International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V72I11P106

Volume 72 Issue 11, 45-54, November 2024

Original Article

Develop a Hybrid Improved Residual Attention with Efficient Net with Mosquito-Cuckoo Search Optimization to Detect the Attack during Network Traffic

A. Senthilkumar[1], L. Kathirvelkumaran[2], K. Kavitha[3], Nithyanantham Sampathkumar[4], S. Sureshkumar[5]

[1]*Department of Computer Science with Data Analytics, Sri Ramakrishna College of Arts and Science, Coimbatore, Tamilnadu, India.*
[2]*Department of Computer Science with Data Analytics, Kongunadu Arts and Science College, Coimbatore, Tamilnadu, India.*
[3]*Department of Electrical and Electronics Engineering, Annamalai University, Annamalainagar, Tamilnadu, India.*
[4]*Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education (Deemed to be University), Virudhunagar, Tamilnadu, India.*
[5]*Department of Computer Science and Engineering, P.A. College of Engineering and Technology, Tamilnadu, India.*

[2]*Corresponding Author : Kathirvelkumaran@gmail.com*

Received: 03 January 2024 Revised: 12 July 2024 Accepted: 14 November 2024 Published: 25 November 2025

Abstract *- Network attacks encompass all forms of unapproved entry into a network, encompassing any endeavor to damage and interfere with the network, usually resulting in serious consequences. The internet facilitates communication and connection, but attackers who aim to breach and harm network security and connections may also violate and endanger the integrity and confidentiality of these relationships and data transfers. The evolution of various network assaults and growth in data interchange between the devices brings the need to safeguard networks and computing devices. The hybrid Improved Residual Attention with EfficientNet (IRAEN) with Mosquito-Cuckoo Search Optimization (MCSO) is proposed for efficient feature extraction from the network traffic. The attention mechanism focuses on something in particular and notes its specific importance. IRAEN-MCSO solves the problem by extracting more features and classifying the attacks accurately. The proposed IRAEN-MCSO model obtains an accuracy of 99.63%, which is higher than the other algorithms such as Gradient-Boost Decision Tree (GBDT), Multilayer Perceptron (MLP), Recurrent Neural Network (RNN), AE-RL and LSTM. Also, the specificity values 99.59%, precision 97.63%, recall 97.78%, and F1 score 97.97% are more efficient than the other existing algorithms.*

Keywords *- Improved Residual Attention with EfficientNet, Mosquito-Cuckoo Search Optimization, Transfer learning, Feature extraction, Security, Network traffic detection, Performance measures, Machine learning.*

1. Introduction

Network attacks are attempts to infiltrate a company's network without permission to obtain information or perform unlawful activities. The possible sources of the attack are both external and internal [1]. Networking methods have been used for decades to improve the transfer and exchange of data. Numerous new services are now feasible as a result of their continuing advances. The emergence of cloud computing methods has led to significant advancements in network technology [2]. Many users can utilize various services and applications through the Internet and calculate and store resources when needed. These benefits include flexibility, low administrative effort, economical high accessibility, reliability, resource utilization, and efficiency [3]. The military, finance, and e-commerce sectors are among the industries with the greatest number of internet-connected systems, which makes them vulnerable to network attacks and increases risk and damage [4]. Attackers can overload the network or target device with traffic until it burns out or stops responding. Emails, webpages, and online bank accounts are among the impacted services. Denial of Service (DoS) assaults can be launched anywhere [5]. Intrusion Detection Systems (IDS) could identify malicious behavior by gathering network behavior, examining security logs, and other data on the network and among connected devices [6]. An IDS is a tool that can safeguard a system in real-time by looking for an abnormal activity that violates the system safety rules and indicators that the system is under assault. IDS are used to find potential cyberattacks that could compromise information systems [7]. The data collected from the root network nodes comprises traffic arriving and departing from many network elements. This amount of traffic capturing could make it easy to identify Internet-related assaults by focusing on as many machines linked to the network as possible [8]. However,

some attacks might not be noticed as malicious activity often occurs horizontally except for the root nodes. Network intrusion detection systems that use root-level capture shall be able to analyze enormous amounts of data at very fast speeds to avoid traffic rejection and prevent detecting delays [9]. In addition, since they cover extensive network interconnections, noise and disturbances can easily be caused if new equipment is added to every network segment. Machine Learning (ML) requires a crucial stage called feature engineering. Along with ML or DL approaches, feature engineering has been employed in feature selection and extraction. Because it involves identifying patterns, highlighting important information, and bringing in a domain expert, it is a fantastic technique to improve predictive algorithms [10].

Consequently, a lack of labeled information is the main obstacle to supervised learning. On the other hand, unsupervised learning makes it much simpler to obtain training data by extracting useful feature data from unlabelled data [11]. However, untrained training techniques do worse at detection than supervised training methods. Some of these techniques have been researched for years, and their approach is well-developed. In addition to the detection effect, they also address real-world issues, including data management and detection effectiveness [12]. Deep Learning (DL) is superior to conventional ML due to developments in advanced techniques such as deep Neural Networks and hardware resources like graphics processing units GPU. The enormous volume of training data, as evidenced by Google and Facebook, can also be partly to blame for today's progress in DL [13]. The goal advantage of DL was its ability to be applied even in networks with few resources because it does not require human feature engineering, unsupervised pre-training, or compression abilities. It implies that DL's capacity for self-learning produces greater accuracy and quicker processing [14].

1.1. Problem Statement

The repetitive data elimination and management of missing values are not considered. Redundancy in the dataset makes the learning algorithms preferential towards frequent records and show no preference towards rare records. Missing values in the dataset result in incorrect classification owing to ineffective input observations. Few attacks like U2L and R2L are not detected efficiently since this one-dimensional distance-based feature representation cannot better represent the pattern these attacks use [15].

1.2. Motivation

Systems located inside the resource's sensor node parameters strength and also sufficiently strong to deal with attacks must be present. Intrusion detection is one of these kinds of defense that sensor networks use, and they are capable of detecting unnamed attacks and discovering ways of fighting against them. Researchers have seen that IDS holds huge compatibility in sensor networks [16]. Hence, intrusion detection has a significant position in the research field for experts. Therefore, acquaintance with this potential research field will benefit the research experts. Considering this, this technical work is designed to identify the intrusion activities occurring in the network [17].

