485	0.9061	0.9775
m_syn	0.9722	0.6796	0.8067	0.8500
b_scan	0.9655	0.9211	0.9428	0.9550
m_ack	0.7910	0.6393	0.7071	0.7249
b_udo	0.8187	0.7568	0.7865	0.7631
b_tcp	0.8413	0.5668	0.6773	0.7692
b_junk	0.7292	0.6512	0.6880	0.6966
b_combo	0.8434	0.8187	0.8309	0.7909
Average	0.8775	0.7695	0.8345	0.8396

Thirdly, the neural network with the novel features selection approach using FGOA-kNN and hyperparameter optimization using IHHO is evaluated using the eleven used classes for attacks and benign.

Since many related NN optimization strategies have been mentioned in the literature, we incorporated additional NN optimizers such as PSO-NN, GWO-NN and BA-NN to provide a more accurate comparison. Tables 9, 10, and 11 evaluate the effectiveness of BA-NN, PSO-NN, and GWO-NN with ten distinct types of attacks and a single class of harmless targets, respectively. Table 12 presents a relative evaluation of the proposed IHHO-NN utilizing the eleven classes of attacks and benign. To improve Recall at the expense of precision, an integrated platform with high traffic predictability can detect any abnormal behaviour. Measures of effectiveness can be uniformly applied to all varieties of botnet attack activities. To conduct empirical validation, we extracted dynamic and static features from the training set and utilized NN to analyze how these attributes affected the

TABLE 8.
EVALUATING THE NEURAL NETWORK'S PERFORMANCE WITH TEN CLASSES OF ATTACKS AGAINST A SINGLE CLASS OF BENIGN OPTIMIZING THE HYPERPARAMETERS WITHOUT FEATURES SELECTION.

Type of attack	Recall	Precision	F1-Score	accuracy
Benign	0.9831	0.9859	0.9845	0.9831
m_scan	0.9943	0.9722	0.9831	0.9831
m_plainudp	0.9589	0.9375	0.9481	0.9336
m_syn	0.9817	0.8974	0.9377	0.9492
m_udp	0.9607	0.9333	0.9468	0.9392
m_ack	0.9602	0.9130	0.9360	0.9492
b_scan	0.9633	0.9502	0.9567	0.9592
b_combo	0.8936	0.9417	0.9170	0.9248
b_udo	0.9722	0.9611	0.9666	0.9696
b_junk	0.9809	0.9438	0.9620	0.9564
b_tcp	0.9502	0.9190	0.9344	0.9340
average	0.9636	0.9414	0.9521	0.9529

TABLE 9.
THE EFFICACY OF THE BAT ALGORITHM (BA-NN) IS MEASURED AGAINST TEN DISTINCT ATTACK TYPES AND A SINGLE BENIGN CATEGORY.

Type of attack	Recall	Precision	F1-Score	Accuracy
Benign	0.9967	0.9091	0.9509	0.9704
m_plainudp	0.9852	0.9868	0.9860	0.9920
m_scan	0.8011	0.6383	0.7105	0.7380
m_udp	0.9174	0.9077	0.9125	0.9148
m_syn	0.8439	0.8451	0.8445	0.8450
b_scan	0.8571	0.8333	0.8451	0.8450
m_ack	0.9375	0.9146	0.9259	0.9292
b_udo	0.9885	0.9772	0.9828	0.9838
b_tcp	0.9036	0.8824	0.8929	0.8940
b_junk	0.8451	0.8108	0.8276	0.8377
b_combo	0.7317	0.5263	0.6122	0.6290
average	0.8916	0.8392	0.8628	0.8708

average precision and Recall from the test set for five different NN configurations.

Using the data in Table 7, we can conclude that the neural network achieves a higher recall rate 0. 9836 and a precision rate of 0.9917 for the benign class. But identifying neural networks does not provide a low false-positive rate when botnet attacks are involved. The classifier's high classification rate suggests that typical CNN with untuned hyperparameters produced low precision rates. The capability of a neural network to detect benign examples as malicious attack net attacks are demonstrated by a lower false-negative rate, which is measured by a greater average recall rate. As shown in Table 8, the new optimization approach with the neural network improved the average rate of precision.

