FMDB *Transactions on Sustainable Computer Letters*

Denial of Service Attack Detection and Mitigation Using WHO-Based Ensemble Classifier in Software-Defined Network

V. Revathi[1,*], Swathi Baswaraju[2], M. Arunadevi Thirumalraj[3], Piyush Kumar Pareek[4]

[1]Department of Research and Development, New Horizon College of Engineering, Bengaluru, Karnataka, India.
[2]Department of Computer Science and Engineering (Data Science), New Horizon College of Engineering, Bengaluru, Karnataka, India.
[3]Department of Computer Science and Engineering, Karunya Institute of Technology and Science, Coimbatore, Tamil Nadu, India.
[3]Department of Computer Science and Business Management, Saranathan College of Engineering, Tiruchirappalli, Tamil Nadu, India.
[4]Department of Artificial Intelligence and Machine Learning, Nitte Meenakshi Institute of Technology, Bengaluru, Karnataka, India.
revshank153@gmail.com[1], baswarajuswathi@gmail.com[2], aruna.devi96@gmail.com[3], piyush.kumar@nmit.ac.in[4]

Abstract: A new paradigm in networking, software-defined networking (SDN) separates control logic from forwarding operations, allowing for faster administration and better setup of network resources. New SDN allows for more security measures and drastically lowers the computational burden on devices associated to the internet of things (IoT) network. However, centralized network design raises security concerns, especially DoS attacks. Since SDN lacks message-verification, attackers can forge source address details to launch a denial-of-service attack. Ensemble Deep Learning and SDN are used to detect and mitigate Distributed Denial-of-Service assaults in this article. The proposed framework uses the bidirectional gated recurrent unit (BiGRU), transformer block, and convolutional neural network to develop an SDN-enabled security apparatus for IoT devices to detect and mitigate DDoS attacks. Wildebeest Herd Optimization (WHO) selects remote SDN controller and features to counter DDoS attacks utilizing Open Flow (OF) switches and reallocate network resources to permitted hosts. The experimental results show that the recommended framework surpasses current state-of-the-art techniques in DDoS detection accuracy and false alarm rate.

Keywords: Centralised Network; Detection and Mitigation; Open Flow; Convolutional Neural Network; Software-Defined Networking; Internet of Things; Optical Character Recognition.

Received on: 05/02/2025, **Revised on:** 15/04/2025, **Accepted on:** 05/06/2025, **Published on:** 22/11/2025

Journal Homepage: https://www.fmdbpub.com/user/journals/details/FTSCL

DOI: https://doi.org/10.69888/FTSCL.2025.000488

Cite as: V. Revathi, S. Baswaraju, M. A. Thirumalraj, and P. K. Pareek, "Denial of Service Attack Detection and Mitigation Using WHO-Based Ensemble Classifier in Software-Defined Network," *FMDB Transactions on Sustainable Computer Letters*, vol. 3, no. 4, pp. 227–241, 2025.

1. Introduction

[*]Corresponding author.

A new era in contemporary worldwide communication infrastructure has dawned with the advent of the Internet of Things (IoT) [1]. It enables interoperability among smart communication technologies, revolutionising many areas of smart city life. As more human-controlled data processing capabilities are integrated into the Internet of Things (IoT) infrastructure, the technology is driving the emergence of numerous practical applications [2]. The smooth integration of the 5G network has enabled many IoT-oriented smart city applications to increase the manufacturing of IoT devices. However, new security research reveals that hackers are gaining access to IoT devices that lack adequate protection [3]. Attacks such as denial-of-service (DoS), DDoS, brute-force, and TCP SYN/UDP flooding are possible on the IoT network due to the large number of devices that are not adequately protected. For example, millions of IoT devices are being subjected to DDoS assaults, which are becoming more common due to the use of botnets [4].

Cybercriminals are becoming increasingly skilled at attacking network security. They may alter data in packet headers and issue valid broad service requests to individual workstations or servers in typically constructed distributed networks [5]. When it comes to forwarding plane network traffic, however, SDN's centralized control architecture enables thorough analysis of the control plane, enabling appropriate security responses to various network security risks [6]. Since SDN integration has been so effective so far, researchers are refocusing their efforts on improving network security to better detect and mitigate DDoS threats [7]. Practical security procedures are made possible by several key properties of SDN-enabled networks. These include a centralised view of the network topology, dynamic reconfiguration of the system, programmability of network devices, and separation of the control plane and data plane. Building effective intrusion detection and prevention systems that SDN enables relies heavily on SDN's integration capabilities [8].

Traditional dispersed network control and management structures are still used by most current administrative units. These structures are not equipped to handle emergencies or new network problems. The goal of SDN is to decouple network control and data forwarding from centralised controller management. However, this makes the controller's network traffic data both an attractive target for hackers and a key indicator for IDS to spot malicious traffic [9]. To meet the security requirements of new network types, we aim to effectively address issues with current intrusion detection systems and propose a software-defined networking architecture that can thoroughly analyse network threats [10]. Traditional network control centres are slow when it comes to device setup and threat prediction. The symmetry notion in this research encompasses the long-term viability of SDN applications and the strong capability of machine learning (ML) models to handle various types of malicious attacks. The former may have centralised administration and communication with OpenFlow switches thanks to the SDN design [11]. This enables a more efficient and adaptable network setup through programming, reducing the overhead of controller-switch communication. As for the second, the ML model must be resilient enough to detect unusual traffic and adapt to new threats; therefore, it needs to achieve a certain level of detection performance across a variety of attacks and models [12].

To enhance decision-making skills and anticipate abnormalities, SDN environments reportedly use ML, DL, and other algorithms. Threats and vulnerabilities pose significant deployment challenges in SDN environments, making IDS monitoring of malicious activities an essential component of these designs. Furthermore, the centralised perspective of SDN presents new practical options [13]. The most popular dataset for testing AI approaches in recent years is one aimed at identifying DDoS attacks; however, the accuracy of predictions from this dataset depends heavily on its quality. To enhance the security of the SDN architecture, predictions and workable solutions are sent back as SDN parameters [14]. Applications can only access all of the network's data via the SDN. The integration of several applications makes load balancing and intrusion detection considerably simpler. Whenever an anomaly is detected, plane [15].

Routers spread across the network operate the control and data planes because these devices have specific open interfaces controlled by software. SDN architecture enables the simultaneous reconfiguration of multiple devices. In the application layer, network device configuration is performed [16]. There is only one controller in the SDN design's control plane. To facilitate communication between the two tiers, APIs are used [17]. DDoS attacks significantly impact the availability of the SDN. Because it is the weakest link, the SDN controller is a primary target of distributed denial-of-service attacks. There is a single potential weak spot in SDN: the controller operated from a central location [18]. A secure south-bound connection allows the data plane and control plane to exchange messages. The network could experience massive delays even with a slight level of channel congestion. The following is a list of the most important things that this study added:

- To create the module for inspecting and extracting features from live network traffic, and to calculate the entry for the present network flow. After that, to identify DDoS attacks, we combine the trained ensemble deep learning module with a dataset specific to the environment that tracks network traffic in real time.
- The WHO model enhances classification accuracy by appropriately selecting the features.
- To conduct various DDoS assaults to evaluate the proposed framework's detection and mitigation capabilities, and to get encouraging results from the simulations.
- To develop an SDN topology design using AI-assisted security prediction and to suggest a prearranged cloud service according to the unit's network characteristics.

