Automatika

Journal for Control, Measurement, Electronics, Computing and Communications

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/taut20

Botnet detection in the internet-of-things networks using convolutional neural network with pelican optimization algorithm

Swapna Thota & D. Menaka

To cite this article: Swapna Thota & D. Menaka (2024) Botnet detection in the internet-of-things networks using convolutional neural network with pelican optimization algorithm, Automatika, 65:1, 250-260, DOI: 10.1080/00051144.2023.2288486

To link to this article: https://doi.org/10.1080/00051144.2023.2288486

Published online: 27 Dec 2023.

AUTOMATIKA
2023, VOL. 65, NO. 1, 250–260
https://doi.org/10.1080/00051144.2023.2288486

Botnet detection in the internet-of-things networks using convolutional neural network with pelican optimization algorithm

Swapna Thota and D. Menaka

Department of Electronics and Communication Engineering, Noorul Islam Centre for Higher Education, Kumaracoil, India

ABSTRACT

Hackers nowadays employ botnets to undertake cyberattacks towards the Internet of Things (IoT) by illegally exploiting the scattered network's resources of computing devices. Several Machine Learning (ML) and Deep Learning (DL) methods for detecting botnet (BN) assaults in IoT networks have recently been proposed. However, in the training set, severely imbalanced network traffic data degrades the classification performances of state-of-the-art ML as well as DL algorithm, particularly in classes with very few samples. The Convolutional Neural Network -Pelican Optimization System (CNN-POA) is a DL relied botnet attack detection algorithm developed in this research. Meanwhile, typical evaluation markers are used to compare the overall performance of the proposed CNN-POA and additional frequently employed algorithms. The simulation results suggest that the CNN-POA method is effective and dependable for detecting IoT network intrusion attacks. Experiments revealed that the suggested CNN-POA approach outperformed a number of current metaheuristic algorithms, with an accuracy of 99.5%.

ARTICLE HISTORY
Received 1 September 2023
Accepted 14 November 2023

KEYWORDS
Botnet detection; internet of things; optimization; convolutional neural networks

1. Introduction

As the volume of IoT equipments deployed expands globally and the frequency of such attacks have indeed achieved new extremes, prompt detection of Distributed Denial-of-Service (DDoS) attacks has become crucial in mitigating security threats. Instant identification improves network security by quickening warnings and removing corrupted IoT devices mostly from the network, stopping the botnet's spread. Inferred from 9 IoT devices affected by BNs, a Local-Global Best Bat Algorithm for Neural Networks (LGBA-NN) is suggested to choose either feature subsets as well as hyperparameters for rapid identification of BN attacks.

From balanced network traffic data, the Deep Recurrent Neural Network (DRNN) develops hierarchical feature description to execute discriminative classification with 99.50% precision and 99.75% recall [1]. Because swarm intelligent algorithms have reduced overall price, highly nonlinear optimization, robustness, easy installation, flexibility, and could even detect the Neris botnet accurately and quickly, through use of two swarm algorithms known as Elephant as well as Dolphin Herding algorithms for optimization for scalable and accurate detection of attacks on the Neris botnet is proposed [2].

Using IoT-specific network characteristics to acknowledge that the feature selection might yield excellent accuracy in the detection of DDoS in the IoT network traffic [3]. Home gateway routers would effectively detect DDoS attack sources on local IoT units employing minimal-cost machine learning methodologies. A software-defined technology-enabled artificial intelligence (AI) relied intrusion detection system (IDS) with two stages that identify attacks intelligently utilizing the Bat algorithm along with swarm division to identify traditional characteristics. Then, used Random Forest to classify by adjusting sample weights utilizing the weighted voting procedures, and the findings demonstrate that the improved intelligent algorithms identify more significant characteristics and perform significantly in flow classification [4].

In comparison to ML strategies for IoT privacy, a random neural network (RaNN) dependent prediction framework had been postulated with an accuracy of 99.20% with a 34.51 msec prediction time [5]. Cyber-attackers have indeed realized out how to utilize Artificial Intelligence (AI) and have already begun to employ adversarial AI to carry out cyber-attacks. This research consolidates information on IoT, AI and attacks with or against AI, as well as examines the correlation among these three discussions, with the goal of extensively introducing and summarizing related literature within those areas [6].

The feature extraction and the selection approaches for the IDS employing conventional neural networks (CNN) method using the newly created SI algorithm, Aquila optimizer (AQU), exhibit great performance employing numerous evaluation indicators [7]. The

CONTACT Swapna Thota swapnathota.rs@rediffmail.com Research Scholar, Noorul Islam Centre for Higher Education, Kumaracoil, Tamil Nadu 629180, India

outcomes of the investigation exhibited that the suggested scheme achieved effective performance for detecting the botnet by incorporating the CNN with a long short-term memory (CNN-LSTM) methodology to detect frequent and severe IoT on four types of security cameras [8].

The effectiveness of a Recurrent Neural Network (RNN) to accurately sort network traffic patterns of extremely imbalanced network traffic data, as well as the several layers of RNN stacked to grasp the representations in a hierarchies, evidenced that RNN model accomplished better generalization ability [9].

When the ML and DL methodologies are deployed in real-world IoT networks, a crucial portion of network traffic in the minority groups is incorrectly categorized, which can result in an invasion of confidentiality, loss of personal data, revenue loss when services and applications are inaccessible, and even death in essential IoT systems.

In this research, an effective DL dependent BN attack detection algorithm that increases detection rates. The following are the research's significant contributions:

1. For extremely unbalanced network traffic data, an effective DL-based botnet attack detection system known as CNN-POA is suggested, which creates additional minority samples to establish class balance.
2. The Bot-IoT dataset is utilized to train, validate and test CNN-POA models to categorize samples of network traffic into the normal class as well as 10 botnet attack classes.
3. The influence of class imbalance on the CNN-POA models' precision, accuracy, recall and F1 score was explored.
4. Experiments revealed that the suggested CNN-POA approach outperformed a number of current metaheuristic algorithms, with an accuracy of 99.5%.

The research is organized as follows: Section 2 explains the relevant works; Section 3 provided a full explanation of the CNN-POA algorithm as well as the model generation process; Section 4 addressed the outcomes; and Section 5 highlighted the key conclusions of the research.