The novel metaheuristics improved algorithm enhances the tuning parameters of the neural network, avoids local minima of values, and uses appropriate parameters for training the neural network.

TABLE 10.
THE EFFICACY OF THE PARTICLE SWARM OPTIMIZATION ALGORITHM (PSO-NN) IS MEASURED AGAINST TEN DISTINCT ATTACK TYPES AND A SINGLE BENIGN CATEGORY.

Type of attack	Recall	Precision	F1-Score	Accuracy
Benign	0.9569	0.9302	0.9434	0.9492
m_plainudp	0.9390	0.9132	0.9259	0.9248
m_scan	0.9434	0.9324	0.9379	0.9384
m_udp	0.9216	0.8929	0.9070	0.9080
m_syn	0.9132	0.8889	0.9009	0.9020
b_scan	0.9479	0.8869	0.9164	0.9248
m_ack	0.9302	0.9009	0.9153	0.9196
b_udo	0.9259	0.9091	0.9174	0.9188
b_tcp	0.9456	0.9174	0.9313	0.9340
b_junk	0.9070	0.7968	0.8484	0.8561
b_combo	0.8264	0.7491	0.7859	0.7884
average	0.9234	0.8834	0.9027	0.9058

TABLE 11.
THE EFFICACY OF THE GREY WOLF OPTIMIZATION ALGORITHM (GWO-NN) IS MEASURED AGAINST TEN DISTINCT ATTACK TYPES AND A SINGLE BENIGN CATEGORY.

Type of attack	Recall	Precision	F1-Score	Accuracy
Benign	0.9701	0.9353	0.9524	0.9574
m_plainudp	0.9665	0.9091	0.9369	0.9485
m_scan	0.9720	0.9286	0.9498	0.9524
m_udp	0.9923	0.9962	0.9943	0.9953
m_syn	0.9886	0.9943	0.9914	0.9935
b_scan	0.9811	0.9028	0.9403	0.9440
m_ack	0.9738	0.9665	0.9701	0.9724
b_udo	0.9028	0.9059	0.9043	0.9075
b_tcp	0.9720	0.9665	0.9692	0.9740
b_junk	0.9630	0.7027	0.8125	0.8075
b_combo	0.8814	0.8414	0.8609	0.8708
average	0.9603	0.9136	0.9348	0.9385

Adapted parameters with practical architectures enhance the classification process. From table 8, the variant achieved a high average maximum Recall of 0.9636 and maximum precision of 0.9414, which improves the prediction process by increasing the rate of false positives.

From Table 9, we can see that the BA-NN has a greater recall rate, but a lower precision rate (0.5263), and the average Recall is 0.8916. Table 10 shows that All target classes, except for ""b_udp", received high precision rates via PSO-NN which only achieved 0.7491. The results for GWO-NN, as shown in Table 11, a superior algorithm for such optimization problems, have a higher average Recall of 0.9603. the algorithm has higher Recall for most attacks except for lower results for the "b_junk" attack of 0.9630. However, compared to the non-optimization version of the experimental data, the variations GOW-NN, PSO-NN and BA-NN have higher accuracy. The proposed model successfully countered each detected botnet attempt (see Table 12).

The proposed model achieved low false-negative and false-positive rates using the novel features selection method with the improved IHHO. Using the clustering process with the new approach, FGOA-kNN enhances the classification process by revealing the unknown pattern inside each cluster for the features selection approach. Also, using both filter and wrapper methods improved the overall accuracy. The proposed model achieves the lowest misclassification rate for all target classes with maximum accuracy of 98.7% compared to all previous neural networks. The results prove the superiority of the proposed model for detecting attacks.

Figure 6 indicates the average Recall, precision, and F-score measure for the proposed model against the recent models. The results prove the superiority of the proposed model for detecting attacks. The radar figure shows that the proposed model outperforms other models in the average case for detecting botnet attacks due to the new improvements applied to the proposed model.