- To assess a security verification scenario, including a simulation, the protocol had to be able to identify and mitigate traffic early if the SDN architecture was subjected to DDoS attacks.

2. Related Work

A thorough investigation of attack findings using the NSL-KDD dataset has been presented by Dash et al. [16]. This dataset encompasses a diverse range of network traffic metrics. This research presents a comparison of two methods: one that utilises Principal Component Analysis (PCA) and another that does not. There are preprocessing processes that utilise robust scaling and encoding algorithms. After combining PCA with Robust Scaler, the results show that IoT device DDoS attack detection becomes much more accurate. It is worth noting that the Random Forest and KNN classifiers outperform Naïve Bayes, achieving accuracies of 99.99% and 90.14%, respectively. The results of this experiment provide valuable insights for enhancing the security of Internet of Things devices against distributed denial-of-service attacks. To achieve robust intrusion detection systems for IoT contexts, the proposed method emphasises the importance of suitable preprocessing procedures. In their discussion of VANET Cloud security, Setia et al. [17] tackled a major issue. Connected cars and the cloud services they rely on are vulnerable to (DDoS) assaults; thus, it's important to be able to predict and counter these attacks. A novel architectural framework is proposed to capture and analyse network flows within the VANET Cloud environment, addressing this problem. It also uses 99.59% accurate machine learning for categorisation and predictive analytics. Security in VANET Cloud installations stands to benefit greatly from the design proposed in this study. Timely responses to security risks and breaches are enabled by its versatility, which ensures practical application in real-world systems.

To effectively identify Distributed Denial of Service (DDoS) attacks in software-defined networking (SDN) settings, Najar and Naik [18] propose a method that combines Balanced Random Sampling (BRS) with Convolutional Neural Networks (CNNs). Filtering, rate limitation, and an iptables rule to prohibit faked IP addresses are just a few of the mitigation strategies we've implemented to address these risks. Furthermore, to ensure the rapid processing of legal data, we utilise a monitoring system that employs rate limiting to track blocklisted IP addresses. Achieving an accuracy of over 99.99% for binary classification and 98.64% for multi-classification, the suggested model exhibits remarkable performance in both contexts. In addition to identifying the assault, our suggested DDoS detection system would also notify a specified email address with extensive contextual information. Using the Area Under the Curve (AUC) metric, we demonstrate that our model outperforms previously published models. In addition, we compared our proposed DDoS mitigation system in three separate scenarios: Attack-Free, Attack-No Mitigation, and Attack-Mitigation, to determine its efficiency and efficacy. These findings demonstrate that our proposed mitigation solution is effective in combating DDoS attacks and maintaining normal network operations.

In their proposal for an improved method of identifying (DDoS) assaults, Hossain and Islam [19] combine an ensemble-based classifier with a hybrid feature selection strategy. The classification accuracy, overfitting reduction, and model resilience are all improved by combining multiple decision trees in an ensemble-based method. To determine which features are most helpful for detecting attacks, the feature selection method employs principal component analysis, correlation analysis, and mutual information. In terms of detection rates, the ensemble-based Random Forest classifier outperforms the other ensemble-based methods when fed the important characteristics. Experimental results demonstrate that the proposed model outperforms state-of-the-art methods across a wide range of datasets for DDoS attack identification, including accuracy, recall, precision, F1-score, and false positive rate. With an error rate of 0%, a true positive rate of 100%, and an accuracy of almost 100%, the suggested method shows great promise as a remedy for DDoS assault detection.

An efficient method for detecting DDoS attacks in both the SDN control and data planes has been proposed by Gadallah et al. [20]. The method utilises new characteristics derived from traffic data to train a Deep Learning (DL) model that can detect DDoS attacks on the control plane. The Autoencoder (AE) with Bidirectional Gated Recurrent Units (GRUs) is a DL approach for DDoS detection. The control plane will include elements such as a transport layer protocol (TLP) header, a type of service (ToS) header, an unknown IP destination address, and the packet's inter-arrival time. The method follows the average arrival bit rate of the switch on the data plane, even while the destination address is unknown. Last but not least, the method employs AE with BGRU in a DL-based model to identify DDoS assaults. On the data plane, we propose the following aspects: the switch's storage capacity, the average flow rate, the IP Options header, and the average rate of packets with addresses. The classifier utilises a dataset created by extracting features and performing calculations on both normal and attack packets. To further enhance detection, additional Machine Learning (ML) approaches are employed. The method updates the user's trust value and blocks questionable senders based on this value to minimise the consequences of DDoS attacks when the system detects one. The proposed approach outperformed similar strategies in terms of accuracy and false alarm rate, as indicated by the experimental data.

To better detect both DDoS and non-DDoS attacks, Nalayini et al. [21] presented an optimised dual intrusion detection system that utilises the best available models. The optimal parameters are determined by Hyper-Tuned parameter optimisation using methods such as Decision Trees, Random Forests, and Logistic Regression. To reduce the 77 features to 4, we apply the RFE

method. Twenty-one models result from an innovative Deep Grid Network that combines hyper-tuned classifiers with seven additional ML methods. To make the most accurate forecast of a DDoS attack, an ensemble approach selects the six best models from 21. Additionally, Mininet generates a new dataset to ensure the model is validated correctly.

A practical dataset, HLD-DDoSDN, was suggested by Bahashwan et al. [22]. It includes the most common DDoS attacks directed against an SDN controller, including ICMP, UDP, and Transmission Control Protocol (TCP). This SDN dataset also includes a range of traffic fluctuation levels, representing the high and low traffic variation rates observed in DDoS attacks. To ensure its superiority, it is statistically tested across all eight scenarios and qualitatively compared with existing SDN datasets. Additionally, it incorporates important elements that significantly enhance the detection of genuine SDN attacks, and it meets the size, attack diversity, and scenario criteria of a benchmark dataset. Our evaluation of HLD-DDoSDN's characteristics is based on a detection technique called Deep Multilayer Perception (D-MLP). The results of the experiments demonstrate that the characteristics used are highly effective in terms of accuracy, recall, and precision in recognising DDoS flooding attacks, regardless of the attack rate.

To identify and categorise DDoS attacks, Effah et al. [23] developed and deployed a hybrid deep learning model called CRNN-Infusion. Our model made use of a convolutional neural network (CNN) besides recurrent models; it was trained using the CICDDoS2019 dataset acquired from the Canadian Institute of Cybersecurity (CIC); and to improve its efficiency and reduce its dimensionality, we used Random Search Hyperparameter Tuning (RSHT) and Feature Selection (FS) approaches. Security for training the model using Cyber Intelligence Centric (CIC) with RSHT and FS methods for dimensionality reduction and improved model efficiency. When tested against competing deep learning (DL) models trained on the same dataset, our proposed model outperformed them all as a DDoS attack classifier. Categorise distributed denial of service (DDoS) assaults on network infrastructures with an accuracy of up to 99.992%. The results show that hybrid deep learning models perform best when hyperparameter tuning and feature selection are performed effectively. Our suggested model achieved 98.92% accuracy, 99.02% precision, 98.92% recall, and 98.93% F1 score.