2. Related works

In addition, for given severely unbalanced network traffic data, the Deep RNN model delivers poor classification outcomes and Synthetic Minority Oversampling Technique (SMOTE) and DRNN for developing a memory-saving DL algorithm, named LS-DRNN method reduced the dimensionality of network traffic features, and data storage memory requirements were lowered. [10]. Relied on the chosen optimal hyperparameters, such as the rectified linear unit (ReLU) activation function, 20 epochs, and Adam optimizer, a deep Bidirectional gated recurrent unit (BGRU) multi-class classifier is established [11], and the outcomes display that the suggested methodology could contribute to the establishment of an effective network IDS for IoT-enabled home automation services with a low false alarm rate (FAR).

Different ML methods were utilized to construct a model where data were trained using the IoT dataset, including Naive Bayes model, K-Nearest Neighbour (KNN). A reference point was used to choose the optimal method relying on accuracy and the area under the receiver operating characteristics curve (ROC AUC) score [12,13]. At the access router level, an IoT botnet detection and also isolation strategy is described that enables IoT devices highly attack resilient and helps to avoid IoT device compromise without the requirement for knowledge of technological administration in depth, making it practical for end users and clients [14].

The Particle Deep Framework (PDF) is an innovative network forensics structure that characterizes the investigation stages for recognizing attack characteristics in IoT networks. It has three innovative features: (1) retrieving data flows in network and validating their integrity (2) utilizing a Particle Swarm Optimization (PSO) algorithm to adjust deep DL variables; and (3) constructing a Deep NN relying on the PDF [15].

Using the Programming language Python and packages including Tensorflow, scikit-learn, an algorithm utilizing a DL method [16] revealed that a DL model could boost accuracy, enabling for the most successful mitigation of attacks on an IoT network. The concern of scalability is acknowledged, and a fog anomaly detection framework is suggested herein utilizing a vector convolutional deep learning (VCDL) framework that could efficiently manage scalable data compared to conventional centralized DL approaches in aspects of accuracy, precision and recall [17].

The hybrid CNN-LSTM algorithm [18], which might employ a real IoT dataset retrieved from an authentic system, which include benign as well as malicious patterns, was suggested to detect botnet attacks. Furthermore, software-defined networking (SDN), which is gradually displacing traditional networking, particularly in the IoT, recommends a detection of botnet technique relied on DL techniques, which was assessed on an innovative, SDN dataset and demonstrated higher accuracy in classification [19].

IoTBoT-IDS [20] is a quantitative learning-dependent structure for botnet detection that safeguards IoT-dependent smart networks from botnet attacks as well as aims to capture the normal behaviour of IoT networks by incorporating statistical learning relied techniques, such as the Correntropy model with the average accuracy of detection 99.2%, which is around 2–5%

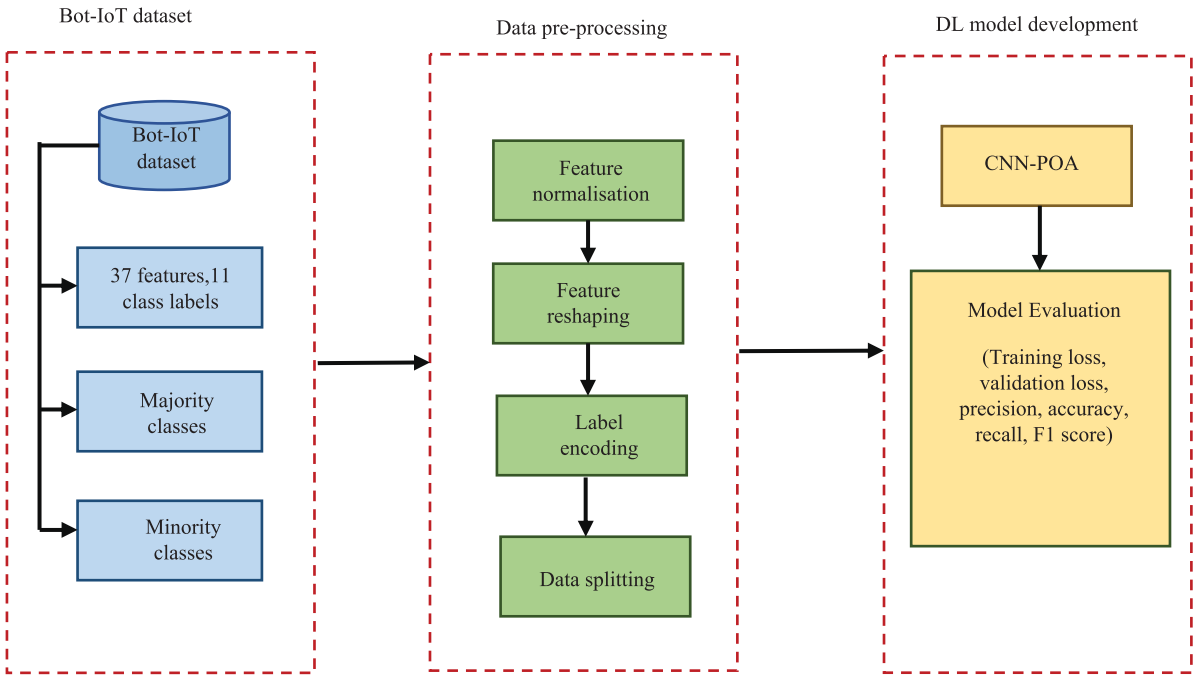

Figure 1. CNN-POA for detecting botnet attacks in IoT networks

higher compared to other IDS. CNN is utilized for data processing and even for feature optimization, while the Bidirectional LSTM (BiLSTM) were utilized for classification in the BiLSTM-CNN model. Among the three separate models, the RNN achieves the highest accuracy [21].

An innovative secure framework as well as attack detection methodology that employs a CNN to retrieve the precise feature description of data and furthermore classifies with 96% accuracy, employs a DL framework, which might then easily detect malicious gadgets [22]. A detection system predicated on anomalies that employs unsupervised DL techniques to identify IoT BN activities to measure its threat detection potential, as well as its FPR reduction and influence on the detection system [23]. A method for detecting potential botnets by examining the behaviours of network traffic from network packets that samples the packets over time and extracts the behavioural features [24–27].

By utilizing the performance of a Grey Wolf Optimization algorithm (GWO) and Genetic algorithm (GA) to optimize the hyperparameters, the recommended approach's principal feature is to identify Internet of Things botnet attacks generated by infected IoT endpoints [28,29].

3. Proposed POA-CNN algorithm

This section incorporates information on the network traffic data, data pre-processing, CNN and POA (Figure 1). Figure 1 incorporates information on the network traffic data, data pre-processing, CNN and POA. CNN's typical POA with a twist. When compared to traditional approaches, this method's unique pattern recognition characteristics include the ability to minimize data dimension, extract features sequentially, and categorize within a single network structure. However, excessively uneven network traffic data in the training set degrades the classification skills of cutting-edge ML and DL algorithms, particularly in classes with tiny sample counts.

