Distributed Approach to detect DDOS attack based on Elephant Herding Optimization and Pipeline Artificial Neural Network

Original Scientific Paper

Yasamin Hamza Alagrash*
Mustansiriyah University
Faculty of Science, Department of Computer Science
Baghdad, Iraq
yhamza@uomustansiriyah.edu.iq
*Corresponding author

Abstract *– Cybersecurity experts widely acknowledge that a Distributed Denial of Service (DDoS) assault poses a grave threat, capable of inflicting substantial financial losses and tarnishing the reputation of enterprises. Conventional detection methods are insufficient for identifying DDoS attacks. Simultaneously, with their vast potential, machine learning solutions play a vital role in this field. This paper presents a distributed approach for identifying distributed denial-of-service threats using the pipeline artificial neural network method, supported by elephant herding optimization for feature selection and extraction. The proposed artificial neural network pipeline-based model for detecting DDoS attacks comprises several key stages: collecting the dataset, preparing the data, implementing a balanced data strategy, selecting relevant features using the swarm optimization method Elephant Herding Optimization (EHO), training the model, testing its performance, and evaluating its effectiveness. Experimental results demonstrate that the proposed approach effectively enhances DDoS detection accuracy while reducing false positives, making it a promising solution for network security. This model demonstrated a remarkably high ability to detect DDoS attacks with a 99% accuracy. Thorough investigations demonstrate that the model is highly skilled in implementing security measures and reducing the risks connected with emerging security threats. The effectiveness of our proposed solution, leveraging a pipeline method in Artificial Neural Network (ANN), is crucial to building a reliable model, which is evident in its ability to deliver effective results in low complexity. The proposed method achieves 99.99% accuracy, 99.80% precision, and a False Positive Rate (FPR) of 0.002%, outperforming recent models. These results demonstrate the model's superior accuracy and robustness in identifying complex attack patterns while minimizing false positives.*

Keywords: *DDoS attack and detection, Auto machine learning, Pipeline ANN, Elephant Herding Optimization (EHO), Artificial Neural Network, AutoML*

Received: February 8, 2025; Received in revised form: June 29, 2025; Accepted: June 30, 2025

1. INTRODUCTION

Distributed Denial of Service (DDoS) attacks are a significant concern in cybersecurity, as extensively reported in the areas of network security, data breaches, and research on malicious activity. The attacks that cause the most denial-of-service disruption send a large number of requests to overload the service. DDoS attacks can lead to a complete or partial service disruption, preventing legitimate users from accessing online services. If attacks do not cause the service to crash completely, they can lead to extremely slow performance and poor user experience[1]. DDoS attacks fall into two main categories: network-level attacks, which overwhelm systems with high-volume data packets (e.g., UDP or SYN floods), and application-level attacks, which exploit weaknesses in applications and online services, such as HTTP flood attacks [2].

The rapidly evolving DDoS area has become so intricate that it is challenging to maintain a clear perspective. This complexity hampers the comprehension of the DDoS phenomenon. Many approaches indicate that the issue is extensive and challenging to investigate and resolve. Existing defense systems employ various methods to address the issue. However, evaluating and comparing their efficacy and cost is daunting, underscoring the critical need for new, innovative approaches [3]. Conventional DDoS attack detection techniques, although efficient with gradual data incre-

ments, are inadequate for effectively analyzing large volumes of high-speed data to detect signs of infiltration. This inadequacy is particularly evident in the face of the evolving nature of DDoS attacks. A DDoS is a crucial cyberattack that aims to disrupt the normal operations of specified servers or networks. This underscores the need for a more robust and efficient approach, which our research aims to provide [4].

Machine learning (ML) is continuously evolving through practice and the application of knowledge [4] [5]. It is regarded as a constituent of artificial intelligence. Depending on the available information, various learning methods exist, such as supervised, semi-supervised, and unsupervised learning [6]. Pipelines and Automated Machine Learning (AutoML) aim to generate algorithmic solutions for machine learning tasks automatically, referred to as machine learning pipelines, that are customized for a specific data set [7]. The ML application in DDoS detection presents challenges in accurately recognizing and preventing attacks while maintaining system efficiency. Several studies have employed classification algorithms to identify and prevent DDoS attacks. DDoS attacks exploit network vulnerabilities to flood a service with excessive requests. Identifying and stopping DDoS attacks in real time can be challenging due to their complex nature and significant consequences [8]. Detection of attacks by anomalies relies on observing differences from standard model usage patterns. These computations depend on easily accessible system parameters, including average CPU utilization, network session rates, user activity frequency, and the type of application being accessed [9]. A variation from a system profile or anomaly could be a sign of a potential intrusion. Yet, the present DDoS attack detection solutions have limitations, such as high detection expenses and an inability to manage substantial network traffic directed toward the server. The packets are examined with classification methods to differentiate DDoS broadcasts from everyday communications [10].

The efficacy of recognition and detection remains challenging, as evidenced by DDoS attacks and the application of machine learning in security evaluation. Several studies have utilized classification algorithms to detect and prevent DDoS attacks. Exploiting network vulnerabilities and sending service requests to the network makes DDoS attacks straightforward. The research created a pipeline machine learning model to identify DDOS attacks in real-time systems. Existing DDoS detection methods, such as rule-based and statistical approaches, often fail under high-traffic loads and adaptive attack strategies. Rule-based systems struggle to detect new attack variations, while statistical models suffer from high false positive rates when traffic patterns fluctuate. The proposed method addresses these issues by utilizing adaptive feature selection, known as EHO, and a deep learning-based classifier, which enables it to adjust to evolving attack patterns while maintaining high accuracy dynamically. The main contributions of this work are as follows:

Developed the EHO for the feature selection phase; most importantly, this work introduced a machine-learning method as a KNN algorithm for evaluating the selected features (fitness function role).

Developed a pipeline ANN model to automate the detection of DDOS attacks using a distributed architecture paradigm. The scalable model is a structured sequence of interrelated data processing and modeling activities created to automate, standardize, and optimize the process of constructing, training, assessing, and implementing machine learning models.

The new detection technique stands out from previous approaches due to its approach of dataset feeding, which combines batch system methodology with streaming.

The performance is improved compared to existing models, as evidenced by several key performance metrics.

The paper is structured as follows: Section 2 describes recent studies on machine learning methods for detecting network attacks and the existing work in this area; Section 3 presents the materials and methods; Section 4 outlines the proposed model; Section 5 describes the dataset splitting and cross-validation. Section 6 displays the proposed pipeline ANN method. Section 7 describes how to evaluate the proposed model, and Section 8 discusses its comparison with previous models . The conclusion is presented in Section 9.

2. RELATED WORK

Several studies have been developed to identify DDoS attacks through machine learning, employing a similar research methodology to ensure consistency in the intensive effort.