Using a feature selection and Bi-LSTM-based honey badger optimisation algorithm in a cloud setting, Pandithurai et al. [24] demonstrated the ability to predict DDoS attacks. Initially, input characteristics are extracted from the DDoS attack dataset. The next step is to send the input features to the preprocessing stages, which may include normalisation using Bayesian and Z-score. Feature selection using Honey Badger Optimisation (HBO) is the next step after data preprocessing. Here, the features are selected by minimising their MSEs to obtain the best features. Then, to forecast DDoS assaults, a Bi-LSTM classifier is given the best characteristics. Additionally, the suggested model is tested using several pre-existing methods, such as ANNs, DBNs, LSTMs, and DNNs. The Bi-LSTM model achieved several impressive results when tested using the current approach. It achieved high accuracy (97%), sensitivity (95%), specificity (90%), error rate (3%), precision (94%), and so on. Finding DDoS in a cloud environment is made easy using the suggested paradigm.

To safeguard the SDN environment, Kaur et al. [25] proposed K-DDoS-SDN, a DDoS attack detection method based on Kafka. The two main components of the K-DDoS-SDN are the NTStorage and NTClassification modules, which are responsible for storing and classifying network traffic, respectively. To categorise traces in real time, the NTClassification module utilises an efficient, well-implemented solution on the two-node Kafka Streams cluster, built with scalable H2O ML approaches in a distributed manner. The NTStorage module systematically saves raw packets, network flows, and 21 key properties in HDFS for retraining existing models. The recently released CICDoS2019 dataset was used for both the design and evaluation of the projected K-DDoS-SDN. When it comes to identifying valid network traces and the most common types of attacks, including DDoS and UDP, the proposed distributed K-DDoS-SDN achieves an average accuracy of 99.22%. Additionally, the results demonstrate that the proposed K-DDoS-SDN successfully categorises traffic traces into five groups with an accuracy of 81% or higher.

3. Proposed System

3.1. SDN-Enabled IoT Topology

At the control layer, the distant SDN floodlights are coordinated with the Mininet emulator, which launches the SDN-enabled IoT system architecture. On the forwarding plane, you'll find the deployed IoT hosts linked to OF switches. Notifying the SDN of any new communication requests is the main function of these OF switches. When in use, OF switches notify the SDN controller of DDoS attacks and the current congestion rate on all available communication lines. In response to a DDoS attack, the SDN activates the built-in congestion management mechanism. Initially, the EV model, which employs adaptive machine learning classification incorporating SVM and LR, is executed by the OF switches. In the mitigation response, the manager calculates an alternative route using operational ports and link flow congestion information. This is done for real host requests.

As intermediate devices, OF switches regularly provide statistics in bytes and execute path returns in compliance with the controller; meanwhile, they use a hash table to record the hierarchical structure of the incoming statistics and the network perspective. The SDN controller also sorts the calculated alternate pathways by network path index. The SDN controller's ability to deploy more alternative pathways across OF switches has an inverse relationship with the predicted paths.

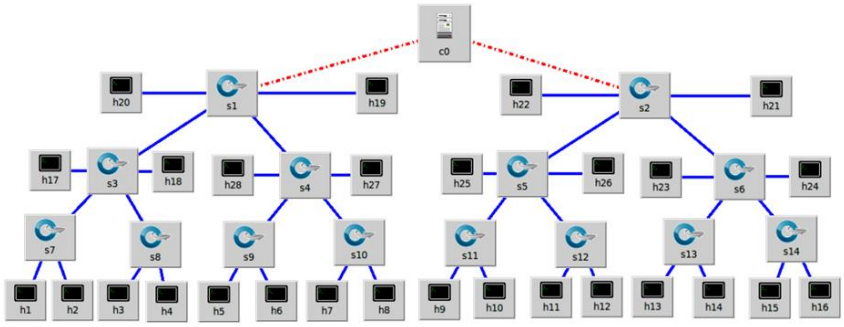

Figure 1: Custom SDN IoT system topology with POX supervisor

In this research, we also employ two topologies—Figures 1 and 2—for experiments that utilise SDN-enabled IoT networks. With a POX Controller, we create a unique SDN-enabled IoT network architecture. In the second network architecture, we include a Floodlight Controller to provide software-defined networking for things. At the forwarding plane, both topologies have OpenFlow switches that are further linked to IoT devices. At the same time, SDN controllers are present in the control plane to counter the DDoS attack.

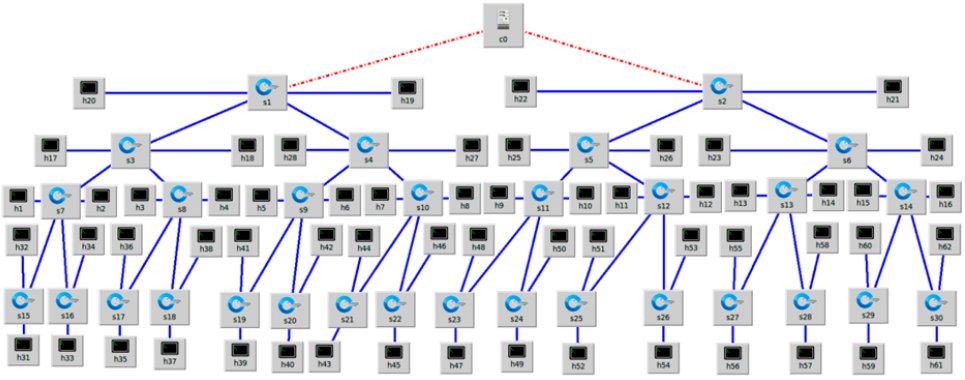

Figure 2: Custom SDN-enabled IoT topology with floodlight controller

Refer to Figure 3 for an illustration of how the proposed model would sit between the controller platform and other controller applications. The three main parts are a mitigator, a detector, and a monitor. By establishing a mapping link among hosts, the monitor enables rapid detection of network irregularities.

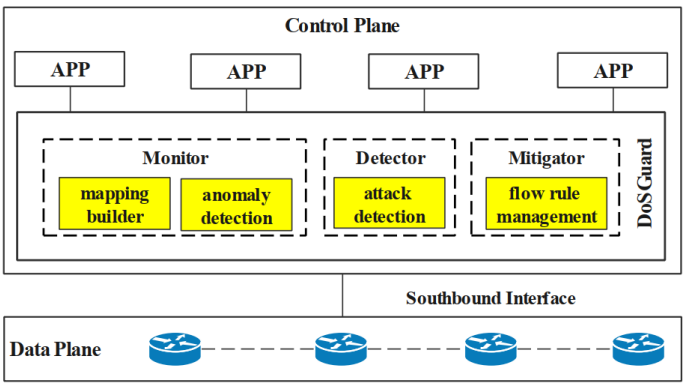

Figure 3: Architecture of the proposed model

3.2. AI Module and Deep Learning Models

We implemented an AI module in the controller to identify and categorise attack types when the SDN network experiences anomalous packets. The models for binary and multiclass classification were trained on SDN public datasets during the study's training phase. Unlike multiclass classification, which focuses on identifying abnormalities, binary classification aims to distinguish between normal and abnormal instances. Before making a forecast for a binary or multiclass classification model, network flows pass through the AI module to extract features. Lastly, the controller determines how to handle network traffic based on the forecasts.