The suggested strategy can significantly reduce attack-related losses if the compromised IoT device is immediately and directly removed from the network upon classifying attack-related irregularities. Figure 7 depicts the loss variation that occurs when training neural networks with the suggested methodology.

The loss curves also show that the proposed model is better than other neural network variations. Additionally, the average execution time was computed during the training of the proposed model using 20% of selected data and using all features, as given in Table 13, to assess the efficacy of the proposed model. The average execution time used to train and test the proposed model using the specified percentage of features is significantly lower than the average execution time used to use all features, as shown in Table 13.

This demonstrates the effectiveness and practicability of the proposed model for the detection response.

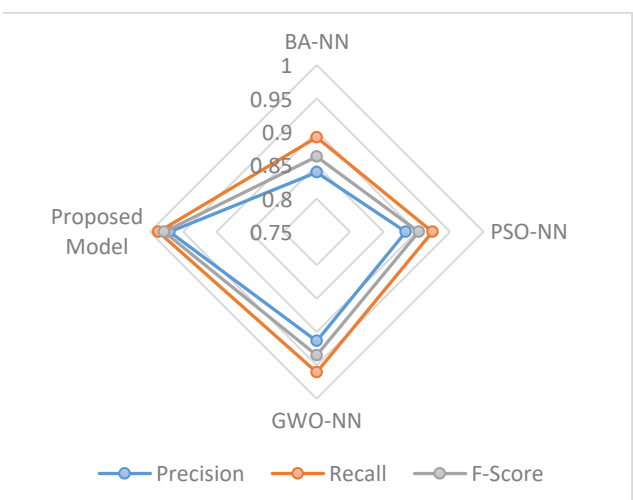

FIGURE 6. Radar graph for proposed model with other models in terms of Recall, precision and F-score

TABLE 12.
THE EFFICACY OF THE PROPOSED MODEL IS MEASURED AGAINST TEN DISTINCT ATTACK TYPES AND A SINGLE BENIGN CATEGORY.

Type of attack	Recall	Precision	F1-Score	Accuracy
Benign	0.9972	0.9890	0.9931	0.9953
m_plainudp	0.9945	0.9783	0.9863	0.9896
m_scan	0.9917	0.9677	0.9796	0.9847
m_udp	0.9783	0.9574	0.9677	0.9696
m_syn	0.9923	0.9756	0.9839	0.9839
b_scan	0.9868	0.9534	0.9698	0.9680
m_ack	0.9890	0.9626	0.9756	0.9890
b_udo	0.9906	0.9600	0.9751	0.9724
b_tcp	0.9641	0.9651	0.9646	0.9761
b_junk	0.9836	0.9809	0.9823	0.9773
b_combo	0.9917	0.9847	0.9882	0.9815
average	0.9873	0.9704	0.9787	0.9807

TABLE 13.
AVERAGE TIMES FOR THE PROPOSED MODEL TO DETECT MIRAI, BASHLITE, AND BENIGN CLASSES ARE TAKEN.

Size of features	Testing Time AVG. (Sec.)	Training Time AVG. (Sec.)
All features	7.316	1129.576
20% selected features	1.981	65.119

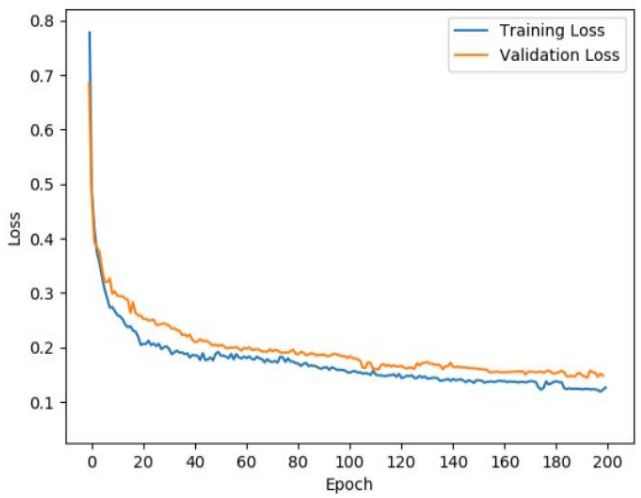

FIGURE 7. The loss during neural network training of the proposed model.