1.3. Main Contribution

- Creating an algorithm to identify and categorize network assaults.
- Examining the features of the network to identify the assault in the network.
- Analyzing the efficiency of the DL methods in the detection of attacks.
- Evaluate the accuracy of the proposed DL algorithms with other DL methods.
- To realize the limited computation complexity, IDS affects intrusion detection results.
- To present the attack-compatible IDS to make the attack detection rate better.

2. Related Works

Various ML methods that could be applied to IDS were presented, including Decision Tree (DT), Random Forest (RF), Naive Bayes (NB), Logistic Regression (LR), Support Vector Machine (SVM), Artificial Neural Network (ANN), and K-Nearest Neighbor (KNN). The Bot-IoT dataset examines machine-learning techniques for binary and multi-class categorization [18]. The accuracy of RF in an HTTP Distributed Denial-of-Service (DDoS) attack is 99%. However, KNN achieves 99% accuracy in multi-class categorization. Compared to the two intrusion detection systems, Apriori employs a data mining association rule approach, while Support Vector Machine utilizes ML techniques. The Network Security Laboratory Knowledge Discovery and Data Mining (NSL-KDD) and University of New South Wales–NB 2015 (UNSW-NB15) database are the two datasets used to evaluate the two technologies' efficacy [19]. [20] Created a novel method for detecting malicious network traffic using an artificial neural network technology that might be applied to deep packet intrusion detection based on inspection. Two techniques exist for identifying DDoS attacks, and an SDN methodology was proposed. [21] Four features were first examined to assess the strength of the attack when it came to the SDN controller due to a DDoS attack.

The upgraded ML method based on K-Nearest Neighbour (KNN) was used in addition to previous methods to determine the DDoS attack. For the first time, a revolutionary concept for identifying DDoS attacks was introduced and explained in terms of attack degree. [22] has proposed an attack prevention system, CSANN, which combines a Cuckoo Search (CS) technique and an Artificial Neural Network (ANN) method to recognize DDoS assaults and enable greater caution on the server side. CS was applied in a way that was motivated by nature to optimize the attributes of the attacker and the cloud

user. [23] the ANN framework was then equipped with these enhanced features. The taught and tested features were compared using the trained attributes kept in the database throughout the testing process. This generated results that ordinary cloud users and attackers could examine. [24] have developed an ML system to identify if an assault has damaged a wireless sensor network. If an assault is discovered, this model must determine what kind of attack it is. Collected the data, cleaned it, and performed the required pre-processing to access the actual data while carrying out such attacks on a restricted network. They were able to create their database as a result. Those data afterwards evaluate an ML model that predicts the possibility of a network attack. [25] the precision of the model has greatly approached the expected output, with 97 percent precision. Next, an adversarial sample of encrypted traffic was generated using Deep Convolution Deep Q Network (DQN) and Generative Adversarial Networks (DCGAN) reinforcement learning. The problem of uneven, insufficient, or little samples was thus handled. The model's accuracy was a high 99.94%. [26] DL techniques to create a mouse behavior-based user authentication paradigm that might be applied to track and detect insider authentication.

First, the network traffic data is pre-processed, and then CNN is used to simulate the data. [27] CNN automatically collects sample characteristics, transforms low-level information about incursion traffic into high-level characteristics, and uses the Random Gradient Descent method to improve the network's variables to converge the model. The KDDTest + results indicate that its identification accuracy is 0.51% and 8.82% greater than DBN and LeNet-5. [28] proposed a unique way to project data from wireless network assaults into a grid-based image that could be used to feed the EfficientNet model of a Convolutional Neural Network (CNN). Specify the precise sequence in which the attribute values are to be placed in a grid so that it can be imaged. It is intended for a lightweight, accurate IDS module used in IoT networks by fusing EfficientNet with the most important subset of properties. A 0.11% false-positive rate and 99.91% F1 score provided the best results. Proposed a two-phase EfficientNet Convolution neural network-based framework to determine whether the identified user sample is authentic or faked [29]. The proposed system is developed on several datasets of Iris biometric samples using an EfficientNet Convolution Neural Network [30][31] to distinguish between a fake and real iris biometric sample.

3. Proposed System

3.1. Dataset

The CICIDS 2023 dataset, sourced from the Canadian Institute for Cybersecurity, comprises both benign and existing common attack traffic, reflecting a real-world network ecosystem. The database was captured on an actual network with a divided architecture. Ten victim machines are present in the second one, which is being assaulted by the attacks of the first four machines.

Table 1. CICIDS 2023 dataset

Day	Dataset Type	Size
Mon	N	12 GB
Tue	N + SFTP + F+ SSH	12 GB
Wed	N + DoS+ HA	14 GB
Thur	N + XSS + WA + Infiltration	7.9 GB
Fri	N + Botnet + PortScan + DDoS	8.5 GB