3.3. Dataset Description

The assessments in the proposed framework are conducted using a dataset specific to the environment. This dataset converts the network statistics to a "CSV" file.

Table 1: Features of the traffic structures

RSIP	SDFB	RFES	RPFES	SDFP
41	119.721586	41	0.516129	0.450748
41	85.944815	41	0.516120	0.279828
41	84.437979	41	0.516129	0.278636

The "tshark" software, run from the terminal, is used to create the CSV file with restricted fields. Also, Tables 1 and 2 demonstrate the attack as mentioned above. Table 2 shows that the packet interval field shows the greatest difference between the two traffic types.

Table 2: Features of the DDoS attack structures

RSIP	RPFES	SDFB	RFES	SDFP
12	1.00000	333.682848	34	0.721688
12	1.000000	313.496126	24	0.821678
12	1.000000	320.257404	25	0.6704407

The entire dataset, comprising 3999 rows and six columns, is presented in Figure 4. The last column determines whether the traffic is considered normal or a distributed denial-of-service attack based on the initial values.

```
      41  0.3897756765675669  75.804607046878985  41.1  0.5161290322580645  1
0     41            0.450748          119.721586    41            0.516129  1
1     13            0.835165          351.018887    26            1.000000  0
2     13            0.714143          307.680791    26            1.000000  0
3     11            0.588235          334.996292    22            1.000000  0
4     12            0.644899          329.904769    24            1.000000  0
...   ..                 ...                 ...   ...                 ...  ..
3994  12            0.721688          333.682848    24            1.000000  0
3995  12            0.821678          313.412966    24            1.000000  0
3996  41            0.279828           85.944815    41            0.516129  1
3997  41            0.279828           84.437979    41            0.516129  1
3998  12            0.670407          330.257404    25            1.000000  0

[3999 rows x 6 columns]
```

Figure 4: Print of the complete dataset over a usual or DDoS attack

3.4. Features Extraction of Network Traffic of Mininet Topology Stage

When the network settings are changed in the framework, the performance of the real-time traffic produced by the topology varies. Figures 1 and 2 illustrate two distinct SDN network topologies, each featuring a unique combination of network density and remote SDN controller. The pace of packet transmission via active communication channels, however, determines the primary aspects of traffic. Such traffic characteristics differentiate between typical traffic patterns and very varied network traffic requests. We created a module to collect data from real network traffic and classified its features. To gather traffic information via the OF switches, our feature extraction module runs a shell script. It generates the "SVC" data files for the

following metrics: packet count, byte size, source IP count, and destination IP count. Afterwards, the designed Python script module takes these files as input and calculates several metrics, Rate of Flow Entries on Switch (RFES). Manipulating network traffic can be achieved by manually initiating distributed denial-of-service (DDoS) attacks against specific hosts within the network. To calculate the following traffic characteristics, this feature extraction module utilises libraries.

- **Rate of Source IP (RSIP):** This function shows the sum of source IPs in a certain amount of time for a specified destination IP address:

$$\mathrm{RSIP} = \frac{\sum \mathrm{SIP}}{\mathrm{T}} \quad (1)$$

Where T is the sample time, which is modifiable based on how well the SDN controller manages the flow of traffic.

- **SDFP, or Standard Packets:** This is the deviation of the sum of packets during the T period:

$$\mathrm{SDFP} = \sqrt{\left((1/\mathrm{n}) * \sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{packets_i} - \mathrm{meanPackets})\right)^2} \quad (2)$$

Given that n is the count of flows, 〖packets〗 _i is the count of packets for flow ith in T period, and mean is the sum of all of them. The total packets across all flows and all T times are averaged to obtain the average packets. Because a distributed denial-of-service (DDoS) attack involves the transmission of many small packets, each with a smaller standard deviation than a typical data packet, this parameter drops dramatically during an attack.

- The Standard Deviation of Flow Bytes (SDFB) represents the quantity of bytes within the standard deviation of the T period:

$$\mathrm{SDFB} = \sqrt{\left((1/\mathrm{n}) * \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{bytes_i} - \mathrm{meanBytes}\right)^2} \quad (3)$$

where $\mathrm{bytes_i}$ is the total bytes of flow ith during the time period, and meanBytes is the average of all the flows' total bytes during that time. Similar to SDFP, SDFB is strongly associated with DDoS attack occurrences. During an assault, this parameter's predicted value is lower than it is during regular traffic flows.

- The sum of flow entries to the switch within a given time period is known as the RFES.

$$\mathrm{RFES} = \frac{\sum \mathrm{F}}{\mathrm{T}} \quad (4)$$

This is a crucial statistic for attack identification because, unlike the SFE value during normal traffic flows, the sum of flows increases significantly within a predetermined time frame during an attack.

- **Pair-Flow Entries on Switch Ratio (RPFES):** The sum of interactively divided flow entries in the switch divided by the total number of flows during the T time

$$\mathrm{RFES} = \frac{\mathrm{IntIP}}{\mathrm{N}} \quad (5)$$

Where N is the overall IP count, and IntIP is the overall count of IPs that are interactive inside the flow. Therefore, there will be a precipitous decline in the volume of interaction flows the moment the assault starts. The feature extraction unit then generates the RSIP, RFES, and RPFES headers and assigns the calculated values once the features mentioned above have been computed. The feature extraction module assigned the value 1 to typical traffic based on the computed values. To accept incoming network traffic in real time, the feature extraction module generates the training data file, “live.csv”, at the end of the process.

The “live.csv” file stores the freshly calculated network traffic and feeds it into the trained adaptive machine learning model. Five characteristics of network traffic can effectively distinguish DDoS attacks from regular traffic: they include an abnormal exponential growth in the values of RSIP, RFES, and RPFES, which stand for Rate of Flow Entries on Standard Deviation of Flow Bytes, respectively. This significantly improves the efficacy and precision of DDoS detection.

.