3.1. Dataset

The Bot-IoT dataset includes 43 network traffic variables as well as three label categories for 11-class classification. Only 37 of such 43 features were revealed to be beneficial in detecting botnet attacks in IoT networks. An 11-class categorization scenario is explored to detect botnet attacks in a detailed manner. DH, DDH, DE, Norm and KE are considered minority classes in this research since they had fewer samples than the majority classes DDU, DDT, DT, SS, OSF and DU.

The Industry IoT (IIoT) smart-home equipment used to capture IIoT traffic samples was built into the Bot-IoT dataset in the Cyber Range Lab of UNSW Canberra Cyber. Smart IIoT goods encompass the thermostats, lighting that is turned on by motion, garages door openers, refrigerator including freezers and meteorological tracking technologies. There are two versions of the data accessible: the entire version, which has about 72 million entries, and the 10% variant, which has over 3.6 million recordings.

3.2. Data preprocessing

Feature normalization, feature reshaping, label encoding and data splitting are all part of the data preprocessing. First, using the min–max normalization technique

described in Equation (1), each one of the traffic feature of network readings was scaled towards a scale of 0 and 1:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where x represents a feature vector of network traffic while x_{min} and x_{max} are the minimum as well as maximum x values. A unit time step was represented by an additional dimension in the set of features. This varies the feature set dimension from $X \in R^{P \times q}$ to $X \in R^{P \times q \times 1}$, where p symbolizes the overall samples and q indicates the overall features. Integers 0–10 were employed to symbolize the numerical values of the 11 classes. To train and assess, the entire data were randomly partitioned into validation (20%), training (60%) and test sets (20%).

3.3. One-dimensional CNN model

CNN is a multi-layer NN, with each layer consisting of many two-dimensional planes. The elements' weighted average in the preceding layer obtains and activates each neurone's output.

(1) The input layer becomes the first layer.

There are five basic attributes of network traffic data, as the characteristic attribute is to acquire 5 time periods with 23 statistical characteristic data. Each sample's input dimension is assigned to the feature plane of 23×5 in this scenario. The outcome of the j^{th} neurone in the i^{th} feature plane in C1 layer specified by $C1_{ij}^{out}$ is given as follows:

$$C1_{ij}^{out} = F\left(\sum_{i=1}^{fl \times 5} W_t^{in} \times f\_raw_t^{in}\right) \quad (2)$$

where W_t^{in} denotes the i^{th} position weight of the convolution kernel, fl indicates the filter's length and $f\_raw_t^{in}$ signifies the characteristic plane position within input layer that corresponds towards the convolution kernel weight. Three forms of nonlinear activation functions tanh sigmoid and relu were highlighted by $F_i(.)(i = 1, 2, 3)$ are shown as follows:

$$F_1(x) = \text{sigmoid}(x) = 1/(1 + \exp^{-x}) \quad (3)$$

$$F_2(x) = \tanh(x) = (\exp^{x} - \exp^{-x})/(\exp^{x} + \exp^{-x}) \quad (4)$$

$$F_2(x) = \text{relu}(x) = \max(0, x) \quad (5)$$

The convolution layer extracts the original data's deeper and more sophisticated features. Furthermore, before the layer is joined to the next layer, the feature plane is compressed. Over fitting happens when there is strong precision occurring on training set but inadequate precision in the validation set during the training network process. 50% of the feature detectors stop operating while training is performed each time, which can increase the network's generalization ability, also referred as dropout. As a result, certain neurones don't really work with a particular probability, while each bunch of samples is learned, the neurone nodes functioning at every training cycle are distinct. Consequently, not all neurones could boost the network's generalization ability.

(2) The first full connection layer seems to be the second layer (F2 layer).

The connective layer controls the amount of neurones and their activation mode. In each batch training, the neurone present within full connection layer is interconnected to the neurones within the preceding layer. The output of the m^{th} neurone $full1_m^{out}$ in F2 layer is the following:

$$full1_m^{out} = F\left(\sum_{n=1}^{nkeep} W_{mn}^{keep} \times \text{keep}_n + b_m^{f2}\right) \quad (6)$$

where n_{keep} represents the volume of neurones following flattening and dropout, W_{mn}^{keep} resembles the connection weight among n^{th} neurone of the surviving working neurone following the preceding layer's processing and the m^{th} neurone of the F2 layer, keep_n signifies the n^{th} neurone of the left-over working neuron and b_m^{f2} specifies the value of the offset of b_m^{f2} neurone of the F2 layer.

(3) The second full connection layer seems to be the third layer (F3 layer). Initially, the quantity of F3 neurones as well as activation mode were specified. This layer's neurones interact in the similar manner as the preceding layer that the F2 layer does. The full connection layer's third and fourth layers are developed to learn the non-linear arrangement of features effectively, as well as the advanced features retrieved by the convolution layer utilizing weight connection.

In order to improve a network's stability, a batch normalization layer, also known as the batch norm, normalizes the information received from a convolutional layer, or FC layer. It usually resides among the convolutional layer and the activation layer. Since it features a minor regularization, the main benefits of this layer are an increase in the network's training speed and a decrease in overfitting.

The output layer constitutes the fourth layer. The amount of neurones within the layer is controlled by amount of categorization of network traffic data classification tasks, and the outcome of F3 in the last layer is transmitted to it. Softmax formulation is being utilized to classify many forms of network attacks, i.e. the final layer reports the probability of every category of traffic

data using the softmax formula:

$$\text{softmax}(y)_i = \exp^{y_i} / \sum_{i=1}^{kind} \exp^{y_i} \tag{7}$$

here y_i implies the i^{th} neurone's output value within the output layer, kind indicates the several forms of network attacks.

The prediction outcomes during training have the highest likelihood. The distance among the real probability distribution p(x) as well as the estimated probability distribution q(x) is measured using the cross-entropy loss function H(p,q).

$$H(p, q) = -\sum_x p(x)\log q(x) \tag{8}$$

Finally, the optimizer chosen throughout the model training cycle is considered.

The SGD offers the highest testing accuracy, however Adam causes a generalization gap in the meanwhile. As a consequence, although Adam accelerates the convergence speed beyond SGD by combining the first-order momentum along with the second-order momentum, its eventual convergence outcome is still inferior to SGD. SGD is subsequently chosen to optimize the model's training procedure for greater detection accuracy. The first-order momentum, set at 0.9, is introduced in order to combat the SGD's frequent oscillation in the neighbourhood's optimal gully, whereas the learning rate was constantly attenuated to enhance the training effect. As a consequence, the optimizer employs the SGD approach predicated on the evaluation presented above, whereby modifies the connection weight as well as bias value of neurones by the cross-entropy loss function value at a predetermined learning rate. It is possible to enhance the overall classification's accuracy through perpetual network training.

