A Stacking Ensemble Model with Enhanced Feature Selection for Distributed Denial-of-Service Detection in Software-Defined Networks

Tariq Emad Ali
Information and Communication Engineering, Al-Khwarizmi College of Engineering, University of Baghdad, Iraq | National Advanced IPv6 Centre, Universiti Sains Malaysia, Gelugor, Penang, Malaysia
tariqemad@kecbu.uobaghdad.edu.iq (corresponding author)

Yung-Wey Chong
School of Computer Sciences, Universiti Sains Malaysia, Gelugor, Penang, Malaysia | National Advanced IPv6 Centre, Universiti Sains Malaysia, Gelugor, Penang, Malaysia
chong@usm.my

Selvakumar Manickam
National Advanced IPv6 Centre, Universiti Sains Malaysia, Gelugor, Penang, Malaysia
selva@usm.my

Mohd Najwadi Yusoff
School of Computer Sciences, Universiti Sains Malaysia, Gelugor, Penang, Malaysia | National Advanced IPv6 Centre, Universiti Sains Malaysia, Gelugor, Penang, Malaysia
najwadi@usm.my

Kok-Lim Alvin Yau
Lee Kong Chian Faculty of Engineering and Science (LKCFES), Universiti Tunku, Selangor, Kuala Lumpur, Malaysia
yaukl@utar.edu.my

Alwahab Dhulfiqar Zoltan
Faculty of Informatics, Eotvos Lorand University, Budapest, Hungary
dolfi@inf.elte.hu

Received: 11 September 2024 | Revised: 9 October 2024 | Accepted: 23 November 2024

 DOI: https://doi.org/10.48084/etasr.8976

ABSTRACT

The proliferation of Distributed Denial of Service (DDoS) attacks poses a significant threat to network accessibility and performance. Traditional feature selection methods struggle with the complexity of network traffic data, leading to poor detection performance. To address this issue, a Genetic Algorithm Wrapper Feature Selection (GAWFS) is proposed, integrating Chi-squared and Genetic Algorithm (GA) approaches with a correlation method to select the most correlated features. GAWFS effectively reduces feature dimensions, eliminates redundancy, and identifies crucial and correlated features for classification. Detection accuracy is further improved by employing a stacking ensemble model, combining Multi-Layer Perceptron (MLP) and Support Vector Machine (SVM) as base models, with Random Forest (RF) as the metamodel. The proposed classifier achieves impressive accuracies of 99.86% for training data and 98.89% for test data, representing improvements of approximately 5% and 40%, respectively, over previous studies. The training time was also reduced to 2,593 s, a substantial improvement of

approximately 29.92%. Validation on various benchmark datasets confirmed the efficacy of the proposed approach, underscoring the importance of the enhanced feature selection method and the stacking ensemble model against DDoS attacks.

Keywords-distributed denial-of-service; software-defined networking; genetic algorithms; stacking ensembles

I. INTRODUCTION

The increasing number of devices connected to the Internet poses significant security challenges, with Distributed Denial of Service (DDoS) attacks emerging as one of the most pervasive and sophisticated threats. DDoS attacks overwhelm systems by flooding them with traffic, leading to degraded performance or even network failure. According to Arbor Networks, approximately 2,000 DDoS attacks occur globally each day, with Q1 2021 alone witnessing a 31% rise compared to 2020, totaling 2.9 million incidents. Notable cases, such as the 2.3 Tbps attack on AWS in 2020 and the 1.3 Tbps attack on GitHub in 2021, underscore the gravity of this threat [1-4]. Common DDoS methods include Internet Control Message Protocol (ICMP) floods, SYN floods, Hypertext Transfer Protocol (HTTP) floods, and User Datagram Protocol (UDP) floods. Each of these methods targets specific vulnerabilities, such as exploiting the TCP handshake protocol (SYN flood) or sending excessive pings to the victim machine (ICMP flood) [5]. With the rise of Software-Defined Networking (SDN), which offers flexibility and centralized control, attackers are increasingly targeting SDN controllers due to their critical role in managing network traffic [6, 7]. The centralized nature of SDN makes it susceptible to DDoS attacks, as attackers can disrupt the normal functioning of controllers by bombarding them with malicious traffic. The SDN architecture comprises three layers: Infrastructure, Control, and Application [8, 9].

Extensive research has been conducted on the application of Feature Selection (FS) methods in combination with Machine Learning (ML) and Deep Learning (DL) algorithms to detect DDoS attacks. These techniques aim to enhance the performance of detection models by selecting the most relevant features from large datasets, thus reducing dimensionality and improving accuracy [10]. However, many studies overlooked the critical importance of FS techniques, focusing on developing classification models alone. In [11], an ensemble framework was proposed for FS that combined the outputs of seven well-known FS methods, demonstrating superior accuracy and minimal false alarms on the NSL-KDD dataset. In [12], a CNN-CNN model was introduced to detect IoT network attacks, outperforming other DL algorithms in the BoT IoT 2020 dataset. In [13], a three-stage DL algorithm was used for DDoS detection on SDN controllers, achieving high accuracy, precision, and F1 scores. In [14], a DL method was developed for detecting ICMPv6 flooding DDoS attacks, achieving notable accuracy and outperforming existing approaches. In [15], Pelican was introduced, a deep neural network that performed exceptionally well on the NSL-KDD and UNSW-NB15 datasets. In [16], a hybrid model was proposed, using Cu-DNNGRU and Cu-BLSTM for DDoS detection and offering a scalable and cost-effective approach for threat detection in IoT environments. However, this model faced challenges with hyperparameter tuning and generalization to unseen data. In [17], a hybrid CNN-LSTM model was proposed for DDoS attack detection, but its complexity posed difficulties. In [18], a hybrid CNN-LSTM model was also deployed for botnet attack detection, outperforming other methods in accuracy and correlation. In [19], an adaptive ensemble learning model was proposed for intrusion detection, showing improved accuracy compared to individual classifiers. In [20], a Performance Accelerator Algorithm (PAA) was developed that used Decision Tree (DT), Random Forest (RF), and K-Nearest Neighbor (KNN) as base models and Deep Neural Network (DNN) as metamodel. Table I provides a comprehensive overview of the methods used in previous studies to detect DDoS attacks.

Despite significant advances in DDoS detection techniques, several challenges remain. Many existing FS methods, such as those proposed in [11, 12] focus primarily on improving accuracy but often employ fixed thresholds for feature selection. This fixed threshold may not adapt well to datasets of varying sizes, potentially leading to the inclusion of irrelevant features or the omission of critical ones. Furthermore, some models rely heavily on CNN for feature selection, which, while powerful, is prone to overfitting and requires large labeled datasets for training [11-14]. In response to these limitations, this study proposes a novel FS method aimed at improving adaptability, robustness, and accuracy. The proposed method systematically selects relevant features based on their correlation, ensuring a balanced approach across different types of data. The proposed FS method leverages both the Chi-squared (Chi2) test and the Genetic Algorithm (GA), followed by Pearson's correlation analysis, to identify highly correlated features while discarding irrelevant or duplicated ones. This enhanced FS technique addresses issues related to overfitting, computational complexity, and adaptability across various datasets. To overcome the challenges outlined in the literature, this study makes the following contributions:

- Develops an enhanced FS method using Chi2 and GA, followed by Pearson correlation, to select highly correlated features.
- Introduces a high-performance stacking ensemble model for DDoS detection, combining multiple base classifiers to improve accuracy and reduce false positives.
- Comprehensively evaluates the model using benchmark datasets, demonstrating its superiority over existing techniques in terms of accuracy, precision, and training time.
- Provides a novel approach to reduce training time and computational costs, making the model suitable for real-time DDoS detection in resource-constrained environments.