3.5. Feature Selection using the WHO Algorithm

The main reason for selecting this approach is that it is relatively new compared to other types of metaheuristic algorithms. Furthermore, it produces better results for the study's benchmark functions as well. Because of this, we decided to employ this metaheuristic approach to enhance the efficiency of the proposed strategy. The World Health Organisation's algorithm is based on wildebeests' behaviour: finding food. Wildebeests are social, energetic creatures who are always on the go in quest of food. As a mating strategy, guys engage in sex challenges with potential suitors. The WHO algorithm began with randomly selected candidates. Among smaller (X_{min}) and upper (X_{max}) borders, the populace was constrained, i.e.,

$$X_i \in [X_{min}, X_{max}] \quad (6)$$

Here, $I = 1,2, \dots, N$

After that, the wildebeest started moving with the milling gait. During this stage, the ideal position was still being sought, with a fixed value (n) representing the small random mobility that depends on location. Those vying for spot X had used a phase. Z_n It should consistently look for the small areas. The duration might be adjusted based on the size of the participants' arbitrary steps. Because of this, the era of focused experiments Z_n was produced formula:

$$Z_n = X_i + \varepsilon \times \theta \times v \quad (7)$$

Here, a random unit vector is characterised by v; a random unchanging value among 1 and 0 is denoted by θ'; the ith applicant sum is represented by X_i; and the learning degree is signified by ε. After assessing a fixed sum (n) of minor random possibilities, the wildebeest adjusted its position to find an ideal random site. It is labelled in Equation (8) below.

$$X_i = a_1 \times Z_n^* + \beta_1 \times (X_i - Z_n^*) \quad (8)$$

Here, the local movement of the candidates is taught by the a_1 and β_1 leader variables.

$$X_i = a_2 \times X_i + \beta_2 \times X_h \quad (9)$$

Here, a_2 besides β_2 stand in the crew's local drive, and X_h denotes a random candidate.

$$X_i = X_i + \theta \times (X_{max} - X_{min}) \times \bar{v} \quad (10)$$

Here, the haphazard unit vector is represented by $\bar{v}$. Another term employed in the algorithm was to simulate crowded locations. There was a population when the grassland was quite productive. Personal pressure is the term used to describe this idea. To complete a job, this word is employed, and the top candidate applies the following formula to outperform all other competitors.

$$\text{if}(\|X^* - X_i\|) < \eta, (\|X^* - X_i\|) > 1 \quad (11)$$

$$\text{Then } X_i = X^* + \varepsilon \times \hat{n} \quad (12)$$

Where the symbol h stands for a congestion-prevention threshold and nˆis the number of reachable sections close to the optimal solution point. After simulating the swarm's social memory in the previous step to improve placements, the following equation was used to calculate it.

$$X = X^* + 0.1 \times \hat{n} \quad (13)$$

At last, the classifier's hyperparameters were initialised using the optimal value of this approach. The ideal parameters are an L2 regularisation dropout rate of 40%, a momentum of 0.8, a learning rate of 0.001, a $1 \times 10{-}6$, and an 8-batch size. The difficulty of the WHO procedure is O ($p \times m$). Here, m besides n signify the populace and problem dimension, correspondingly.

3.6. Proposed Multichannel Deep Learning Framework

Three models were used for this study, all of which contributed to the final forecast, and the research made use of an integrated multichannel method:

- The transformer block,

- Typical CNN architecture.
- Bidirectional GRU (BiGRU)

Our suggested approach differs from the current state of the art by combining the capabilities of three state-of-the-art deep learning representations: the block, BiGRU, and CNN architectures. As a consequence, it can accurately extract relevant information and provide reliable results. More information on each of them is provided in the sections that follow.

3.6.1. Transformer Block

The transformer accelerates training by enabling fast parallel computation, unlike RNNs and LSTMs. Faster learning and improved model performance on any prediction task are enabled by the attention mechanism, which addresses the limitations of the encoder-decoder model with lengthy sequences. The transformer is primarily composed of multihead attention (MHA) and scaled dot-product attention units. Also included in the model are embeddings, a fully connected network, encoder and decoder stacks, and a softmax. It is possible to determine the transformer's scaled dot-product as

$$a(q, k, v) = SM\left(\frac{qk^T}{\sqrt{d_k}}v\right) \quad (14)$$

Where a represents each attention key, v is a charge, and SM is SoftMax. d_k is the dimensionality of the vector. The square root of the dimensions of the key vectors is used to split the attention weights. To make the weights equal to one, the function is used in the equation. To find MHA, use the formula in (15).

$$MHA(q, k, v) = Concat(hd_1, \dots, hd_h)W^O \quad (15)$$

where hd_i is intended as follows:

$$hd_i = a\left(qw_i{}^Q, kw_i{}^K, vw_i{}^V\right) \quad (16)$$

and $w_i{}^Q$, $w_i{}^K$, $w_i{}^V$ agree to the weight media to be learned. The transformer nevertheless surpasses NLP deep representations, such as LSTMs and gated recurrent units (GRUs), on several tasks, while utilising only the attention mechanism and not an RNN, and setting the embedding size.

3.6.2. Bidirectional Recurrent Neural Networks

The bidirectional network in our perfect begins with a layer that applies dropout to entire feature maps rather than specific regions. Then, the GRU-based bidirectional RNN (BiRNN) layer [32] receives the output from this layer. This layer links two hidden layers, one facing ahead and one facing backwards, to the same output. Afterwards, the BiRNN layer's output is concurrently passed to pooling layers, which then aggregate their outputs to produce fresh input for the following level. There are several windows or partitions within each input feature map. Following these steps, the average pooling function may determine the average of an n-by-n window:

$$\frac{x_1+x_2+\cdots+x_n}{n} \quad (17)$$

The max-pooling of the sum that has a window $\{x_1, \dots x_n\}$. The goal of both max pooling and average pooling is to reduce the dimensionality of the data while retaining all relevant information.

$$z_{ti} = \sigma(W_{xz}x_{ti} + W_{sz}s_{ti-1} + b_z) \quad (18)$$

$$e_{ti} = \sigma(W_{xe}x_{ti} + W_{se}s_{ti-1} + b_e) \quad (19)$$

$$g_{ti} = \tanh\left(W_{xg}x_{ti} + W_{hg}(e_{ti} \odot h_{ti-1}) + b_g\right) \quad (20)$$

$$s_t i = (1 - z_{ti}) \odot s_{ti-1} + z_{ti} \odot g_{ti} \quad (21)$$

where x_{ti} is the contribution to the GRU cell at period ti. W_{xg}, W_{xz} and W_{xe} are the weight matrices that receives input X_{ti}. W_{sg}, W_{se} and W_{sz} are matrices, the previous cell state vector. tanh is a function, and σ is a purpose. b_e, b_g, and b_z are units. s_{ti} is the output at time ti.⊙ mentions the Hadamard creation. In BiRNNs, each GRN cell computes the hidden direction $\overrightarrow{s_{ti-1}}$ and

the backward direction $\overleftarrow{s_{ti+1}}$. As a result, the BiGRU can benefit from features in both directions. The following equation explains the concept of BiRNNs.

$$s_{ti} = \overrightarrow{s_{ti-1}} \oplus \overleftarrow{s_{ti+1}} \quad (22)$$

Where $\oplus$ suggests the element-wise sum for the vectors from both commands, left and right.

3.6.3. Basic CNN architecture

Another CNN we have is simple and has only one CNN layer. The ReLU activation is applied, and the layer contains 32 filters, each with a size of 4. Every CNN filter uses the filter map W to convolve the input x with the feature map CV.

$$F = CV(W, X) \quad (23)$$

Then, the bias unit b enhances the chin map activation function, as exposed in (24)

$$Relu(F + b) \quad (24)$$

At the outset of training, the CNN layer's filters are randomly initialised using the Glorot normal initialiser. The output is then sent to the film, which uses a pool size of 2 and applies dropout at 50%.