N- Normal; F-Force; HA- Heart bleed Attacks; WA - Web Attack

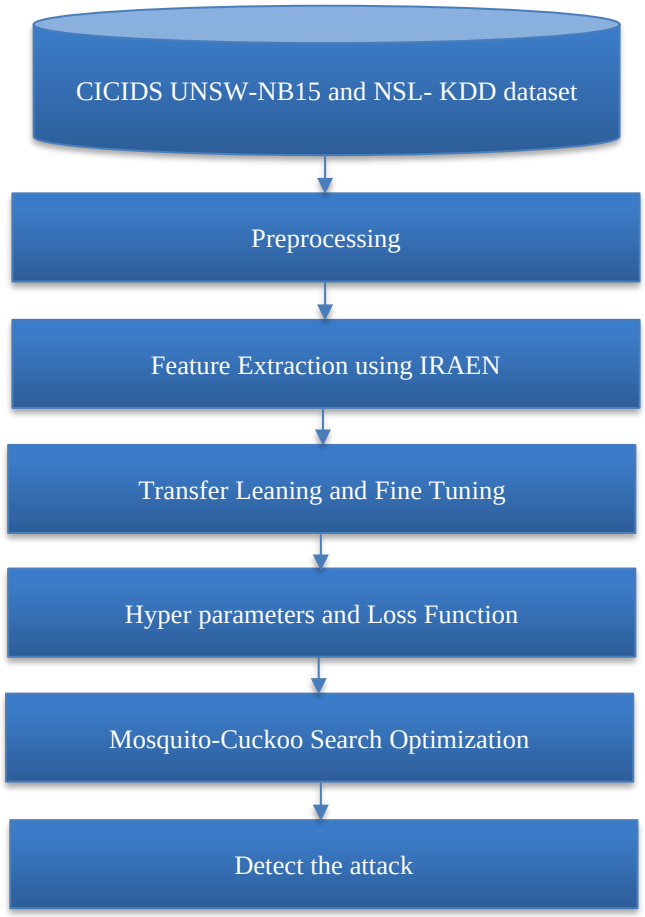

Fig. 1 Proposed architecture

A total of 84 characteristics in CSV files are included, with 50 GB of raw data in PCAP files. Over five days, network traffic was captured, yielding 2,830,743 instances in total. For a total of 5 days, July 3, 2023, at 9 a.m., was the start of the data-gathering period, which concluded on July 7, 2023, at 5 p.m. The typical day is Monday, when there is only moderate traffic. As indicated in Table 1, the executed assaults consist of DoS, Botnet, Brute Force FTP, Heartbleed, DDoS, Web Attack, Infiltration, and Brute Force SSH. The advantage of this database is that it is offered in two formats: a list of network flow parameters extracted from the Ethernet frames in CSV files and a record of Ethernet frames in Packet CaPture (PCAP) files. While the goal of CICIDS 2023 was to overcome the shortcomings of earlier IDS datasets, relatively few investigations have been performed to find any

difficulties. After a thorough assessment of the data set, four deficiencies varied in severity were identified. To use the CICIDS 2023 data, users have to create all of their files since they are spread over several files. This will result in a lot of difficult-to-process data. These three issues are not, as is usual in ML, affecting the performance of intrusion detection. It also indicates that there are a lot of missing values in the samples and that the classes are quite unbalanced. Figure 1 shows the step-by-step procedure of the proposed system.

3.2. Pre-Processing

A CSV file containing partial traffic data and the original dataset in PCAP format is provided by the CICIDS 2023 dataset. As seen in Figure 2, time division, matrix generation, PKL file labeling, PKL file generation, traffic segmentation, and one-hot encoding are carried out to convert the original data into the algorithm's input format. Time separation is the process of dividing the original PCAP file into relevant period PCAP files based on the kind and timing of the attack.

- Traffic Segmentation: The CICIDS 2023 dataset is formatted as a single daily PCAP file. The large amount of information in these PCAP files makes it difficult to train the machine.
- Generate and label the PKL file: The Python pickle tool bundled the traffic to speed up the data reading. Following traffic division, many sessions are created and saved to a PCAP file. Then, for every session, a label relating to the type of assault was generated. There are multiple data flows inside each session, and each data flow includes a packet.
- Matrix generation - The next stage is to standardize the duration of each session since the model's input needs to have a set length.
- One_hot encoding - A method to gather data for an algorithm and obtain a better forecast is to use one hot encoding. DL techniques are typically employed when using sequential classification issues. In essence, binary vectors represent category data in one-hot encoding. One-hot encoding processes the data and turns qualitative qualities into quantitative information so that the model can be learned and classified efficiently.

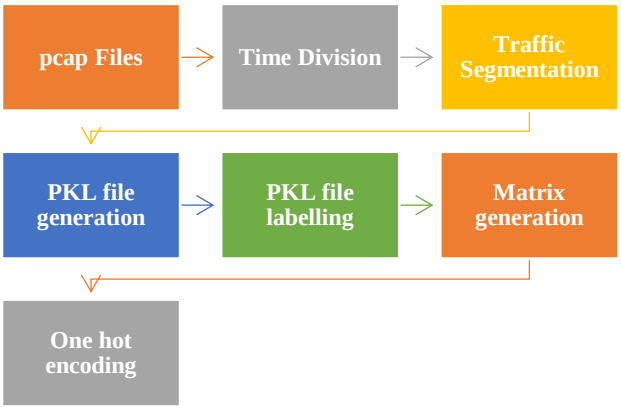

Fig. 2 Data Pre-processing steps

A traffic image is created from the data packets extracted from the pre-processed data. A two-dimensional matrix containing some bit value from a network communication packet is called a "traffic image".Created a traffic image for further processing by combining the x and y bytes from a packet's header and material. The input vector is the network traffic following pre-processing and one-hot coding.

3.3. Feature Extraction using IRAEN

Features are important in the realm of image processing. Before features are obtained, the sampled image is subjected to various image pre-processing techniques, such as scaling, thresholding, normalizing, binarization, etc. Subsequently, feature extraction strategies extract useful characteristics for image recognition and classification. Character identification is one of the many image-processing applications that benefit from feature extraction approaches. Feature extraction provides a formal approach that describes the relevant shape information in a pattern, making identifying it easier. A technique's input data will be converted into a smaller group of characteristics (also known as a characteristics vector) when it is deemed redundant (lots of data, but not much information) and too big to handle. The structure of a feature extraction network is shown in this section. First, the image data is fed into EN, an ML method that was transfer learning pre-trained on the ImageNet dataset. Second, to create IRAEN, a new function module called RA block was introduced after EfficientNet.

Any CNN model feature map can be used with the attention layers. Assuming that the 2D map size was N×N×C and the number of channels was C, we can say that the size of the input feature map is N×N×C. The attention module first uses two consecutive convolutional layers to squeeze the feature map until it is N×N×16 in size. After that, it learns N×N weights using a sigmoid activation function and a locally connected 2D layer. The weights are then replicated over the channel dimensional times using a second convolutional layer. It is significant to notice that a linear activation function follows this layer, meaning that a large range of values can be assigned to the weights. Nevertheless, divide the outcome by the average weight vector to scale the new feature map after averaging it with attention into a single feature vector of length C. This guarantees that the resultant process functions as a weighted average, maintaining values similar to the initial feature vectors. First, we intended to utilize the EffecientNet-B7 model version since larger EfficientNet models yield higher accuracy than smaller models. However, given limited computational resources, chose the EffecientNet-B0 variation as a fair balance between parameter count and accuracy. A new compound scaling technique called EfficientNet is presented. It uses an elements coefficient to uniformly scale the network depth, width, and resolution to increase the model's accuracy when its variables and computation volume are maximized. Equation (1) is the specific formula, and the greatest accuracy of the model is shown by.

$$N(r,w) = \odot \hat{f}_x^{di\hat{L}_x}\left(I_{[r\times\hat{h}_{x,r}\times\hat{w}_{x,w}\times\hat{c}_x]}\right) \quad (1)$$