VI. Conclusion and future work

This paper introduces a proposed model to detect botnet attacks in an IoT environment. The proposed model is presented to catch the botnets using a novel feature selection approach with an optimized neural network. The novel feature selection approach integrates unsupervised clustering with a filter-wrapper-based approach. A novel metaheuristic improvement to the standard HHO is proposed to enhance the tuning process of the neural network, empowering the classification process. The evaluation showed that the novel feature selection approach with the new model outperformed other related models in terms of precision and Recall. The selected percentage of features is also evaluated for different classifiers in the N-BaIoT dataset. In the future, other new optimization algorithms will be used to improve the detection process.

References

[1] F. Jameel, U. Javaid, W. U. Khan, M. N. Aman, H. Pervaiz, and R. Jäntti, "Reinforcement learning in blockchain-enabled IIoT networks: A survey of recent advances and open challenges," *Sustainability*, vol. 12, no. 12, p. 5161, 2020.

[2] K. A. Abuhasel and M. A. Khan, "A secure industrial Internet of Things (IIoT) framework for resource management in smart manufacturing," IEEE Access, vol. 8, pp. 117354–117364, 2020.

[3] A. Al-Abassi, H. Karimipour, A. Dehghantanha, and R. M. Parizi, "An ensemble deep learning-based cyber-attack detection in industrial control system," *IEEE Access*, vol. 8, pp. 83965–83973, 2020.

[4] W. U. Khan, M. A. Javed, T. N. Nguyen, S. Khan, and B. M. Elhalawany, "Energy-efficient resource allocation for 6G backscatter-enabled NOMA IoV networks," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[5] M. Azmat, S. Kummer, L. T. Moura, F. Di Gennaro, and R. Moser, "Future outlook of highway operations with implementation of innovative technologies like AV, CV, IoT and Big Data," Logistics, vol. 3, no. 2, p. 15, 2019.

[6] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "Botnet in DDoS attacks: trends and challenges," IEEE Communications Surveys \& Tutorials, vol. 17, no. 4, pp. 2242–2270, 2015.

[7] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in IoT security: Current solutions and future challenges," *IEEE Communications Surveys \& Tutorials*, vol. 22, no. 3, pp. 1686–1721, 2020.

[8] M. Antonakakis *et al.*, "Understanding the mirai botnet," in *26th USENIX security symposium (USENIX Security 17)*, 2017, pp. 1093–1110.

[9] O. Osanaiye, K.-K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: Review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147–165, 2016.

[10] C. D. McDermott, F. Majdani, and A. V Petrovski, "Botnet detection in the internet of things using deep learning approaches," in *2018 international joint conference on neural networks (IJCNN)*, 2018, pp. 1–8.

[11] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, "Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using spark," *IEEE Access*, vol. 6, pp. 59657–59671, 2018.

[12] T. A. Tuan, H. V. Long, L. H. Son, R. Kumar, I. Priyadarshini, and N. T. K. Son, "Performance evaluation of Botnet DDoS attack detection using machine learning," *Evol Intell*, vol. 13, no. 2, pp. 283–294, 2020.

[13] V. R. Kebande and H. S. Venter, "A cognitive approach for botnet detection using Artificial Immune System in the cloud," in *2014 Third International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec)*, 2014, pp. 52–57.

[14] R. A. Khurma, I. Aljarah, A. Sharieh, and S. Mirjalili, "Evolopy-fs: An open-source nature-inspired optimization framework in python for feature selection," in *Evolutionary machine learning techniques*, Springer, 2020, pp. 131–173.

[15] K. Da, "A method for stochastic optimization. arXiv," *arXiv preprint arXiv:1412.6980*, 2014.

[16] H. T. Rauf, S. Malik, U. Shoaib, M. N. Irfan, and M. I. Lali, "Adaptive inertia weight Bat algorithm with Sugeno-Function fuzzy search," *Appl Soft Comput*, vol. 90, p. 106159, 2020.