3.4. Proposed pelican optimization algorithm (POA)

3.4.1. Inspiration and behaviour of pelican during hunting

The pelican is a huge bird with a long mouth and a capacious bag in its throat for catching and swallowing prey. This bird likes to be in groupings of several hundred pelicans and survives in them. Pelicans have the following appearance: they weigh 2.75 to 15 kg and 1.07 to 1.82 m tall,. Pelicans consume primarily fish, with a few frogs, turtles and crustaceans sprinkled in for excellent measure; if those were hungry, they'll perhaps consume shrimp. Pelicans frequently hunt in groups. Pelicans plunge to their prey from a height of 10–20 metres after detecting its location. Of course, some species hunt their prey at lower elevations as well. Then it extend their wings on the water's surface to pressurize the fish into shallow water, where it can easily catch their prey. When a pelican catches a fish, a considerable volume of water that penetrates its beak, causing it to push forward prior consuming the fish to remove the extra water. Pelicans' hunting behaviour and approach is a sophisticated strategy that have turned these birds into excellent predators.

To update candidate solutions, the proposed POA replicates pelican behaviour and strategy when attacking and maybe even hunting prey. In two phases, this hunting method is simulated:

(i) Moving towards prey (exploration stage).
(ii) Winging on the water surface (exploitation stage).

3.4.2. Phase 1: Moving towards prey

In the initial stage, the pelicans locate the prey and migrate to that position. The exploration stage of the suggested POA in uncovering diverse regions of search space is enhanced by modelling this pelican's method. The crucial aspect of POA is that the prey's location in the search space is produced at random. POA's exploration ability in specific search of resolving a dilemma space is increased as a result of this. The foregoing notions, as well as the pelican's strategy for approaching the prey, are mathematically represented in Equation (9).

$$x_{i,j}^{P_1} = \begin{cases} x_{i,j} + \text{rand.}(p_j - I.x_{i,j}), & F_p < F_i; \\ x_{i,j} + \text{rand.}(x_{i,j} - p_j), & \text{else}, \end{cases} \tag{9}$$

where $x_{i,j}^{P_1}$ represents the i^{th} pelican's new status in the j^{th} dimension, I denotes random which equals to one or either two, p_j indicates the position of prey in the j^{th} dimension, and F_p symbolizes its objective function. The variable I specifies a number which might be either 1 or 2 at random. This parameter is arbitrarily chosen for each member as well as iteration. When this parameter's value is equivalent to two, a member experiences greater displacement, which could result to newer locations. As a result, parameter I has an impact on the POA's ability to precisely examine the search space.

The pelican's new position is acknowledged in the proposed POA if the objective function's value improves in that location. The algorithm is stopped from migrating towards non-optimal places in this sort of updating, known as effective updating. Equation is used to model this process (10).

$$X_i = \begin{cases} x_i^{P_1}, & F_i^{P_1} < F_i; \\ X_i, & else \end{cases} \tag{10}$$

where $x_i^{P_1}$ is the i^{th} pelican's new status and $F_i^{P_1}$ is premised on stage 1's objective function value.

3.4.3. Phase 2: Winging on the water surface

After reaching the water's surface, pelicans expand their wings to lift the fish higher, then gather the catch

in their pouch in throat. Pelicans catch more additional fish in the attacked region as a result of this tactic. The proposed POA converges to better places in the hunting region after modelling pelican behaviour. This technique improves POA's local search as well as exploitation capabilities. To converge to a better result, the algorithm should investigate the points near the pelican's location mathematically. Equation mathematically simulates the behaviour of pelicans throughout hunting (11).

$$x_{i,j}^{P_2} = x_{i,j} + R.\left(1 - \frac{t}{T}\right).(2.\text{rand} - 1).x_{i,j} \qquad (11)$$

where $x_{i,j}^{P_2}$ symbolizes i^{th} pelican's new status in the j^{th} dimension, R implies constant, which was equivalent to 0.2, R.(1 − t/T) implies the neighbourhood radius of $x_{i,j}$, The iteration counter seems to be t, and the maximal number of iterations was T. The coefficient "R.(1 − t/T)" signifies the radius of the population members' neighbourhood to explore locally around every member to discover a best solution. This coefficient has an effect on the exploitation of the POA in order to become nearer to the best global solution. The value of this coefficient is significant in the first iterations, so a broader area over each member is investigated. The "R•(1 t/T)" coefficient falls as the method replicates, leading in lesser radii of every member's neighbourhood. This permits us for scanning the region over each population member in smaller, more precise steps, permitting the POA to converging to solutions that are nearer to (and even precisely) the global (and also accurate) ideal depending on the utilization notion.

Effective updating was indeed employed at this stage to approve or discard the updated pelican position, which would be modelled in Equation (12).

$$X_i = \begin{cases} x_i^{P_2}, & F_i^{P_2} < F_i; \\ X_i, & else \end{cases} \qquad (12)$$

where $x_i^{P_2}$ is the i^{th} pelican's new status and $F_i^{P_2}$ indicates its objective function value dependent on stage 2.

After most members had been updated dependent through first as well as second phases, the best candidate solution would indeed be updated relying on the new population status and the values of the goal function. The algorithm moves on to the subsequent iteration, and the recommended POA based on Equations (9)–(12) is replayed until the entire execution is completed. Finally, as an optimal solution for the issue and the best candidate solution produced throughout the iterations of the algorithm is displayed. Algorithm 1 demonstrates its pseudo-code.

4. Results and discussion

The experiment was performed using an Intel Core i7-6700HQ processor (2.59 GHz along with 64 GB).

Algorithm 1. POA Pseudo-code

Start POA.
1. Input the optimization information of the problem.
2. Define the amount of iterations (T) and the size of the POA population (N).
3. Pelican's position were initialized and calculate the objective function.
4. For t = 1:T
5. Prey position is generated at random.
6. For i = 1:N
7. Phase 1: Moving towards prey.
8. For j = 1:m
9. Calculate the j^{th} dimension's new status with Equation (9).
10. End.
11. The j^{th} population member is updated using Equation (10).
12. Phase 2: Winging on the water surface.
13. For j = 1:m.
14. Evaluate new status of the j^{th} dimension using Equation (11).
15. End.
16. The j^{th} member of population were updated (7).
17. End.
18. best candidate solution were updated.
19.End.
20. Output optimal solution.
End POA.

