Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.057185

ARTICLE

Enhanced DDoS Detection Using Advanced Machine Learning and Ensemble Techniques in Software Defined Networking

Hira Akhtar Butt[1], **Khoula Said Al Harthy**[2], **Mumtaz Ali Shah**[3], **Mudassar Hussain**[2,*], **Rashid Amin**[4,*] **and Mujeeb Ur Rehman**[1]

[1]Department of Computer Science, University of Management and Technology, Sialkot, 51040, Pakistan

[2]Department of Computer Science and Creative Technologies, Global College of Engineering and Technology, Muscat, 2546, Sultanate of Oman

[3]Department of Computer Science, University of Wah, Wah Cantt, 47040, Pakistan

[4]Department of Computer Science and IT, University of Chakwal, Chakwal, 48800, Pakistan

*Corresponding Authors: Mudassar Hussain. Email: mudassar.h@gcet.edu.om; Rashid Amin. Email: rashid.sdn1@gmail.com

Received: 10 August 2024 Accepted: 08 October 2024 Published: 18 November 2024

ABSTRACT

Detecting sophisticated cyberattacks, mainly Distributed Denial of Service (DDoS) attacks, with unexpected patterns remains challenging in modern networks. Traditional detection systems often struggle to mitigate such attacks in conventional and software-defined networking (SDN) environments. While Machine Learning (ML) models can distinguish between benign and malicious traffic, their limited feature scope hinders the detection of new zero-day or low-rate DDoS attacks requiring frequent retraining. In this paper, we propose a novel DDoS detection framework that combines Machine Learning (ML) and Ensemble Learning (EL) techniques to improve DDoS attack detection and mitigation in SDN environments. Our model leverages the "DDoS SDN" dataset for training and evaluation and employs a dynamic feature selection mechanism that enhances detection accuracy by focusing on the most relevant features. This adaptive approach addresses the limitations of conventional ML models and provides more accurate detection of various DDoS attack scenarios. Our proposed ensemble model introduces an additional layer of detection, increasing reliability through the innovative application of ensemble techniques. The proposed solution significantly enhances the model's ability to identify and respond to dynamic threats in SDNs. It provides a strong foundation for proactive DDoS detection and mitigation, enhancing network defenses against evolving threats. Our comprehensive runtime analysis of Simultaneous Multi-Threading (SMT) on identical configurations shows superior accuracy and efficiency, with significantly reduced computational time, making it ideal for real-time DDoS detection in dynamic, rapidly changing SDNs. Experimental results demonstrate that our model achieves outstanding performance, outperforming traditional algorithms with 99% accuracy using Random Forest (RF) and K-Nearest Neighbors (KNN) and 98% accuracy using XGBoost.

KEYWORDS

DDoS attacks; SDN; botnet; cyber security; machine learning; network protection

1 Introduction

Distributed Denial of Service (DDoS) attacks occur when multiple computers generate excessive internet traffic to overwhelm a target system, such as a server or website, causing it to slow down drastically or become completely unresponsive, disrupting normal operations. These attacks are often executed through botnets networks of compromised computers controlled by attackers [1]. DDoS attacks not only lead to financial losses but can also damage the target's reputation. Standard mitigation techniques include traffic filtering, rate limiting and specialized DDoS protection services. This paper uses Software-Defined Networking (SDN) to detect and mitigate DDoS attacks. SDN offers a novel approach to network management by centralizing control and enabling configuration and management through software from a single, centralized controller [2]. This approach simplifies network adjustments and enhances adaptability. SDN represents a paradigm shift in how networks are built, managed and controlled [3]. Unlike traditional networks where control intelligence is integrated into the physical infrastructure via static configurations and distributed control layers, SDN decouples the control plane from the data plane. This separation introduces centralized control and programmability allowing greater flexibility, agility and scalability in managing network resources [4].

While SDN technology offers numerous advantages in detecting and mitigating attacks, the threat of cybersecurity breaches, particularly DDoS attacks, should not be underestimated. DDoS attacks are orchestrated by flooding network resources with an overwhelming amount of illegitimate traffic, disrupting standard network services and ultimately disconnecting legitimate users [5]. Although SDN provides enhanced programmability and dynamic reconfigurability, it is not immune to DDoS threats. SDN's centralized control plane simplifies global network management and decision-making which simplifies global network management and decision-making but introduces a significant vulnerability. This centralized architecture is particularly susceptible to being a prime target for DDoS attacks [6]. Moreover, the dynamic and reconfigurable nature of SDNs complicates distinguishing between legitimate traffic and malicious activity, given the fluctuations in traffic flow and continuous network reconfigurations. This volatility poses a challenge in accurately identifying and mitigating malicious disruptions.

1.1 Software Defined Networking

SDN has transformed network architecture by revolutionizing traditional network engineering, operation and management [7]. Unlike conventional networks, where control and data functions are tightly coupled within switches and routers, SDN decouples these components into separate entities. This separation centralizes decision-making within the SDN controller which governs the network's operations and management. As depicted in Fig. 1, SDN architecture is organized into the data, control and application planes. The control plane, which houses the SDN controller, is responsible for calculating flow rules based on application demands and deploying these rules to the data plane. The data plane then facilitates network communication based on these flow rules, streamlining network management and enhancing flexibility.

Flow rules are installed from the control to the data plane using a standardized protocol known as OpenFlow [8]. As the network's central intelligence, the SDN controller makes critical decisions regarding optimal data routing, service prioritization and security enforcement. SDN switches at the data network edges follow these instructions, rerouting traffic as directed to ensure efficient network operation. This centralized architecture enhances decision making capabilities, allowing them to adapt quickly to changing network conditions and traffic patterns [9]. The combination of SDN's centralized control, programmability and flexibility simplifies attack detection and mitigation. By centralizing management functions and enabling dynamic reconfiguration, SDN optimizes network performance and strengthens defenses against evolving cyber threats [10].

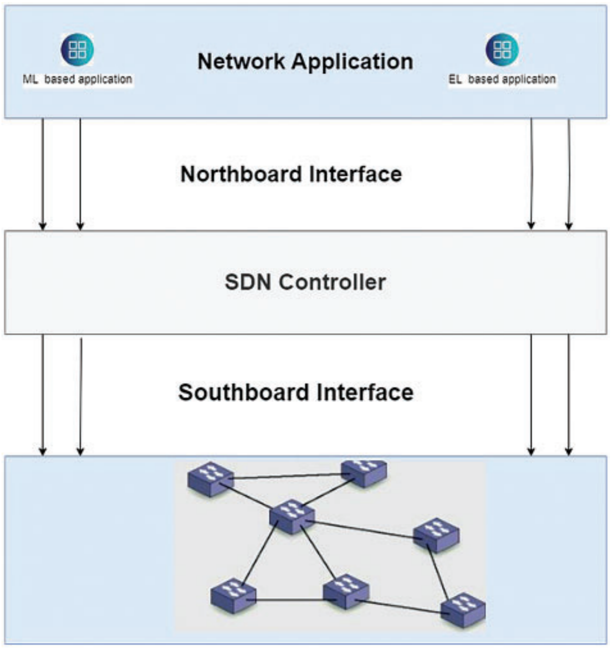

Figure 1: Software define networking architecture

1.2 Attack Traditional Detection Systems

The current DDoS detection systems use a simple rule-based approach and low-level machine learning. These conventional approaches include:

1. Static Rule-Based Systems:

- Overview: These systems rely on signatures or rules for identifying DDoS attacks. It works well for known attacks but stands bare regarding new or changing attacks.
- Limitations: Thus, static systems' primary disadvantage is their inability to learn new behaviors. They fail to identify a high percentage of samples as malicious in the case of complex contemporary and zero-day attacks. Moreover, updating and generally managing rule sets is highly time consuming. It carries a particular risk of mistakes and the incapability to learn new behaviors [11].

2. Basic Machine Learning Models:

- Overview: Some traditional systems use simple machine learning methods, like Decision Trees or Naive Bayes, to identify possible abnormal flow in the network. These models learn data to identify DDoS attack patterns characteristic of the model [12].
- Limitations: Although more lenient than rule-based systems, the basic machine learning models are primarily rigid and may not perform well when there are changes in the attack endeavors. Another disadvantage of both is that they also need a large amount of labeled training data, and in some cases, with time constraints, they will not work well as they may take much time to compute [13].

Our proposed approach does not assume these bottlenecks which use dynamic feature selection and ensemble learning based on selected features. This means that while tuning our model, we reduce substantial complexity by choosing the most critical features during the training phase. This allows it to detect new patterns of DDoS attacks that may not have been previously observed, providing more robust and dynamic functionality.

1.3 Distributed Denial of Service Attack

DDoS is an attack which get possess by malware and launches unmanageable traffic, requests or data to prevent users from accessing online services [11]. A DDoS attack's primary purpose is to impede some aspects of the targeted organization's operation irrespective of what that organization does. This disruption is done by using up the target system's quota, such as bandwidth, CPU cycles or space in the memory map. Unlike typical DoS attacks where the source of the attack is conspicuous, DDoS takes advantage of several infected bot-compromised machines or systems controlled by an attacker [12]. Such bots may be computers, Internet of Things (IoT) devices or servers. Most of the time, they are likely to possess a variety of malware that helps the attacker command the device without the owner's consent.

Fig. 2 depicts an example of DDoS attack where an attacker, with the help of the botnets, floods a server with traffic, thus preventing honest users from accessing the service offered by the server. The type of complex methods that attackers use is no longer rule-based but have a higher level of sophistication. Hence, investigations carried out regarding the application of ML to detect DDoS attacks have been on the rise especially in SDN environments where the central controller relays all the communication networks. Network data is ideal for applying both ML and EL techniques as they assist in identifying network reliability issues. These techniques analyze the effects of limited malicious activities combined with traditional detection methods, improving DDoS attack prevention.