Hnamte *et al*. [11] present a new approach to DDoS attack identification using a deep neural network (DNN) model based on deep learning (DL) concepts. This method is designed to be scalable and adaptive for monitoring network traffic and identifying patterns related to DDoS attacks. This paper evaluated the DNN model performance using different datasets, including SDN, CICIDS2018, and Kaggle DDoS. Results show that, in terms of detection accuracy, their proposed DNN-based methodology was 99.98% for the SDN dataset, 100% for the CICIDS2018 dataset, and 99.99% for the Kaggle DDoS dataset. Disadvantages: The paper gives more advantages of the DNN-based methodology than is necessary. It also notes the challenge of implementing DNNs practically in an SDN setting without discussing the details of the technique. Mustapha *et al*. [12] proposed a hybrid method combining Machine Learning (ML) and Deep Learning (DL) algorithms. They utilized Generative Adversarial Networks (GAN) to generate realistic data and employed a Long Short-Term Memory (LSTM) model for DDoS detection. The system achieved a detection accuracy of between 91.75% and 100%. However, the solution is characterized by complexity and overhead.

Anley *et al*. [13] developed a methodology that utilizes deep learning for DDoS detection, employing adaptive architectures within a transfer-learning framework. It discusses transferring information between disparate datasets to enhance classification accuracy in adaptive architectures for DDoS detection. The methodology utilizes tailored CNN architectures with varied layer configurations and pre-trained models (VGG16, VGG19, and ResNet50) while adaptively optimizing hyperparameters. The model was evaluated using four publicly accessible datasets: KDDCup'99, UNSW-NB15, CSE-CIC-IDS2018, and CIC-DDoS2019. The suggested adaptive transfer learning technique proficiently distinguishes between benign and malignant activities and specific attack classifications. Custom CNN models demonstrated exceptional accuracy in distinguishing between benign and DDoS attack traffic, with Conv4 achieving 99.90%, Conv8 attaining 99.94%, and Conv18 reaching 99.88% on the CIC-DDoS2019 dataset. The Conv18 model, adapted from CIC-DDoS2019 to the CSE-CIC-IDS2018 dataset, was archived. The approach attains better results relative to single-domain training. Adaptive designs and hyperparameter optimization enhance the robustness and efficiency of DDoS attack detection. This approach has certain drawbacks, including the use of multiple CNN architectures, transfer learning, and hyperparameter optimization, which likely increase the system's complexity. This may result in elevated computing expenses and necessitate substantial resources for execution, particularly in a real-time context.

Ouhssini *et al*. [14] introduced a Deep Defend approach, a system for the real-time detection and prevention of DDoS attacks in cloud environments. It utilizes deep learning methodologies, notably CNN-LSTM-Transformer architectures, to forecast traffic entropy and identify potential assaults. The framework employs a genetic approach for optimal feature selection to improve the effectiveness of the CNN-DT model in differentiating between regular and attack traffic. This methodology utilizes entropy-based forecasting to predict potential DDoS attack periods, thereby reducing the computational burden associated with preprocessing and classification. The approach was evaluated using the CIDDS-001 network traffic dataset. Results: The proposed system exhibits exceptional accuracy in entropy predictions. Additionally, it facilitates the swift and precise identification of DDoS attacks.

The primary drawback is that the study indicates that the CIDDS-001 dataset contains constraints, such as a restricted number of characteristics, class imbalance (notably for attacks aside from DDoS), and a significant volume of duplicated data, which may cause biases and compromise the accuracy of the conclusions.

Beshah *et al*. [15] presented a different accuracy update weighted Probability Averaging Ensemble (AUWPAE) framework introduced for DDoS attack detection utilizing real-time data streaming. The proposed system utilizes the dynamic characteristics of incoming streaming data to construct a model that identifies idea drifts. The AUWPAE methodology assigns dynamic weights based on their real-time performance to base learners.

Solution Evaluated On: IoTID20 and CICIoT2023 datasets comprising benign and DDoS traffic data. The suggested adaptive online DDoS attack detection framework achieves detection accuracies of 99.54% and 99.33% for the relevant datasets. However, the study doesn't show a real scenario for how the proposed model works in real-time data streaming.

Ashraf *et al*. [16] employed a DDoS detection model that utilized machine learning algorithms, including Random Forest, SVM (Support Vector Machine), Naïve Bayes, KNN (K-Nearest Neighbors), XGBoost, and AdaBoost, on the CICDDoS2019 dataset. The study enhances dimensionality reduction and feature selection methods for efficient DDoS detection, identifying essential elements within. The machine learning methods AdaBoost and XGBoost exhibited outstanding performance, with 100% accuracy in DDoS attack detection. Alternative algorithms, such as KNN and Random Forest, demonstrated higher accuracy, while SVM and Naïve Bayes showed comparatively lower accuracy. Naïve Bayes exhibited the shortest training duration but yielded the lowest F1-score, indicating constraints in DDoS attack identification due to a high incidence of false positives. The research focuses on the Port map segment of the CICDDoS2019 dataset, which may limit the generalizability of the results.

Suarez *et al*. [17] This study evaluates six different machine learning models: Random Forest (RF), Decision Tree (DT), AdaBoost (ADA), XGBoost (XGB), Multilayer Perceptron (MLP), and Deep Neural Network (DNN). This paper presents a preprocessing and feature selection approach using the CICDDoS2019 dataset. The authors examined features using Principal Component Analysis (PCA) and Pearson correlation; subsequently, Tree of Parzen Estimators (TPE) was employed for hyperparameter optimization. This comprehensive methodology, which encompasses assessing various machine learning models with sophisticated preprocessing and feature selection approaches, enabled the authors to attain elevated accuracy in DDoS attack detection while minimizing the number of features. The Random Forest (RF) classifier demonstrated superior performance, attaining an accuracy of 99.97%, an F1 score of 99.98%, and an AUC score of 99.96%.

Other classifiers were considerably accurate; nevertheless, RF surpassed them. The research identifies a shortcoming in the model's capacity to adapt to rapidly developing DDoS attacks. Examining response time during real-time DDoS attacks is vital for future study consideration.

The solution proposed by Elsadig *et al*. [18] presents a streamlined machine learning methodology that employs the XGBoost model to detect DoS attacks in wire-

less sensor networks (WSNs). It utilizes the latest WSN-DS dataset, which is specifically designed for evaluating DoS attacks in WSNs. The methodology emphasizes high precision, effective feature selection, thorough assessment metrics, and minimized processing duration, rendering it appropriate for real-time detection in WSN settings.

The approach was evaluated using the WSN-DS dataset. This dataset comprises both regular and abnormal traffic, featuring four types of DoS attacks: black hole, gray hole, TDMA, and flood.

The proposed XGBoost model attained exceptional performance, with a maximum accuracy of 99.73%.

Compared to other examined classifiers, XGBoost exhibited a 68% reduction in processing time. The results underscore the efficacy of ensemble approaches such as XGBoost. The paper identifies constraints, including issues associated with real-time implementation, scalability concerns, and dataset limitations that may inadequately reflect the enormous diversity of contemporary DoS attacks.

Silivery *et al*. [19] presents a deep learning based multi-classification system to detect DoS and DDoS attacks, which consists of DCGAN to generate synthetic samples, ResNet-50 to extract deep features, and a modified version of AlexNet as a classifier that is trained with the help of the Atom Search Optimization (ASO) algorithm. This multi-component pipeline achieved an accuracy of 99.37% and 99.33% on the UNSW-NB15 and CICIDS2019 datasets, respectively. The problem of class imbalance was adequately addressed with the help of GANs, and feature representation and classification accuracy were enhanced using ResNet and AlexNet.

The model, however, presents enormous architectural complexity that can prove challenging to execute in real-time or on the edge due to the computational and training overheads.