TABLE I. METHODS EMPLOYED IN PREVIOUS STUDIES FOR DETECTING DDOS ATTACKS

Ref.	Dataset	Feature selection	Model classification	Contributions	Gaps
[11]	NSL-KDD	Proposed EnFS that combined the outputs of cfs, ReliefF, SU, GR, Chi2, IG, and OneR using the voting technique.	SVM, NB, DT, NN, and LR for individual data classification with various ensemble techniques, voting, LR, NB, NN, DT, and SVM.	Reduced the dimensionality of the dataset by identifying important or relevant features from the original dataset.	Used a fixed threshold to determine the final set of selected features. The fixed threshold can be extended to a dynamic threshold to accurately determine the final set of selected features.
[12]	BoT-IOT	Leveraged CNN to select feature sets.	CNN	Minimize data dimensionality.	Can be extended to address overfitting, sensitivity to hyperparameters, the need for large amounts of labeled data, and computational complexity.
[13]	DDOS-attack-SDN	Proposed IGR and Chi2	RNN	Captured complex feature interactions and reduced dimensionality	Used a fixed threshold to determine the final set of selected features. The fixed threshold can be extended to a dynamic threshold to accurately determine the final set of selected features.
[14]	Synthetic dataset	Ensemble FS technique that uses Chi2 and IGR methods.	LSTM	Reduce dataset dimensionality.	Used a fixed threshold to determine the final set of selected features. The fixed threshold can be extended to a dynamic threshold to accurately determine the final set of selected features.
[15]	NSL-KDD and UNSW-NB15	N/A	Hybrid model using SimpleRNN + CNN.	Tuned a DNN classifier	Can be extended to address the insufficiency of training data, low capacity of computing resources, CNN processing non-image datasets, and high FP. Did not discuss the FS process.
[16]	CICIDS2018	N/A	Cu- BLSTM	Proposed a highly cost-effective and scalable solution for threat detection in IoT, compared with existing works for better performance.	Can be extended to address the high complexity of hybrid models, overfitting, and generalization, and reduce large data and resource requirements. FS can be added.
[17]	UNSW-NB15	N/A	CNN + LSTM	Proposed a detection model using DL and compared it with CNN and LSTM methods.	Can be extended to address the insufficiency of training data, low capacity of computing resources, and CNN processing non-image datasets. Did not discuss the FS process.
[18]	N/A	N/A	CNN+LSTM	Detect malware of IoT devices using emerging technology.	Can be extended to address the insufficiency of training data, low capacity of computing resources, and processing of non-image datasets. Did not discuss the FS process.
[19]	NSL-KDD	N/A	Ensemble learning DT, RF, KNN, and DNN using voting for output.	Combined the advantages of different algorithms to improve detection.	DNN requires a long time for detection. Dataset uses voting which is the prediction of every lower-level classifier, and the winner is the prediction with the most votes. This approach can be extended to address the high complexity of hybrid models, overfitting, and generalization, and reduce large data and resource requirements. FS can be added.
[20]	Synthetic	N/A	DT, RF, and KNN, and the final classifier was DNN	Compared the performance between the individual and the ensemble model.	Did not use an FS method, DNN required a long time for detection, and did not mention the method of prediction for the base models.

II. METHODOLOGY

A. Dataset and Preprocessing

This study utilized a dataset that contained various features related to DDoS attacks. Several preprocessing techniques were implemented to ensure the data is suitable for analysis and modeling, such as the imputation of missing values with zeros and the elimination of rows containing non-numeric values (NaN). Both the label encoder and one-hot encoding techniques were used to transform categorical data into a numerical format appropriate for ML, resulting in a machine-readable version of the dataset. These methods were crucial for ensuring interoperability with different DL models. Furthermore, FS and scaling were essential data preparation methods to create a successful model. Z-score normalization is a standardization technique that is used to scale data packets before their input into various deep-learning models. Using this procedure, the features are rescaled to have standard characteristics with a standard deviation of 1 and an average value of 0 [21]. Using the scikit-learn standard scaler module, feature scaling was achieved by converting independent components into a distribution with mean (μ) and standard deviation (σ) as shown in (1) and (2):

$$\mu = \frac{1}{N}\sum_{1=1}^{N}(x_i) \quad (1)$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad (2)$$

In this study, we used the public dataset called SDN_Dataset [22], containing a deferent type of DDoS attacks. Additionally we generated a dataset on the testbed. These two datasets were used to evaluate the proposed model and measure its performance in detection with seen and unseen data.

TABLE II. FEATURE LIST WITH IDS

Feature ID	Feature name	Feature ID	Feature name
1	dt	13	byteperflow
2	switch	14	pktrate
3	src	15	Pairflow
4	dst	16	Protocol
5	pktcount	17	port no
6	bytecount	18	tx bytes
7	dur	19	rx bytes
8	dur nsec	20	tx kbps
9	tot dur	21	rx kbps
10	flows	22	tot kbps
11	packetins	23	SYN Flag Count
12	pktperflow		

After training and testing of the proposed model with public data, it needs to be evaluated with unseen data close to realtime traffic. Therefore, we generated a new dataset using the testbed. Many tools were used in this step like the Tshark tool that is used to capture the traffic pass through and save it as pcap file and CICFlowMeter to convert this pcap file to csv format that can be read by the model in both training and testing stages. Using these datasets will increase model performance and make it more reliable with live networks.

B. Proposed Feature Selection Method

Most of the previous studies focused on the detection method rather than Feature Selection (FS). Therefore, in this study we developed a hybrid FS method called Genetic Algorithm Wrapper Feature Selection (GAWFS) (Figure 1). This method reduces the space size and the number of features needed to model the feed.

Initially, two feature selection algorithms, Chi2 and GA, to determine the priority of each feature in the dataset. These filters are chosen for their high computational efficiency, scalability to high-dimensional data, and low risk of overfitting. Despite their advantages, when used individually, these filters have limitations since they do not rely on a classifier, leading them to overlook feature dependencies by evaluating each feature in isolation. For each ranking algorithm, the average ranks (denoted R1 and R2) are calculated, and feature sets (FS1 and FS2) that exceed the mean priority value are selected. After selecting the features using each algorithm, the second step involves retaining only those features common to the two FS algorithms. After this step, the feature set is reduced to 15 of the original 23 features, as shown in Table III.

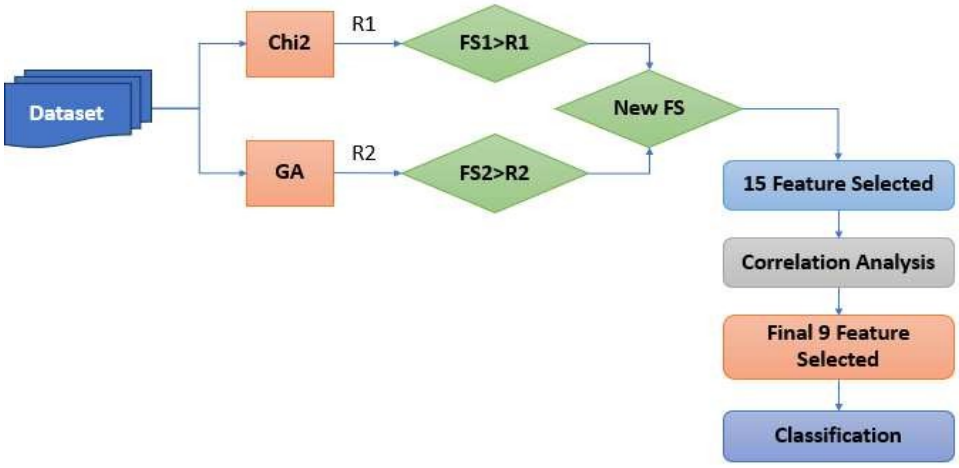

Fig. 1. Hybrid feature selection model.

TABLE III. FEATURE ANALYSIS RESULTS FOR THE SECOND STEP

Method	Mean value	Feature IDs above mean	Common features
Chi2	0.15352	23, 20, 11, 9, 8, 7, 6, 5, 4, 2, 10, 21, 1	20, 11, 9, 8, 7, 6, 5, 2, 10
GA	0.09768	14, 16, 15, 17, 18, 19, 10, 1, 21, 8, 2, 6	

The final step assesses whether these 15 features are correlated with each other, identifying and removing any redundant features. Selecting features that are correlated and are free from redundancy aims to improve classification performance. In the third step of filtering to select the most efficient features, correlation analysis is used to determine the relationships between them. Based on the correlation values obtained, three cases are identified:

- Case 1: A correlation coefficient near 0, whether positive or negative, indicates minimal or no relationship between the two features.
- Case 2: A correlation coefficient near +1, but not equal to +1, indicates a strong positive relationship between the features. If the correlation coefficient is exactly +1, it means that these two features are similar, and, hence, one of the features is dropped.
- Case 3: A correlation coefficient near -1 indicates a negative relationship between the two features, implying an inverse relationship between them.