MATLAB R2021b is the development framework of the Windows 10 operating system.

4.1. Performance metrics

The classification performance of the CNN-POA model with the NN model and the other ML/DL methodologies relied on training and validation loss, detection rate, accuracy, recall, precision as well as F1 score were represented by Equations (13)–(16):

Accuracy: This metric is used to calculate the rate at which data are classified correctly.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \qquad (13)$$

Precision: This metric describes the classifier's ability to predict normal data without conditions. It is defined as the number of TPs divided by the number of TPs, plus the number of FPs as follows.

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\% \qquad (14)$$

Recall: Recall is the ratio of the number of records correctly classified to the number of all corrected events and can be computed as follows

$$\text{Recall} = \frac{TP}{TP + FN} \times 100\% \qquad (15)$$

F1-Score: This is defined as the harmonic mean of recall and precision, which can be computed as follows.

$$F_1 = \frac{2 \times TP}{(2 \times TP) + FP + FN} \times 100\% \qquad (16)$$

where TP symbolizes the total count of BN attack samples accurately classified, FP exemplifies the number of normal samples misclassified, TN denotes the total

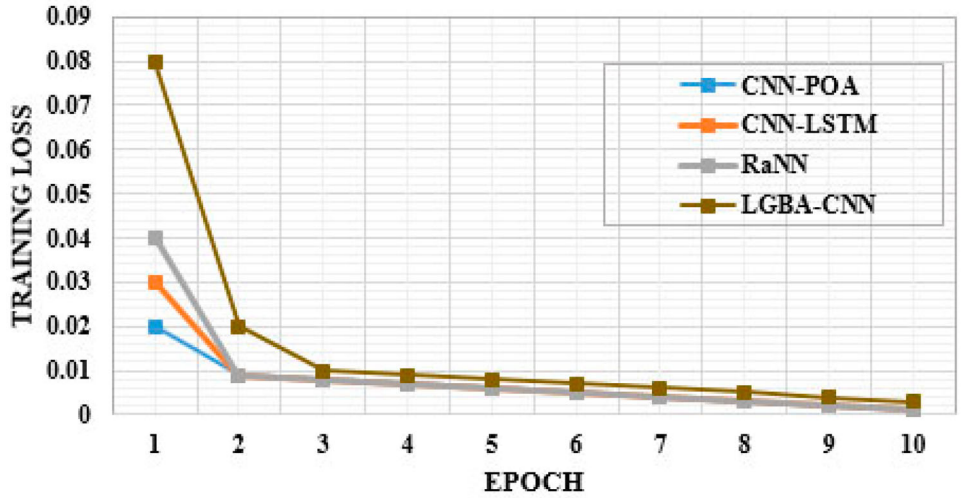

Figure 2. Training losses of the CNN-POA and other models.

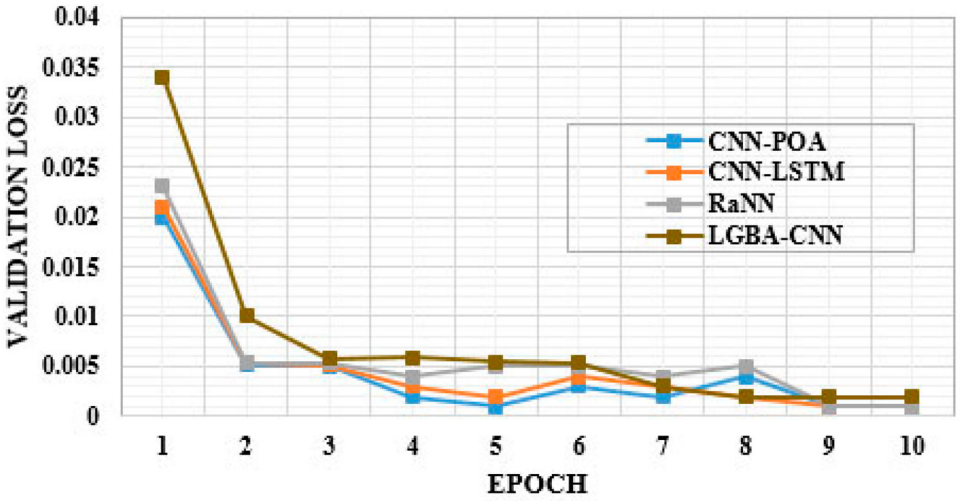

Figure 3. Validation losses of the CNN-POA and other models.

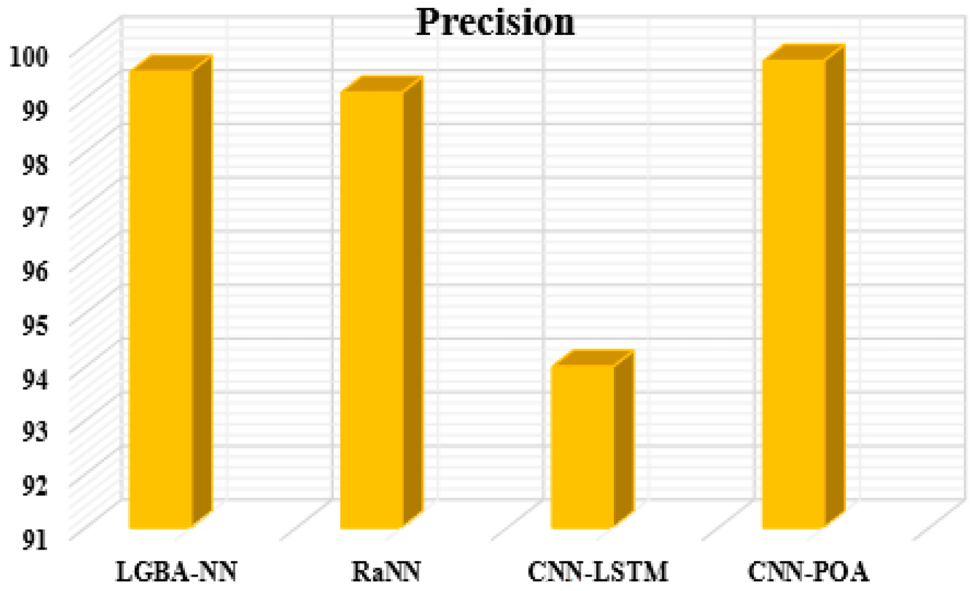

Figure 4. Comparison of precision in different models.