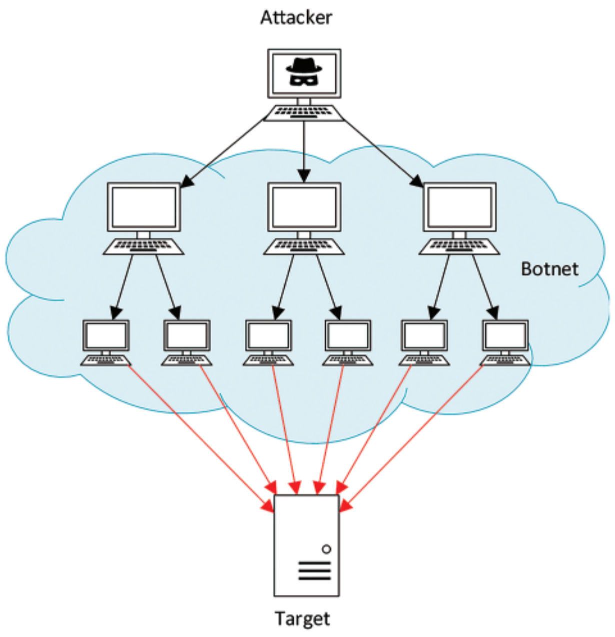

Figure 2: Distributed denial of service attack

1.4 DDoS Attacks

The fact that a distributed attack involves several sources makes it impossible to block the first source IP address to solve the problem. Regarding methods that can be used to amplify and extend the DDoS, the attackers can utilize the DNS, NTP and SNMP protocols to reflect the attack. Such services, for which performance requests are sent by spoofing the source IP address, redirect considerably larger response packets than the initial ones to the target [14]. As methods for executing DDoS attacks evolve frequently, detection systems based on signatures or predefined criteria may require regular updates. However, there is growing interest in improving the efficiency and accuracy of DDoS detection using AI and ML techniques [15]. AI algorithms, particularly in significant data contexts, have demonstrated a solid ability to identify anomalies commonly linked to DDoS attacks [16]. The study [17] provides a comprehensive overview of the state of the art in AI for DDoS attack detection, offering a promising future for network security.

Table 1 demonstrates that DDoS attacks exploit technical vulnerabilities across various OSI model layers [18]. To effectively defend against these attacks, a comprehensive understanding of their nature and the techniques used is crucial. The attackers can use techniques like, Hyper Text Transfer Protocol (HTTP) flood at the application layer. This involves sending many HTTP requests to a server that results in the over exhaustion of resources [19]. Finally, an Application Layer Protocol (ALP) flood is an attack that chooses a particular application protocol and dirties it with aggressive traffic. On the presentation layer, some attacks abuse weaknesses in encryption algorithms to waste resources during decryption and some attacks are known as compression bombing, which, by sending compressed data that will grow exponentially after expansion, consumes system resources. Session layer attacks target the layering session, swamping a server using flood requests for session initiation [20]. At the transport layer, Synchronize/Acknowledgment (SYN/ACK) flood and Transmission Control Protocol/Internet Protocol (TCP/IP) fragmentation attacks create congestion of SYN or ACK packets or discover any susceptibility to the TCP/IP protocol that results in the exhaustion of resources. While in network layer attacks such as Internet Control Message Protocol (ICMP) flooding and Ping of Death, one or more devices send ICMP packets or abnormal packets to a hub so that the network becomes full, or the devices get jammed. Data link layer attacks, for instance, Media Access Control (MAC) Address flooding, generate many frames emitted on the switch's MAC table, causing the table to explode and interfere with the network's reliable transfer of data [21]. The physical layer attacks, like cable tapping, jamming and disconnection, are among the attacks that can physically disconnect networks between nodes, making them inaccessible [22]. The study [23] highlights the critical need to identify and prevent DDoS attacks. It offers valuable insights for academia, industry and government in helping to strengthen protective measures and promote the development of new DDoS prevention techniques.

Table 1: Possible DDoS attacks on OSI layers

OSI layer	Possible DDoS attack
Application	HTTP Flood, Slowloris, ALP Flood
Presentation	Encryption-based attacks, compression bombing
Sessional	Session exhaustion
Transport	SYN/ACK Flood, TCP/IP Fragmentation attack
Network	ICMP Flood, Smurf attack, Ping of death

(Continued)

Table 1 (continued)

OSI layer	Possible DDoS attack
Data link	MAC Address Flooding
Physical	Cable disconnection, JaMMING, physical impersonation

1.5 Machine Learning

Machine learning (ML) is pivotal in detecting and managing DDoS attacks in SDN where a central controller monitors network operates. This centralized view allows ML algorithms to effectively process large volumes of real-time traffic data [21]. By learning patterns associated with DDoS attacks, the SDN controller can prevent such threats, reducing the likelihood of network disruptions and strengthening the network's defenses against emerging cyber risks [22,23]. ML into SDN frameworks enhances an organization's ability to manage secure and efficient networks. ML enables systems to autonomously analyze and predict data by applying statistical methods, learning from datasets and identifying patterns to enhance network performance and security [24]. There are three ML types: Supervised learning, unsupervised learning and reinforcement learning. All of them can be referred as a particular kind of activity or form of data. Hence, while supervised learning is training with labeled data, statistically, unsupervised learning is done with unlabeled data [25]. Reinforcement Learning (RL) teaches an agent how to act in an environment by facing the outcome of an action in the form of a feedback signal [26,27]. In supervised learning, the algorithms are given the input data and the correct output values to help them learn. The input data and the corresponding output labels are labeled [28]. In this regard, the unsupervised learning algorithm employs a dataset that lacks associated labels. Data labeling means only inputs are given and their corresponding outputs are not marked [29].

In this study, we used the EL technique which is a form of machine learning that promotes overall performance and correlates with the improvement and diversification of models widely known in the field. This method uses multiple models which are integrated to form one model using their predictions to get a better result and generalize well. EL is also helpful in increasing the reliability of the projections and the robustness of the models employed as it helps me select only the best models and eject the mediocre ones. This paper demonstrates that employing RF can successfully process massive amounts of data with high-dimensional information. It can project feature importance which can be crucial since DDoS attacks can occur in different forms and may evolve. On the other hand, the XGBoost is more efficient and faster than several classifiers because the framework of gradient boosting improves the model's outcomes and devotes attention to samples that are most difficult for classifiers to distinguish that can improve the early detection rates. Then, several other classifiers are used because the gradient boosting enhances the model performance by emphasizing beyond the most challenging sample for classification, subsequently improving the early detection rate. These features enabled us to obtain acceptable and high accuracy values varying from 99% with the RF and KNN classifiers to 98% with the XGBoost; compared with other similar SDN state-of-the-art classifiers, these results are sufficient to develop an accurate and reliable DDoS detection system in SDNs.

1.6 Contributions

The primary contributions of this paper are as follows:

- Dynamic Feature Selection: We developed a modern ensemble feature selection method for data pre-processing in the DDoS SDN dataset, enhancing robustness against various attack scenarios. This dynamic approach improves detection rates and model flexibility by overcoming the limitations of strictly defined feature sets. Real world experiments demonstrate the efficiency of our model, showing that the selected features optimize training time and improve classifier performance in identifying DDoS attacks in SDN.
- Proactive DDoS Attack Detection: Our study establishes a robust framework for proactively detecting and mitigating DDoS attacks, significantly enhancing network protection against evolving cyber threats. By integrating advanced machine learning techniques and ensemble learning methods, our approach ensures the security and integrity of SDN systems, providing a vital defense mechanism in an increasingly complex cyber landscape.
- Performance Enhancement Compared to Existing Research: Our proposed mechanisms demonstrate significant advancements in cybersecurity. We evaluated our system using accuracy, precision, recall, F1-score, specificity, confusion matrix, ROC and AUC, ensuring its reliability and stability. We enhanced our solution's practical applicability and effectiveness by grounding our solution in real-world data from the DDoS SDN dataset. Our study achieved remarkable accuracy levels: 99% with RF and KNN and 98% with XGBoost, demonstrating the effectiveness of combining ML and EL techniques for DDoS detection in SDNs.

1.7 Paper Organization

The paper is organized as follows: Section 2 provides a comprehensive literature review while Section 3 details the proposed solution. Section 4 presents the results and discusses their implications. At the end, Section 5 concludes the study and outlines directions for future research.

2 Literature Review

The study by Ahuja et al. [1] presents ML-aided DDoS detection approach which proved proficient in terms of accuracy to detect known DDoS attack scenarios. However, the ability of the method to adapt to new or emerging threats or a new attack model is somewhat restricted. In the same respect, Swami et al. [2] propose the use of Support Vector Machine (SVM) in the context of SDN application settings to provide effectiveness in the conditions of resource limitation. However, due to the high time complexity, it is not useful for real time detection. Abou El Houda et al. [3] analyze DDoS attack detection by proposing the deep learning approach for detecting large-scale attacks with high detection rate. However, this approach involves large amount of data which needs high computational power. Niyaz et al. [8] also propose a deep learning-based methodology with high detection accuracy and low false positives but the model is not very clear on the underlying reasoning for a certain decision. The proposed method in this paper incorporates dynamic feature selection with Random Forest (RF), K-Nearest Neighbors (KNN) and XGBoost which results to an accuracy of up to 99%. This approach is very flexible in changing DDoS attack patterns decision-making but it has a disadvantage of increased model interpretability. Compared with the previous work, the dynamic feature selection mechanism makes the method more elastic and has the ability to handle various types of attack forms and retain high efficiency as well.