The approach suggested by Naiem et al. [20] is an iteration feature selection-based and Cloud-specific approach to DDoS mitigation. It utilizes the Pearson Correlation Coefficient (PCC) and Random Forest Feature Importance (RFFI) to reduce the feature space, which is then fed to machine learning classifiers comprising Support Vector Machines (SVM) and Decision Trees. Their model achieved 99.27% accuracy and 97.6% precision on cloud-specific datasets, with minimal feature dependency and reduced latency in processing as its key priorities.

However, this framework has not been demonstrated to apply in real-time and to dynamic traffic, which is crucial in real-world cloud-based deployment situations.

Akinwale et al. [21] propose a model of HTTP regeneration (HReg) to counter attacks on mobile HTTP servers. Based on the OMNeT++ simulation platform, the system dynamically detects and regenerates corrupted HTTP sessions to maintain service availability. The measured performance is 73% throughput, 68.8% delivery ratio, and 69.4% goodput under DDoS attack conditions.

Although the approach of regeneration is new and promising in mobile and wireless settings, it has yet to be tested in a real deployment scenario or against deep learning-based detection baselines.

Hussein proposes a deep learning model, which combines a denoising autoencoder (DAE) and a 1D convolutional neural network (CNN) to detect DDoS attacks on the NSL-KDD dataset. The noisy features cleaned up by the DAE present clearer signals before classification. The model achieves a high level of accuracy of 97.7%, a recall of 98.1%, and an F1-score of 97.8% [22].

Nonetheless, the commonly used NSL-KDD dataset lacks freshness and comprehensiveness in reflecting contemporary and emerging DDoS threats. The study also does not consider the system latency and streaming deployment.

The analyzed publications on machine learning and deep learning-based DDoS detection demonstrate high detection accuracy, typically exceeding 99 percent, particularly with the latest methods, including GANs, ensemble models, CNN-LSTM-Transformer architectures, and transfer learning. Most of them, however, have drawbacks such as overly complex models, heavy computation, the use of outdated or unbalanced datasets, and have not been tested in real-time or resource-constrained settings, including those found in IoT, cloud, and wireless networks. Although some of the approaches are either streaming or cloud-specific, the majority lack evidence of real-world deployment. The gaps mentioned underscore the need for lightweight, efficient, and adaptive DDoS detection systems that can be effectively applied in the real world. This research aims to address this need by optimizing the preferred streamlined pipeline ANN model for minimal resource consumption and real-time detection capability.

This study develops a new approach to detecting DDOS attacks based on auto ANN, specifically pipeline ANN, which can deal with real-time schemes and resource-constrained systems.

3. MATERIALS AND METHODS

The materials and methods required in the proposed method are represented as follows:

3.1. DISTRIBUTED DENIAL OF SERVICE (DDOS)

Various compromised devices are utilized in a DDoS attack to target and disrupt a service. DDoS attacks are carried out through botnets. In a DDoS attack, users usually send a server authentication request to establish a connection. The server responds with the outcome of the authentication process. Once the asking user grants this authorization, a connection is established, and access to the server is provided [23]. Also, the attacker floods the server with many authentication requests. Since the requests have fake return addresses, the server needs assistance in identifying a

user to provide authentication approval. The session ends automatically after a set duration during this authentication process. The server typically prolongs the session for over a minute before terminating it. The attacker's continuous requests overwhelm the server, resulting in numerous open connections and a denial of service [24][25].

It frequently occurs when many systems overwhelm a victim's bandwidth or capacity. Such an attack occurs because numerous hacked systems (for instance, a botnet) bombard the targeted system and produce a lot of network traffic, which is performing a task [26]. Hostile botnets infiltrate computers with malicious scripts and programs. Once the botnet gains control of the system, it alerts the master computer. An attacker can take control of the system and send commands to try a DoS attack using this master machine [27].

The proposed approach operates in a distributed environment by deploying a detection method across multiple network nodes. Each node independently analyzes incoming traffic and shares anomaly reports with a central decision system. This distributed processing improves scalability and ensures real-time DDoS detection by reducing the computational burden on a single detection point. Unlike centralized methods, this framework enhances resilience against targeted attacks on a single detection server. Fig. 1 illustrates the threat model of a DDoS attack.

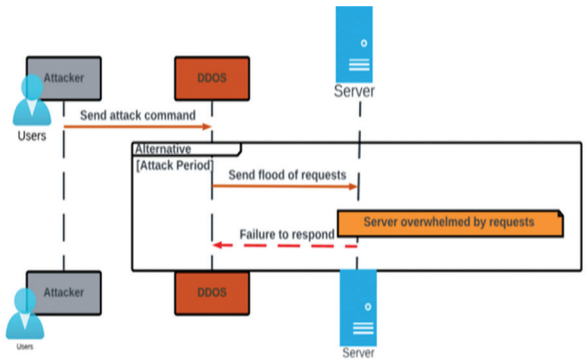


Fig. 1. Threat Model of DDOS Attack

3.2. ELEPHANT HERDING OPTIMIZATION (EHO)

The concept of the behavior of herding elephants can be summarized as follows:

The elephant population is partitioned into a predetermined number of clans, each customized for female elephants. Every elephant in a clan is under the leadership of the master female, known as the matriarch. This behavior aims to find the best solution within a smaller search space, known as a local search. Furthermore, male elephants depart from their clans after puberty and establish independent lives. This behavior is employed to guarantee a global search [28].

The solutions of every male elephant are regarded as wrong solutions. Conversely, all female elephants' solutions are considered good, and the matriarch possesses the best solution within each clan. The EHO algorithm can be characterized based on elephant herding behavior [29].

The population of elephants is partitioned into j clans. The matriarch's influence determines the new place for each elephant in the ci. The jth elephant within the ci clan can be computed by Equation (1):

$$e_{new,\,ci,\,j} = e_{cij} + \alpha \times (e_{best,\,ci} - e_{ci,\,j}) \times r \quad (1)$$

The $e_{new,\,ci,\,j}$ represents the updated placement while e_{cij} Represents the previous placement for Elephant J in the clan ci. $e_{best,\,ci}$ refers to Matriarch CI, who is considered the best elephant. The scaling factor "α" belongs to the interval [0,1], while the variable "r" belongs to the interval [0,1], which is generated for each individual in each iteration [30]. The best elephant for each clan is computed by Equation (2):

$$e_{new,\,ci,\,j} = \beta \times e_{center,\,ci} \quad (2)$$

The factor β, between 0 and 1, defines the impact $e_{center,\,ci}$. In the new individual $e_{new,\,ci,\,j}$. The $e_{center,\,ci}$ indicates the ci clan's central individual (matriarch). It can be computed using Equation (3) for the dth dimension.

$$e_{center,ci,d} = \frac{1}{n_{ci}} \times \sum_{j=1}^{n_{ci}} e_{ci,j,d} \quad (3)$$

Where $1 \le d \le D$, and n_{ci} indicates the number of elephants in clan ci. $e_{ci,\,j,\,d}$. it is the dth dimension of the individual. $e_{ci,\,j,\,d}$, while the center (matriarch) of clan ci ($e_{center,\,ci,\,d}$) can be modified using Equation (3). When addressing optimization problems, male elephants leaving their families can be represented as separating operators. The individual with the lowest fitness in every iteration executes the separation operator, as demonstrated in Equation (4).

$$e_{worst,d} = e_{min} + (e_{max} - e_{min} + 1) \times rand \quad (4)$$

In the search space, the lower and upper limits are denoted by emin and emax, respectively. The variable "$rand$" represents a randomly generated number from 0 to 1 [31].