3.6.4. Multichannel

Three networks—Convolutional Neural Networks (CNNs), BiRNNs, and a transformer block—are used in this paper's multichannel deep learning model to process the input concurrently. The output is aggregated and sent to two fully coupled dense layers, one from each network. There are 60 neurons in the first dense layer and 30 in the second. The softmax classifier is then given the output. Softmax categorises the input data into two groups: cyberbullying and non-cyberbullying. The outputs of the three nets are mutually using the same layer. If vectors U, I, O, where *U* is as follows: $U = \{U_1, U_2, .. U_i\}$, the vector I is as follows: $I = \{I_1, I_2, .. I_i\}$, and the vector O is as follows: $O = \{O_1, O_2, .. O_i\}$, then combine them into one vector as shadows:

$$V = Conc(U, I, O) \quad (25)$$

The consequence V would be as shadows: $V = \{U_1, U_2, U_i, I_1, I_2, I_i, O_1, O_2, O_i\}$
Dense layer computes its output DL as shadows:

$$DL = F(\sum_i w_i . x_i + b) \quad (26)$$

Where w is the activation function, it is performed, after which b is added. Given that this model addresses a binary classification problem, a binary loss function was used. To understand how the entropy loss function works, look no further than Equation (27).

.

$$BC = -\sum_c y_c . \log(s_\theta(x)_c) \quad (27)$$

Where c represents the class index, this problem is divided into two classes. With the current input data (x) and the anticipated probability (s) for class c, we can determine the proper value (y) for class c. Additionally, we optimise the network using the Adam optimiser.

4. Result and Discussion

This section presents the outcomes of the suggested model. A desktop computer equipped with a 3.20 GHz Intel Core (i7) 8700U CPU, 4 GB of NVIDIA GeForce GTX 1050 Ti graphics, and 16 GB of main memory was used to perform the experimental research for this work. The program was developed using Python 3.7 and the associated libraries. By allocating 70% of the data for training and 30% for testing, we were able to boost the network's learning capacity while preventing overfitting and deterioration. Using L2 regularisation, the hyperparameters that the WHO procedure optimises are: a learning rate of 0.4. To make sure the model learned consistently, we set the training duration to 100 epochs. After fifty epochs, the learning and training curves converged, indicating that the model had reached a stable state.

4.1. SDN Simulation

Using the commercial edition of EstiNet, which combines the benefits of a simulator and an emulator (https://www.estinet.com/ns/?page_id = 21140), this study's SDN version. This means that the SDN controller allocates resources to mimic the host's test network traffic. It is also possible to programmatically control the occurrence of network events. This means that the simulated environment's timing resources are synchronised with the real system time. The software's graphical user interface enables the modelling of the actual network's design and packet transmission behaviour. In its software imitation, EstiNet displays the actual network setup and packet transmission characteristics. The EstiNet simulator includes OpenFlow, which supports the SDN protocol. In software-defined networking, controllers and switches communicate via the OpenFlow protocol. There are two possible configurations for the control plane in an OpenFlow switch: one in which the control plane and data plane overlap (in-band control plane), and the other in which they are distinct (out-of-band control plane). In the EstiNet simulation's software-defined networking (SDN) environment, the open-source controller software may run and take control of the OpenFlow virtual switch. Within the simulated environment, it adheres to the OpenFlow protocol and supports activities. With the simulator's execution and playback modes, users can observe how FlowTable, GroupTable, and MeterTable are updated. Whether it's the packet transmission latency or loss rate of a wired network, or the energy consumption or distance covered by a wireless network, EstiNet can programmatically regulate these network conditions.

Using a variety of wireless and cable communication protocols, it transmits packets over a network. The most common application of time dilation is to model changes in the rate of time passage. Reducing the system clock while maintaining accuracy allows for the simulation to run. With the help of a packet event translator or the event processing core, the simulated network's virtual network time is automatically updated, making it more accurate and efficient than traditional time dilation approaches. To execute the program code, EstiNet utilises Linux containers. Construct virtual networks to build hybrid virtual/physical simulations. The EstiNet simulator offers a testing environment for networks, encompassing both real and virtual devices. For testing purposes, both types of devices can communicate with each other. Our ML, in addition to DL representations, is deployed in the SDN for specific purposes, as it supports the Python language. As a tool for DDoS attack simulation under Linux, the Hing tool simulates SYN flood and random-source attacks. It is a packet assembler/analyzer. An attacker may launch a (DDoS) assault by sending several connection requests and a flood of random packets to the target system from various sources. The environmental setting is shown in Figure 5.

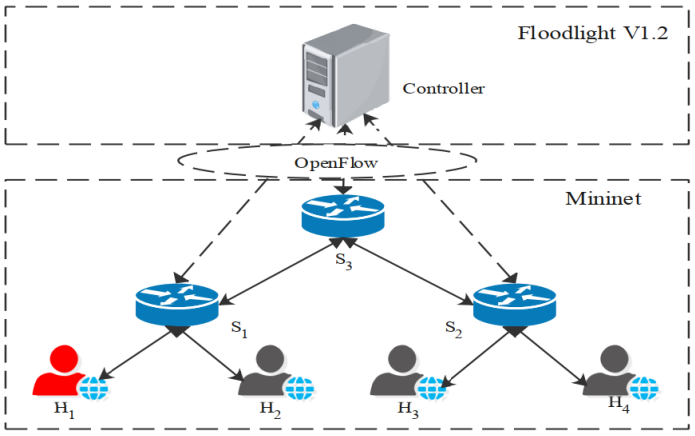

Figure 5: Environment setup

4.2. Performance Measure

A performance indicator is a reliable way to assess the proposed system's efficiency by regularly evaluating results and outcomes. Additionally, the procedure of educating, assembling, and analysing data regarding the outcomes of group or solo activities is a measure of efficacy. See Table 3 for the confusion measure used to assess the binary input; it's explained in the following words:

Table 3: Confusion matrix

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Positive (FP)
Actual Negative	False Negative (FN)	True Negative (TN)

True positives are actual occurrences that have been appropriately labelled as positive. Once a genuine positive is recognised as a false negative, the FP will state as much. Just as FN are actual negative occurrences, TN are true negatives that were incorrectly classed as positive. The mathematical equations of ACC, F-m, PR, and RC are meant in Eqs. (28), (29), (30), and (31).

$$\text{Accuracy} = \frac{\text{TN+TP}}{\text{TP+TN+FN+FP}} \times 100 \quad (28)$$

$$\text{F} - \text{measure} = \frac{\text{2TP}}{\text{(2TP+FP+FN)}} \times 100 \quad (29)$$

$$\text{Precision} = \frac{\text{TP}}{\text{(FP+TP)}} \times 100 \quad (30)$$

$$\text{Recall} = \frac{\text{TP}}{\text{(FN+TP)}} \times 100 \quad (31)$$

Where true positive is symbolised as TP, and TN expresses true negative, and then FP is spoken as false positive, and FN is uttered as false negative.