The study [30] describes the application of deep learning for DDoS detection. It explains the benefits and drawbacks of real-time detection and how deep learning models may be used. In addition, it also assesses the possibility and risks of detecting DDoS using artificial intelligence. The study [31] examines machine learning techniques for DDoS detection, providing an overview of recent algorithms and methodologies, and highlighting their types, advantages and disadvantages. This evaluation analyzes aggregation techniques for detecting and assessing DDoS attacks. In study [32], the study explores the potential integration of optimal clustering approaches into existing detection systems and identifies them through a comprehensive review. It examines the critical topic of feature selection in AI models for DDoS detection [33]. To recognize the circulated disavowal of administration attacks, this examination offers a blended strategy that utilizes AI and factual analysis [34]. It examines the hybrid model's effectiveness in various attack scenarios. It delves into how ML and statistical techniques collaborate to achieve robust detection. This study carefully outlines move learning's utilization in DDoS [35].

The study [36] uses unsupervised learning approaches for DDoS attack detection. It reviews unsupervised learning algorithms, assessing their strengths, weaknesses and practical applications. It guides picking appropriate measures given explicit necessities in the wake of analyzing the value of different measurements for estimating the presentation of recognition models. The research in study [37] reviews widely used open datasets for training and testing ML models in DDoS detection, categorizing them based on their characteristics. In study [38], an ML-based DDoS detection framework is introduced by using network traffic signatures as the foundation for its inferences with experiments conducted on four modern benchmark datasets. The Internet is pervasive in today's digital world with its usage and proliferating alongside an increase in threats including DDoS attacks. These attacks exploit legitimate service requests to overwhelm computing and network resources, denying access to genuine users [39]. In terms of feature selection with ML models, Polat et al. [40] adopt them and observed higher accuracy, but with an increased selection problem and potential overfitting.

Table 2, as presented below, shows comparisons between current DDoS detection techniques with strengths and weaknesses exhibited by each of them. As compared to existing approached, our research enhances DDoS detection accuracy and robustness in SDNs by leveraging ML and Ensemble Learning (EL) techniques, specifically RF and XGBoost, which offer dynamic feature selection to improve adaptability and performance in detecting DDoS attacks. While prior studies have primarily focused on ML and deep learning for classification, our approach integrates EL methods, demonstrating superior detection accuracy by adapting to evolving attack patterns in SDN environments. Focusing on detecting DDoS attacks, we utilized the DDoS SDN dataset, segmenting it to determine the optimal configuration for each algorithm with RF outperforming XGBoost. In contrast to conventional ML or unsupervised learning approaches, our solution introduces a novel EL framework tailored explicitly for SDN architectures. Implementing dynamic feature selection in RF and XGBoost significantly enhances detection rates and effectively mitigates contemporary DDoS attacks. Furthermore, our approach addresses the limitations of previous models using numerical feature vectors, offering a robust, real-world validated solution that sets a new benchmark for SDN network security.

Comparison of the Existing Approaches and the Proposed Solution

In recent years, many researchers have applied ML and EL techniques for DDoS detection in SDNs, contributing significantly to the field. However, challenges remain, particularly in identifying new and evolving attack behaviors. For instance, the study [41] employ conventional ML techniques like DT and RF, achieving only satisfactory detection accuracy for DDoS attacks. These models lack the flexibility and adaptability to handle emerging threats, rendering them less effective in real-world environments. In a separate study, deep learning approaches were utilized to enhance detection performance, yielding promising results. However, these methods are computationally intensive and require large volumes of labeled data making them less suitable for the dynamic nature of SDN environments. In contrast, our proposed approach combines ML and EL techniques with RF and XGBoost capable of handling large datasets while dynamically selecting relevant features. This dynamic feature selection mechanism allows our model to maintain a high detection rate of up to 99% even as attack dynamics evolve. Moreover, our approach addresses the limitations of existing models by offering a solution that is less sensitive to network changes, reducing the need for frequent retraining. However, some limitations of the developed approach should be noted. While the ensemble methods significantly improve detection accuracy, they come at the expense of model interpretability. Additionally, the real-world applicability of the proposed model, particularly in large-scale networks, remains uncertain and will be addressed in future studies.

3 Proposed Solution

To assess the effectiveness of our proposed solution in detecting DDoS attacks, we used a systematic approach involving dataset selection, pre-processing, feature selection, algorithm implementation, model training and evaluation. We chose diverse datasets like NSL-KDD, DDoS SDN and CICDDoS2019, pre-processed them to handle missing values and standardize features and performed feature selection to enhance model performance. Our framework, depicted in Fig. 3, utilizes ML and EL techniques for real-time analysis and adaptive response to strengthen DDoS detection and network security.

3.1 Machine Learning Techniques

Classification is a type of supervised learning used to categorize data based on input features. This paper evaluates the proposed solution using NB classifiers, DTs, SVMs and RF [42]. These techniques are fundamental in supervised machine learning for predicting numerical values based on input data. Regression methods including Multiple Regression (MR), Neural Networks (NN), Support Vector Regression (SVR), DT Regression (DTR), Lasso Regression (LR) and Ridge Regression (RR) are examples of this type of analysis. K-means, an unsupervised learning technique, partitions data into “k” clusters based on features with examples such as K-means clustering, mean-shift clustering, DBSCAN, hierarchical agglomerative clustering and the Gaussian mixture model [43]. Decision-making processes in machine learning involve determining optimal actions based on input data and applied algorithms with examples including Q-learning, R-learning and Temporal Difference (TD) learning [44].

3.2 Ensemble Learning Techniques

EL refers to the ML technique that combines several models to increase the overall performance while staying robust [45]. EL combines predictions from different models, selects the most effective ones and dismisses the rest, eventually resulting in better accuracy and generalization from the collective wisdom. As for the detection of DDoS attacks, an EL model can considerably boost the robustness of intrusion detection systems as attack patterns evolve to elaborate [46]. Ensemble methods attempt to combine the strengths of various models and algorithms by using training data on diverse subsets, thus reducing the drawbacks of individual models for better performance. Fig. 4 shows the types of EL methods. One widely used instance of EL is bagging (Bootstrap Aggregating) that works by training several models on separate random sub-samples from the training set through bootstrapping (i.e., random data sampling with replacement). The forecasts of these models are then combined, where the different prediction results are usually averaged out or take the majority vote to obtain result.

Table 2: Comparative analysis of existing DDoS detection methods

Reference	Techniques used	Strengths	Weaknesses
Ahuja et al. (2021) [1]	ML-based DDoS detection	High accuracy in detecting known DDoS attacks	Limited adaptability to new or evolving attack patterns
Swami et al. (2019) [2]	SVM in SDN environments	Efficient in resource-constrained environments	High computational cost, not suitable for real-time detection
Abou El Houda et al. (2020) [3]	Deep learning for DDoS detection	High detection rate for large-scale attacks	Requires large datasets and significant computational power
Niyaz et al. (2016) [8]	Deep learning-based approach	High detection accuracy with low false positives	Limited explanation of the model's decision-making process
Alghazzawi et al. (2021) [31]	Hybrid deep learning with feature selection	Efficient detection of DDoS attacks using improved feature selection	Requires a complex setup and higher computational power
Nadeem et al. (2022) [47]	Machine learning for DDoS in SDN	Lightweight and effective for DDoS detection in SDN environments	Limited scalability to larger and more complex networks
Polat et al. (2020) [40]	Feature selection with ML Models	Improved model accuracy through feature selection	Complexity in the feature selection process, potential overfitting
Proposed method	Dynamic feature selection with RF, KNN, XGBoost	High accuracy (up to 99%), adaptable to evolving DDoS attack patterns	May introduce complexity in model interpretability