EHO differs from other optimization algorithms because it does not utilize the prior individuals in the subsequent updating phase. EHO is an algorithm inspired by a swarm that handles global optimization tasks involving clan updates and searching operations. EHO does not focus on relaxation techniques because it is more noise resistant. EHO works very well in constrained and optimized environments. The key characteristics of EHO include a rapid convergence rate, the lowest mistakes in determining localizations, and efficient execution time. The algorithm can address ML problems that are not convex directly [32].

4. PROPOSED MODEL

This paper aims to develop an auto ANN system to identify DDoS attacks. The proposed system consists of four stages: loading the dataset, preprocessing, and

feature selection based on the EHO method. Then, a comprehensive pipeline process is established by building and evaluating an ANN model until the best model is identified and exported. Fig. 2 illustrates the generic framework of the detection model.

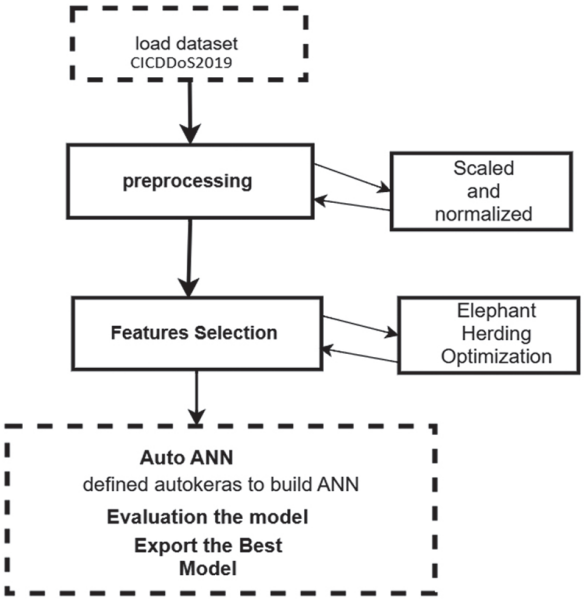


Fig. 2. Proposed Framework to detect DDOS attack

4.1. DATASET ASSEMBLING

The Canadian Institute for Cybersecurity generated the CICDDoS2019 dataset in an authentic network setting, including current genuine data. The collection includes a range of modern DDoS assaults targeting SYN, LDAP, Port Map, UDP, NetBIOS, UDP-Lag, SNMP, MSSQL, DNS, and NTP [33]. Analysis of the CICDDoS2019 dataset reveals that out of 1,048,575 network flow records, more than 58% were classified as attacks and around 42% as legitimate network traffic flows [34]. This study gathers and compiles DDOS attack data, a subset of the CICDDoS2018 dataset, as the proposed model focuses on DDOS attacks.

4.2 DATA PREPROCESSING

The dataset fed to a proposed model was created using preprocessing techniques with batch data preprocessing restriction. It was cleaned by removing null values, and normalization procedures were applied to scale and balance it. The essential characteristics relevant to the DDoS attack flows were collected from the datasets to enable the effective and efficient application of the proposed models. The focus is on differentiating between assaults and standard traffic patterns, rather than individual packets. The preprocessing phase procedure is explained in Figure 3. To address class imbalance, Synthetic Minority Over-sampling. Technique (SMOTE) is applied to generate additional attack samples, ensuring a balanced dataset. This helps in improving the classifier's performance by preventing bias towards the majority class.

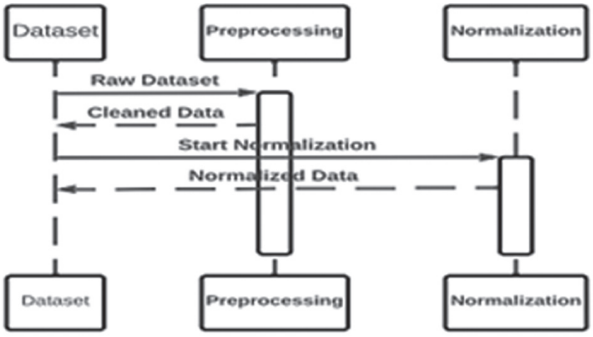


Fig. 3. Dataset pipeline preprocessing

4.3. FEATURE SELECTION

First, the features used in DDOS attacks are normalized, putting all the feature values on the same scale, thus helping the model learn more efficiently and train faster. Each column (feature) is normalized except the last column by converting each value into a float number; this can be done by subtracting the mean from each element in the column and then dividing the result by the column standard deviation. EHO is chosen for feature selection due to its ability to reduce dimensionality while maintaining classification performance. The algorithm iteratively selects the most relevant features based on their contribution to detection accuracy. After optimization, the retained features included packet size, flow duration, source port, destination port, and protocol type, which were identified as key indicators of DDoS attack patterns.

4.3.1. Population Initialized

A population is produced randomly within a search space. This population comprises multiple elephants (solutions), both females and males. Each solution has multiple features, expressed as floating-point values assigned during the normalization process.

4.3.2. Selecting Features Based on Threshold

Based on the predetermined threshold, the float numbers for features in each solution are converted into binary numbers. "Zero" indicates that this feature was not selected. "One" means choosing the corresponding column from the original dataset.

4.3.3. Find Accuracy

Initially, the newly converted dataset is split into training and test sets. At each iteration, the KNN model is trained on the training set (comprising 80% of the total dataset size), and its performance is evaluated based on the test set (comprising 20% of the total dataset size). Then, the model's accuracy is measured by comparing the model predictions to the actual class values, and the accuracy for each solution represents the fitness value for that solution. Table 1 displays the fitness value for each solution in a single iteration.

Table 1. The fitness value (accuracy using KNN) is in one iteration

#	No. of features	Fitness Value (accuracy)
1	43	76.381%
2	40	77.891%
3	36	77.548%
4	43	79.234%
5	37	78.543%
6	43	84.089%
7	43	83.749%
8	41	77.770%
9	41	76.476%
10	40	80.122%

4.3.4. Sorting the Solutions

After computing the fitness value (accuracy) for each solution, these solutions and their corresponding positions are sorted in descending order (from highest to lowest) based on their fitness values to define all the solutions, from the best-performing solution to the worst-performing solution, and their respective positions. Table 2 presents the fitness values in descending order.

Table 2. Order of the fitness function

New index	Previous Index	No. of features	Fitness
1	6	43	84.089%
2	7	43	83.749%
3	10	40	80.122%
4	4	43	79.234%
5	5	37	78.543%
6	2	40	77.891%
7	8	41	77.770%
8	3	36	77.548%
9	9	41	76.476%
10	1	43	76.381%

4.3.5. Clans Creation

Female elephants and their calves live in a group of many clans; each has one Matraich, also called the clan leader, who represents the best solution in that clan. The number of female solutions per clan is determined during each iteration by dividing the total population (ten solutions) by the predefined three clans. The remaining solutions are categorized as male solutions, representing male elephants that separate from their family upon reaching maturity and live independently. The female solutions indicate the randomly selected reasonable solutions, while the male ones indicate the worst ones.