[17] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE communications surveys \& tutorials*, vol. 21, no. 1, pp. 686–728, 2018.

[18] S. Dwivedi, M. Vardhan, and S. Tripathi, "Defense against distributed DoS attack detection by using intelligent evolutionary algorithm," *International Journal of Computers and Applications*, vol. 44, no. 3, pp. 219–229, 2022.

[19] F. Amiri, M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1184–1199, 2011.

[20] A. H. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," in *2003 Symposium on Applications and the Internet, 2003. Proceedings.*, 2003, pp. 209–216.

[21] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans Pattern Anal Mach Intell*, vol. 27, no. 8, pp. 1226–1238, 2005.
[22] R. Qaddoura, M. Al-Zoubi, H. Faris, I. Almomani, and others, "A multi-layer classification approach for intrusion detection in iot networks based on deep learning," *Sensors*, vol. 21, no. 9, p. 2987, 2021.
[23] A. Kannan, G. Q. Maguire Jr, A. Sharma, and P. Schoo, "Genetic algorithm based feature selection algorithm for effective intrusion detection in cloud networks," in *2012 IEEE 12th International Conference on Data Mining Workshops*, 2012, pp. 416–423.
[24] S. M. Kasongo, "An advanced intrusion detection system for IIoT based on GA and tree based algorithms," *IEEE Access*, vol. 9, pp. 113199–113212, 2021.
[25] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021.
[26] M. Mayuranathan, M. Murugan, and V. Dhanakoti, "Best features based intrusion detection system by RBM model for detecting DDoS in cloud environment," *J Ambient Intell Humaniz Comput*, vol. 12, no. 3, pp. 3609–3619, 2021.
[27] A. Javadpour, S. K. Abharian, and G. Wang, "Feature selection and intrusion detection in cloud environment based on machine learning algorithms," in *2017 IEEE international symposium on parallel and distributed processing with applications and 2017 IEEE international conference on ubiquitous computing and communications (ISPA/IUCC)*, 2017, pp. 1417–1421.
[28] R. Qaddoura, A. Al-Zoubi, I. Almomani, and H. Faris, "A multi-stage classification approach for iot intrusion detection based on clustering with oversampling," *Applied Sciences*, vol. 11, no. 7, p. 3022, 2021.
[29] A. Firdaus, N. B. Anuar, A. Karim, and M. F. A. Razak, "Discovering optimal features using static analysis and a genetic search based method for Android malware detection," *Frontiers of Information Technology \& Electronic Engineering*, vol. 19, no. 6, pp. 712–736, 2018.
[30] Y. Meidan *et al.*, "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," *IEEE Pervasive Comput*, vol. 17, no. 3, pp. 12–22, 2018.
[31] B. Abdollahzadeh, F. Soleimanian Gharehchopogh, and S. Mirjalili, "Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems," *International Journal of Intelligent Systems*, vol. 36, no. 10, pp. 5887–5958, 2021.
[32] W. Wang, J. Xi, A. Chong, and L. Li, "Driving style classification using a semisupervised support vector machine," *IEEE Trans Hum Mach Syst*, vol. 47, no. 5, pp. 650–660, 2017.
[33] L. L. C. Kasun, Y. Yang, G.-B. Huang, and Z. Zhang, "Dimension reduction with extreme learning machine," *IEEE transactions on Image Processing*, vol. 25, no. 8, pp. 3906–3918, 2016.
[34] P. Devan and N. Khare, "An efficient XGBoost--DNN-based classification model for network intrusion detection system," *Neural Comput Appl*, vol. 32, no. 16, pp. 12499–12514, 2020.
[35] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future generation computer systems*, vol. 97, pp. 849–872, 2019.
[36] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature inspired cooperative strategies for optimization (NICSO 2010)*, Springer, 2010, pp. 65–74.
[37] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: theory and application," *Advances in Engineering Software*, vol. 105, pp. 30–47, 2017.
[38] Q. Gu, Z. Li, and J. Han, "@article{gu2012generalized, title={Generalized fisher score for feature selection}, author={Gu, Quanquan and Li, Zhenhui and Han, Jiawei}, journal={arXiv preprint arXiv:1202.3725}, year={2012} }," *arXiv preprint arXiv:1202.3725*, 2012.
[39] M. I. Zhou, "A hybrid feature selection method based on fisher score and genetic algorithm," *Journal of Mathematical Sciences: Advances and Applications*, vol. 37, no. 1, pp. 51–78, 2016.
[40] Q. Gu, Z. Li, and J. Han, "@article{gu2012generalized, title={Generalized fisher score for feature selection}, author={Gu, Quanquan and Li, Zhenhui and Han, Jiawei}, journal={arXiv preprint arXiv:1202.3725}, year={2012} }," *arXiv preprint arXiv:1202.3725*, 2012.