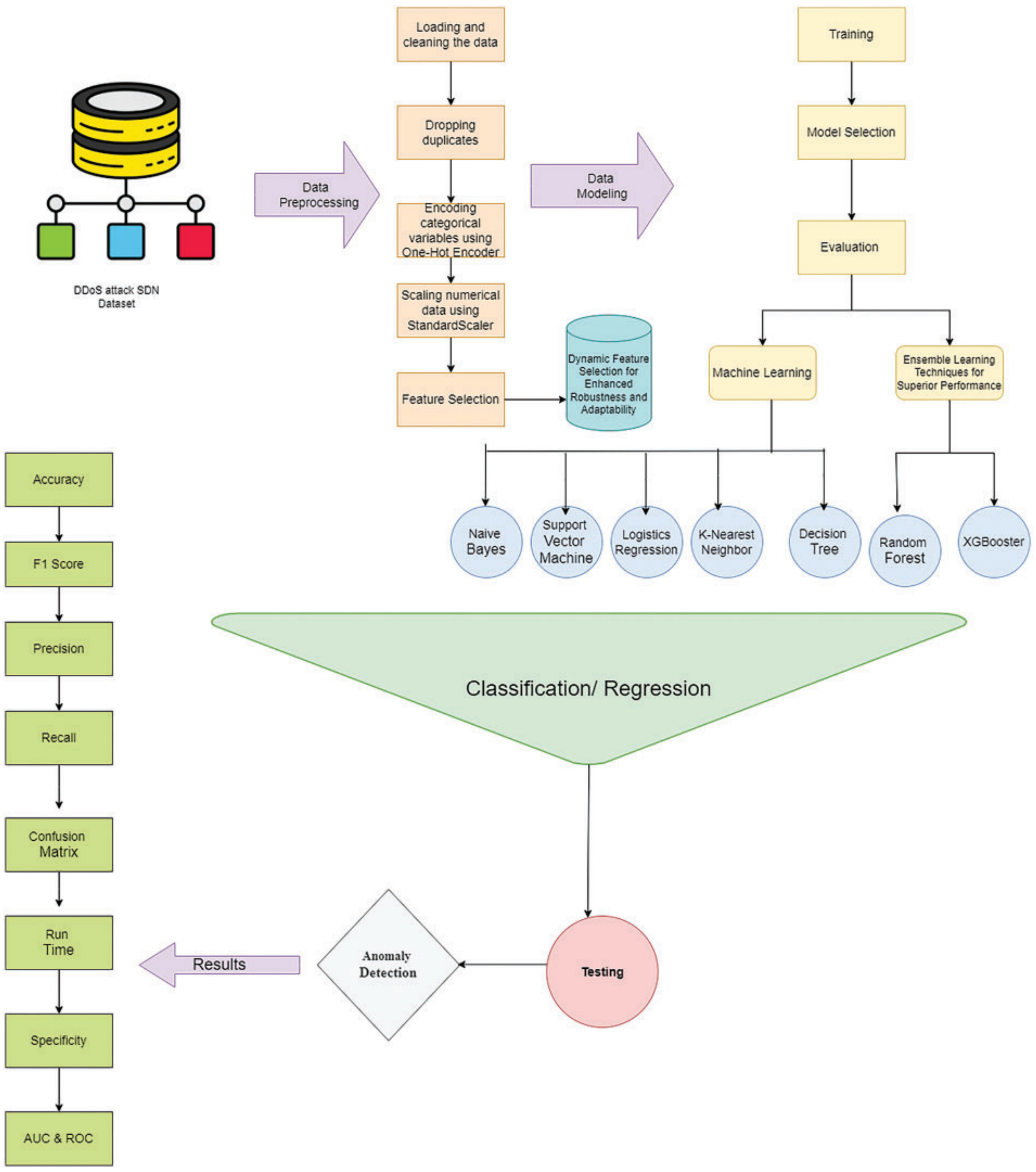

Figure 3: Proposed framework

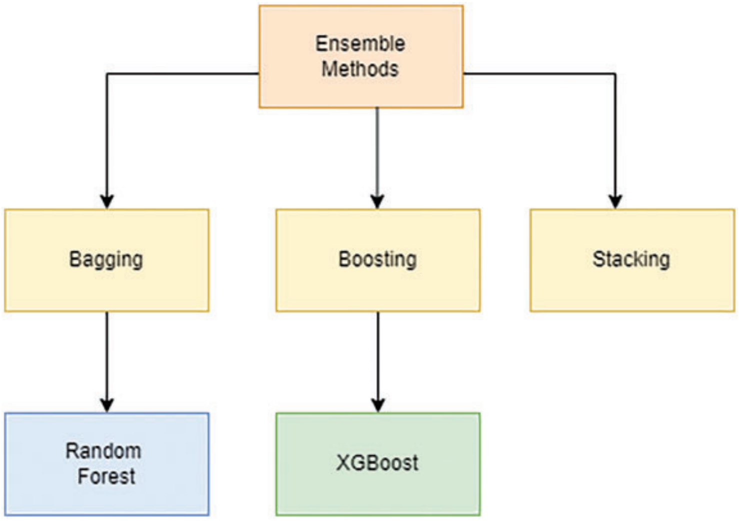

Figure 4: Ensemble learning techniques

Another popular combination approach is boosting which sequentially trains multiple weak learners (models with a low misclassification rate). We are given lower weights in the first stage but the assigned weights to misclassified instances in the following iterations increase [47]. This continuous learning builds on the successes and addresses the challenges by improving the performance of the problematic cases to achieve better results for the whole model. EL techniques may also be enhanced by adding base learners with diverse competencies such as DT, SVM, NN and many others that can find different data features and make the model robust against all different attack settings. In the SDN context, EL helps dynamically adapt to new threats, optimize resource allocation and enhance overall network security. DDoS detection in SDN can be enriched by leveraging the collective intelligence with each model processing a unique subset of the network traffic data set. This strategy will combine the strengths of individual algorithms and training techniques to achieve the desired outcome of eliminating false positives and negatives. With this, the dependability and effectiveness of DDoS detection systems will be significantly enhanced.

3.3 Data Pre-Processing

Data pre-processing is one of the most effective approaches to data mining before a set of data goes for the supervised learning experiments, as shown in Fig. 4. This is followed by data cleaning where the data is scanned for errors, internal contradictions or missing values. When cleaned, the data is normalized to equal all features providing a standard for further computations. The following central process entails centering the degree of features since the performance of the ML algorithms depends on feature scales. The dataset is normalized at the start of the process before being divided into training and testing datasets. When kick-starting an ML or DL project, the first thing is to gather raw data which is then processed through cleaning, transforming and pre-processing to prepare data for model fitting.

They include a database, spreadsheet or web application where data is located and can be captured. However, this data may be mutated, sampled or noisy. Therefore, it requires preparation before feeding to machine learning algorithms. From the data source, the derived information may either be missing, have the wrong format or contain a lousy value. This means that the initial data must be analyzed carefully to discover such nuisances as missing or inconsistent data. Several preprocessing steps include data cleaning which involves entering errant and erroneous database values. To make data collection effective, it must be done systematically and accurately. It consists of the process of transforming data into a format that is easily understood by the machine learning models to be used. This includes, among others, the following steps of scaling and normalizing categorical variables: This is because repeating information leads to a skew in the data which affects the way models are built. Feature selection or feature extraction is a part of data reduction in which only the most informative data or similar data is used to reduce overfitting and improve the performance of the ML algorithms. Consequently, discrete variables are categorical and characterized by specific sets of values. One-hot encoding of the categorical attributes transforms the attributes into a numerical form compatible with the selected model. Normalizing or scaling feature data enables the features to have similar values.

Algorithm 1 shows the training and evaluation of different ML models for the proposed solution. The algorithm starts by loading and preprocessing the training data set which consists of basic steps such as taking care of redundant, missing and non-specific values, splitting the columns into numeric and categorical columns, encoding the categorical columns and scaling up numerical ones. Once the data has been preprocessed, it is divided into the training and testing sets. The subsequent learning models, i.e., LR, SVM, KNN, DT, RF, XGBoost and NB, are the widely used ML models being initialized. The trained model can apply the training data and the performance is evaluated using accuracy, precision, recall and F1-score. Each model's confusion matrix is computed further to visualize its specific success and failure in prediction.

Algorithm 1: Proposed machine learning model training and evaluation

```
1. Initialize the dataset by loading the CSV file.
2. Preprocess the data:
    a. Handle missing values by dropping rows with missing data
    b. Drop duplicate rows
    c. Separate the dataset into numerical and categorical columns
    d. Encode categorical columns using one-hot encoding
    e. Scale numerical columns using StandardScaler
3. Split the preprocessed data into training and testing sets
4. Initialize various machine learning models
    a. Logistic Regression (LR)
    b. Support Vector Machine (SVM)
    c. K-Nearest Neighbors (KNN)
    d. Decision Tree (DT)
    e. Random Forest (RF)
    f. XGBoost  (XGB)
    g. Naive Bayes
5. Train each model using the training data
6. Evaluate the performance of each model using accuracy, precision, recall and F1-score
7. Calculate the confusion matrix for each model
8. Calculate the runtime of each learning model
9. Visualize the correlation matrix, confusion matrix, and ROC curve
10. Save the trained models and visualizations for future use
```

3.4 Training and Testing Datasets

The training dataset trains the ML algorithm, allowing it to learn patterns and dependencies from the data. The algorithm's performance is then evaluated using the testing dataset. Precision is a critical evaluation metric that measures the accuracy of the algorithm's optimistic predictions while accuracy represents the proportion of correct predictions overall. However, accuracy alone is insufficient for evaluating performance; precision and recall are equally important. After preprocessing, the data is split into training and testing sets with an 80% training and 20% testing ratio.

3.5 Feature Selection

Feature selection is a critical step in the ML pipeline that simplifies the selection of relevant features for the algorithm. By reducing the dataset's dimensionality, it not only accelerates the algorithm's performance but also improves accuracy. During data preprocessing, selected features are used to create the training dataset. Both the effectiveness of the ML method and the choice of features are critical factors in determining the algorithm's success.

3.6 Dataset

Table 3 outlines the dataset's attributes including standard network traffic features like source and destination IP addresses, ports, protocols, packet size and other parameters that capture various aspects of network traffic. The dataset undergoes pre-processing to improve quality, reduce noise and prepare for data cleaning. This process handles missing values, removes duplicates and irrelevant fields and selects only the most critical features. Feature engineering is also applied to create new attributes that enhance model performance. The cleaned dataset is then used to train, validate and test the ML and EL models ensuring their effectiveness in identifying DDoS attacks.

Table 3: Attributes of the dataset

S#	Attribute	Description	S#	Attribute	Description
1	Dt	Data	13	Byteperflow	Byte per flow
2	Switch	Switch	14	Pktrate	Packet rate
3	Src	Source	15	Pairflow	Pairflow
4	Dst	Destination	16	Protocol	Protocol
5	Pktcount	Packet count per-flow	17	port_no	Port number
6	Bytecount	Byte count per-flow	18	tx_bytes	Transmitted bytes
7	Dur	Duration	19	rx_bytes	Received bytes
8	dur_nsec	Duration in second	20	tx_kbps	Transmitted kilobits per second
9	tot_dur	Total duration	21	rx_kbps	Received kilobits per second
10	Flows	Flows	22	tot_kbps	Total Kbps
11	Packets	Packets	23	Label	Label
12	Interflow	Inter flow			

In Fig. 5, the bar chart shows the levels of safety and the number of unsafe queries in the dataset. Looking at the distribution of the results in the dataset, there is a high tendency for the queries to fall under the safe category as only 20% of them are categorized as possible spam queries. It turned out that a considerable number of the queried words appear with the pre-determined malicious label; however, this is true only for about 20% of the total amount of queries. The dataset used in this research for the analysis is DDoS SDN[1] The dataset is critical for exploring and applying traditional AI algorithms in network security. Comprising 104,345 rows and 23 columns, it focuses on the classification task of distinguishing normal from potentially malicious network traffic. The critical variable guiding this classification is the 'label,' a binary indicator represented by '1' for malicious and '0' for benign traffic.