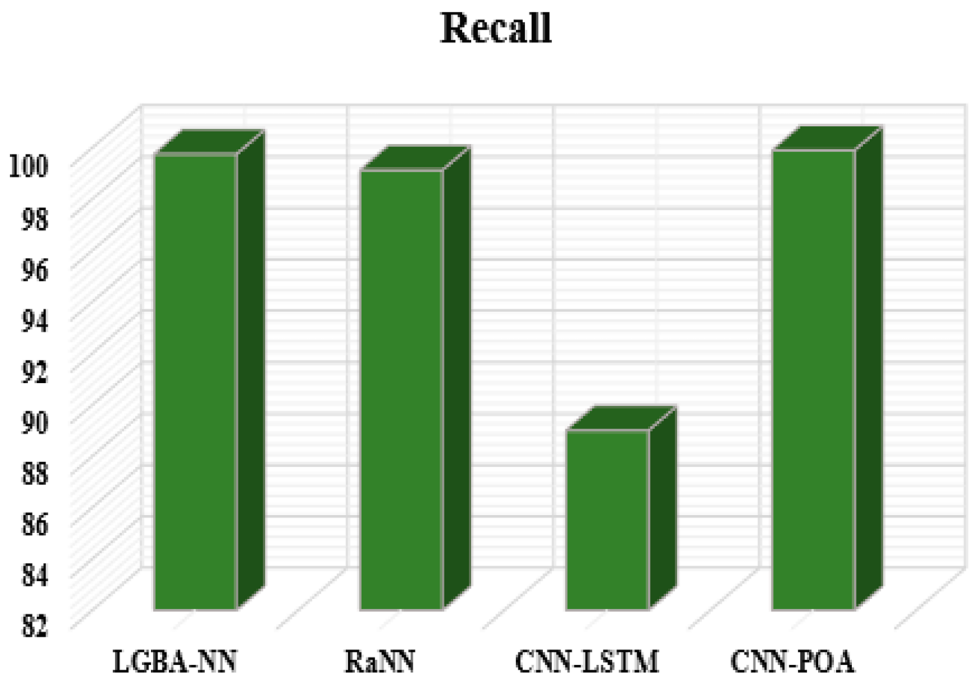

Figure 5. Recall comparison of different models.

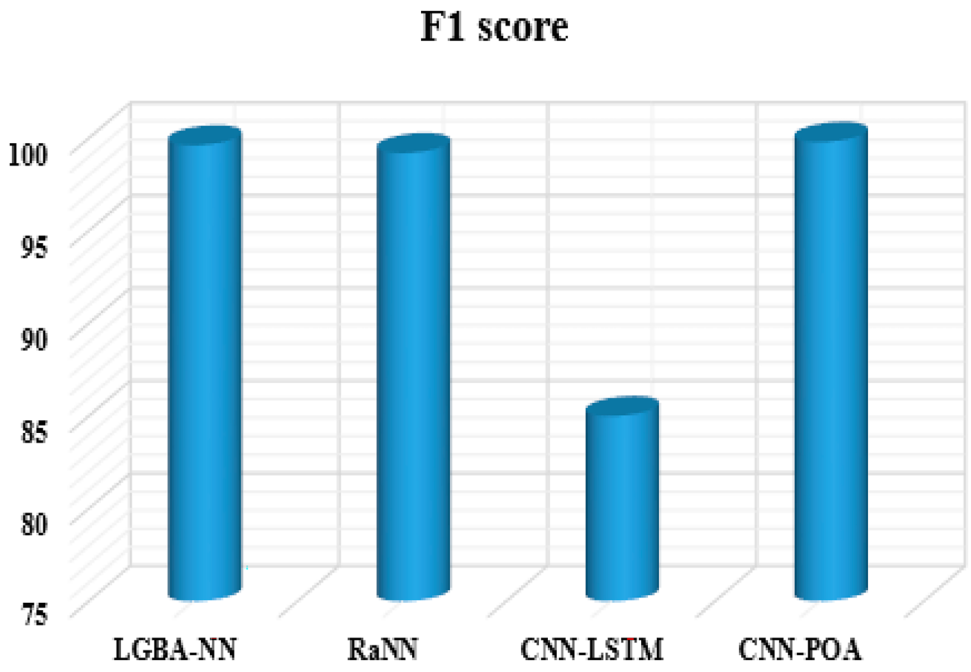

Figure 6. F1 score comparison of different models.

count of normal samples correctly categorized as normal traffic, and FN depicts the volume of attack samples misclassified to be normal traffic.

This subsection, analyse the cross-entropy losses of the CNN-POA, CNN-LSTM, RaNN and LGBA-CNN models during training and validation to evaluate their robustness/reliability against underfitting and overfitting, respectively.

Figure 2 shows the training losses of CNN-POA and compared with other models. As the number of epochs continued to increase between 1 and 10, the training losses decreased in all circumstances. When POA were utilized, however, the CNN model exhibited the minimal training losses. The CNN-LSTM model was decreased from 0.0804 to 0.0023 at the completion of the 10-epoch training, whereas the CNN-POA model was lowered from 0.0387 to 0.0008. The CNN-POA model had a 66.34% lower training loss than the conventional models at the end of the research. This implies that the SMOTE-DRNN model is robust against overfitting compared to the DRNN model.

Figure 3 depicts the validation losses of CNN-POA and other methodologies. However, proposed model have lower validation losses when POA was applied. The outcomes of validation losses generated by CNN-POA were very close towards CNN-LSTM.

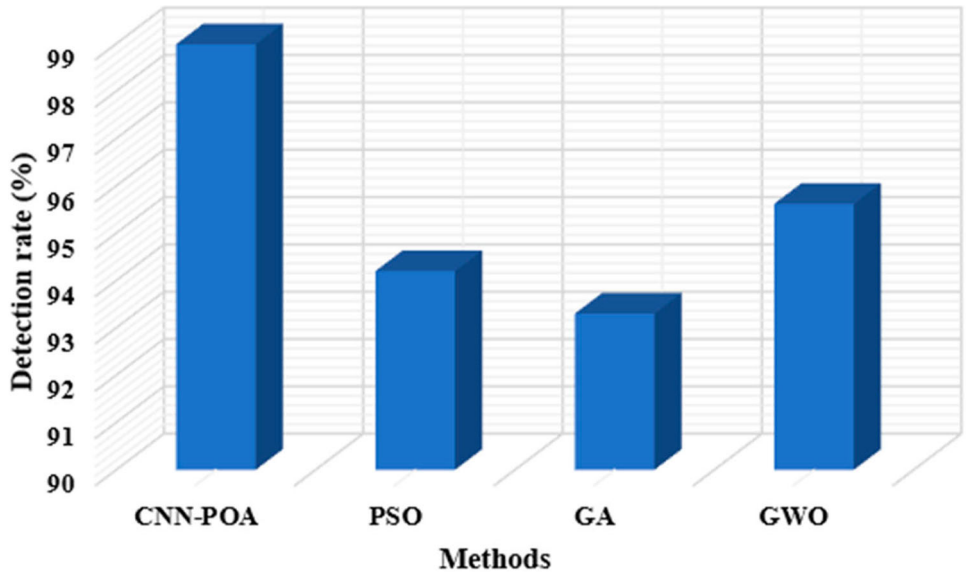

Figure 7. Detection rate comparison of different algorithms.