4.3.6. Update Clan Operator

First, the centroid (best solution) of each clan is computed by taking the mean of all solutions within the clan, which is calculated separately for each feature. Second, each clan's best solution position is determined based on the highest fitness value. Third, for each solution in each clan, if the current solution doesn't equal the best, the position of the current solution is updated using Equation (1). More specifically, Equations (2) and (3) are used to update the position of the best solution that is currently accessible. The clan solutions aim to update the current solution randomly and optimize their positions sequentially in each iteration, thereby increasing the probability of reaching the optimal solution.

4.3.7. Update Separate Operator

The position of the worst solution is optimized by adding to the male solutions, allowing them to explore new regions in the search space, which helps them avoid local solutions. The Updating aims to move the male solutions towards a better position in the search space.

4.3.8. Best Features

When the stopping condition is met, the best solution with the highest fitness value is selected in each iteration. Then, these best solutions from all iterations are ranked in descending order to choose the optimal solution. This solution comprises many features, denoted as 0s and 1s, and only features with a value of 1 are considered optimal. Table 3 shows the selected features.

Table 3. Selected features using EHO

fet1	fet2	fet3	feat4	fet5	fet6
'Flow Duration'	'Total Fwd Packets'	'Fwd Packet Length Mean'	'Fwd Packet Length Std'	'Bwd Packet Length Max'	'Bwd Packet Length Min'
fet7	**fet8**	**fet9**	**fet10**	**fet11**	**fet12**
'Bwd Packet Length Mean'	'Flow IATStd'	'Flow IAT Max'	'Flow IAT Min'	'Fwd IAT Total'	'Fwd IAT Mean'
fet13	**fet14**	**fet15**	**fet16**	**fet17**	**fet18**
'Fwd IAT Std'	'Fwd IAT Max'	'Fwd IAT Min'	'Bwd IAT Total'	'Bwd IAT Mean'	'Bwd IAT Std'

5. SPLIT DATASET AND CROSS VALIDATION

The data splitting process, represented by confirming examples of 1500 regular and anomalous DDoS attacks, is categorized into significant subgroups. The initial dataset is the training subset, comprising data selected randomly from the training dataset. Our solution utilizes the K-fold methodology [35]. To analyze the influence of various k values on the model performance estimation and compare it to the optimal test scenario. This can help determine the appropriate value of K. To determine the algorithm that is substantially associated, compare the distribution of the data with that of the optimal test scenario.

With the result distribution from an assessment of the same algorithms under ideal test conditions. The chosen configuration is a reliable approximation for the ideal test scenario, provided the outcomes are correlated at 30% for testing and 70% for training. To evaluate model performance, this work employed 10-fold cross-validation. Figure 4 demonstrates k-fold accuracy.

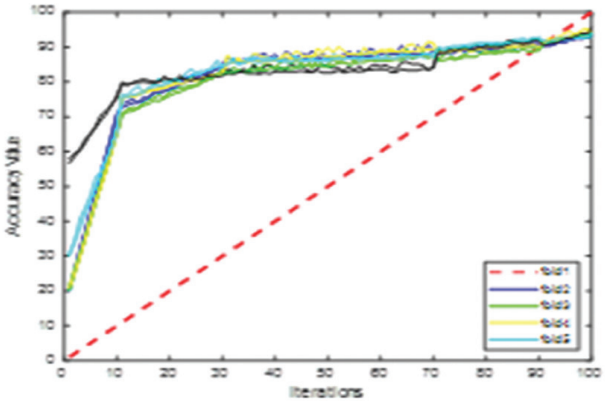


Fig. 4. K-fold Accuracy result

6. PIPELINE ARTIFICIAL NEURAL NETWORK

A machine learning pipeline is an extensive process that encompasses data collection, preprocessing, feature engineering, model training, hyperparameter optimization, evaluation, and deployment. Every phase is essential for constructing a proficient and effective machine learning model, and automation technologies such as Auto ANN help optimize these procedures. Fig. 5 illustrates the construction of a pipeline ANN.

Building an auto ANN, an automated design search is activated using the AutoKeras library to identify the optimal neural network design efficiently.

Hyperparameter optimization: It additionally automates the refinement of hyperparameters. The ANN is optimized using grid search tuning to select the best hyperparameters. The final model has a learning rate of 0.001, comprising 3 hidden layers with 128, 64, and 32 neurons, respectively. The ReLU activation function is used, and the Adam optimizer is employed with a batch size of 32 and 100 epochs. This configuration is chosen based on experimental results, which maximize the F1-score and minimize false positives. Table 4 shows the ANN design structure.

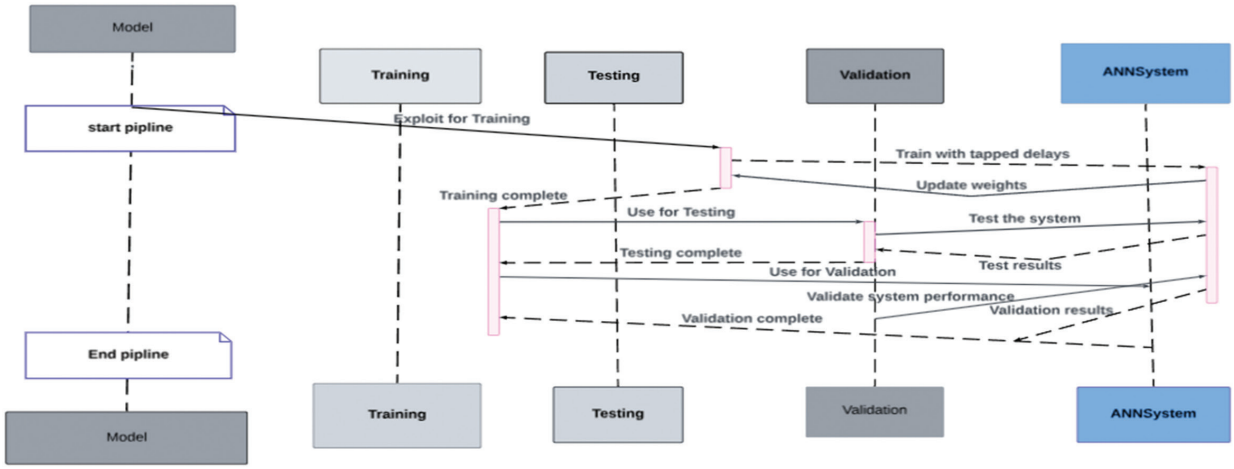


Fig. 5. Pipeline ANN

Table 4. ANN parameters

ANN Parameters	Value
Neurons	(10,100)
Activation function	"ReLu"
optimizer	"ADAM"
Learning rate	0.01
Batch size	200
Epochs	100
Dropout rate	0.2
Normalization	(0,1)

7. EXPERIMENTS AND RESULTS

This section focuses on evaluating the performance of the proposed model. The experiments were conducted on a system equipped with an Intel i7-12700K processor, 32 GB of RAM, and an NVIDIA RTX 3080 GPU. The software environment used is Python 3.9, which utilizes TensorFlow 2.10 and Scikit-learn 1.1 modules. With the ray[tune] python library, a distributed hyperparameter across multiple nodes.

The training time for 100 epochs is 45 minutes, and the average inference time is 3.2 milliseconds per sample, making it suitable for real-time deployment. The outputs are evaluated by comparing the confusion matrix and prediction time to analyze the performance differences between the classifiers.