X = 1,2,3,…,s

Equation (1) $\odot$ uses predetermined parameters in the baseline network to describe the classification network, convolution operation, and number of convolutional layers, $\hat{f}_x, \hat{L}_x, \hat{h}_x, \hat{W}_x, \hat{c}_x$represented by N. The coefficients for scaling the network's width, depth, and resolution are denoted by w, d, and r, respectively, and their computations are the following:

$$depth: d = \alpha^{\emptyset} \quad (2)$$

$$width: w = \beta^{\emptyset} \quad (3)$$

$$resolution: r = \gamma^{\emptyset} \quad (4)$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

EfficientNet's activation parameter is modeled as a Swish activation operation, which is a Self-Gated activation function denoted by Equation (5):

$$swish(I) = I\sigma(\beta I) \quad (5)$$

Where σ is the logistic function, and β might have fixed hyperparameters or be learned. The Swish activation characteristic shares some excellent performance advantages with the ReLU and Leaky ReLU functions because of their similar shapes. However, its activation process is smoother than that of ReLU and Leaky ReLU.

3.4. Transfer Learning and Fine Tuning

This part shows how the model is created and developed. Initially, a pre-trained EfficientNet-B0 base method evaluated on the ImageNet database has been imported via the Keras library. The base model quickly improved its image recognition capabilities by utilizing its characteristics thanks to the pre-initialized weights from ImageNet. Weights acquired during training on the ImageNet dataset comprise characteristics that can help identify forms, edges, and other crucial elements needed for classifying images.

This approach reduced labor requirements and expedited the procedure. With over 14 million photos and 1000 distinct classes, the EfficientNet-B0 basic method was evaluated using ImageNet data. As a result, fine-tuning was necessary because the existing EfficientNet-B0 structure can be used for the task selected. The base model's layers were frozen before the traffic image training data to fine-tune the proposed end layers. With this technique, they could avoid the feature selection capacity from being overridden throughout the training rounds and preserve the information contained in the weights that the extraction layers developing with the ImageNet dataset yielded. The resultant model was then validated using the test data.

Table 2. Hyperparameters and values

Hyperparameters	Values
Epochs	100
Batch Size	32
Learning Size	0.08
Optimizer	Adam
Activation Operation	Swish

3.5. Hyperparameters and Loss Function

This section explains the hyperparameter and loss function parameters chosen to provide useful outcomes. Equation (6), where y represents binary values of 0 or 1, and p represents the probability, shows the loss measurement for the binary categorization.

$$CE = -(jlog(p) + (1-j)\log(1-p)) \quad (6)$$

The optimal loss reduction during training is accomplished by using the Adam optimizer. An adaptive gradient descent function is used in this optimization strategy to help the weights approach the local minima faster. Choose Adam over other optimization techniques like SGD or RMSProp because of its rapid learning curve, efficient memory utilization, and ease of use. Adam currently submitted some really good DL applications. The hyperparameter values are displayed in Table 2, along with a minor Learning Rate (LR) tuned to work in tandem with the other hyperparameters. Adam worked more productively and swiftly to achieve quicker integration. Able to impart information over the network without exhausting computing memory, mainly to the batch size of 32. In addition, each model has defined durations to see its response after 100 epochs.

3.6. Improved Residual Attention with EfficientNet with Mosquito–Cuckoo Search Optimization (IRAEN-MCSO)

Once the feature reduction is made, optimal feature selection is performed using the Hybrid IRAEN-MCSO to choose the input dataset's best characteristics. Based on studies on the social behavior of mosquitoes, the IRAEN-MCSO is a distributed or parallel algorithm that uses meta-heuristics. It was modeled after certain cuckoo species compulsory brood parasitism, in which the cuckoos deposit their eggs in the nests of host birds from other species. Some host birds may directly combat the intrusive cuckoos. The hybridized IRAEN-MCSO will result in the optimally selected features, which are then forwarded to the classifier for accurate attack detection. A kind of metaheuristic algorithm inspired by nature, CS was developed in response to the demanding reproductive tactics of certain cuckoo species. The final rule necessitates giving the program some new, random solutions. Tree model rules are delimited. It may be represented by the host nests rubbing against one another to develop new nests. The fundamental phases of the CS may be identified by studying the known cuckoo breeding habits. The functional form in the D-dimensional space represents the optimization issue that has to be addressed. Identifying the

new nest population $X_{i(t+1)}$ is one of the primary tasks in the CS. Additionally, the following is how Levy's battle is used to gain the new nests:

$$I_{xy}(t+1) = I_{xy}(t) + \sigma \oplus Levy(\lambda) \quad (7)$$

Where λ is a Levy flight variable, ⊕ denotes the entry-wise multiplication procedure, and α is the step size. The entry-wise output is comparable to the one employed in mosquitoes. However, the randomized walking through Levy combat is more effective in traversing the searching area since its step length is significantly greater over time. Therefore, as power distribution is frequently connected with specific scale-free qualities, the Levy fight may display personality and cyclic activity in combat patterns. Of course, the random seeking method used by the traditional mosquito algorithm is supposed to be replaced by the Levy fight to enhance mosquito efficiency.

Pseudo Code: Mosquito-Cuckoo Search Optimization (MCSO)

Initialize the population of mosquitoes (solutions) N

```
Initialize parameters:
  - Max iterations (Iter_max)
  - Step size (α)
  - Discovery rate of new nests (Pa)
  - Probability of Mosquitoes' following behavior (P_mosquito)
  - Probability of Cuckoo laying eggs (P_cuckoo)
Evaluate the fitness of each mosquito (solution)
while (iter < Iter_max):
  for each mosquito, i:
    # Cuckoo Search Phase (Exploration)
    Generate a new solution (nest) using Lévy flight:
      new_solution = current_solution + α * Lévy(step)
        Evaluate the fitness of new_solution
    if (fitness(new_solution) > fitness(current_solution)):
      Replace current_solution with new_solution
        # Mosquito Swarming Behavior (Exploitation)
    if (rand() < P_mosquito):
      Identify the best mosquito (leader) in the swarm
      For each mosquito j in the swarm:
        Move towards the leader with random variation:
          new_position = mosquito_j + rand() * (leader - mosquito_j)
        Evaluate the fitness of new_position
        if (fitness(new_position) > fitness(mosquito_j)):
          Replace mosquito_j with new_position
        # Discovery of New Nests (Pa)
    if (rand() < Pa):
      Generate new solutions randomly
      Evaluate the fitness of these solutions
      Replace worst solutions with new solutions if they are better
  Select the best solution from the population
  Update iteration counter iter = iter + 1
Return the best solution found
```

Algorithm IRAEN-MCSO

Step 1: Initialized previous values for parameters include the minimum and maximum weights, as well as the population size of the swarm (N)(W_{min}, W_{max}) and acceleration coefficients (c_1, c_2).