[41] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.," 2022.
[42] S. Kasim, S. Deris, and R. M. Othman, "Multi-stage filtering for improving confidence level and determining dominant clusters in clustering algorithms of gene expression data," *Comput Biol Med*, vol. 43, no. 9, pp. 1120–1133, 2013.
[43] T. M. Kodinariya and P. R. Makwana, "Review on determining number of Cluster in K-Means Clustering," *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.
[44] M. Greenacre and R. Primicerio, *Multivariate analysis of ecological data*. Fundacion BBVA, 2014.
[45] N. Sánchez-Maroño, A. Alonso-Betanzos, and M. Tombilla-Sanromán, "Filter methods for feature selection--a comparative study," in *International Conference on Intelligent Data Engineering and Automated Learning*, 2007, pp. 178–187.
[46] J. Kohavi, "Kohavi R., John GH," *Wrappers for feature subset selection, Artificial Intelligence*, vol. 97, no. 1–2, pp. 273–324, 1997.
[47] J. Kohavi, "Kohavi R., John GH," *Wrappers for feature subset selection, Artificial Intelligence*, vol. 97, no. 1–2, pp. 273–324, 1997.
[48] S. Arora and P. Anand, "Chaotic grasshopper optimization algorithm for global optimization," *Neural Comput Appl*, vol. 31, no. 8, pp. 4385–4405, 2019.
[49] G. Kaur and S. Arora, "Chaotic whale optimization algorithm," *J Comput Des Eng*, vol. 5, no. 3, pp. 275–284, 2018.
[50] H. R. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," *Proceedings - International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA 2005 and International Conference on Intelligent Agents, Web Technologies and Internet*, vol. 1, pp. 695–701, 2005, doi: 10.1109/cimca.2005.1631345.
[51] M. Abd Elaziz, D. Oliva, and S. Xiong, "An improved opposition-based sine cosine algorithm for global optimization," *Expert Syst Appl*, vol. 90, pp. 484–500, 2017.
[52] W. Long, J. Jiao, X. Liang, S. Cai, and M. Xu, "A random opposition-based learning grey wolf optimizer," *IEEE Access*, vol. 7, pp. 113810–113825, 2019.
[53] Y. Meidan *et al.*, "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," *IEEE Pervasive Comput*, vol. 17, no. 3, pp. 12–22, 2018.
[54] Z. Hussain, A. Akhunzada, J. Iqbal, I. Bibi, and A. Gani, "Secure IIoT-enabled industry 4.0," *Sustainability*, vol. 13, no. 22, p. 12384, 2021.
[55] M. Shahin, "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.
[56] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. den Hartog, "ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets," *IEEE Internet Things J*, vol. 9, no. 1, pp. 485–496, 2021.
[57] D. Javeed, T. Gao, M. T. Khan, and D. Shoukat, "A hybrid intelligent framework to combat sophisticated threats in secure industries," *Sensors*, vol. 22, no. 4, p. 1582, 2022.
[58] M. M. Alani, "BotStop: Packet-based efficient and explainable IoT botnet detection using machine learning," *Comput Commun*, vol. 193, pp. 53–62, 2022.
[59] T. Hasan *et al.*, "Securing industrial internet of things against botnet attacks using hybrid deep learning approach," *IEEE Trans Netw Sci Eng*, 2022.
[60] R. Abu Khurma, I. Almomani, and I. Aljarah, "IoT botnet detection using salp swarm and ant lion hybrid optimization model," *Symmetry (Basel)*, vol. 13, no. 8, p. 1377, 2021.
[61] Y. Masoudi-Sobhanzadeh and S. Emami-Moghaddam, "A real-time IoT-based botnet detection method using a novel two-step feature selection technique and the support vector machine classifier," *Computer Networks*, vol. 217, p. 109365, 2022.
[62] M. Al-Sarem, F. Saeed, E. H. Alkhammash, and N. S. Alghamdi, "An aggregated mutual information based feature selection with machine learning methods for enhancing IoT botnet attack detection," *Sensors*, vol. 22, no. 1, p. 185, 2022.
[63] M. G. Karthik and M. B. M. Krishnan, "Hybrid random forest and synthetic minority over sampling technique for detecting internet of things attacks," *J Ambient Intell Humaniz Comput*, pp. 1–11, 2021.
[64] B. Pang, E. Nijkamp, and Y. N. Wu, "Deep learning with tensorflow: A review," *Journal of Educational and Behavioral Statistics*, vol. 45, no. 2, pp. 227–248, 2020.
[65] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Machine learning proceedings 1992*, Elsevier, 1992, pp. 249–256.
[66] M. A. Hall and G. Holmes, "Benchmarking attribute selection techniques for discrete class data mining," *IEEE Trans Knowl Data Eng*, vol. 15, no. 6, pp. 1437–1447, 2003.