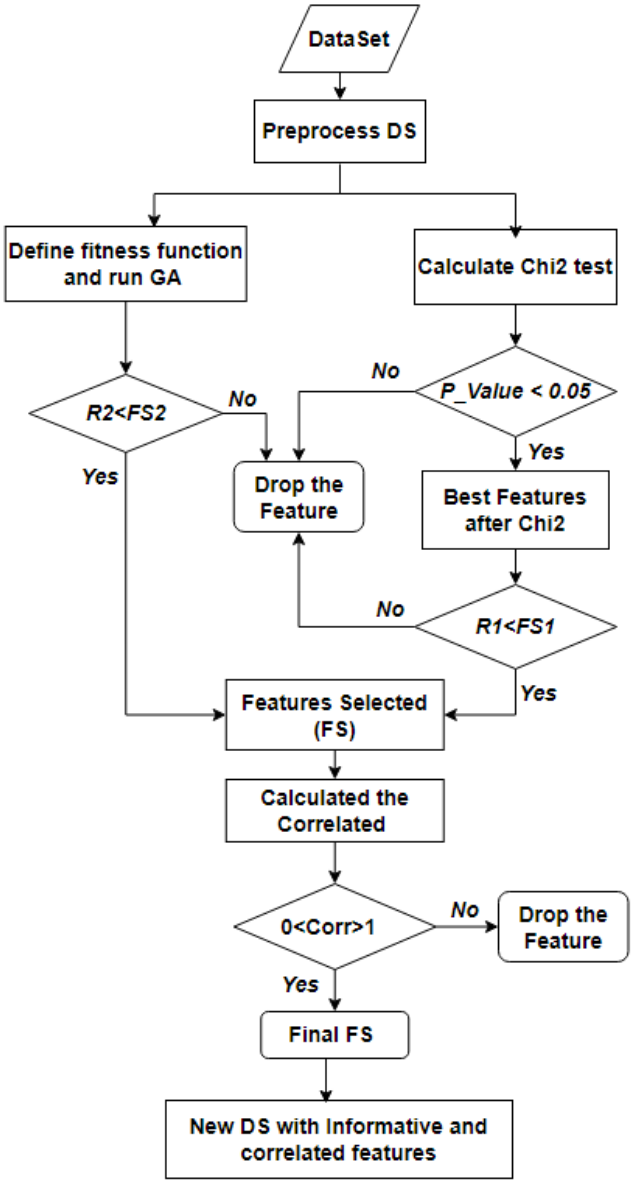

Fig. 2. Flowchart for the proposed GAWFS feature selection.

This approach examined the correlations between the features selected using the Pearson correlation method [23]. This method is utilized to quantify the strength of correlation between characteristics that are linearly connected. The Pearson correlation can be determined using:

$$P_{r(x,y)} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\,\Sigma x_i^2 - (\Sigma x_i)^2} * \sqrt{n\,\Sigma y_i^2 - (\Sigma y_i)^2}} \quad (3)$$

where $P_{r(x,y)}$ is the Pearson correlation coefficient between x and y, n is the number of observations, x_i is the value of x for the i^{th} observation, and y_i is the value of y for the i^{th} observation. The ultimate set of features for the classification model input is reduced to just 9 features, which are informative and more correlated with each other, as shown in Table IV, which shows the calculated correlated values using Pearson correlation. The proposed GAWFS method effectively addresses the limitations of adaptive correlated and informative FS, overfitting, sensitivity to hyperparameters, data scarcity, and computational complexity, rendering it a valuable tool for enhancing ML model performance in various real-world applications. Each step of the filter to produce the final set of features is illustrated in Figure 2 and described in Algorithm 1.

TABLE IV. THIRD STEP FOR FEATURE SELECTION USING PEARSON CORRELATION

FS_ID	20	11	9	8	7	6	5	2	10
20	1.00	0.307	0.320	0.302	0.278	0.278	0.271	0.307	0.125
11	0.307	1.00	0.804	0.836	0.612	0.612	0.825	0.804	0.004
9	0.320	0.804	1.00	0.907	0.854	0.854	0.893	0.804	0.069
8	0.302	0.836	0.907	01.00	0.717	0.717	0.977	0.836	0.029
7	0.278	0.612	0.854	0.717	1.00	0.706	0.706	0.612	0.149
6	0.278	0.612	0.854	0.717	0.706	1.00	0.706	0.612	0.149
5	0.271	0.825	0.893	0.977	00.706	0.706	1.00	0.825	0.031
2	0.307	0.804	0.804	0.836	0.612	0.612	0.825	1.00	0.004
10	0.125	0.004	0.069	0.029	0.149	0.149	0.031	0.004	1.00

```
Algorithm 1: Final Feature Selection
Require: A set of 23 features from the
  original dataset
Ensure:
1: A set of features whose value is
   greater than the mean priority value
2: A set of 15 features
3: A set of 11 final features
BEGIN:
For each of the five filters do
  Calculate the priority of each feature
  Assign the features with the highest
  priority value as Rank 1 and lowest
  priority as Rank 22
  Calculate the mean of the priority
  values obtained
end for
For each of the two filters do
  if Priority value of each feature > Mean
  Priority value then
    Select the feature
  else
    Discard the feature
  end if
end for
Output 1
For all the two filters do
  if a feature is present in all the two
  filters, then
    Select the feature
  else
    Discard the feature
  end if
end for
Output 2
For all the 15 features do
  Calculate the correlation using Pearson
  correlation
```

```
  if 0 < Correlation value < 1 then
    Select the feature
  else
    Discard the feature
  end if
end for
Output 3
END
```

C. Proposed Stacking Ensemble Model

This phase is dedicated to building a DDoS attack detection model within the SDN framework. Following the completion of the steps that involve FS, scaling, and dataset splitting for training and testing, the proposed classifier model is fed these datasets. The training of the detection model involves applying a classifier to a flow dataset containing various DDoS attack scenarios along with normal traffic. This comprehensive training dataset enables the classifier to learn and adapt to diverse attack possibilities and normal network patterns. As a result of this stage, a trained model is obtained to detect the presence of DDoS attacks within the SDN framework. Figure 3 shows the processes involved in the construction and testing of these detection models.

In this study, periodic retraining is proposed to enrich the system with new possible scenarios. Retraining should occur either when an increase in false positive alarms is noticed or when a new network configuration is implemented. The primary focus is to propose a novel model that detects DDoS attacks across different scenarios. This model aims to improve accuracy and robustness while addressing limitations found in traditional models. Figure 4 shows the processes of the stacking detection model and testing.

This study employed two different submodels, the Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP) algorithms, in the stacking ensemble framework as base models to leverage their strengths in capturing complex relationships and learning intricate patterns in the data.

As a first step both SVM and MLP models were trained, then the output from both models were fed to the meta model which uses the Sigmoid Activation Function (SAF). This model considers the final stage for classification from the output from SVM and MLP. In this study, we preferred RF as a final classifier than XGBoost, because the RF can work with nonlinear entry and its adaptable for any change in the data model and type.

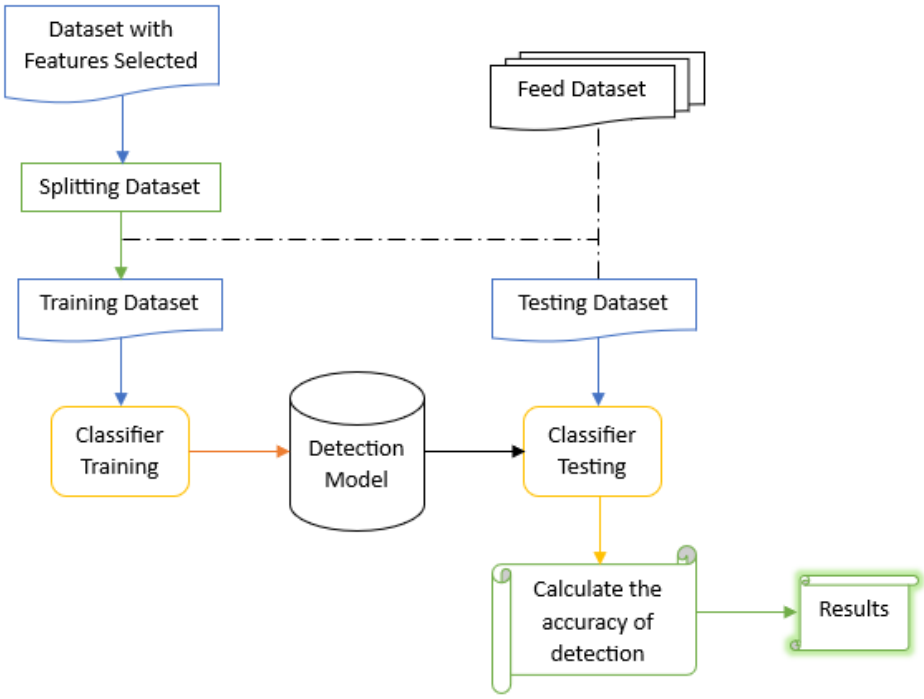

Fig. 3. Detection models training and testing processes.