Step 2: Each cuckoo's lower and higher boundaries, as well as its pace, are determined in distinct neighbourhoods.

Step 3: Within the defined space, a random initialization process is used to start the first generation of cuckoos, $I_x = [i_x, \dots, i_{xD}], x = 1$

Step 4: The fitness operation of each cuckoo $f(I_x) = f_x$ is evaluated. The optimal location Pbest and the swarm's overall optimal location Gbest(t) are determined.

Step 5: In terms of the fitness operation, a friction pa of the lowest performing cuckoo is selected. Within the designated search area, the chosen cuckoos should be abandoned and replaced with new ones produced randomly (similar to CS).

Step 6: An update is made to the inertia weight w.

$$w = w_{max} - \frac{w_{max} - w_{min}}{T} \times t \quad (8)$$

Step 7: The number of iterations is increased (t = t + 1). As with other metaheuristic algorithms, the ultimate fitness function value or the process that ends is always the termination condition. Proceed to Step 4 if the termination requirement is not satisfied. Otherwise, return I_{best} the final solution to the optimization issue.

Step 8: Display the outcomes of the optimization. Plotting the optimal adaptation parameter values individually while making improvements.

4. Results and Discussion

The Python implementation of the proposed model was built on top of the Keras and TensorFlow frameworks. The performance is assessed using three datasets: NSLKDD, UNSW-NB15, and CICIDS 2023. 20% of the database is used for training, and the remaining 80% is used for assessment. Three cutting-edge CNN algorithms, including ResNet-50, Inception-V3, and Xception model, are used to demonstrate the efficacy of the proposed technique. Evaluation of IDSs was based on four widely used metrics: True Positive Rate (TPR), False Positive Rate (FPR), Accuracy (ACC), and F1-score. The accuracy of the proposed work is compared with other algorithms for the CICIDS 2023 dataset, as shown in Figure 3. The proposed work obtained an accuracy of 99.63%, and ResNet-50, Inception-V3, and Xception obtained an accuracy of 97.68%, 98.65%, and 98.94%, respectively. The TPR of the algorithms are compared as illustrated in Figure 4, and it shows that the proposed model has a higher TPR of 99.56%, Xception obtains 98.34%, Inception-V3 has 97.63% and ResNet-50 has 97.47%. Figure 5 shows that the FPR of the proposed work is 0.21%, which is low compared to other algorithms. The Inception-V3, ResNet-50, and Xception models obtain 0.64%, 0.56%, and 0.43% of FPR, respectively.

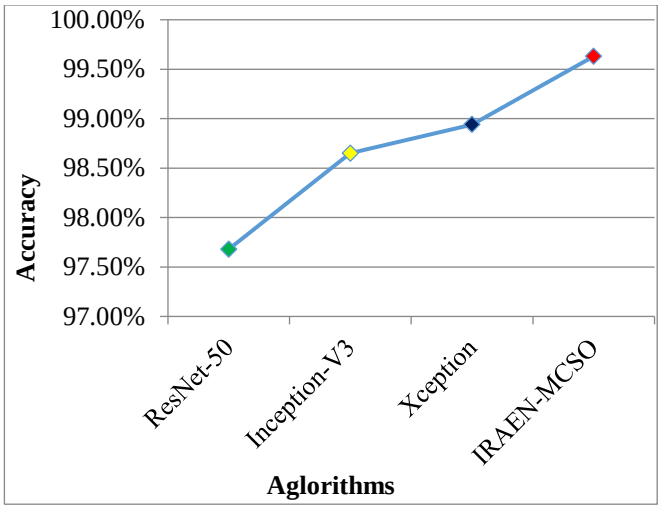

Fig. 3 Comparison of accuracy of IRAEN-MCSO with order models for the CICIDS-2023 dataset

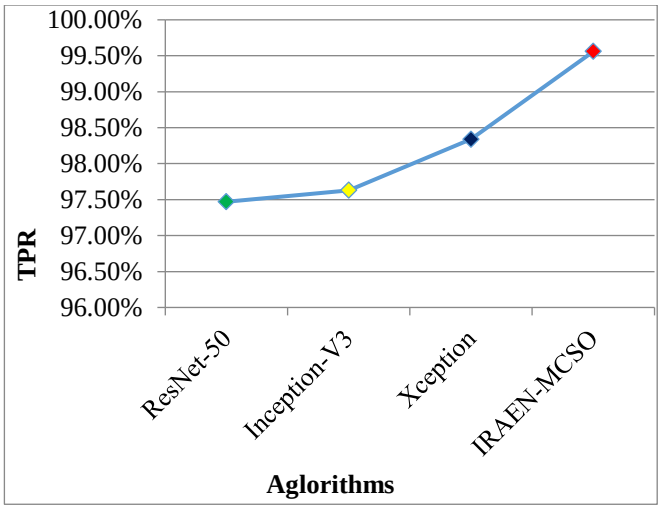

Fig. 4 Comparison of TPR of IRAEN-MCSO with order models for the CICIDS-2023 dataset

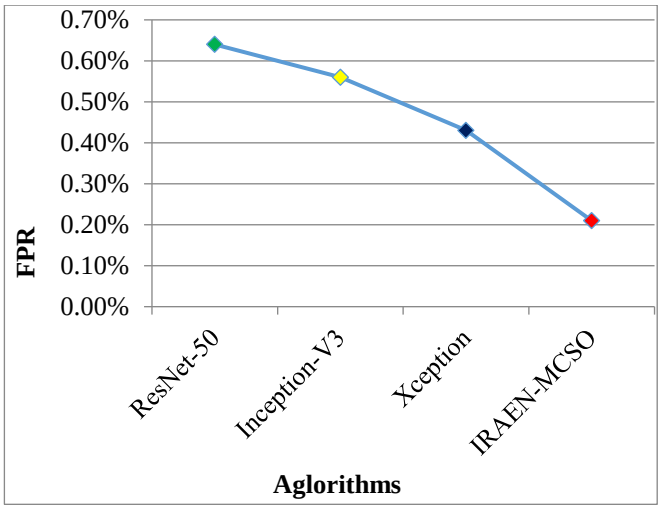

Fig. 5 Comparison of FPR of IRAEN-MCSO with order models for the CICIDS-2023 dataset