Table 4: Validation analysis of proposed feature selection

Methodology	Parameter Evaluation			
	Precision (%)	Recall (%)	Accuracy (%)	F-measure (%)
PSO	78.14	89.92	88.67	80.72
ACO	70.91	72.69	72.33	75.17
MBO	64.17	86.66	80.24	68.28
GWO	91.94	95.61	76.86	95.78
WHO	96.84	97.24	94.13	98.97

Table 4 represents the Validation Analysis of the Proposed Feature Selection. In the analysis of the PSO scheme, a precision of 78.14, a recall of 89.92, an accuracy of 88.67, and an F-measure of 80.72 correspondingly. Then, the ACO scheme achieved precision of 70.91%, recall of 72.69%, accuracy of 72.33%, and an F-measure of 75.17%. Then, the MBO scheme achieved a precision of 64.17%, a recall of 86.66%, an accuracy of 80.24%, and an F-measure of 68.28% (Figure 6).

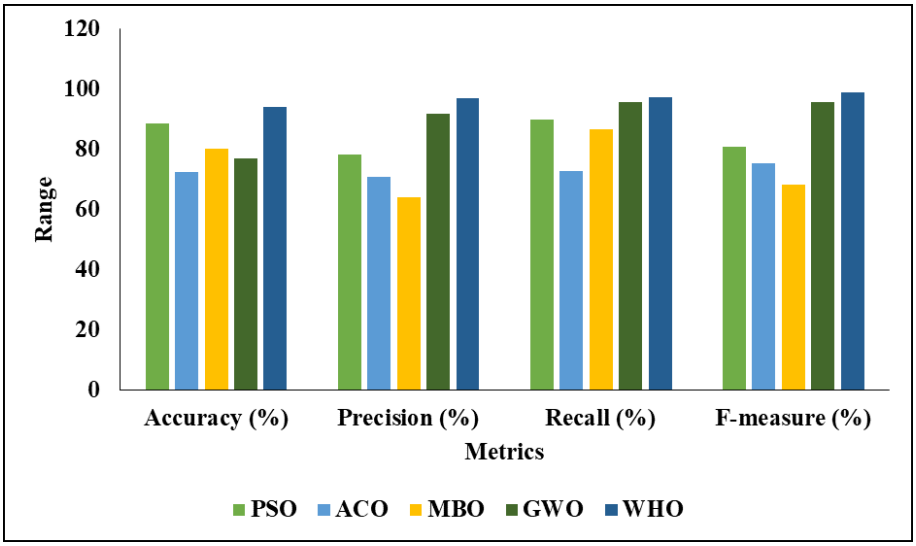

Figure 6: Visual representation of proposed feature selection models

Then the GWO scheme achieved a precision of 91.94, a recall of 95.61, an accuracy of 76.86, and an F-measure of 95.78. Then, the WHO scheme achieved a precision of 96.84, a recall of 97.24, an accuracy of 94.13, and an F-measure of 98.97 (Figure 7).

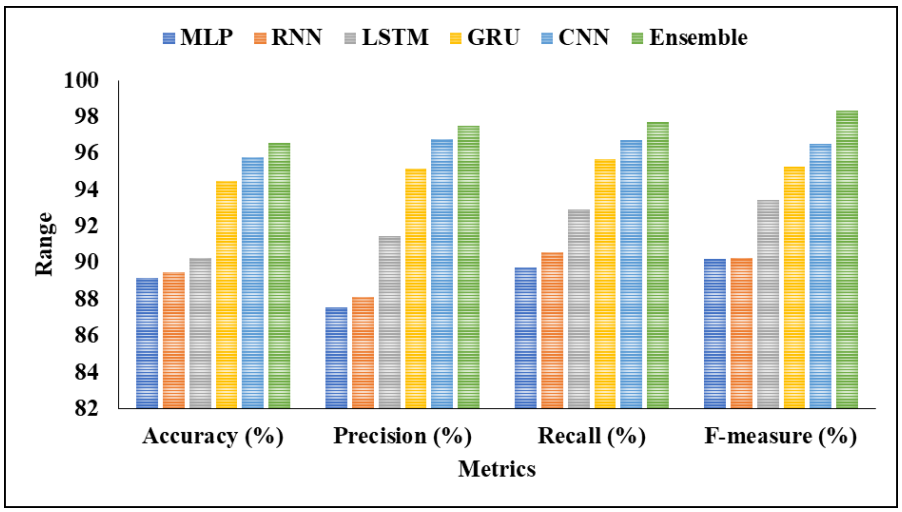

Figure 7: Graphical description of various classifiers

4.3. Advantages and Restrictions

This section deliberates the compensations and limits of the proposed model.

4.3.1. Advantages

The projected scheme has the subsequent compensations:

- No additional hardware is needed; the suggested model is an add-on module for the SDN controller that blocks DoS attacks targeting SDN. It acts as a barrier between the controller and other controller programs, and its design aligns with OpenFlow standards. It is unique in that it requires no changes to the data plane or supplementary hardware.
- Quick anomaly detection: the watchdog establishes connections between hosts and switches for mapping purposes. Using statistical models enables the quick identification of network irregularities.
- Low overhead and effective: the monitor only activates the detector to identify attacks when it detects an abnormality, which drastically reduces the system's overhead compared to other DoS detection methods. Concurrently, the detector considers the characteristics of the address assault and extracts the most indicative features to enhance the detection algorithm's performance.

4.3.2. Limitations

The projected scheme also has some limits, which are briefly discussed as shadows:

- Doesn't target particular flows—instead, after an assault has been verified, the proposed model's mitigator will set up flow rules to prevent further communication from the infected host.

Second, it can't identify attacks on the application plane; the suggested model was primarily designed to safeguard data-plane and control-plane components, such as flow tables and controllers. Malicious apps or data leaks that occur on the request plane or via northbound interfaces (e.g., RESTful API) are not protected.

5. Conclusion

In this article, we examine ways to protect Software-Defined Networking (SDN) settings against denial-of-service (DoS) attacks that exploit fake or falsified source addresses. These attacks are quite dangerous because SDN systems separate the control and data planes, making the controller an ideal target for overload attacks. To fix this problem, the research suggests an adjustable, protocol-free security framework that uses ensemble deep learning. This model uses several learning techniques to make it more robust and accurate at distinguishing real from fake traffic. The WHO technique is a way to extract and select features that help the model identify the most important flow characteristics, thereby improving its ability to detect. The system is a small add-on for SDN controllers that works with OpenFlow messages and detailed flow statistics. The model looks for anomalous patterns in data exchanges between switches and hosts that could indicate an ongoing attack. When the model

detects suspicious activity, it automatically configures flow rules at the switch level to block hazardous packets from reaching the controller. This method greatly reduces the controller's workload and mitigates the effects of DoS attacks. The study's practical results show that the proposed ensemble deep learning model achieves high accuracy, low overhead, and low computational cost. It can detect and stop many types of DoS attacks targeting SDN infrastructure. Overall, the solution provides SDN security with a framework that can grow, perform well, and adapt to protect against spoofed-source DoS attacks.

Acknowledgment: N/A

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request, subject to approval from all contributing authors.

Funding Statement: The authors affirm that this research received no external funding of any kind.

Conflicts of Interest Statement: All authors declare that they have no conflicts of interest associated with this work.