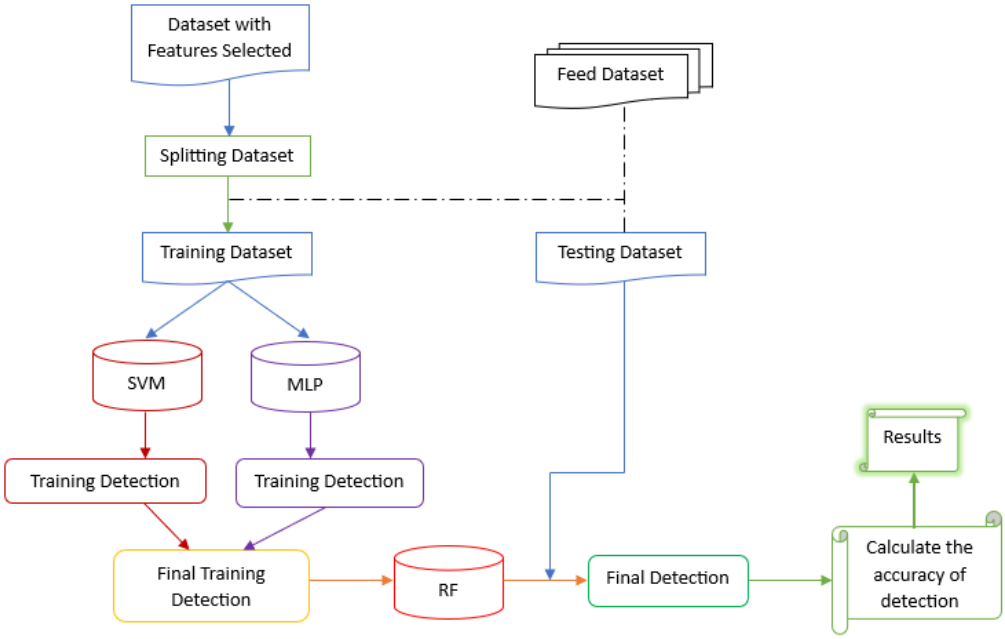

Fig. 4. Flowchart for stacking detection model for both training and testing processes.

III. EXPERIMENTAL SETUP

A. Hardware and Software Specifications

The experimental setup for this research included a Dell PC featuring a 64-bit architecture with an Intel Core i5-4200U CPU running at 1.60GHz and 12 GB of DDR4 RAM. The operating system was Ubuntu 22.04 LTS, and the system storage capacity was facilitated by a 1TB hard drive. The software environment included Python version 3.10.0 and Mininet version 2.3.0. Two different datasets were employed: a public dataset for training and testing the model, and a generated dataset to evaluate it.

B. Dataset

This study used two datasets to ensure a comprehensive evaluation of the proposed DDoS attack detection method: a public dataset and a synthetic dataset. First, the proposed model was trained and tested using the public SDN dataset, which is designed for SDN. The second step was to test the proposed model with an unseen synthetic dataset. As seen in Figure 5, the network topology consists of eight hosts and six switches arranged hierarchically.

The main reason for choosing this architecture is its compatibility with hierarchical network structures that require a root node. To avoid a loop between the OpenFlow (OF) switches [34], the POX controller [25, 26], which is Spanning Tree Protocol (STP) enabled, represents the root node in this instance. Furthermore, two OF vSwitches, designated s1 and s2, function as the core/distribution layer within the network architecture. A comprehensive mesh topology connects these OF vSwitches, improving redundancy and connection. The strategic design ensures that s1 and s2 effectively control network traffic and maintain network stability. Additionally, four more OF vSwitches (s3, s4, s5, s6) serve as the network access layer. Each of these vSwitches connects directly to two host devices, improving scalability and network management capabilities while optimizing network access for end-user devices.

In our topology IP address range 10.0.0.1-8 is assigned to each host. All hosts are connected to vSwitch version 3.2.1 with 10 Gbps connection between switches.

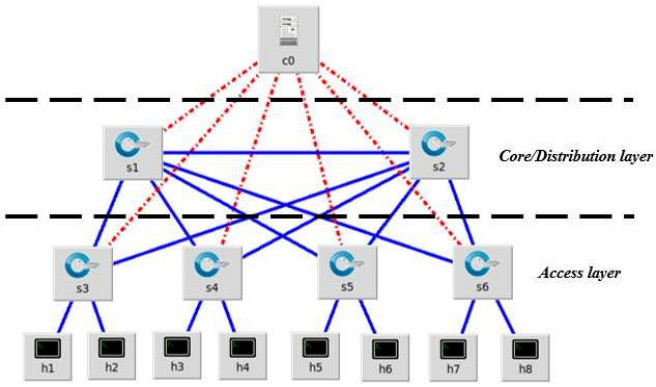

Fig. 5. Collapsed core design network topology.

POX 0.7.0 is the network controller employed, developed in Python, adhering to the OF standard and functioning as an open-source development platform. Additionally, the OF protocol version 1.3.0 was set up to be used with OpenFlow vSwitches, providing optimal compatibility with the POX controller and ensuring seamless control and communication between the controller and the switches. The DDoS attack dataset was created using Scapy [27]. The proposed model was evaluated using this newly created dataset to assess its performance. Creating realistic datasets is essential to validate ML models in the context of SDN, where dynamic threat

landscapes require creative defenses. Two unique situations, each with two separate forms of traffic, were developed to evaluate the detection mechanism:

- The initial focus of the investigation lies on "Normal traffic" as the first category under examination. To meticulously assess the impact of the proposed technique on this specific type of network traffic, a deliberate approach was followed to intentionally craft representative normal traffic scenarios alongside the concurrently simulated attack traffic. This intentional creation of normal traffic instances allows for a detailed and context-specific analysis, aiming to uncover any potential influences or alterations introduced by the suggested technique on regular network activities.
- "Attack traffic" refers to the second kind of traffic examined. It is identified by spoofing IP addresses and is directed at a particular IP address on the network.

IP addresses are assigned sequentially in the Mininet environment, starting with 10.0.0.1. The destination IP address for standard traffic is dynamically created between the ranges designated for that type of communication (e.g. 10.0.0.1 → 10.0.0.8). However, the RandIP() random function is used to create fake source IP addresses. These fake source IP addresses are used by one or more attackers on the network. The victim machine's IP address is explicitly set as the target IP address to emulate attack communications. This enables the generation of attack traffic focused on specific network targets. For instance, specifying 10.0.0.2 as the destination IP address redirects all packets to Host 2, as shown in Figure 6.

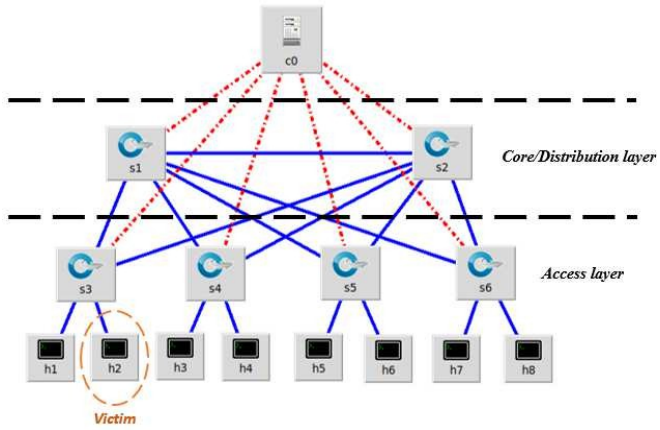

Fig. 6. Attack generation testbed.

With this configuration, traffic can be deliberately redirected to Host 2, simulating an attack scenario against that particular destination. To demonstrate the proposed model's performance in DDoS attack detection, two scenarios are employed to generate traffic in the network topology.