7.1. EXPERIMENT MODEL

The testing bed is in a distributed structure for DDoS attack detection, as shown in Figure 6. To evaluate the system's performance, this test deploys a primary virtual machine (VM) that simulates legitimate traffic, while an attacker VM generates malicious traffic. A Server Cluster VMA is a multi-node virtual machine that simulates a server cluster. A DDoS Detection System VM runs a Python script to identify DDoS attacks. Client virtual machine: Use the Python scripting to create HTTP requests for the Server Cluster virtual machine. Several instances share traffic with the Server Cluster virtual machine. Use the suggested approach to find questionable traffic patterns and forward them to the virtual machine to detect DDoS attacks.

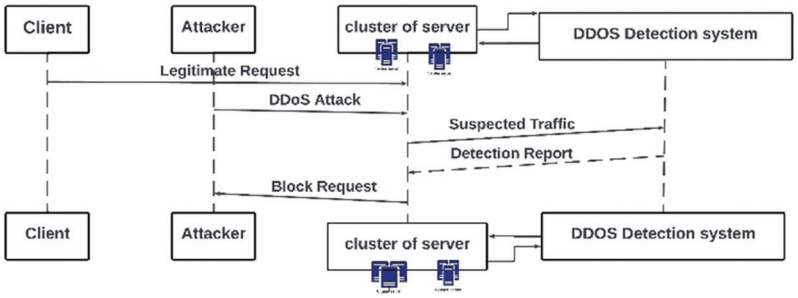


Fig. 6. The experiment testbed

The methodology has been tested in a live environment using QuasarRAT, JRat, and Black Shades, some of the most prevalent remote administration tools (RATs) readily available from both public and academic paper repositories. The results showed that none of these programs managed to evade our models and tools, as their operations consumed resources different from those of the user tasks. Detecting a DDoS attack is a binary classification with labels for benign and DDoS attacks. In this work, benign is seen as a standard class. An attack is considered a positive class because the interest is in finding out an assault, while an innocuous event is considered a negative class.

Analysis of the CICDDoS2019 dataset reveals that out of 1,048,575 network flow records, over 58% were classified as attacks, and approximately 42% were identified as legitimate network traffic flows. Fig. 7 illustrates how network traffic is visualized in the CICDDoS2019 dataset, as well as the unbalanced dataset.

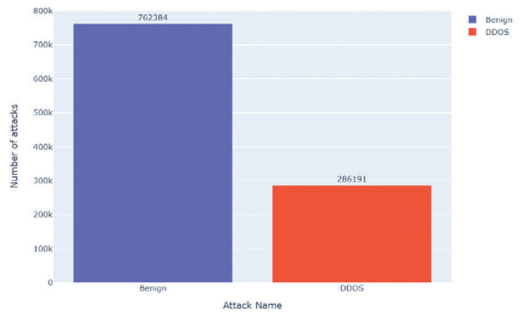


Fig. 7. Unbalanced CICDDoS2019 dataset

Synthetic samples are created from minority populations. Existing samples can be used to create new ones. This work discusses the issue of class imbalance in use.

Examples from minority groups. This should be done on the training set before the model is fitted. The class imbalance problem can be efficiently solved by implementing SMOTE, a model that requires no additional details. As a result, SMOTE is a technique for augmenting minority-specific data. Minority classes in the CIC 2019 dataset contain attack traffic. Fig. 8 displays a balanced dataset.

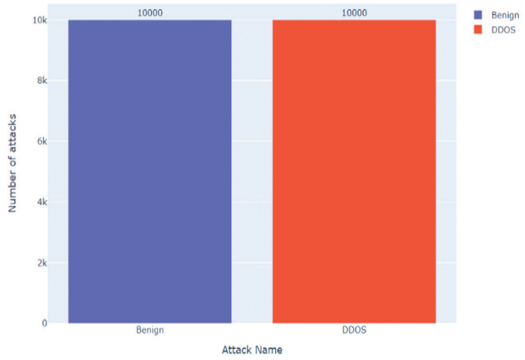
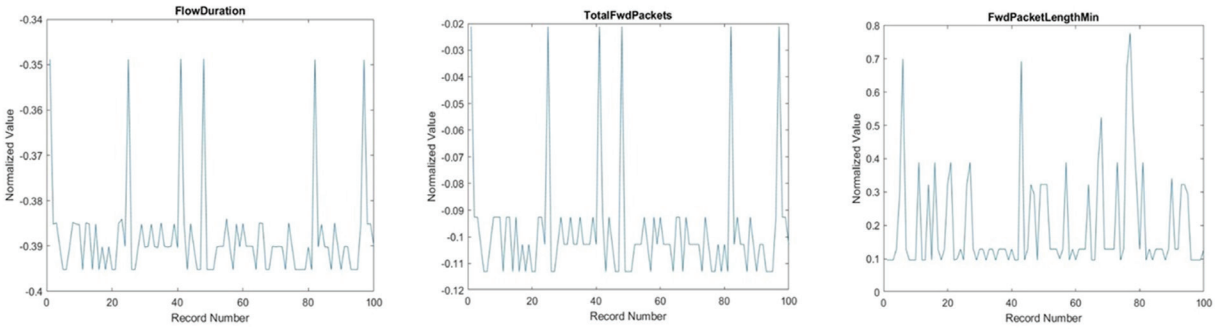


Fig. 8. Balanced dataset

Fig. 9 displays a sample of features that will be input into our model. These characteristics represent the outcomes of the preprocessing and feature selection stages. To provide a TCP connection activity, including the flags used during the handshake, the traffic direction, and the connection's current state.



(a) (b) (c)

Fig. 9. Some samples of the normalization final features: (**a**) Flow duration Feature (**b**) forward packet total (**c**) Forward packet length min feature

The proposed approach begins with 100 and increases to 500 in 100-step increments, with other default settings. To evaluate the effectiveness of the splitting criterion, the dataset was partitioned into 70% for training and 30% for testing. Enumeration of hyperparameters is used in our proposed approach. Fig. 10 demonstrates a successful training process where the model's error (measured by MSE) significantly reduces and then stabilizes, indicating that the model has learned effectively and reached a point of convergence.

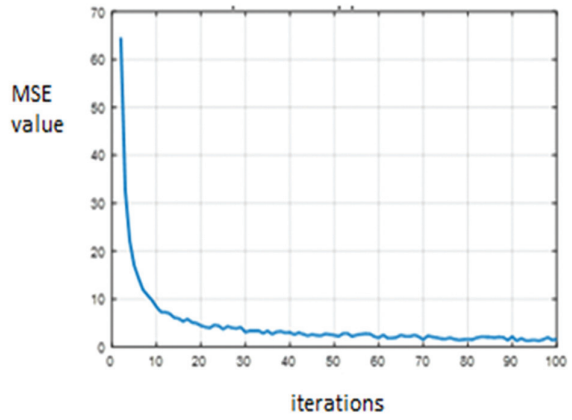


Fig.10. The MSE of ANN (training iterations)

Since our solution is based on pipeline ANN, Table 5 explains the hyperparameter tuning process. This process utilizes resources using the FIFO scheduling algorithm, emphasizing low complexity.