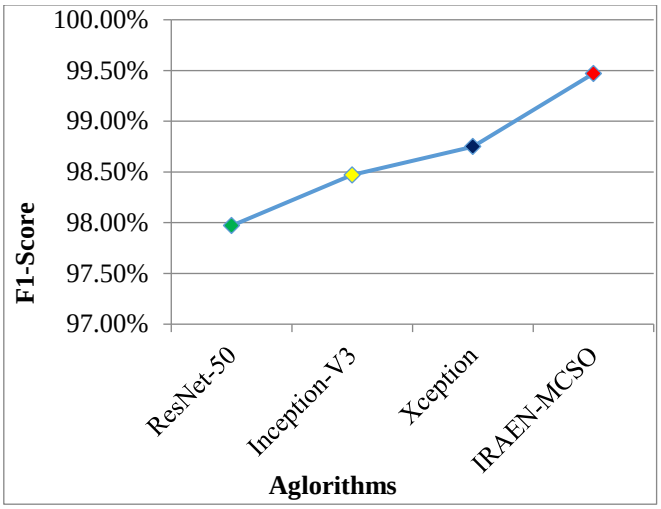

Fig. 6 Comparison of F1-Score of IRAEN-MCSO with order models for the CICIDS-2023 dataset

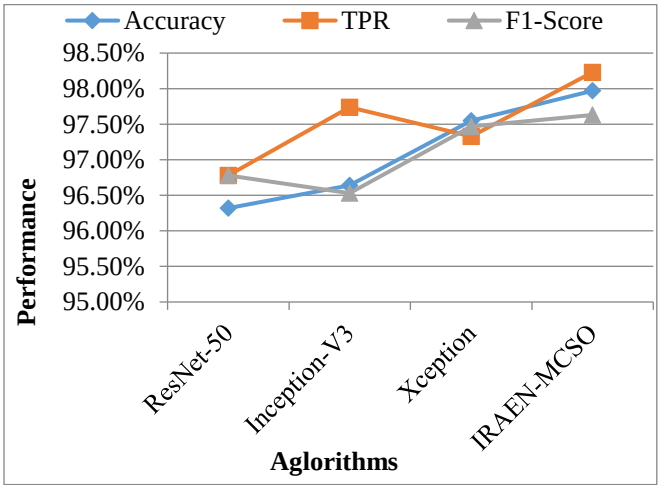

Fig. 7 Comparison of Performance of IRAEN-MCSO with order models for NSL-KDD dataset

The F1 score of the proposed work was high compared to the other algorithms, as illustrated in Figure 6. The F1 scores of ResNet-50, Inception-V3, Xception, and IRAEN-MCSO are 97.97%, 98.47%, 98.75%, and 99.47% respectively. From the above Figures 7 and 8, the performance of the proposed work is better when compared with different classifiers for the NSL-KDD dataset. The proposed IRAEN-MCSO obtained 97.97% accuracy, 96.64%, 97.55%, and 96.32% accuracy by Xception, ResNet-50, and Inception-V3 models. The TPR obtained by ResNet-50, Xception, Inception-V3, and IRAEN-MCSO is 96.78%, 97.33%, 97.74%, and 98.23%, respectively. The F1 score is 96.53%, 96.78%, 97.63%, and 97.47% for ResNet-50, Inception-V3, Xception, and IRAEN-MCSO, respectively.

The FPR of the classifiers are 0.8, 0.7, 0.55, and 0.5 as shown in Figure 8. It shows that the proposed IRAEN-MCSO obtains a lower FPR compared with other models. The efficiency of the classifiers for the UNSW-NB15 database is

shown in Figure 9. The accuracy, TPR, and F1 scores of the ResNet-50 classifier are 95.52%, 95.89%, and 96.78% respectively. For Inception-V3 95.77%, 96.46%, and 96.37%. For Xception, 96.89%, 97.24%, and 96.73%.For RA-EfficientNet, 97.35%, 97.86%, and 97.35%.

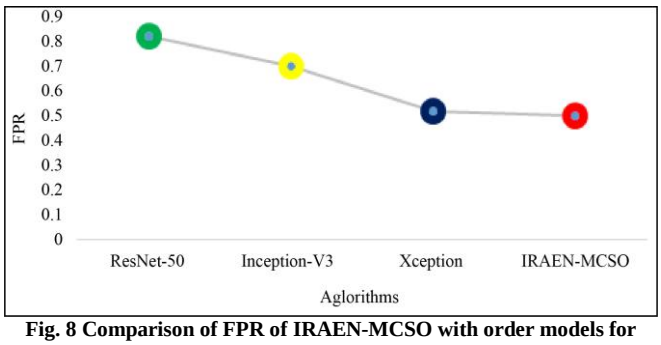

Fig. 8 Comparison of FPR of IRAEN-MCSO with order models for NSL-KDD dataset

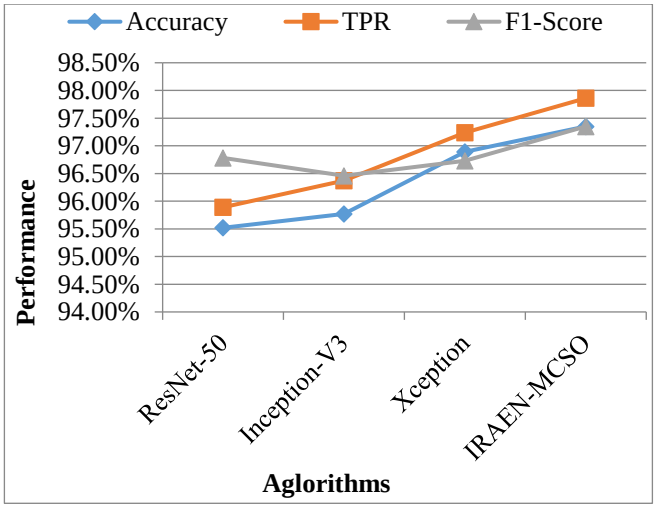

Fig. 9 Comparison of Performance of IRAEN-MCSO with order models for UNSW-NB15 dataset