1) Test Case 1: Single Victim Under Attack Scenario

In this case, the attacker(s) use fake IP addresses to launch a three-flooding assault on a single host on the network. Host 2 is assigned the IP address 10.0.0.2. Host 3 (10.0.0.3) acts as an insider attacker and contributes 20% of the overall traffic flow, as seen in Figure 7. The inside attacker's assault flow accounts for 20%. Parallel regular traffic is also produced simultaneously to determine whether the proposed technique negatively affects it. This comprehensive method enables the evaluation of network security during an attack and its impact on normal network traffic. To ensure statistical reliability, the detection mechanism in this situation is evaluated using four distinct simulations with ten repetitions of each iteration. To assess the accuracy of the mechanism in distinguishing between malicious and normal traffic, the first simulation generates only normal traffic in the network, which means that all data traffic is normal with no attack traffic. In the second simulation, attack traffic is introduced into Host 2 at a rate of 25% of the total volume of traffic, along with regular traffic. The purpose of this test is to evaluate how well the system recognizes low-rate assault circumstances.

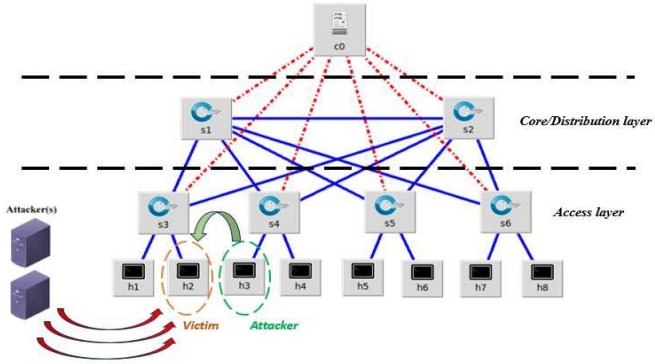

Fig. 7. Single victim under attack scenario.

In the third stage, the attack traffic rate was upgraded to 50 from total traffic. While in the 4th stage the attack traffic upgrade was more than 75% of the overall. This difference in the percentage of attack traffic will give an idea about the proposed model performance in detection accuracy with different attack rates.

2) Test Case 2: Multiple Victims under Attack Scenario

In this test, as shown in Figure 8, a wrong or fake IP address is used to attack the victim with many requests on the network.

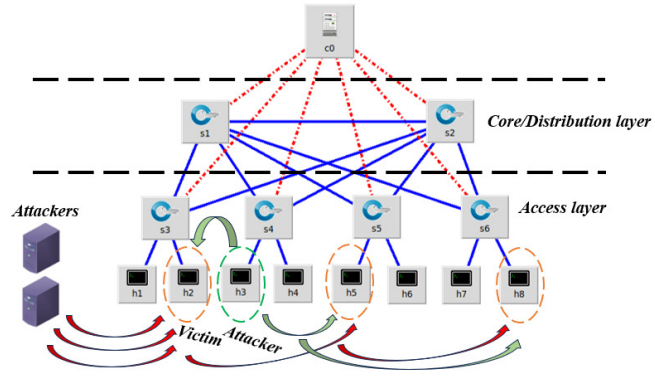

Fig. 8. Multiple victims under attack scenario.

We worked with this test type to test the effect of attack on multiple victims and the ways this will affect the performance of the proposed model for DDoS attack detection rather than signal attack detection. This scenario gives close evaluation of the model detection performance.

IV. EXPERIMENTAL RESULTS

A. Proposed Feature Selection Results

In this study, a dataset with (104,345 rows × 22 columns) was used and fed to the proposed FS method to find the informative and more correlated features that effect the proposed detection model.

For each ranking algorithm, the average rank is calculated, and feature sets that exceed the mean priority value are selected. Once the features are selected by each algorithm, only the features that are common across the two feature selection algorithms are selected. After this step, the feature set is reduced to 15 of the original 23 features. The final step assesses whether these 15 features are correlated with each other using the Pearson correlation method. The final output features $FS_{(Final)}$ are (104,345 rows × 11 columns) including dt, switch, dst, bytecount, due, dur_nsec, tot_dur, packetins, pktperflow, tx_bytes, tot_kbps, which are more correlated and informative. Therefore, the proposed GAWFS identifies effective, informative features and most correlated with each other. This combination aims to efficiently select features that are both correlated and informative for DDoS attack detection, enhancing the accuracy of the ML model. Following the application of the GAWFS approach, Figure 9 demonstrates an example of an output feature dataset, illustrating the dimensionality reduction from input features (104,345 rows × 22 columns) to output features (104,345 rows × 11 columns).

Out[87]:

	dt	switch	dst	bytecount	dur	dur_nsec	tot_dur	packetins	pktperflow	tx_bytes	tot_kbps
0	11425	1	0.0	48294064	100	716000000	1.010000e+11	1943	13535	143928631	0.0
1	11605	1	0.0	134737070	280	734000000	2.810000e+11	1943	13531	3842	0.0
...	...	...	...	...	...	...	...	...	...	...	...
104343	5262	3	0.0	3038	31	805000000	3.180500e+10	10	30	15209	2.0
104344	5262	3	0.0	3038	31	805000000	3.180500e+10	10	30	15099	2.0

104345 rows × 11 columns

Fig. 9. Output features using the proposed GAWFS method.

B. Performance Evaluation for Individual and the Proposed Stacking Models

To evaluate the performance improvements achieved by the stacking ensemble model, individual tests were performed on each model (SVM, RF, MLP) both with and without feature selection. The rationale behind this approach is to investigate the impact of feature selection on the models' performance and demonstrate how the proposed feature selection method can enhance the overall effectiveness of the models. Comparing the models' performance with and without feature selection can illustrate the benefits of the feature selection method. Additionally, feature selection becomes particularly useful when dealing with large datasets. Large datasets often contain a significant amount of redundant or irrelevant information that can hinder model training and performance. Reducing the dataset to only the most important features can decrease computational costs, speed up training times, and potentially improve the model's generalizability. This dataset is specifically designed for SDN networks, providing a relevant context for this study. Although it contains a relatively small number of features, it effectively reflects the behavior of the proposed model's performance, even when applied to larger datasets.

Tables V and VI show the comparison between different models using different matrices with and without the use of FS, respectively.

TABLE V. PERFORMANCE METRICS OF INDIVIDUAL MODELS WITH FEATURE SELECTION

Model	Accuracy	Precision	Recall	F1-score
SVM	97.0%	98.96%	95.0%	96.94%
RF	97.5%	97.98%	97.0%	97.49%
MLP	96.5%	96.97%	96.0%	96.48%

TABLE VI. PERFORMANCE METRICS OF INDIVIDUAL MODELS WITHOUT FEATURE SELECTION

Model	Accuracy	Precision	Recall	F1-score
SVM	95.0%	96.50%	95.80%	94.72%
RF	95.7%	95.00%	95.2%	95.60%
MLP	94.8%	95.20%	94.0%	94.59%

The performance of the stacked ensemble model, which combines SVM, RF, and MLP, was then evaluated. Table VII shows the performance metrics for the stacking model compared to the individual models.

TABLE VII. PERFORMANCE METRICS OF INDIVIDUAL MODELS AND STACKING MODELS (WITH FEATURE SELECTION)

Model	Accuracy	Precision	Recall	F1-score
SVM	97.0%	98.96%	95.0%	96.94%
RF	97.5%	97.98%	97.0%	97.49%
MLP	96.5%	96.97%	96.0%	96.48%
Stacking	99.86%	99.82%	99.71%	99.71%

It is important to note that the comparison of the stacking model was performed only with the proposed FS method. This is because FS significantly improves the performance of

individual models, making the stacking model built on these optimized models more effective and reliable. Comparing the stacking model without the proposed FS would not provide meaningful insight, as it would involve suboptimal individual models, leading to less accurate conclusions. As shown in Table VII, the stacking model achieves higher precision, recall, and F1-score compared to individual models. These improvements, although currently minor, are critical in improving the overall robustness and reliability of DDoS attack detection. The stacking model benefits from the strengths of individual models, leading to a more accurate and reliable detection system. The performance metrics are currently very close, primarily due to the size and complexity of the dataset used in this study. In practical applications with larger and more diverse datasets, the differences in performance between models with and without FS, as well as between individual models and the stacking model, will likely be more pronounced. FS can help manage the complexity of large datasets, reduce computational costs, and improve overall performance. The stacking model, which uses the strengths of multiple models, will provide even greater improvements in detection accuracy and reliability.