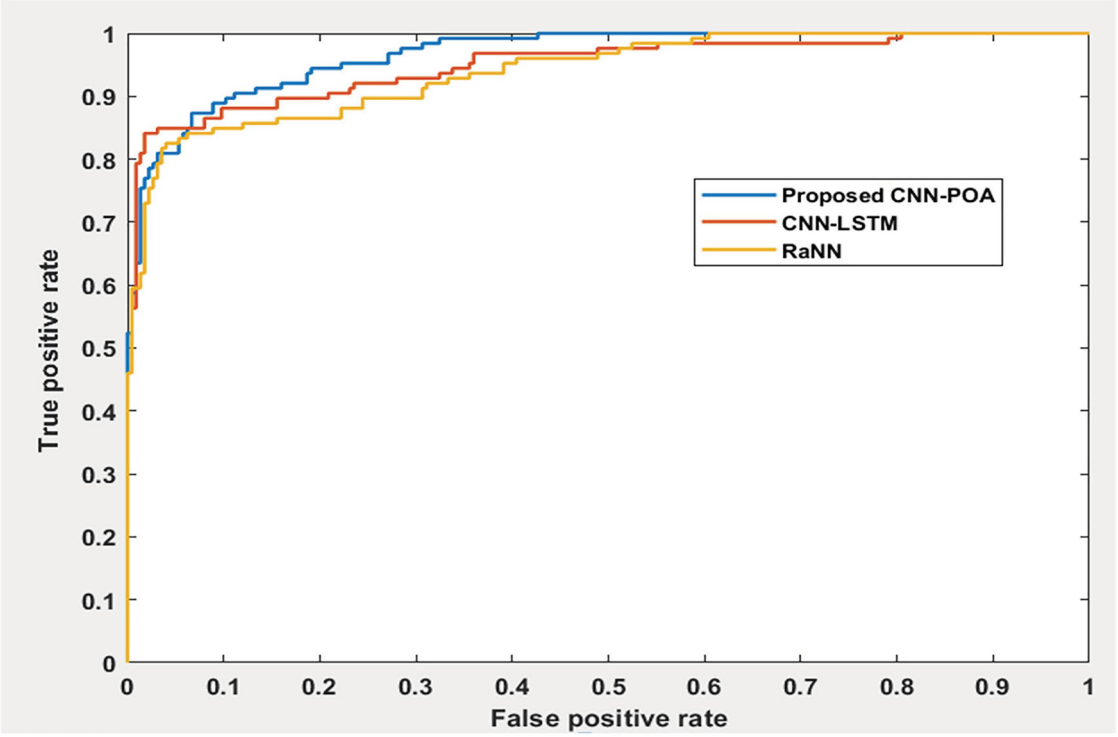

Figure 8. ROC comparison of different algorithms.

At the end of the 10-epoch validation, proposed CNN-POA achieved a validation loss of 0.0321 while that of CNN-LSTM was 0.0325. On the other hand, LGBA-CNN, RaNN produced relatively high validation losses respectively. As the overall epochs extended from 1 to 10, the validation losses dropped in all scenarios.

Precision as well as recall of different models are highlighted in Figures 4 and 5. By decreasing the feature space, the suggested model CNN-POA outperforms its predecessors in terms of precision and recall. Furthermore, the results reveal that the suggested model is adaptive and consistent in a variety of IoT contexts. Feature selection strategies, which might reduce the number of features for training phase, were overlooked by preceding research. Therefore, in this case the training time is reduced and employs minimal processing power.

F1-score for various models is highlighted in Figure 6. From the outcomes, it could be examined that the suggested CNN-POA ensures consistent F1-score compared to three relevant models. As a result, the suggested method could detect malicious patterns without relying on the environment. The recommended method's accuracy is contrasted with other approaches in Table 1.

Table 1. Comparison of accuracy with other methodologies.

References	Methodology	Accuracy (%)
Alharbi et al.[30]	LGBA-NN	85.2
Letteri et al.[19]	SDN	97
Ashraf et al.[20]	BMM	99.2
Hezam et al.[21]	BiLSTM-CNN	89.77
Proposed methodology	CNN-POA	99.5

The minimum loss directly indicates the CNN-POA approach maximizes the overall attack detection accuracy and detection rate shown in Figure 7. The proposed CNN-POA approach recognizes the network attacks with a maximum detection rate (98.99%).

Figure 8 depicts the suggested CNN-POA model's ROC AUC curve for the dataset. It demonstrates that the CNN-POA model efficiently differentiates botnet traffic from typical traffic having an AUC of approximately 99.5%. The botnet detection system should utilize the CNN-POA algorithm alternatively.

5. Conclusion

The nature of IoT applications, which include millions of sensors, results in tremendous amounts of data being generated. Furthermore, a crucial concern arises from these applications in terms of ensuring the confidentiality as well as protection of these data. The research provides the Convolutional Neural Network-Pelican Optimization System (CNN-POA), a DL-based botnet attack detection algorithm. For extremely unbalanced network traffic data, an effective DL-based botnet attack detection system CNN-POA is suggested, which creates additional minority samples to establish class balance. Adopting POA, which has a strong ability to locate viable regions that provide the optimal solution and improved CNN's performance. The influence of class imbalance on the CNN-POA models' precision, accuracy, recall and F1 score was explored. Experiments revealed that the suggested CNN-POA approach outperformed a number of current metaheuristic algorithms, with an accuracy of 99.5%.

Future work might involve techniques to simplify the algorithm and minimize its computational complexity. It is feasible to combine and analyse the existing network attack data sets as well as determine the most useful aspects in the context of the heterogeneous issues associated with various platforms.

Disclosure statement

No potential conflict of interest was reported by the author(s).