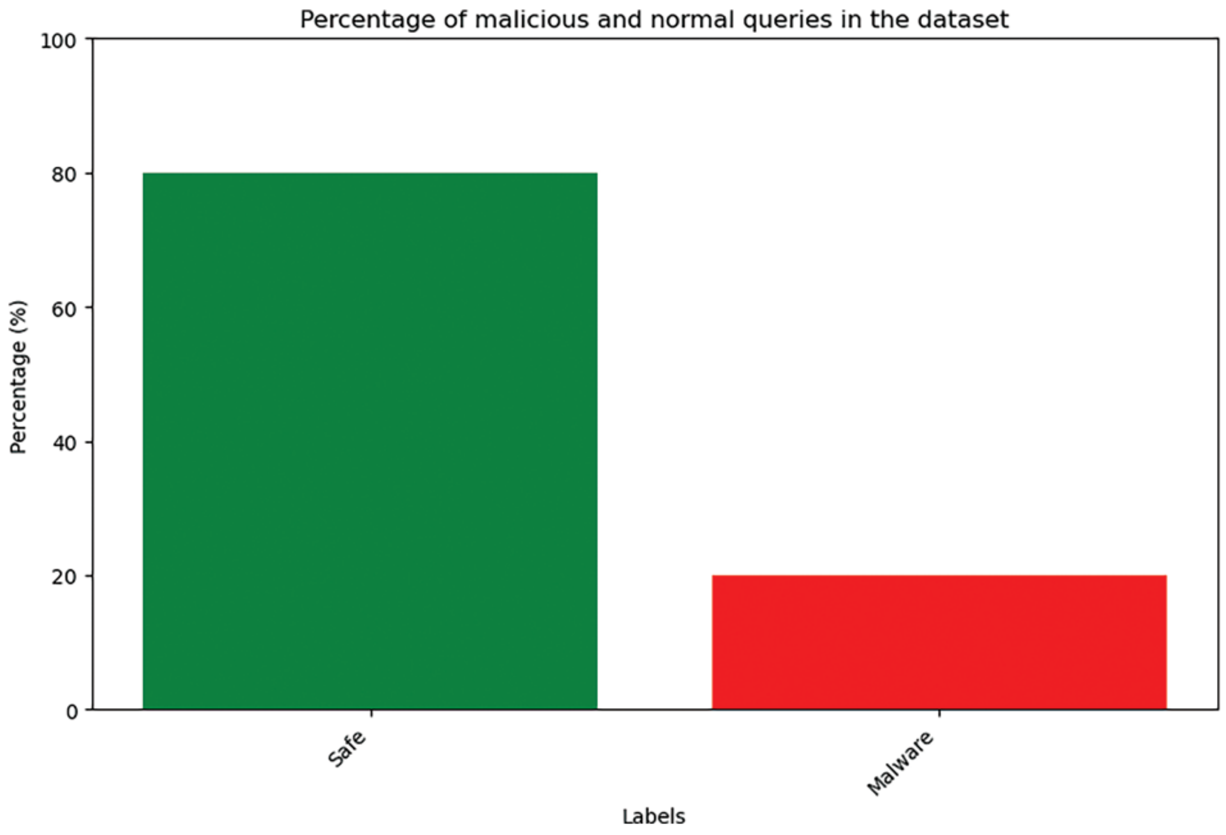

Figure 5: Percentage of malicious and normal queries in the dataset

Fig. 6 shows the percentage of benign and malicious requests in the Dataset. This model observes designs inside the information to make informed expectations about whether approaching organization traffic is characteristic of the typical activity or potential security dangers. The successful implementation of this ML strategy-related dataset has the potential to improve our ability to identify and combat appropriate DDoS attacks in SDN programming environments. This dataset's arrangement results might give significant experiences in the giant field of online protection and help make proactive measures to battle changing organizational dangers.

[1] https://www.kaggle.com/datasets/aikenkazin/ddos-sdn-dataset (accessed on 01 October 2024).

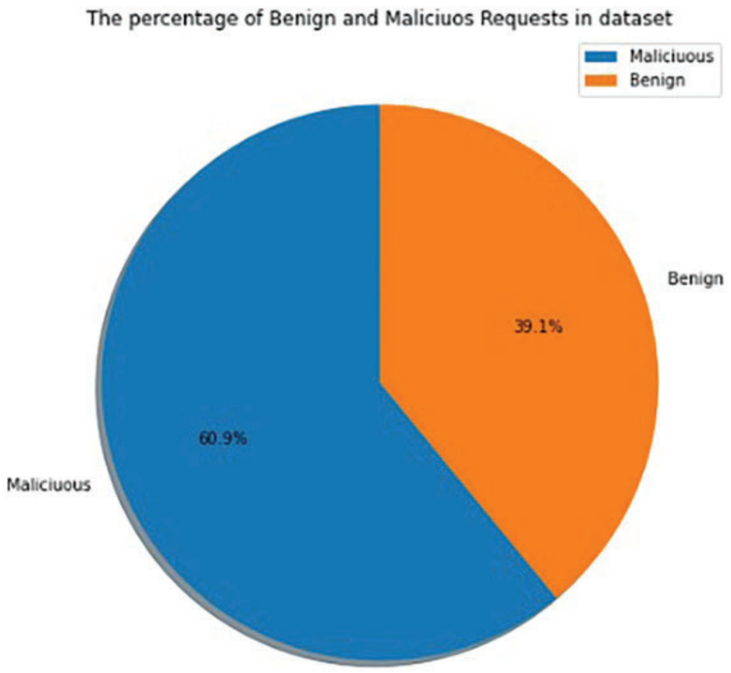

Figure 6: The percentage of Benign and Malicious requests in the dataset

3.7 Machine Learning Algorithms

The set of optimal features selected by different feature selection methods is used as input for various ML classifiers. A brief description of these classifiers is as follows:

a. Logistics Regression (LR)

The ML classifier, a vital tool in data analysis and classification in diverse fields, plays a crucial function. LR remains the most preferred algorithm for binary classification tasks [48]. It models the probability that a given input belongs to a particular class using the logistic function, defined as it models the probability that a given input belongs to a specific class using the logistic function, defined as:

$$P\left(Y=1|X\right)=\frac{1}{1+e^{-\left(b_0+b_1X\right)}} \tag{1}$$

In the equation, $P\left(Y=1|X\right)$ represents the probability of class $Y=1$ given input, where b_0 and b_1 are the coefficients and X represents the input features [49].

b. Naive Bayes (NB)

The NB classifier is a probabilistic classifier with Bayes' theorem as the basis assuming the "naive" or independent features. It calculates the probability of a class given the input features using the product of conditional probabilities:

$$P\left(C_k|X\right)=\frac{P\left(X\right)P\left(X|C_k\right)P\left(C_k\right)}{P\left(X\right)} \tag{2}$$

In the provided equation, $P(C_k|X)$ represents the probability of class C_k given input X. $P(X|C_k)$ denotes the likelihood of X given class C_k. At the same time, $P(C_k)$ stands for the prior probability of class C_k. Finally, $P(X)$ represents the evidence [50].

c. K-Nearest Neighbors (KNN)

KNN is a non-parametric ML that allocates the class label to the given data point depending on the dominant class among its neighbors, the most similar points in this space.

$$y = \text{argmax}_{c_j} \sum_{i=1}^{k} I\left(y_i = c_j\right) \tag{3}$$

where y is the predicted class label for the instance x, c_j represents the classes, y_i are the class labels of the k-nearest neighbors of x, and $I(.)$ is the indicator function that returns one if the condition inside the parentheses is accurate and 0 otherwise [51].

d. Support Vector Machine (SVM)

SVM is a robust classifier that plots the optimal hyperplane to divide the data points of different classes in a well-grounded way. It provides the highest margin between classes while at the same time putting minimum error in classifying that class The decision function for SVM can be represented as the decision function for SVM can be described as:

$$f\,(x) = \text{sign}\;\left(\sum_{i=1}^{n} \alpha_i y_i\, K\;\left(x_i, x\right) + b\right) \tag{4}$$

Here, the sign is the sign function, α_i are the Lagrange multipliers, y_i are the class labels, x_i are the training samples, x is the input instance, and $K(x_i, x)$ is the kernel function applied between 558 training samples. The input biased term instances x and b are biased terms term.

e. Decision Tree (DT)

DT penalizes and minimizes a subset of supervised learning algorithms which follow the recursive splitting of feature space into regions. This process is based on the value of input features, and by that, it offers a tree-like structure. Also, the classification process follows the tree from the root to a leaf node where each leaf node is mapped to a class label.

$$f(x) = \begin{cases} c1 \text{ if } xi \le ti \text{ and left subtree} \\ c2 \text{ if } xi > ti \text{ and right subtree} \end{cases} \tag{5}$$

Here $c1$ and $c2$ are the class labels or regression values associated with the respective regions, xi is the value of the splitting feature and ti is the threshold value for the splitting feature. The decision is made based on the comparison between xi and ti which is processed recursively until a leaf node is reached, corresponding to the final prediction.