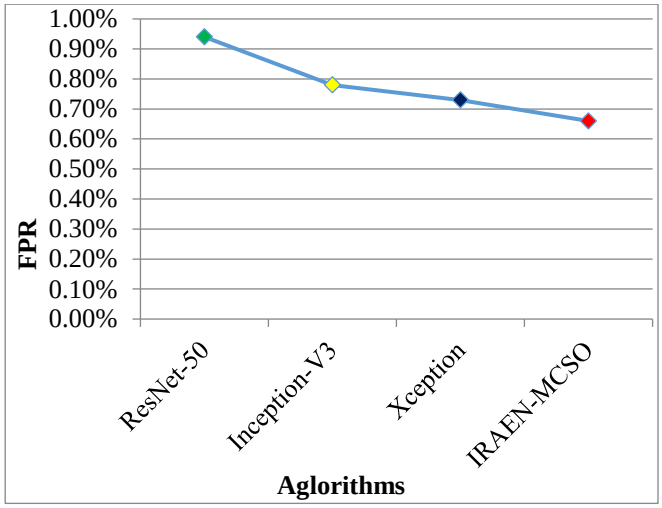

Fig. 10 Comparison of FPR of IRAEN-MCSO with order models for UNSW-NB15 dataset

The FPR of the classifiers are 0.94, 0.78, 0.73, and 0.66, as shown in Figure 10. It shows that the proposed IRAEN-MCSO obtains a lower FPR than other models. In terms of performance, the proposed IRAEN-MCSO outperforms the others, requiring fewer computer-based computations and performing with more precision. The results of each performance evaluation metric were satisfactory.

The efficacy of a proposed method to separate the existing flooding assaults from the overall number of DoS flooding attacks is known as the DoS flooding attack likelihood. In other words, it is the proportion of flooding assaults that have been caught to those that have been detected overall in the environment. The performance chance for the proposed and existing techniques was compared and evaluated in Figure 11. This evaluation demonstrates that the proposed IRAEN-MCSO approach functions better than the earlier methods with the precise assailant node identification. The performance rate is the proportion of times the system successfully recognizes the attacking nodes. Figure 12 compares and contrasts the success rates of the proposed and existing approaches. This evaluation demonstrates that the proposed IRAEN-MCSO approach functions better than the earlier methods with the precise assailant node identification.

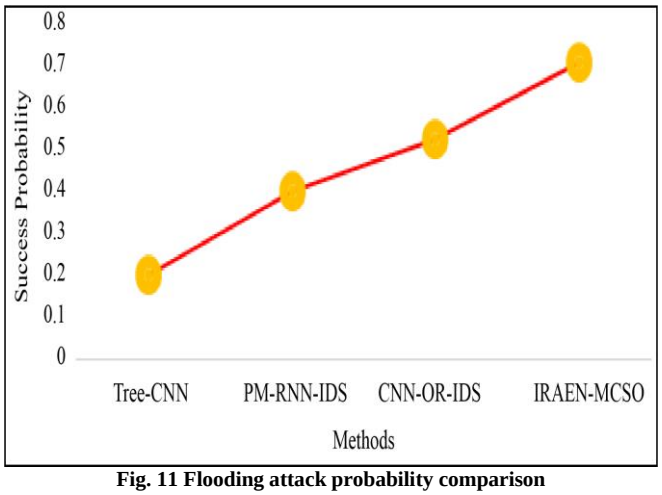

Fig. 11 Flooding attack probability comparison

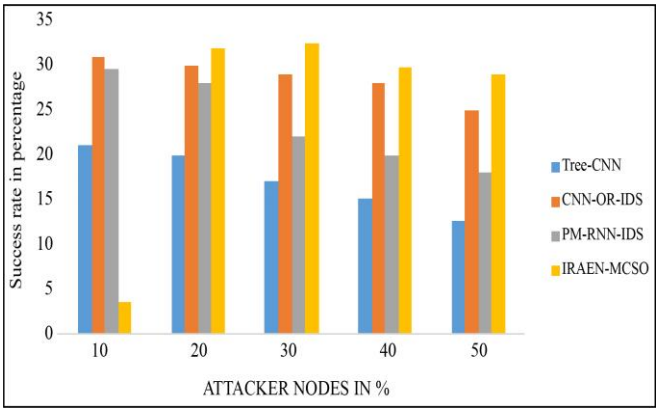

Fig. 12 Comparison of success rate

5. Conclusion

In contrast to cutting-edge methods, the previously mentioned methodical assessment of the proposed IRAEN-MCSO model validates that the model is successfully applicable to real-world situations, such as those involving new attack kinds, fluctuating attack patterns, and adjustments to performance requirements. As a result, when the proposed approach is implemented in actual situations, its performance should improve. Anomaly identification allows us to address potentially dangerous situations arising from anomalies and comprehend the unknown underlying phenomenon. The challenge of distinguishing abnormal network flows and activities that could negatively affect information system security is known as "network anomaly detection".

The IRAEN-MCSO technique for feature extraction from data that aids in network attack detection was covered in this paper. Various DL algorithms are compared with the proposed work to analyze the efficiency, and various datasets are used in the experiment. According to the acquired results, the proposed design performs better in terms of classification in trials using the CICIDS 2023 dataset. The IRAEN-MCSOobtains the accuracy for the CICIDS 2023 dataset is 99.63%, which is high compared to other techniques.