In summary, while the results show close performance metrics, the real benefits of the proposed FS method and stacking ensemble model will be more evident when applied to larger datasets. The reduction in dimensionality and focus on the most relevant features will lead to performance improvements in complex and large-scale data environments.

C. Comparison of Proposed Stacking Model with Previous Studies

The performance of the proposed stacking model was compared with other existing methods using the same dataset. Table VIII shows the performance metrics along with False Positives (FP) and False Negatives (FN).

TABLE VIII. COMPARISON OF THE PROPOSED MODEL WITH PREVIOUS STUDIES

Model	Accuracy	Precision	Recall	F1-score	FP	FN
[11]	98.6344%	98.642%	98.63%	98.63%	250	200
[12]	97.6616%	97.6651%	97.662%	97.663%	268	220
[13-14]	95.39%	95.394%	95.390%	95.3921%	499	463
Proposed	99.86%	99.82%	99.71%	99.7076%	23	34

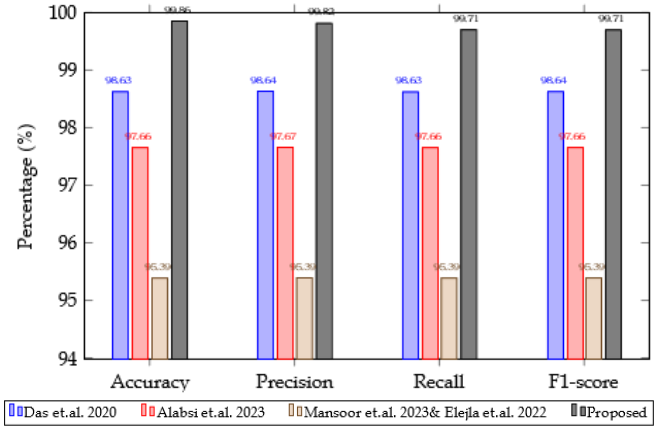

Fig. 10. Comparison of the proposed model with previous studies.

As shown in Table VIII and Figure 10, the proposed model demonstrated superior performance in all metrics compared to previous studies. Lower FP and FN values indicate a more reliable and accurate detection system. The proposed model's high accuracy and precision reflect its ability to correctly identify DDoS attacks, minimizing false alarms and missed detections. The substantial reduction in FP and FN values compared to other methods underscores the effectiveness of the approach in real-world applications. Using the same dataset as previous studies ensures a fair and accurate comparison. Although this dataset has a small number of features, it is specifically designed for SDN networks and adequately reflects the behavior of DDoS attack detection models. This choice allows us to demonstrate the potential of the proposed model, even with larger datasets, as the improvements in metrics are expected to be more pronounced with more complex data. For unseen data, as shown in Table IX and Figure 11, the proposed model also demonstrated superior performance in single-victim scenarios.

TABLE IX. PERFORMANCE METRICS FOR SINGLE ATTACK SCENARIO

Method	Accuracy	Precision	Recall	F1-score	FP	FN
[17]	69.75%	68.28%	67.90%	68.61%	530.75	524.25
[28]	79.47%	79.56%	78.98%	79.38%	444.75	441
[29]	78.31%	78.62%	78.33%	78.09%	446.75	429.25
[30]	63.44%	64.18%	63.01%	62.81%	615.25	610.75
Proposed	99.02%	98.84%	99.05%	99.02%	39.5	44.75

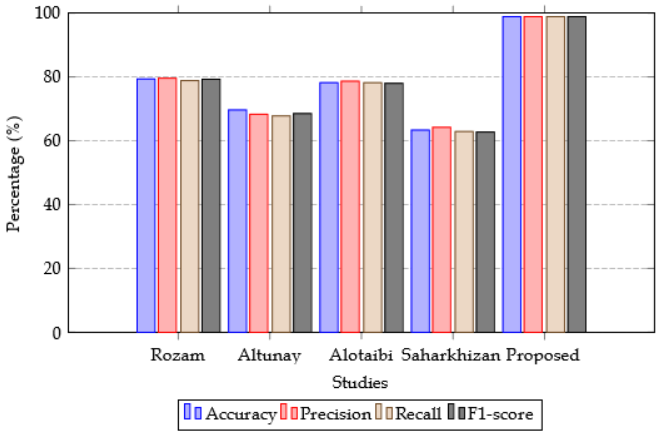

Fig. 11. Performance metrics for single-attack scenarios with unseen data.

Furthermore, in unseen data, as shown in Table X and Figure 12, the proposed stacking model also demonstrated superior performance in multiple attack scenarios.

TABLE X. PERFORMANCE METRICS FOR MULTI-ATTACK SCENARIO

Method	Accuracy	Precision	Recall	F1-score	FP	FN
[17]	75.92%	74.68%	75.10%	74.47%	494.5	483.75
[28]	83.38%	83.41%	83.27%	83.42%	385	379.25
[29]	82.41%	82.29%	83.06%	82.34%	422.5	391
[30]	71.56%	75.63%	71.81%	71.72%	587.5	578.25
Proposed	99.53%	99.59%	99.53%	99.54%	26	33.5

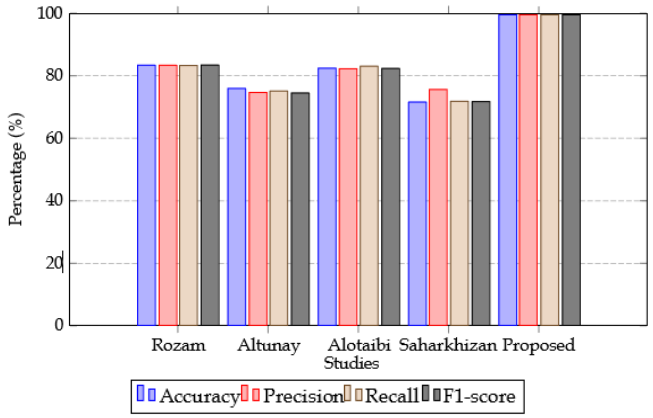

Fig. 12. Performance metrics for multiple victim scenarios.

As shown in Tables IX and X, the proposed model outperformed existing methods in single- and multiple-attack scenarios. In the single-attack scenario, the proposed model achieved significantly higher metrics across all measures compared to previous studies. Specifically, it achieved an accuracy of 99.02%, a precision of 98.84%, a recall of 99.05%, and an F1 score of 99.02%, with minimal FP and FN. This indicates its robustness in correctly identifying victims and non-victims. In the multiple attack scenario, the proposed model continued to outperform other methods with an accuracy of 99.53%, precision of 99.59%, recall of 99.53%, and F1-score of 99.54%. The numbers for FP and FN were low, demonstrating the model's ability to maintain high accuracy even in complex scenarios involving multiple victims.

D. Training Time Analysis

Furthermore, as shown in Table XI and Figure 13, the proposed model exhibited superior performance in terms of training time compared to other studies.

TABLE XI. TRAINING TIMES OF VARIOUS METHODS

Reference	Training time (s)
[11]	4280.61823
[12]	3249.60771
[13]	3699.132496
[14]	3699.132496
Proposed	2593.37563

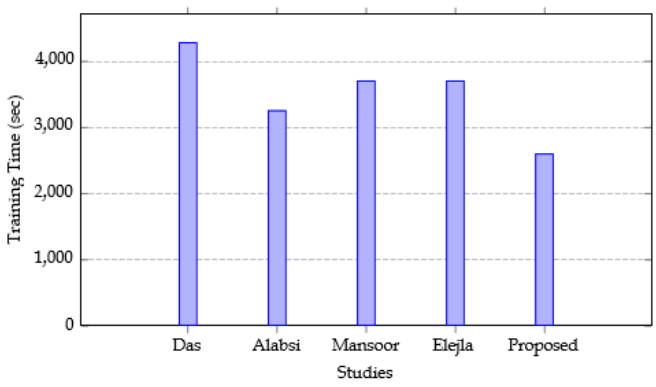

Fig. 13. Training times of various methods.

This advantage can be attributed to the dynamic FS method, which efficiently selects informative and correlated features. Unlike other studies that rely on CNN or fixed threshold FS, this method avoids restrictions on selecting diverse informative and correlated features, reducing training time and enhancing overall performance.