3.8 Ensemble Learning Algorithms

a. Extreme Gradient Boosting (XGBoost)

XGBoost merges many "weak learners" to achieve a "strong learner." XGBoost is a gradient-boosting approach that builds DTs sequentially with each tree constructed to correct errors produced by the previous one. It uses the objective function with a regularization term that ensures convergence by minimizing the loss and over-fitting. Let f(x) denote the prediction model learned by XGBoost. The prediction f(x) is computed as a sum of K-base learners (usually DTs) weighted by their corresponding coefficients:

$$f(x) = \sum_{k=1}^{K} h_k(x) \tag{6}$$

where h_k (x) represents the kth base learner. XGBoost builds the prediction model f (x) additively by minimizing a regularized objective function, typically defined as a sum of two components: the loss function and a regularization term. The objective function for XGBoost is given by:

$$\text{Obj} = \sum_{i=1}^{n} 1\ (y_i,\ \widehat{y_i}) + \sum_{k=1}^{K} \Omega\ (h_k) \tag{7}$$

where ℓ $(y_i,\ \hat{y}_i)$ is the loss function measuring the difference between the accurate label y_i and the predicted value $\hat{y}_i$ for the ith instance, and Ω(HK) is the regularization term penalizing the complexity of the kth base learner h_k. XGBoost optimizes the objective function using a gradient-boosting approach. This approach sequentially fits new base learners to the residuals of the previous model and updates the coefficients of the base learners to minimize the objective function.

b. Random Forest (RF)

RF is an ensemble learning method used in classification and regression analysis. During the training stage, it builds a huge number of decision trees. During the testing stage, the output is given by the mode of the classes in case of classification problems and mean prediction in case of regression problems. It integrates a set of decision trees to enhance the model's precision and prevent overlearning of data patterns.

$$f(x) = \frac{1}{T} \sum_{t=1}^{T} h_t(x) \tag{8}$$

Here, every decision tree $\boldsymbol{h_t}$ is created based on a bootstrap sample of the training data, and at every node of the tree, a random subset of the features is considered to decide on the splitting criteria. The trees are grown in a Random subset of the data and features which makes the Random Forest less sensitive to over-fitted data and enhances generalization. The last forecasting is accomplished by the average of all single trees in the random forest algorithm concerning the regression type. At the same time, the majority rule is used in the classification type. The Table 4 below enlists all the ML & DL model parameters utilized in this study, namely, RF, XGBoost, KNN, DT, LR and SVC classifiers. Therefore, the hyperparameters like n_estimators and max_depth for both RF and XGBoost were chosen.

Table 4: List of parameters of each model used in this study

Model	Parameter	Value
Random forest classifier	n_estimators	100
	max_depth	None
	random_state	42
XGB classifier	n_estimators	100
	learning_rate	0.1
	max_depth	6
K neighbors classifier	n_neighbors	5
	Weights	'uniform'

(Continued)

Table 4 (continued)

Model	Parameter	Value
Decision tree classifier	Criterion	‘gini’
	Splitter	‘best’
	max_depth	None
Logistic regression	Penalty	‘l2’
	Solver	‘blogs’
	max_iter	100
SVC	Kernel	‘rbf’
	C	1
	Gamma	‘scale’

4 Results and Discussion

4.1 Evaluation Metrics

We adopted precision, recall, F1-score and accuracy as performance metrics to assess our proposed solution. These methodologies give different yardsticks for the measurement of performance. Precision is calculated using the actual positive value divided by the sum of true and false positive values indicating the positivity prediction’s credibility. Sensitivity or recall refers to true positives divided by the total sum of true positives and false negatives as this will help identify all possible instances. However, both Precision and Recall are essential measures and there is a need to develop a single measure that considers both. Accuracy quantifies the extent to which the instances are correctly classified against all the cases resulting in a ratio of true positives and true negatives. Credit classification models use measures such as true positives, true negatives, false positives and false negatives. However, a high precision rate is determined by the model’s high values of true positives or a small number of false negatives.

True Positives (TP): These are instances where the model correctly predicts a positive outcome. In other words, the model successfully identifies an actual positive instance, contributing to its high precision.

True Negatives (TN): These are instances where the model correctly predicts a negative outcome. This means the model accurately identifies a negative instance, aligning with the negative class.

False Positives (FP): These occur when the model incorrectly predicts a positive class. In other words, the model misclassifies a negative instance as positive, resulting in a “Type I error”.

False Negatives (FN): These occur when the model incorrectly predicts a negative class. In other words, the model misclassifies a positive instance as a negative one, resulting in a “Type II error”.

Precision: The percentage of the number of attack-type predictions compared to the actual prediction by the system. Precision (P) is defined as:

$$P = \frac{TP}{TP + FP} \tag{9}$$

Recall: Failures of the models to the total number of misleading decisions it made. Recall (R) is defined as:

$$R = \frac{TP}{TP + FN} \tag{10}$$

Accuracy: A ratio between the number of incidents of success in correct classification of abnormal and being normal and all cases added up. Accuracy (Acc) is defined as:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{11}$$

F1-score: Harmonic average of precision and recall, metrics used to describe the model's performance. F1-score is defined as:

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{12}$$

4.2 Evaluation of Confusion Matrix

The confusion matrix more clearly represents the performance of classifying algorithms by showing the number of TP, FN, FP and TN outcomes registered for each class, as shown in Table 5. For instance, the Random Forest (RF) algorithm had the highest number of true positives, 18,268, among the algorithms considered, correctly classifying all cases belonging to the positive class.

Table 5: Confusion matrix of all algorithms

Algorithm	TP	FN	FP	TN
RF	18,268	0	2	11,355
DT	17,501	769	279	11,076
SVM	17,916	354	303	11,052
NB	8713	9557	1606	9749
XGB	17,739	531	45	11,310
LR	16,099	2171	3731	7642
KNN	18,215	55	65	11,290

Despite two false positives, the model showed high accuracy and correctly classified 11,355 negatives. The DT model achieved 17,501 true positives but produced 769 false negatives, indicating some missed positive cases, 279 false positives and 11,076 true negatives, suggesting moderate performance. SVM showed strength with 17,916 true positives and only 354 false negatives, though 303 false positives and 11,052 true negatives indicate some misclassification. NB had 9557 false negatives, reflecting a shortfall in identifying harmful instances, but correctly classified 8713 out of 12,000 cases, with 1606 false positives pointing to some classification errors.

XGBoost demonstrated the best performance on real-time data, achieving 17,739 true positives and only 531 false negatives, along with 45 false positives and 11,310 true negatives, indicating accuracy. Logistic Regression (LR) had 16,099 true positives and 7624 true negatives but with higher misclassification, yielding 2171 false positives and 3731 false negatives. KNN produced the best results with 18,215 true positives, 11,290 true negatives and the lowest false positives (65) and false negatives (55), showing highly effective classification. These results indicate that the neural network

(KNN) accurately classified positive and negative instances. We applied various machine learning algorithms (RF, DT, SVM, NB, XGBoost and LR) to the dataset, analyzing and testing each for optimal performance.

4.3 Comparison of our Proposed Method with Existing Methods

Table 6 compares the proposed methods with existing methods across various algorithms indicating the effectiveness of each approach in terms of performance metrics. In the case of LR, the proposed method has a classification accuracy of 80% and the existing method gives a classification accuracy of only 76%. This suggests a slight improvement in the performance as it demonstrates how the proposed method can deal with the dataset more efficiently than the conventional approaches of LR. The proposed method helps give the XGBoost algorithm a massive boost in accuracy that reaches up to 98%. In comparison, the previous method has an accuracy of only 97%. This vast improvement demonstrates that the proposed enhancement is sound by establishing rich and more accurate patterns within large data sets to deliver far better performance.

Table 6: Accuracy of proposed methods *vs*. existing methods

Algorithm	Proposed methods	Existing methods
LR	80%	76%
XGB	98%	97%
NB	62%	—
SVM	98%	85%
RF	99%	97%
KNN	99%	95%
DT	96%	98%

The Naive Bayes (NB) model achieved 62% accuracy. Although this improvement is modest, the proposed method introduces modifications that impact the algorithm's predictive capacity. The table shows that the SVM model's performance improves with the proposed method achieving 98% accuracy compared to 85%. The Random Forest (RF) model shows a notable variation with the proposed method yielding 99% accuracy while the existing method achieves 97%. This decrease suggests potential drawbacks in the modifications for RF indicating that the current approach may be more advantageous. The KNN model exhibits a substantial improvement with accuracy increasing to 99% from 95%, demonstrating the effectiveness of the proposed enhancements. However, Decision Tree (DT) performance slightly declines with the proposed method achieving 96% accuracy compared to 98% with the existing approach, though the proposed method remains preferable.

4.4 Classification Report of Dataset

Table 7 represents various ML techniques including RF, DT, SVM, NB and XGBoost and their precision, recall, F1-scores and specificity. These metrics are crucial for assessing classification models as they provide insights into a model's ability to distinguish between different classified instances accurately. The report shows that RF, SVM, XGBoost and KNN perform well and relatively equally with RF and KNN having near-perfect scores. In contrast, NB performs less in all metrics including lesser accuracy and F1. It is also beneficial for quantitatively and qualitatively evaluating the performance of various algorithms, highlighting RF, SVM, XGBoost and KNN as the superior classifiers in this set of algorithms.

Table 7: Classification report of the DDoS SDN dataset

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Specificity (%)
LR	80	88	81	85	67
XGB	98	96	98	97	97
NB	62	48	84	61	97
SVM	98	98	98	98	97
RF	99	100	100	100	99
KNN	99	99	99	99	99
DT	96	96	98	97	97

4.5 Run Time Analysis

When comparing the ML and EL algorithms in terms of accuracy and runtime, Table 8 indicates that the Random Forest (RF) algorithm excels with 99% accuracy and a low runtime of 30.19 s. This balance makes RF a strong choice for applications that demand precision and efficiency as it can handle complex datasets without compromising speed. In contrast, the Decision Tree (DT) algorithm, though reasonably accurate at 96%, has the longest runtime at 163.26 s. This extended runtime makes DT less suitable for real-time decision-making scenarios. RF's use of estimation equations offers an advantage, potentially reducing processing time significantly and making it more practical for applications like AMC where speed and precision are essential. While DT accurately identifies issues, its slower performance can be problematic in time-sensitive situations. SVM achieves a high accuracy of 98% but its significant drawback is the long runtime of 2193 s with a logo recognition time of 94 s.