Ethics and Consent Statement: This study was conducted in strict accordance with established ethical standards, and informed consent was obtained from all participants with full agreement from all authors involved.

References

1. M. Aslam, D. Ye, A. Tariq, M. Asad, M. Hanif, D. Ndzi, S. A. Chelloug, M. A. Elaziz, M. A. A. AI-Qaness, and S. F. Jilani, "Adaptive machine learning based distributed denial-of-services attacks detection and mitigation system for SDN-enabled IoT," *Sensors*, vol. 22, no. 7, p. 2697, 2022.
2. K. Sritharan, R. Elagumeeharan, S. Nakkeeran, A. Mohamed, B. Ganegoda, and K. Yapa, "Machine learning based distributed denial-of-services attacks detection and mitigation testbed for SDN-enabled IoT devices*," in Proc. 2022 13th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT)*, Kharagpur, India, 2022.
3. A. A. Alashhab, M. S. M. Zahid, A. Muneer, and M. Abdullahi, "Low-rate DDoS attack detection using deep learning for SDN-enabled IoT networks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 11, p. 371, 2022.
4. P. Kumari and A. K. Jain, "SDN-Enabled IoT to Combat the DDoS Attacks," *in Communication and Intelligent Systems (ICCIS 2022), H. Sharma, V. Shrivastava, K. K. Bharti, and L. Wang, Eds., Lecture Notes in Networks and Systems, Springer*, Singapore, 2023.
5. M. S. A. Muthanna, R. Alkanhel, A. Muthanna, A. Rafiq, and W. A. M. Abdullah, "Towards SDN-enabled, intelligent intrusion detection system for Internet of Things (IoT)," *IEEE Access*, vol. 10, no. 2, pp. 22756–22768, 2022.
6. W. G. Negera, F. Schwenker, T. G. Debelee, H. M. Melaku, and Y. M. Ayano, "Review of botnet attack detection in SDN-enabled IoT using machine learning," *Sensors*, vol. 22, no. 24, p. 9837, 2022.
7. M. Maray, H. M. Alshahrani, K. A. Alissa, N. Alotaibi, A. Gaddah, A. Meree, M. Othman, and M. A.Haza, "Optimal deep learning driven intrusion detection in SDN-enabled IoT environment," *Comput. Mater. Continua*, vol. 74, no. 3, pp. 6587–6604, 2022.
8. M. A. Razib, D. Javeed, M. T. Khan, R. Alkanhel, and M. S. A. Muthanna, "Cyber threats detection in smart environments using SDN-enabled DNN-LSTM hybrid framework," *IEEE Access*, vol. 10, no. 3, pp. 53015–53026, 2022.
9. S. Muzafar and N. Jhanjhi, "DDoS attacks on software defined network: Challenges and issues," *in Proc. 2022 Int. Conf. Bus. Anal. Technol. Secur. (ICBATS)*, Dubai, United Arab Emirates, 2022.
10. D. Javeed, T. Gao, M. T. Khan, and D. Shoukat, "A hybrid intelligent framework to combat sophisticated threats in secure industries," *Sensors*, vol. 22, no. 4, p. 1582, 2022.
11. H. M. Chuang, F. Liu, and C. H. Tsai, "Early detection of abnormal attacks in software-defined networking using machine learning approaches," *Symmetry*, vol. 14, no. 6, p. 1178, 2022.
12. A. A. Ibrahim, A. R. Atayee, and I. A. Lawal, "SDN multi-domain supervisory controller with enhanced computational security count," *J. Telecommun. Electron. Comput. Eng. (JTEC)*, vol. 14, no. 2, pp. 23–29, 2022.
13. K. Bobrovnikova, O. Savenko, S. Lysenko, and I. Hurman, "IoT cyberattack detection approach based on energy consumption analysis," *in Proc. 2022 12th Int. Conf. Dependable Syst. Services Technol. (DESSERT)*, Athens, Greece, 2022.
14. S. Asaithambi, L. Ravi, H. Kotb, A. H. Milyani, A. A. Azhar, S. Nallusamy, V. Varadarajan, and S. Vairavasundaram, "An energy-efficient and blockchain-integrated software defined network for the industrial Internet of Things," *Sensors*, vol. 22, no. 20, p. 7917, 2022.
15. M. K. Mohammed, Z. A. Abod, and A. A. Abdullah, "Secure SDN traffic based on machine learning classifier," *LC Int. J. STEM*, vol. 3, no. 1, pp. 118–128, 2022.

16. S. K. Dash, S. Dash, S. Mahapatra, S. N. Mohanty, M. I. Khan, M. Medani, S. Abdullaev, and M. Gupta, "Enhancing DDoS attack detection in IoT using PCA," *Egypt. Inform. J.*, vol. 25, no. 3, p. 100450, 2024.
17. H. Setia, A. Chhabra, S. K. Singh, S. Kumar, S. Sharma, V. Arya, B. B. Gupta, and J. Wu, "Securing the road ahead: Machine learning-driven DDoS attack detection in VANET cloud environments," *Cyber Secur. Appl.*, vol. 2, no. 1, p. 100037, 2024.
18. A. A. Najar and S. M. Naik, "Cyber-secure SDN: A CNN-based approach for efficient detection and mitigation of DDoS attacks," *Computers and Security*, vol. 139, no. 4, p. 103716, 2024.
19. M. A. Hossain and M. S. Islam, "Enhancing DDoS attack detection with hybrid feature selection and ensemble-based classifier," *Measurement: Sensors*, vol. 32, no. 4, p. 101037, 2024.
20. W. G. Gadallah, H. M. Ibrahim, and N. M. Omar, "A deep learning technique to detect distributed denial of service attacks in software-defined networks," *Computers and Security*, vol. 137, no. 2, p. 103588, 2024.
21. C. M. Nalayini, J. Katiravan, S. Geetha, and J. C. Eunaicy, "A novel dual optimized IDS to detect DDoS attack in SDN using hyper tuned RFE and deep grid network," *Cyber Security and Applications*, vol. 2, no. 2, p. 100042, 2024.
22. A. A. Bahashwan, M. Anbar, S. Manickam, G. Issa, M. A. Aladaileh, B. A. Alabsi, S. D. A. Rihan, "HLD-DDoSDN: High and low-rates dataset-based DDoS attacks against SDN," *PLOS One*, vol. 19, no. 2, p. e0297548, 2024.
23. E. Q. Effah, E. O. Osei, and A. Tetteh, "Hybrid approach to classification of DDoS attacks on a computer network infrastructure," *Asian J. Res. Comput. Sci.*, vol. 17, no. 4, pp. 19–43, 2024.
24. O. Pandithurai, C. Venkataiah, S. Tiwari, and N. Ramanjaneyulu, "DDoS attack prediction using a honey badger optimization algorithm based feature selection and Bi-LSTM in cloud environment," *Expert Syst. Appl.*, vol. 241, no. 5, p. 122544, 2024.
25. A. Kaur, C. R. Krishna, and N. V. Patil, "K-DDoS-SDN: A distributed DDoS attacks detection approach for protecting SDN environment," *Concurrency Comput.: Pract. Exper.*, vol. 36, no. 3, p. e7912, 2024.