V. SIMPLIFICATION OF THE PROPOSED STACKING FOR RESOURCE-CONSTRAINED ENVIRONMENTS

The computational demands and overhead of the proposed stacking SVM-MLP-RF model are driven by the complexities inherent in each of the base classifiers (SVM and MLP) and the ensemble method (RF) used for final classification. The SVM component is computationally intensive due to its reliance on solving quadratic optimization problems, particularly when dealing with a large number of support vectors or features. The complexity of the SVM scales with the number of training samples and the dimensionality of the feature space, often resulting in significant memory and computational resource consumption during both training and testing. For the MLP component, computational requirements are influenced by the number of layers and neurons per layer, where deeper architectures with more neurons lead to higher computational complexity. Moreover, the training time for MLPs is often prolonged by the need for extensive hyperparameter tuning, including learning rates, batch sizes, and activation functions. The RF component, which aggregates the output from the SVM and MLP, also introduces additional computational overhead, particularly with a large number of trees and deep tree structures.

The proposed stacking model was simplified to address these challenges, called Sim. Several strategies are employed to optimize the model for resource-constrained environments. For the SVM component, the key hyperparameters were optimized by reducing the size of the feature space through FS, thus minimizing the number of support vectors required. The regularization parameter C was adjusted to balance the trade-off between the margin width and classification error, reducing computational costs while maintaining the model performance. Additionally, a linear kernel was used to avoid computationally expensive non-linear kernel calculations. For the MLP component, the network architecture was simplified by reducing the number of hidden layers between 1 and 3 and limiting the number of neurons per layer between 10 and 50. Dropout rates between 10% and 20% were applied to prevent overfitting while keeping the network compact and efficient. The learning rate was carefully set to balance between quick convergence and stable learning, and early stopping techniques were employed to stop training once performance plateaued, further reducing computational time. Finally, the RF component was also optimized by limiting the number of trees between 10 and 50 and pruning less significant trees.

Table XII shows the comparison between the proposed stacking model's performance before and after the simplification.

In particular, AUC values and overlapping confidence intervals suggest that the ability to distinguish between classes is preserved, further supporting the suitability of the proposed

Sim stacking SVM-MLP-RF model for deployment in resource-constrained scenarios.

VI. DISCUSSION

Table IX and Figure 11 showed the accuracy and other metrics for single-attack scenarios. The proposed model showed a significant increase in accuracy (99.021%), precision, recall, and F1-score. Furthermore, Table X and Figure 12 show the performance in multiple attack scenarios, where the model maintains high accuracy and other metrics. The overall comparison summary in Figure 14 illustrates the average enhancement for unseen data between the proposed model and previous studies.

TABLE XII. PERFORMANCE COMPARISON OF PROPOSED STACKING AND SIM PROPOSED STACKING MODELS

Metric	Stacking SVM-MLP-RF	Sim stacking SVM-MLP-RF
Memory usage (MB)	1350.5	780.2
CPU utilization	88%	60%
Accuracy	99.035%	98.95%
Precision	98.97%	97.70%
Recall	99.02%	98.80%
F1-score	99.02%	98.81%
Training time (s)	2593.38	1802.965
FP	33	40
FN	40.84	35.46

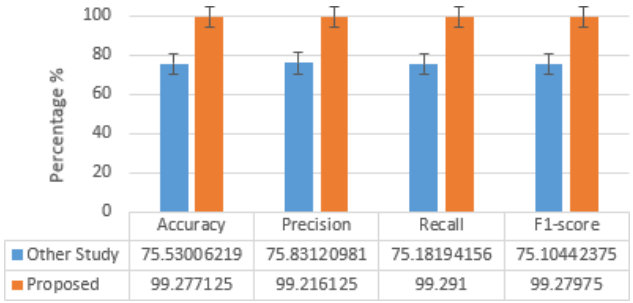

Fig. 14. Comparison summary between the proposed model and previous studies.

The utilization of unseen data showcases the remarkably higher accuracy achieved by the proposed DDoS attack detection model, marking a notable improvement compared to previous studies such as [17], [28], [29], and [30]. In [28], the utilization of individual algorithms led to limitations in capturing diverse DDoS attack patterns, resulting in lower accuracy. Similarly, in [17], a hybrid approach that integrated CNN and LSTM models faced challenges in handling complex attack patterns, yielding 69.7484% accuracy. In [29], ensemble voting encountered complexities and overfitting, especially when the ensemble models lacked diversity. In [30], a set of LSTM models with a Decision Tree as the final classifier struggled to capture intricate traffic patterns, leading to 63.443% accuracy.

The remarkable accuracy and overall performance of the proposed model across various attack rate scenarios are attributed to its diverse ensemble approach, meticulous hyperparameter tuning, adept handling of class imbalances, and the efficacy of selected models and algorithms in capturing diverse data patterns. Furthermore, the considerable impact of class imbalance in the training dataset, where normal traffic constitutes 60% and attack traffic represents 40%, is evident in shaping the model's performance under different attack rate scenarios. In situations with a singular attack rate, the challenge arises from the imbalance between the frequency of malicious attacks and the predominance of the majority class. This imbalance hinders the model's ability to distinguish between normal and attack traffic, leading to a noticeable decline in accuracy. However, using GAWFS and the stacking approach, an impressive accuracy of 99.021% was achieved, showcasing the effectiveness of this method in addressing this challenge. Moreover, when faced with multiple attacks, the metrics display consistent values across different attack rates. This uniformity in accuracy across diverse attack scenarios can be attributed to the distribution of attack packets among multiple hosts. This distribution helps to maintain a proportion of normal and attack traffic that aligns with the characteristics of the public dataset used for model training. Harmony between the attack distribution in the training data and the test scenarios underscores the model's robustness and its ability to perform DDoS attack detection with varying attack rates. This highlights the adaptability of the model to real-world scenarios, where attacks may be distributed across multiple sources. The model achieved an accuracy of 99.533%, which shows reliable detection, regardless of the distribution of malicious traffic.

Overall, the results show that the proposed model enhanced all metrics compared to other studies in both single- and multiple-attack scenarios using unseen data. For a single attack, there was an increase in accuracy of 36.13%, precision of 36.03%, recall of 37.47%, and F1-score of 37.11%. For multiple attacks, there was an increase in accuracy of 27.09%, precision of 26.06%, recall of 27.1%, and F1-score of 27.63%.

VII. CONCLUSION AND FUTURE WORK

The proposed stacking ensemble model with enhanced feature selection has the potential to enhance DDoS attack detection and classification efficiency in network traffic. The dataset was preprocessed before being input into two different ML/DL models specifically designed for attack detection. The stacking ensemble model was preferred over traditional ML/DL techniques, encompassing FS and classification processes, due to its superior performance. The introduction of the novel FS technique, GAWFS, becomes crucial to identifying and selecting the most important, correlated, and informative properties of the target variable across various attack detection scenarios. These selected features are incorporated into the proposed stacking ensemble model, surpassing alternative approaches in both attack scenarios. The stacking ensemble model, utilizing RF as the ultimate metamodel and SVM and MLP as base models, capitalizes on the unique characteristics of multiple models to capture diverse data patterns and excel in various circumstances.

From the collected results, it is shown that after retuning the model parameters, the model performance in the detection of balanced and unbalanced data is improved and is adaptable to different traffic mode and type.

In conclusion, these results underscore the critical importance of the proposed technique in preserving network stability, preventing malfunctions, and safeguarding network integrity and performance, especially for DDoS attacks in SDNs. The model achieved a remarkable precision of 99.82%, a recall of 99.71%, and an F1-score of 99.7076%, with low FP (23) and FN (34). It also attained an impressive accuracy of 99.86% with seen data and 98.89% on average with unseen data for both single and multiple attack rates. Moreover, the proposed model outperformed existing approaches in terms of training time, reducing it to 2,593 s, which represents an improvement of approximately 29.92% compared to other studies.

ACKNOWLEDGMENT

This study was supported by the eköp-24 University Excellence Scholarship Program of the Ministry of Culture and Innovation from the source of the National Research, Development, and Innovation Fund.

REFERENCES

[1] T. Emad Ali, F. Imad Ali, A. Hussein Morad, and M. A Abdala, "Diabetic Patient Real-Time Monitoring System Using Machine Learning," *International Journal of Computing and Digital Systems*, vol. 16, no. 1, pp. 189–199, 2024.

[2] S. V. Ramani and R. H. Jhaveri, "SDN Framework for Mitigating Time-Based Delay Attack," *Journal of Circuits, Systems and Computers*, vol. 31, no. 15, Oct. 2022, Art. no. 2250264, https://doi.org/10.1142/S0218126622502644.