Table 5. Distributed hyperparameter tuning

Resource request	Memory Usage	Number of Trials	Learning Rates
Job requested 3 out of 5 available CPUs	3.5 GB of memory is currently used out of 29.00GB available on the node.	Total of 4 trials: 3 are running, and 1 has an error.	0.01
Ratio or a percentage (50%).	Ratio: 3.4 / 28.8 ≈ 0.118 (approximately 11.8%)	Ratio of Running Trials to Error Trials: 3/ 1 = 3	Ratio:10
Macro average	0.50	0.50	0.50
Weight average	1.00	1.00	1.00

After numerous experiments in the testing phase, the ANN classifier was applied to detect DDoS attacks, and our model achieved a 99% accuracy (MSE of approximately 0.01). The classifiers exhibit nearly identical accuracy, with only a negligible discrepancy. To conclude, the topic at hand is resource consumption: Time Consumed: 51.5697112083435 seconds, Memory Consumed: 1.8 GB, and CPU Usage: 15.7%. Based on this comprehensive throughput, Pipelines can improve the efficiency of the development process and minimize redundant activities, enabling data scientists to spend more time on advanced tasks such as model selection and optimization. This can expedite overall advancement.

8. DISCUSSION

The outcomes achieved using pipeline machine-learning techniques are comparable to those reported by other researchers. Both models exhibit an accuracy of over 0.98, with a minimal false positive rate. Furthermore, our results demonstrate a very competitive false positive rate compared to models yielding the best outcomes.

Several of the models being compared lack the use of cross validation and, in certain instances, the confusion matrix, which is a fundamental approach to evaluation. Moreover, this model demonstrates substantial resource utilization and yields low-complexity outcomes by implementing a pipeline of machine learning approaches. This comparison demonstrates that the identical data set determines the classification accuracy. Table 6 presents the disparities between classification systems based on several parameters.

Table 6. compares recent DDOS detection models based on the CES-CICIDS2019 dataset

RF	Year	Methods	ML	Accuracy
[16]	2024	DDOS attack detection based on ML	Random Forest, SVM, Naive Bayes, KNN, XGBoost, and AdaBoos	Most accurate RF=99.9, AdaBoost and XGBoost =100
[36]	2024	Detection approach based on several MLalgorithms	Logistic Regression SVM, DT, RF, ANN, KNN	RF most accurate %99
Proposed model	2025	DDOS detection EHO features selection, and pipeline ANN	Pipeline ANN	%99.995 with a low complexity

While the proposed model's accuracy (99.99%) is comparable to existing approaches, its significant advantage lies in its lower false positive rate (FPR) (0.002%) and higher Precision (99.80%). As shown in Table 7, a lower FPR ensures fewer false alarms, making the system more reliable in real-world scenarios where excessive false positives can lead to service disruptions.

Table 7. Performance Comparison Table

Model	Accuracy	Precision	Recall	F1-score	False positive rate (FPR)
Proposed system	99.99%	99.80%	99.70%	99.75%	0.002%
Light Weiwei ML model [37]	98.72%	98.40%	97.80%	98.10%	0.005%
Deep learning [38]	98.55%	98.00%	97.50%	97.75%	0.006%
Transformer [39]	98.40%	97.80%	97.30%	97.55%	0.008%
CNN-LSTM [40]	98.30%	97.50%	97.00%	97.25%	0.009%

The proposed pipeline ANN + EHO model achieves the highest accuracy and the lowest false positive rate (FPR). Table 7 presents a comparative analysis of the proposed model using different performance measures with recent DDoS detection models. The results show that the proposed approach achieves an accuracy of 99.99% and a false positive rate of 0.002%. In contrast, the existing deep learning-based methods, such as CNN-LSTM [14] and transformer-based classifiers [39], demonstrate slightly lower precision and recall, resulting in a marginally higher false positive rate (FPR). This highlights the effectiveness of the proposed EHO-ANN model in distinguishing malicious traffic patterns while minimizing false alarms.

9. CONCLUSION

The DDOS attack is a risk that must be detected and prevented from harming the network system. This paper develops a pipeline ANN approach to detect a DDOS attack in a distributed manner.

Feature selection is used to reduce the dimension of the DDOS features and retain only relevant features that affect the attack's detection. A machine learning classifier, such as K-Nearest Neighbors (KNN), is used to evaluate the selected features. The final features are used for final training, while the pipelines are utilized to minimize the resources required for training techniques. The pipeline machine learning model has been shown to significantly impact both binary classifications. Moreover, the proposed solution demonstrates the effect of resource utilization in both the testing and training stages. The ANN pipeline model suggested in this paper can detect DDoS attacks that share comparable attributes. The model was evaluated using the CICDDoS2019 dataset. Though the proposed approach gives high detection accuracy, it also has certain limitations. First, the model has been primarily tested on the benchmark dataset CICDDoS2019, which may not fully represent real-world attack scenarios. Next, computational efficiency remains an issue, particularly in large-scale network deployments. Hence, future research should focus on optimizing model performance for real-time applications. This work will integrate adaptive learning techniques to enhance the detection of emerging attack patterns and validate the approach in real-world environments.

10.REFERENCE:

[1] L. Eliyan, R. Di Pietro, "DoS and DDoS attacks in Software Defined Networks: A survey of existing solutions and research challenges", Future Generation Computer Systems, Vol. 122, 2021, pp. 149-171.

[2] K. Adedeji, A. Abu-Mahfouz, A. Kurien, "DDoS Attack and Detection Methods in Internet-Enabled Networks: Concept, Research Perspectives, and Challenges", Journal of Sensor and Actuator Networks, Vol. 12, No. 4, 2023.

[3] T. Ali, Y. Chong, S. Manickam, "Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review", Applied Sciences, Vol. 13, No. 5, 2023, p. 3183.

[4] K. Aggarwal, M. Mijwil, S. Sonia, A. Al-Mistarehi, S. Alomari, M. Gök, A. Alaabdin, S. Abdulrhman. "Has the Future Started? The Current Growth of Artificial Intelligence, Machine Learning, and Deep Learning", Iraqi Journal for Computer Science and Mathematics, Vol. 3, No. 1, 2022, pp. 115-123.

[5] S. Muhamed, "Detection and Prevention WEB-Service for Fraudulent E-Transaction using APRIORI and SVM", Al-Mustansiriyah Journal of Science, Vol. 33, No. 4, 2022, pp. 72-79.

[6] M. Mijwil, I. Salem, M. Ismaeel, "The Significance of Machine Learning and Deep Learning Techniques in Cybersecurity: A Comprehensive Review", Iraqi Journal for Computer Science and Mathematics, Vol. 4, No. 1, 2023, pp. 87-101.

[7] X. He, K. Zhao, X. Chu, "AutoML: A survey of the state-of-the-art", Knowledge-Based Systems, Vol. 212, 2021.

[8] I. Salem, M. Mijwil, A. Abdulqader, M. Ismaeel, A. Alkhazraji, A. Alaabdin, "Introduction to The Data Mining Techniques in Cybersecurity", Mesopotamian Journal of CyberSecurity, Vol. 2022, 2022, pp. 28-37.

[9] P. Saini, S. Behal, S. Bhatia, "Detection of DDoS attacks using machine learning algorithms", Proceedings of the 7th International Conference on Computing for Sustainable Global Development, New Delhi, India, 12-14 March 2020, pp. 16-21.