References

[1] Theyazn H. H. Aldhyani, and Hasan Alkahtani, "Cyber Security for Detecting Distributed Denial of Service Attacks in Agriculture 4.0: Deep Learning Model," *Mathematics*, vol. 11, no. 1, pp. 1-19, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Bhawana Sharma et al., "Anomaly Based Network Intrusion Detection for IOT Attacks Using Deep Learning Technique," *Computers and Electrical Engineering*, vol. 107, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[3] Mahdi Soltani et al., "An Adaptable Deep Learning-Based Intrusion Detection System to Zero-Day Attacks," *Journal of Information Security and Applications*, vol. 76, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[4] Srinath Venkatesan, "Design an Intrusion Detection System Based on Feature Selection Using ML Algorithms," *Mathematical Statistician and Engineering Applications*, vol. 72, no. 1, pp. 702-710, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[5] Mahmood A. Al-Shareeda, Selvakumar Manickam, and Murtaja Ali Saare, "DDoS Attacks Detection Using Machine Learning and Deep Learning Techniques: Analysis and Comparison," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 2, pp. 930-939, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Vanlalruata Hnamte, and Jamal Hussain, "DCNNBiLSTM: An Efficient Hybrid Deep Learning-Based Intrusion Detection System," *Telematics and Informatics Reports*, vol. 10, pp. 1-13, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[7] Ahmet Sardar Ahmed Issa, and Zafer Albayrak, "DDOS Attack Intrusion Detection System Based on Hybridization of CNN and LSTM," *Acta Polytechnica Hungarica*, vol. 20, no. 2, 105-123, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[8] Albara Awajan, "A Novel Deep Learning-Based Intrusion Detection System for IOT Networks," *Computers*, vol. 12, no. 2, pp. 1-17, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[9] Afnan Alotaibi, and Murad A. Rassam, "Adversarial Machine Learning Attacks Against Intrusion Detection Systems: A Survey on Strategies and Defense," *Future Internet*, vol. 15, no. 2, pp. 1-34, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Jay Kumar Jain, and Akhilesh A. Waoo, "An Artificial Neural Network Technique for Prediction of Cyber-Attack using Intrusion Detection System," *Journal of Artificial Intelligence, Machine Learning and Neural Network*, vol. 3, no. 2, pp. 33-42, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] Andrew McCarthy et al., "Defending Against Adversarial Machine Learning Attacks Using Hierarchical Learning: A Case Study on Network Traffic Attack Classification," *Journal of Information Security and Applications*, vol. 72, pp. 1-14, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[12] Afnan Alotaibi, and Murad A. Rassam, "Enhancing the Sustainability of Deep-Learning-Based Network Intrusion Detection Classifiers against Adversarial Attacks," *Sustainability*, vol. 15, no. 12, pp. 1-25, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Vinod Varma Vegesna, "Secure and Reliable Designs for Intrusion Detection Methods Developed Utilizing Artificial Intelligence Approaches," *International Journal of Current Engineering and Scientific Research*, vol. 10, pp. 1-7, 2023. [Google Scholar] [Publisher Link]

[14] Rajasekhar Chaganti et al., "Deep Learning Approach for SDN-Enabled Intrusion Detection System in IOT Networks," *Information*, vol. 14, no. 1, pp. 1-7, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Vikash Kumar, Ditipriya Sinha, and Ayan Kumar Das, *Cyber-Attack Detection Applying Machine Learning Approach*, In Applications of Mathematical Modeling, Machine Learning, and Intelligent Computing for Industrial Development, 1st ed., pp. 159-178, 2023. [Google Scholar] [Publisher Link]

[16] Kamaldeep, Manisha Malik, and Maitreyee Dutta, "Feature Engineering and Machine Learning Framework for DDOS Attack Detection in The Standardized Internet of Things," *IEEE Internet of Things Journal*, vol. 10, no. 10, pp. 8658-8669, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[17] Harshit Shah et al., "Deep Learning-Based Malicious Smart Contract and Intrusion Detection System for IoT Environment," *Mathematics*, vol. 11, no. 2, pp. 1-22, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[18] G. Logeswari, S. Bose, and T. Anitha, "An Intrusion Detection System for SDN Using Machine Learning," *Intelligent Automation & Soft Computing*, vol. 35, no. 1, pp. 867-880, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Subhan Ullah et al., "Machine Learning-Based Dynamic Attribute Selection Technique for DDoS Attack Classification in IoT Networks," *Computers*, vol. 12, no. 6, pp. 1-17, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Rubayyi Alghamdi, and Martine Bellaiche, "A Cascaded Federated Deep Learning Based Framework for Detecting Wormhole Attacks in IOT Networks," *Computers & Security*, vol. 125, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[21] Rahma Meddeb et al., "A Deep Learning-Based Intrusion Detection Approach for Mobile Ad-Hoc Network," *Soft Computing*, vol. 27, pp. 9425-9439, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ankit Thakkar, and Ritika Lohiya, "A Review on Challenges and Future Research Directions for Machine Learning-Based Intrusion Detection System," *Archives of Computational Methods in Engineering*, vol. 30, pp. 4245-4269, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[23] D. Shankar, "Deep Analysis of Risks and Recent Trends towards Network Intrusion Detection System," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 1, pp. 262-276, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[24] B. Jothi, and M. Pushpalatha, "WILS-TRS - A Novel Optimized Deep Learning Based Intrusion Detection Framework for IoT Networks," *Personal and Ubiquitous Computing*, vol. 27, no. 3, pp. 1285-1301, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Khattab M. Ali Alheeti et al., "Intelligent Detection System for Multi-Step Cyber-Attack Based on Machine Learning," *Proceedings 2023 15th International Conference on Developments in eSystems Engineering (DeSE)*, Baghdad & Anbar, Iraq, pp. 510-514, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[26] Ke He, Dan Dongseong Kim, and Muhammad Rizwan Asghar, "Adversarial Machine Learning for Network Intrusion Detection Systems: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 538-566, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[27] Trifa S. Othman, and Saman M. Abdullah, "An Intelligent Intrusion Detection System for Internet of Things Attack Detection and Identification Using Machine Learning," *Aro-The Scientific Journal of Koya University*, vol. 11, no. 1, pp. 126-137, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[28] Safa Mohamed, and Ridha Ejbali, "Deep SARSA-Based Reinforcement Learning Approach for Anomaly Network Intrusion Detection System," *International Journal of Information Security*, vol. 22, no. 1, pp. 235-247, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[29] Shiming Li et al., "EIFDAA: Evaluation of an IDS with Function-Discarding Adversarial Attacks in the IIoT," *Heliyon*, vol. 9, no. 2, pp. 1-18, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[30] S. Murugan, and M. Jeyakarthic, "An Efficient Bio-Inspired Algorithm Based Data Classification Model for Intrusion Detection in Mobile Adhoc Networks," *The International Journal of Analytical and Experimental Modal Analysis*, vol. 11, no. 11, pp. 834-848, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[31] A. Thirumalairaj, and M. Jeyakarthic, "Hybrid Cuckoo Search Optimization Based Tuning Scheme for A Deep Neural Network for Intrusion Detection Systems in the Cloud Environment," *Journal of Research on the Lepidoptera*, vol. 51, no. 2, pp. 209-224, 2020. [Google Scholar]