[3] C. Verma, Z. Illés, and D. Kumar, "TCLPI: Machine Learning-Driven Framework for Hybrid Learning Mode Identification," *IEEE Access*, vol. 12, pp. 98029–98045, 2024, https://doi.org/10.1109/ACCESS.2024.3428332.

[4] T. E. Ali, Y.-W. Chong, and S. Manickam, "Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review," *Applied Sciences*, vol. 13, no. 5, Jan. 2023, Art. no. 3183, https://doi.org/10.3390/app13053183.

[5] T. E. Ali, Y. W. Chong, and S. Manickam, "Comparison of ML/DL Approaches for Detecting DDoS Attacks in SDN," *Applied Sciences*, vol. 13, no. 5, Jan. 2023, Art. no. 3033, https://doi.org/10.3390/app13053033.

[6] C. Verma, "NextGen Learning: Hybrid Mode Prediction with Machine Learning," in 2024 11th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, Mar. 2024, pp. 1–8, https://doi.org/10.1109/ICRITO61523.2024.10522445.

[7] F. I. Ali, T. E. Ali, and Z. T. Al_dahan, "Private Backend Server Software-Based Telehealthcare Tracking and Monitoring System," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 19, no. 1, pp. 119–134, Jan. 2023, https://doi.org/10.3991/ijoe.v19i01.32433.

[8] T. Emad Ali, A. Hussein Morad, and M. A. Abdala, "Load Balance in Data Center SDN Networks," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 5, Oct. 2018, Art. no. 3084, https://doi.org/10.11591/ijece.v8i5.pp3084-3091.

[9] F. I. Ali, T. E. Ali, and A. H. Hamad, "Telemedicine Framework in COVID-19 Pandemic," in *2022 International Conference on Engineering and Emerging Technologies (ICEET)*, Kuala Lumpur, Malaysia, Oct. 2022, pp. 1–8, https://doi.org/10.1109/ICEET56468.2022.10007389.

[10] C. Verma, "Machine Learning Model for Applicability of Hybrid Learning in Practical Laboratory," *Procedia Computer Science*, vol. 235, pp. 1600–1607, Jan. 2024, https://doi.org/10.1016/j.procs.2024.04.151.

[11] S. Das, D. Venugopal, S. Shiva, and F. T. Sheldon, "Empirical Evaluation of the Ensemble Framework for Feature Selection in DDoS Attack," in *2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, New York, NY, USA, Aug. 2020, pp. 56–61, https://doi.org/10.1109/CSCloud-EdgeCom49738.2020.00019.

[12] B. A. Alabsi, M. Anbar, and S. D. A. Rihan, "CNN-CNN: Dual Convolutional Neural Network Approach for Feature Selection and Attack Detection on Internet of Things Networks," *Sensors*, vol. 23, no. 14, Jan. 2023, Art. no. 6507, https://doi.org/10.3390/s23146507.

[13] A. Mansoor, M. Anbar, A. A. Bahashwan, B. A. Alabsi, and S. D. A. Rihan, "Deep Learning-Based Approach for Detecting DDoS Attack on Software-Defined Networking Controller," *Systems*, vol. 11, no. 6, Jun. 2023, Art. no. 296, https://doi.org/10.3390/systems11060296.

[14] O. E. Elejla, M. Anbar, S. Hamouda, S. Faisal, A. A. Bahashwan, and I. H. Hasbullah, "Deep-Learning-Based Approach to Detect ICMPv6 Flooding DDoS Attacks on IPv6 Networks," *Applied Sciences*, vol. 12, no. 12, Jan. 2022, Art. no. 6150, https://doi.org/10.3390/app12126150.

[15] P. Wu, H. Guo, and N. Moustafa, "Pelican: A Deep Residual Network for Network Intrusion Detection," in *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, Valencia, Spain, Jun. 2020, pp. 55–62, https://doi.org/10.1109/DSN-W50199.2020.00018.

[16] D. Javeed, T. Gao, M. T. Khan, and I. Ahmad, "A Hybrid Deep Learning-Driven SDN Enabled Mechanism for Secure Communication in Internet of Things (IoT)," *Sensors*, vol. 21, no. 14, Jan. 2021, Art. no. 4884, https://doi.org/10.3390/s21144884.

[17] H. C. Altunay and Z. Albayrak, "A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks," *Engineering Science and Technology, an International Journal*, vol. 38, Feb. 2023, Art. no. 101322, https://doi.org/10.1016/j.jestch.2022.101322.

[18] M. S. Akhtar and T. Feng, "Detection of Malware by Deep Learning as CNN-LSTM Machine Learning Techniques in Real Time," *Symmetry*, vol. 14, no. 11, Nov. 2022, Art. no. 2308, https://doi.org/10.3390/sym14112308.

[19] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An Adaptive Ensemble Machine Learning Model for Intrusion Detection," *IEEE Access*, vol. 7, pp. 82512–82521, 2019, https://doi.org/10.1109/ACCESS.2019.2923640.

[20] P. K. Mondal, L. P. Aguirre Sanchez, E. Benedetto, Y. Shen, and M. Guo, "A dynamic network traffic classifier using supervised ML for a Docker-based SDN network," *Connection Science*, vol. 33, no. 3, pp. 693–718, Jul. 2021, https://doi.org/10.1080/09540091.2020.1870437.

[21] S. Haider *et al.*, "A Deep CNN Ensemble Framework for Efficient DDoS Attack Detection in Software Defined Networks," *IEEE Access*, vol. 8, pp. 53972–53983, 2020, https://doi.org/10.1109/ACCESS.2020.2976908.

[22] N. Ahuja, "DDOS attack SDN Dataset." Mendeley, Sep. 27, 2020, https://doi.org/10.17632/JXPFJC64KR.1.

[23] J. Liu, Y. Zhang, and Q. Zhao, "Video stabilization algorithm based on Pearson correlation coefficient," in *2019 International Conference on Advanced Mechatronic Systems (ICAMechS)*, Kusatsu, Shiga, Japan, Aug. 2019, pp. 289–293, https://doi.org/10.1109/ICAMechS.2019.8861649.

[24] R. Rahimi *et al.*, "A high-performance OpenFlow software switch," in *2016 IEEE 17th International Conference on High Performance Switching and Routing (HPSR)*, Yokohama, Japan, Jun. 2016, pp. 93–99, https://doi.org/10.1109/HPSR.2016.7525645.

[25] A. Dhulfiqar, N. Pataki, and M. Tejfel, "Chatbot-Based Querying of IoT Devices in EdgeX," in *SQAMIA 2023: Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications,* Bratislava, Slovakia, Sep. 2023, vol. 1613.

[26] M. H. H. Khairi, S. H. S. Ariffin, N. M. A. Latiff, A. S. Abdullah, and M. K. Hassan, "A Review of Anomaly Detection Techniques and Distributed Denial of Service (DDoS) on Software Defined Network (SDN)," *Engineering, Technology & Applied Science Research*, vol. 8, no. 2, pp. 2724–2730, Apr. 2018, https://doi.org/10.48084/etasr.1840.

[27] A. Dhulfiqar, M. A. Abdala, N. Pataki, and M. Tejfel, "Deploying a web service application on the EdgeX open edge server: An evaluation of its

viability for IoT services," *Procedia Computer Science*, vol. 235, pp. 852–862, Jan. 2024, https://doi.org/10.1016/j.procs.2024.04.081.

[28] N. F. Rozam and M. Riasetiawan, "XGBoost Classifier for DDOS Attack Detection in Software Defined Network Using sFlow Protocol.," *International Journal on Advanced Science, Engineering & Information Technology*, vol. 13, no. 2, 2023.

[29] Y. Alotaibi and M. Ilyas, "Ensemble-Learning Framework for Intrusion Detection to Enhance Internet of Things' Devices Security," *Sensors*, vol. 23, no. 12, Jan. 2023, Art. no. 5568, https://doi.org/10.3390/s23125568.

[30] M. Saharkhizan, A. Azmoodeh, A. Dehghantanha, K.-K. R. Choo, and R. M. Parizi, "An Ensemble of Deep Recurrent Neural Networks for Detecting IoT Cyber Attacks Using Network Traffic," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8852–8859, Sep. 2020, https://doi.org/10.1109/JIOT.2020.2996425.