[10] R. Doriguzzi-corin, S. Millar, S. Scott-hayward, J. Mart, "LUCID : A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection", IEEE Transactions on Network and Service Management, Vol.17, No. 2, 2020, pp. 876-889.

[11] V. Hnamte, A. Najar, H. Nguyen, J. Hussain, M. Sugali, "DDoS attack detection and mitigation using deep neural network in SDN environment", Computers & Security, Vol. 138, 2024.

[12] A. Mustapha. R. Khatoun, S. Zeadally, F. Chbib, "Detecting DDoS attacks using adversarial neural network", Computers & Security, Vol. 127, 2023.

[13] M. Anley, A. Genovese, D. Agostinello, V. Piuri, "Robust DDoS attack detection with adaptive transfer learning", Computers & Security, Vol. 144, 2024, p. 103962.

[14] M. Ouhssini, K. Afdel, E. Agherrabi, M. Akouhar, A. Abarda, "DeepDefend: A comprehensive framework for DDoS attack detection and prevention in cloud computing", Journal of King Saud University - Computer and Information Sciences, Vol. 36, No. 2, 2024, p. 101938.

[15] Y. Beshah, S. L. Abebe, H. Melaku, "Drift Adaptive Online DDoS Attack Detection Framework for IoT System", Electronics, Vol. 13, No. 6, 2024, p. 1004.

[16] U. Ashraf, H. Sharif, S. Usman, M. Hasnain, "A Machine Learning Based Approach for the Detection of DDoS Attacks on Internet of Things Using CICDDoS2019 Dataset - PortMapLahore Garrison University Research Journal of Computer Science and Information Technology, Vol. 8, No. 2, 2024.

[17] F. Becerra-Suarez, I. Fernández-Roman, M. Forero, "Improvement of Distributed Denial of Service Attack Detection through Machine Learning and Data Processing", Mathematics, Vol. 12, No. 9, 2024, p. 1294.

[18] M. Elsadig, "Detection of Denial-of-Service Attack in Wireless Sensor Networks: A Lightweight Machine Learning Approach", IEEE Access, Vol. 11, No. 2, 2023, pp. 83537-83552.

[19] A. Silivery, K. Rao, L. Kumar, "An effective deep learning based multi-class classification of DoS and DDoS attack detection", International Journal of Electrical and Computer Engineering Systems, Vol. 14, No. 4, 2023, pp. 421-431.

[20] S. Naiem, A. Kheder, A. Idrees, M. Marie, "Iterative feature selection-based DDoS attack prevention approach in cloud", International Journal of Electrical and Computer Engineering Systems, Vol. 14, No. 2, 2023, pp. 197-205.

[21] A. Akinwale, E. Olajubu, A. Aderonmu, "A Regeneration Model for Mitigation Against Attacks on HTTP Servers for Mobile Wireless Networks", International Journal of Electrical and Computer Engineering Systems, Vol. 15, No. 5, 2024, pp. 395-406.

[22] T. Hussein, "Deep Learning-based DDoS Detection in Network Traffic Data", International Journal of Electrical and Computer Engineering Systems, Vol. 15, No. 5, 2024, pp. 407-414.

[23] P. Kumari, A. Jain, "A comprehensive study of DDoS attacks over IoT network and their countermeasures", Computers & Security, Vol. 127, 2023.

[24] M. Mittal, K. Kumar, S. Behal, "Deep learning approaches for detecting DDoS attacks: a systematic review", Soft Computing, Vol. 27, No. 18, 2023, pp. 13039-13075.

[25] S. Lee, Y. L. Shiue, C. Cheng, Y. Li, Y. Huang, "Detection and Prevention of DDoS Attacks on the IoT", Applied Sciences, Vol. 12, No. 23, 2022.

[26] M. Gelgi, Y. Guan, S. Arunachala, M. Samba, N. Dragoni, "Systematic Literature Review of IoT Botnet DDOS Attacks and Evaluation of Detection Techniques", Sensors, Vol. 24, No. 11, 2024, p. 3571.

[27] Z. Shah, I. Ullah, H. Li, A. Levula, K. Khurshid, "Blockchain Based Solutions to Mitigate Distributed Denial of Service (DDoS) Attacks in the Internet of Things (IoT): A Survey", Sensors, Vol. 22, No. 3, 2022, p. 1094.

[28] N. Ahmed, Y. Mohialden, D. Abdulrazzaq, "A new method for self-adaptation of genetic algorithms operators", International Journal of Civil Engineering and Technology, Vol. 9, No. 11, 2018, pp. 1279-1285.

[29] M. Ali, K. Balasubramanian, G. Krishnamoorthy, S. Muthusamy, S. Pandiyan, H. Panchal, S. Mann, S. Thangaraj, N. El-Attar, L. Abualigah, D. Abd Elminaam, "Classification of Glaucoma Based on Elephant-Herding Optimization Algorithm and Deep Belief Network", Electronics, Vol. 11, No. 11, 2022, p. 1763.

[30] A. Malki, A. Mohamed, Y. Rashwan, R. El-Sehiemy, M. Elhosseini, "Parameter identification of photovoltaic cell model using modified elephant herding optimization-based algorithms", Applied Sciences, Vol. 11, No. 24, 2021.

[31] A. Ismaeel, I. Elshaarawy, E. Houssein, F. Ismail, A. Hassanien, "Enhanced Elephant Herding Optimization for Global Optimization", IEEE Access, Vol. 7, 2019, pp. 34738-34752.

[32] Y. Duan, C. Liu, S. Li, X. Guo, C. Yang, "Gradient-based elephant herding optimization for cluster

analysis", Applied Intelligence, Vol. 52, No. 10, pp. 11606-11637, 2022.

[33] A. Hagar, B. Gawali, "Deep Learning for Improving Attack Detection System Using CSE-CICIDS-2018", NeuroQuantology, 2022.

[34] J. Leevy, T. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data", Journal of Big Data, Vol. 7, No. 1, 2020.

[35] L. Yates, Z. Aandahl, S. Richards, B. Brook, "Cross validation for model selection: A review with examples from ecology", Ecological Monographs, Vol. 93, No. 1, 2023, pp. 1-24.

[36] R. Gautam, R. Padmavathy, "Distributed denial of service attack detection using machine learning classifiers", International Journal of Ad Hoc and Ubiquitous Computing, Vol. 46, No. 3, 2024, pp. 123-149.

[37] S. Sadhwani, B. Manibalan, R. Muthalagu, P. Pawar, "A lightweight model for DDoS attack detection using machine learning techniques", Applied Sciences, Vol. 13, No. 17, 2023, p. 9937.

[38] A. Alahmadi, M. Aljabri, F. AL Haidari, D. Alharthi, "DDoS attack detection in IoT-based networks using machine learning models: A survey and research directions", Electronics, Vol. 12, No. 14, 2023, p. 3103.

[39] G. Kirubavathi, I. Sumathi, J. Mahalakshmi, "Detection and mitigation of TCP-based DDoS attacks in cloud environments using a self-attention and intersample attention transformer model", The Journal of Supercomputing, Vol. 81, 2025, p. 474.

[40] M. L. Viñuela, J.-A. R. Gallego, "Systematic Literature Review of Machine Learning Models for Detecting DDoS Attacks in IoT Networks", Advances in Distributed Computing and Artificial Intelligence Journal, Vol. 13, 2024.