Fatma Taher (Member, IEEE) received the Ph.D. degree from the Khalifa University of Science, Technology and Research, United Arab Emirates, in 2014. She is currently the Assistant Dean of the College of Technological Innovation, Zayed University, Dubai, United Arab Emirates. She has published more than 40 articles in international journals and conferences. Her research interests are in the areas of signal and image processing, pattern recognition, deep learning, machine learning, artificial intelligence, medical image analysis, especially in detecting of the cancerous cells, kidney transplant, and autism. In addition to that, her researches are watermarking, remote sensing, and satellite images. She served as a member of the steering, organizing, and technical program committees of many international conferences.

Mahmoud Abdel-Salam received the B.S. degree in web engineering from Mansoura University, Egypt, in 2019. His research interests are web, cyber-security, artificial intelligence, optimization problems, evolutionary algorithms, web service composition, big data, data mining and cloud computing. He published several papers a journals and conferences.

Dr. Mohamed Elhoseny is an Associate Professor at the University of Sharjah, UAE. Dr. Elhoseny is an ACM Distinguished Speaker and IEEE Senior Member. His research interests include Smart Cities, Network Security, Artificial Intelligence, Internet of Things, and Intelligent Systems. Dr. Elhoseny is the founder and the Editor-in-Chief of IJSSTA journal published by IGI Global. Also, he is an Associate Editor at several Q1 journals such as IEEE Journal of Biomedical and Health Informatics, IEEE Access, Scientific Reports, IEEE Future Directions, Remote Sensing, International Journal of E-services and Mobile Applications and Human-centric Computing and Information Sciences. Moreover, he served as the co-chair, the publication chair, the program chair, and a track chair for several international conferences published by recognized publishers such as IEEE and Springer. Dr. Elhoseny is the Editor-in-Chief of The Sensors Communication for Urban Intelligence CRC Press-Taylor& Francis Book Series, and the Editor-in-Chief of The Distributed Sensing and Intelligent Systems CRC Press-Taylor& Francis Book Series.

Dr. Ibrahim M. EL-Hasnony received his B.Sc., M.Sc., Ph.D degrees in information systems from Mansoura University, Egypt in 2010, 2016 and 2021, respectively. His research interests include cloud computing, big data, data analysis, smart city, the Internet of Things, neural networks, artificial intelligence, web service composition, blockchain, and evolutionary algorithms. He has published several papers in international and local conferences, journals and proceedings in different fields of information technology.