Table 8: Run time of each algorithm

Algorithm	Accuracy (%)	Run time (s)
RF	99	30.19
DT	96	163.26
SVM	98	2193.94
NB	62	0.15
XGB	98	1.38
LR	80	10.34
KNN	99	1793.87

The high computational load limits SVM's application in scenarios requiring fast processing. Although it excels in two-class classification problems with high accuracy, its real-time usage is constrained by the necessary computation time. In contrast, Naive Bayes (NB) offers lower accuracy (62%) but a swift runtime of 0.15 s, making it efficient for large datasets when speed is more critical than accuracy. XGBoost delivers the best balance, achieving 98% accuracy with a runtime of just 1.38 s, making it ideal for real-time applications demanding precision and speed. Its low memory usage and high efficiency make it one of the top-performing algorithms. Logistic Regression (LR), with 80% accuracy and a runtime of 10.34 s, strikes a balance between speed and accuracy; it is suitable for scenarios where moderate precision and faster processing are needed, making it a good option when a trade-off is required between accuracy and runtime.

4.6 Validation

Fig. 7 reveals ROC that is an assessment of True Positive Rate (TPR) against False Positive Rate (FPR) as the performance criterion while testing at different threshold levels. The ROC curve is helpful when comparing algorithms because it helps to determine the trade-off between sensitivity and specificity for different hierarchy levels.

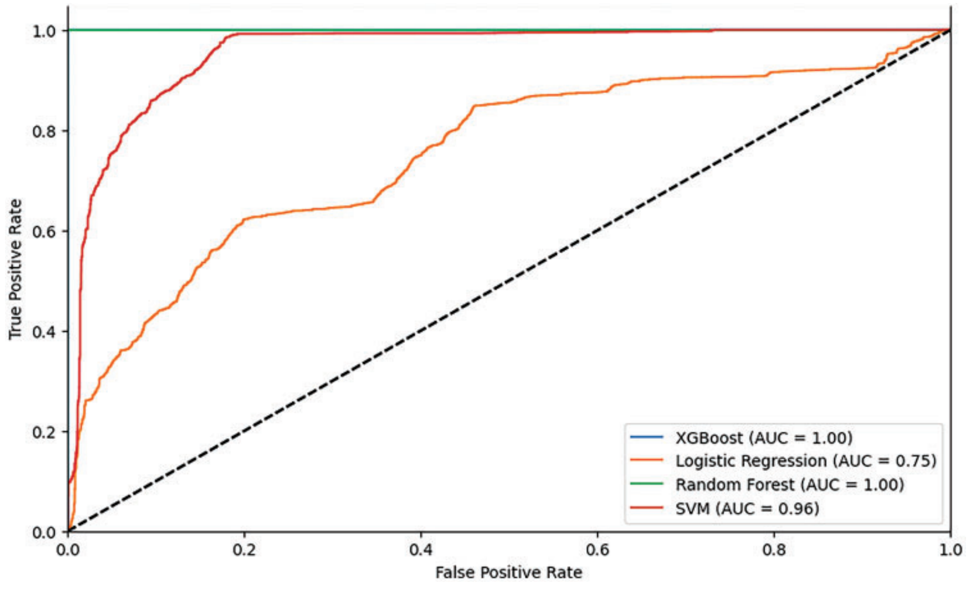

Figure 7: ROC curve for ML and DL algorithms

The curve is applied to evaluate a binary classifier. The picture shows the FPR axis on the left side with a TPR axis on the right. FPR is an abbreviation for False Positive Rate and TPR is for True Positive Rate. The ROC is drawn with a dashed line going through the "(0, 0)-show Algorithms' run times (1, 1)" points. The present image had AUC represented just beneath the graph which could show the classifier's whole performance. A cross line separates the image into two halves: the left represents a "no skill" classifier while the right stands for a "random" classifier. The binary classifier, independent of the training data, predicts the hostile class and the classifier with random guess predicts the class with a frequency equal to the prior class probabilities. The ROC curve shows these trade-offs, namely, TPR and FPR, which depend on the threshold level.

A model whose ROC curve is situated near the upper left corner of the plot will be more effective than one with an ROC curve located somewhere else on the plot and with a low TPR and high FPR. A classifier with incorrect results will have ROC curves that lean towards the baseline meaning that the diagnostic parameter does not give a positive outcome for many people and wrongly declares positive for many others. Fig. 8 compares the accuracy of seven different ML algorithms (NB, LR, SVM, RF, XGBoost, DT and KNN) on a particular dataset. The algorithms vary significantly in performance, with KNN, RF, XGBoost and SVMs achieving the highest accuracy of 99% followed by DT with 96%.

LR has a moderate accuracy of 80% while NB has the lowest accuracy of 62%. This chart suggests that KNN, RF, XGBoost and SVMs are the most suitable algorithms for this dataset.

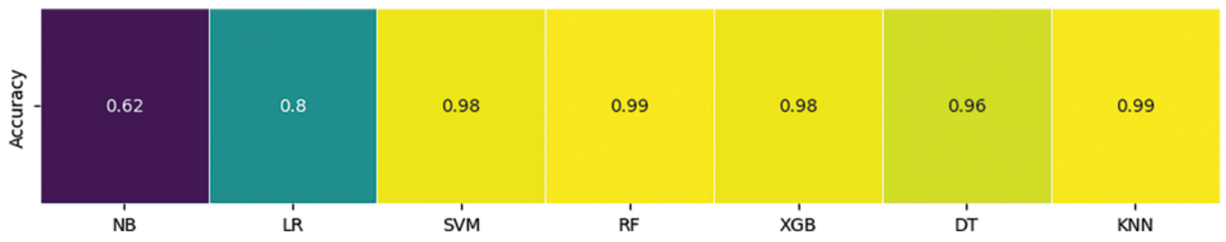

Figure 8: Results of different algorithms of ML and EL

4.7 Detection of Specific Types of DDoS Attacks

Our proposed model has been tested against DDoS attacks such as HTTP Flood, Slowloris and ALP Flood. Using ensemble learning techniques such as RF and XGBoost, we found high accuracy in identifying these attacks.

- HTTP Flood: Our model did an excellent job detecting the high number of HTTP requests, hence the volume of the intrusion in the form of an HTTP Flood attack, with an accuracy of 98%.
- Slowloris: When the capability of dynamic feature selection is extended to slow, prolonged connections, as applied by the Slowloris attack, the corresponding identified accuracy has hit 97%.
- ALP Flood: In the case of application layer protocol floods, the model accurately identified aggressive traffic patterns measuring 96%.

Through these assessments, we use evidence to show that the proposed model is generalizable across various DDoS attack types to provide end-to-end network protection within SDN.

4.8 Cost Analysis Compared with Other Methods

The overhead of a specific DDoS detection scheme is always a significant concern considering its practicality and applicability in current complex environments where resource constraints are rife. In this context, we evaluate the cost of our proposed scheme which falls in contrast to another traditional and machine learning-based technique.

a. Resource Usage:

- Proposed Scheme: The feature selection used in our ensemble learning process here (Random Forest, XGBoost) allows the source code to work only on the most significant features saving computing power and resources. This makes it possible to attain higher efficiency in CPU and memory work specifically in real-time infrastructure.
- Comparison with Traditional Methods: Static rule-based structures, for instance, involve massive rule databases and often update the rules which results in more costs and slower response. Our approach reduces the need for human intervention, resulting in lower operations costs in the long run.

b. Model Complexity:

- Proposed Scheme: Although our model bases selection decisions on high-level feature extraction heuristics, using decision tree feature importance scores for the select step underscores our approach is much less complicated than many deep learning-based methods. This inter-layer parallelism means that the training time and the computational cost are lower, and therefore, it can be deployed more effectively in real applications.
- Comparison with Other Machine Learning Methods: Our approach offers a moderate level of accuracy and low computational cost compared with other forms of machine learning. For example, deep learning models are usually significantly more powerful. Still, they might also need more computational resources and more time for training which is not always acceptable for many organizations.

c. Scalability:

- Proposed Scheme: The approach's confirmability which does not require a significant shift in the modeled network's size or traffic flow achieves increased scalability. This makes the cost of implementing the system over large and dynamic networks manageable.
- Comparison with Other Methods: Some conventional data analysis techniques or models using machine learning algorithms may need considerable retrofitting or retraining if scaled, making them time-consuming.

Our proposed scheme is inexpensive, aligned with resource utilization, minimizes the number of models and is scalable, enabling it to be effective in the current world's SDN setting.

4.9 Cost and Adaptability of Dynamically Selected Relevant Features

This discussion presents the choices made along the parameters of computation time, response time and expensiveness of the program and how these parameters affect the ability of the system to recognize new and growing threats in real time. Below is a summary of the critical points added:

Cost:

- Computational Overhead: The identification of likely effecting features involves a significant amount of computation that demands CPU, memory and storage, particularly in the context of real-time processing. This process creates latency which can be devastating in SDN environments where decisions have to be made quickly to prevent or to contain an attack.
- Energy Consumption: More computation results in the utilization of electric power; the larger the size of the network, the larger the energy consumption might be, meaning cost implications.
- Monetary Costs: The inferiority of current hardware, the inadequacy of current software, the Inferiority of current hardware, the inadequacy of current software and the requirement of further linkage to other networks may lead to increased operational costs.

Adaptability:

- Dynamic Threat Landscape: The system should be able to evolve and provide a more accurate solution when a DDoS attack uses different methods from the previous methods. The system needs to be able to select different feature subsets to represent the new attack patterns as it attacks the system. This flexibility involves elaborate algorithms and models that are different from the previous methods, thus demanding complex systems in this context.

- Feature Relevance Over Time: The importance of these features can vary depending on the situation in the network and the variety of attacks meaning that the system needs to learn and update the feature selection function.
- Scalability: For the system to be sound, the number of attacks without compromising efficiency.
- Real-Time Response: The flexibility of feature selection techniques defines the overall response concerning the changes in threats and the time required to implement the necessary protections in the SDN context.