References

[1] Popoola SI, Adebisi B, Ande R, et al. smote-DRNN: a deep learning algorithm for botnet detection in the internet-of-things networks. Sensors. 2021;21(9):2985. doi:10.3390/s21092985

[2] Maisirreem a Kamal, Ibrahim Laheebm, Al-Alusi Abdulsattara. Dolphin and elephant herding optimization swarm intelligence algorithms used to detect Neris botnet. J Eng Sci Technol. 2020;15(5):2906–2923.

[3] Doshi R, Apthorpe N, Feamster N. Machine learning DDOS detection for consumer internet of things devices. 2018 IEEE Security and Privacy Workshops (SPW). 2018:29–35. IEEE.

[4] Li J, Zhao Z, Li R, et al. Ai-based two-stage intrusion detection for software defined IoT networks. IEEE Internet Things J. 2018;6(2):2093–2102. doi:10.1109/JIOT.2018.2883344

[5] Latif S, Zou Z, Idrees Z, et al. A novel attack detection scheme for the industrial internet of things using a lightweight random neural network. IEEE Access. 2020;8:89337–89350. doi:10.1109/ACCESS.2020.2994079

[6] Kuzlu M, Fair C, Guler O. Role of artificial intelligence in the Internet of Things (IoT) cybersecurity. Discover Internet Things. 2021;1(1):1–14. doi:10.1007/s43926-020-00001-4

[7] Fatani A, Dahou A, Al-Qaness MA, et al. Advanced feature extraction and selection approach using deep learning and Aquila optimizer for IoT intrusion detection system. Sensors. 2021;22(1):140. doi:10.3390/s22010140

[8] Alzahrani MY, Bamhdi AM. Hybrid deep-learning model to detect botnet attacks over internet of things environments. Soft Comput. 2022: 1–15.

[9] Popoola SI, Adebisi B, Hammoudeh M, et al. Stacked recurrent neural network for botnet detection in smart homes. Comput Electr Eng. 2021;92:107039. doi:10.1016/j.compeleceng.2021.107039

[10] Popoola SI, Adebisi B, Ande R, et al. Memory-efficient deep learning for botnet attack detection in IoT networks. Electronics. 2021;10(9):1104. doi:10.3390/electronics10091104

[11] Popoola SI, Ande R, Fatai KB, et al. Deep bidirectional gated recurrent unit for botnet detection in smart homes. In: Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics. Cham: Springer; 2021. p. 29–55.

[12] Alshamkhany M, Alshamkhany W, Mansour M, et al. Botnet attack detection using machine learning). 2020 14th International Conference on Innovations in Information Technology (IIT). 2020;203–208. IEEE.

[13] Pokhrel S, Abbas R, Aryal B. "IoT Security: Botnet detection in IoT using Machine learning." *arXiv preprint arXiv:2104.02231*. 2021.

[14] Dietz C, Castro RL, Steinberger J, et al. IoT-botnet detection and isolation by access routers. 2018 9th International Conference on the Network of the Future (NOF). 2018;88–95, IEEE.

[15] Koroniotis N, Moustafa N, Sitnikova E. A new network forensic framework based on deep learning for Internet of Things networks: a particle deep framework. Future Gener Comput Syst. 2020;110:91–106. doi:10.1016/j.future.2020.03.042

[16] Susilo B, Sari RF. Intrusion detection in IoT networks using deep learning algorithm. Information. 2020;11(5):279. doi:10.3390/info11050279

[17] Bhuvaneswari Amma NG, Selvakumar S. Anomaly detection framework for Internet of Things traffic using vector convolutional deep learning approach in fog environment. Future Gener Comput Syst. 2020;113:255–265. doi:10.1016/j.future.2020.07.020

[18] Alkahtani H, Aldhyani TH. Botnet attack detection by using CNN-LSTM model for Internet of Things applications. Secur Commun Networks. 2021;2021. doi:10.1155/2021/3806459

[19] Letteri I, Della Penna G, De Gasperis G. Security in the internet of things: botnet detection in software-defined networks by deep learning techniques. Int J High Perform Comput Networking. 2019;15(3–4):170–182. doi:10.1504/IJHPCN.2019.106095

[20] Ashraf J, Keshk M, Moustafa N, et al. IoTBoT-IDS: a novel statistical learning-enabled botnet detection framework for protecting networks of smart cities. Sustain Cities Soc. 2021;72:103041. doi:10.1016/j.scs.2021.103041

[21] Hezam AA, Mostafa SA, Baharum Z, et al. Combining deep learning models for enhancing the detection of botnet attacks in multiple sensors Internet of Things networks. JOIV: Int J Inform Visualization. 2021;5(4):380–387. doi:10.30630/joiv.5.4.733

[22] Sahu AK, Sharma S, Tanveer M, et al. Internet of Things attack detection using hybrid Deep Learning Model. Comput Commun. 2021;176:146–154. doi:10.1016/j.comcom.2021.05.024

[23] Apostol I, Preda M, Nila C, et al. IoT Botnet anomaly detection using unsupervised deep learning. Electronics. 2021;10(16):1876. doi:10.3390/electronics10161876

[24] Shi W-C, Sun H-M. DeepBot: a time-based botnet detection with deep learning. Soft Comput. 2020;24(21):16605–16616. doi:10.1007/s00500-020-04963-z

[25] Saif S, Yasmin N, Biswas S. Feature engineering based performance analysis of ML and DL algorithms for Botnet attack detection in IoMT. Int J Syst Assur Eng Manag. 2023;14(Suppl 1):512–522.

[26] Fraihat S, Makhadmeh S, Awad M, et al. Intrusion detection system for large-scale IoT NetFlow networks using machine learning with modified Arithmetic Optimization Algorithm. Internet Things. 2023: 100819. doi:10.1016/j.iot.2023.100819

[27] Hosseini F, Gharehchopogh FS, Masdari M. MOAEOSCA: an enhanced multi-objective hybrid artificial ecosystem-based optimization with sine cosine algorithm for feature selection in botnet detection in IoT. Multimed Tools Appl. 2023;82(9):13369–13399. doi:10.1007/s11042-022-13836-6

[28] Al Shorman A, Faris H, Aljarah I. Unsupervised intelligent system based on one class support vector machine and Grey Wolf optimization for IoT botnet detection. J Ambient Intell Humaniz Comput. 2020;11:2809–2825. doi:10.1007/s12652-019-01387-y

[29] Liu X, Du Y. Towards effective feature selection for IoT botnet attack detection using a genetic algorithm. Electronics. 2023;12(5):1260. doi:10.3390/electronics12051260

[30] Alharbi A, Alosaimi W, Alyami H, et al. Botnet attack detection using local global best bat algorithm for industrial internet of things. Electronics. 2021;10(11):1341. doi:10.3390/electronics10111341