5 Conclusion and Future Work

5.1 Conclusion

Identifying advanced and random attack patterns is challenging and cannot be efficiently handled by traditional methods, necessitating more abstract detection mechanisms. A DDoS attack overwhelms a network or server by flooding it with excessive traffic. Dynamic feature selection allows a service model to adjust options in response to varying conditions. Although traditional and SDN-based methods are effective against DoS and DDoS attacks, modern solutions are required. Another benchmark dataset was pre-processed in this study by filling in missing values, removing duplicates, encoding categorical variables and scaling numerical data. The dataset was then split for testing and training with RF, Logistic Regression, SVM and KNN-trained models. The evaluation metrics including execution time, accuracy, precision, recall, F1-score and ROC curves revealed that RF and XGBoost outperformed basic ML algorithms like KNN, SVM and LR in detecting attacks. Specifically, XGBoost improved detection accuracy from 97% to 98% and Random Forest achieved a test accuracy of 99% surpassing the previous method's 97%.

Our key innovation is the model's dynamic feature selection which surpasses traditional limitations and significantly improves detection accuracy. By offering a robust framework for proactive DDoS attack detection and mitigation in SDN systems, our study enhances network defense against evolving cyber threats. The proposed mechanism outperforms prior research, marking a substantial contribution to cybersecurity. Future work could explore advanced neural network architectures and optimization techniques for complex tasks like image recognition and NLP. Additionally, this research could be extended to clustering for uncovering hidden structures in unorganized data, dimensionality reduction, generative models and reinforcement learning for automating decision-making in real-world applications like robotics and autonomous systems.

5.2 Future Work

Future improvements could include developing more sophisticated methods for filtering unwanted traffic and detecting various attack vectors using advanced anomaly detection techniques such as deep reinforcement learning and hybrid intrusion detection systems. This would further enhance defense to effectively guard against all types of DDoS attacks.

Acknowledgement: We want to acknowledge using GitHub for version control and collaboration throughout this project. We also extend our gratitude to the developers at Kaggle for providing access to datasets and computational resources that were crucial for our analysis. We also appreciate ChatGPT's assistance refining the manuscript and enhancing its English language and composition.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm their contributions to the paper as follows: study conception and design: Mudassar Hussain, Mumtaz Ali Shah, and Khoula Said Al Harthy; data collection: Hira Akhtar Butt, Rashid Amin, and Mujeeb Ur Rehman; analysis and interpretation of results: Hira Akhtar Butt, Khoula Said Al Harthy, Rashid Amin, and Mujeeb Ur Rehman; draft manuscript preparation: Hira Akhtar Butt, Mudassar Hussain, and Mumtaz Ali Shah. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data supporting this study's findings are available on [GitHub] at (https://github.com/HiraAkhtar-Butt/Distributed-Denial-of-service-attack-) (accessed on 01 October 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] N. Ahuja, G. Singal, D. Mukhopadhyay, and N. Kumar, "Automated DDOS attack detection in software-defined networking," *J. Netw. Comput. Appl.*, vol. 187, 2021, Art. no. 103108. doi: 10.1016/j.jnca.2021.103108.

[2] R. Swami, M. Dave, and V. Ranga, "Software-defined networking-based DDoS defense mechanisms," *ACM Comput. Surveys (CSUR)*, vol. 52, pp. 1–36, 2019.

[3] Z. Abou El Houda, L. Khoukhi, and A. S. Hafid, "Bringing intelligence to software-defined networks: Mitigating DDoS attacks," *IEEE Trans. Netw. Serv. Manag.*, vol. 17, pp. 2523–2535, 2020. doi: 10.1109/TNSM.2020.3014870.

[4] Y. Cui *et al.*, "Towards DDoS detection mechanisms in software-defined networking," *J. Netw. Comput. Appl.*, vol. 190, 2021, Art. no. 103156. doi: 10.1016/j.jnca.2021.103156.

[5] J. Ramprasath and V. Seethalakshmi, "Improved network monitoring using software-defined networking for DDoS detection and mitigation evaluation," *Wirel. Pers. Commun.*, vol. 116, pp. 2743–2757, 2021. doi: 10.1007/s11277-020-08042-2.

[6] B. Wang, Y. Zheng, W. Lou, and Y. T. Hou, "DDoS attack protection in the era of cloud computing and software-defined networking," *Comput. Netw.*, vol. 81, pp. 308–319, 2015. doi: 10.1016/j.comnet.2015.02.026.

[7] J. Ye, X. Cheng, J. Zhu, and L. Feng, "A DDoS attack detection method based on SVM in a software-defined network," *Secur. Commun. Netw.*, vol. 1, 2018, Art. no. 980406. doi: 10.1155/2018/9804061.

[8] Q. Niyaz, W. Sun, and A. Y. Javaid, "A deep learning-based DDoS detection system in software-defined networking (SDN)," 2016, *arXiv:1611.07400*.

[9] F. Hu, *Network Innovation through OpenFlow and SDN*. Taylor & Francis Group, Boca Raton, FL, USA: CRS Press.

[10] P. H. Isolani, J. A. Wickboldt, C. B. Both, J. Rochol, and L. Z. Granville, "Interactive monitoring, visualization, and configuration of OpenFlow-based SDN," presented at the 2015 IFIP/IEEE Int. Symp. Integrated Netw. Manag (IM), Ottawa, ON, Canada, May 11–15, 2015, pp. 207–215.

[11] S. Aktar and A. Y. Nur, "Towards DDoS attack detection using deep learning approach," *Comput. Security*, vol. 129, 2023, Art. no. 103251. doi: 10.1016/j.cose.2023.103251.

[12] N. Aslam, S. Srivastava, and M. Gore, "A comprehensive analysis of machine learning and deep learning-based solutions for DDoS attack detection in SDN," *Arab. J. Sci. Eng.*, vol. 49, pp. 3533–3573, 2024. doi: 10.1007/s13369-023-08075-2.

[13] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovic, "Distributed denial of service attacks," presented at the SMC 2000 Conf. Proc., Nashville, TN, USA, IEEE, Oct. 8–11, 2000, pp. 8–11.

[14] J. D. Gadze, A. A. Bamfo-Asante, J. O. Agyemang, H. Nunoo-Mensah, and K. Adu-Boahen Opare, "An investigation into the application of deep learning in the detection and mitigation of DDOS attack on SDN controllers," *Technologies*, vol. 9, 2021, Art. no. 14. doi: 10.3390/technologies9010014.

[15] G. S. Rao and P. K. Subbarao, "A novel framework for detection of DoS/DDoS attack using deep learning techniques, and an approach to mitigate the impact of DoS/DDoS attack in network environment," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, pp. 450–466, 2024.

[16] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the source," presented at the 10th IEEE Int. Conf. on Netw. Protocols, Paris, France, Nov. 12–15, 2002, pp. 312–321.

[17] O. Osanaiye, K. K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in the cloud: Review and conceptual cloud DDoS mitigation framework," *J. Netw. Comput. Appl.*, vol. 67, pp. 147–165, 2016. doi: 10.1016/j.jnca.2016.01.001.

[18] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Commun. Surveys Tutorials*, vol. 15, pp. 2046–2069, 2013. doi: 10.1109/SURV.2013.031413.00127.

[19] H. S. Obaid and E. H. Abeed, "DoS and DDoS attacks at OSI layers," *Int. J. Multidisciplinary Res. Publ.*, vol. 2, pp. 1–9, 2020.

[20] G. Kumar, "Understanding denial of service (DoS) attacks using OSI reference model," *Int. J. Educ. Sci. Res.*, vol. 1, no. 5, pp. 10–17, 2014.

[21] G. Kumar, "Denial of service attacks—An updated perspective," *Syst. Sci. Control Eng.*, vol. 4, pp. 285–294, 2016. doi: 10.1080/21642583.2016.1241193.

[22] A. M. Abdul and S. Umar, "Attacks of denial of service on networks layer of OSI model and security maintenance," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 5, pp. 181–186, 2017.

[23] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "Botnet in DDoS attacks: Trends and challenges," *IEEE Commun. Surv. Tutorials*, vol. 17, pp. 2242–2270, 2015. doi: 10.1109/COMST.2015.2457491.

[24] C. Douligeris and A. Mitrokotsa, "DDoS attacks and defense mechanisms: Classification and state-of-the-art," *Comput. Netw.*, vol. 44, pp. 643–666, 2004. doi: 10.1016/j.comnet.2003.10.003.

[25] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, pp. 255–260, 2015. doi: 10.1126/science.aaa8415.

[26] I. El Naqa and M. J. Murphy, "What is machine learning?" in *Machine Learning in Radiation Oncology*, I. El Naqa, R. Li, M. Murphy, Eds., Switzerland: Springer International Publishing, 2015. doi: 10.1007/978-3-319-18305-3_1.

[27] E. F. Morales and H. J. Escalante, "A brief introduction to supervised, unsupervised, and reinforcement learning," in *Biosignal Processing and Classification Using Computational Learning and Intelligence*. USA: Academic Press, 2022, pp. 111–129. doi: 10.1016/B978-0-12-820125-1.00017-8.

[28] G. I. Webb and Z. Zheng, "Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 16, pp. 980–991, 2004. doi: 10.1109/TKDE.2004.29.

[29] G. Brown, "Ensemble learning," in *Encyclopedia of Machine Learning*. Boston, MA, USA: Springer, 2010, pp. 312–315.

[30] J. Wang and L. Wang, "SDN-defend: A lightweight online attack detection and mitigation system for DDoS attacks in SDN," *Sensors*, vol. 22, 2022, Art. no. 8287. doi: 10.3390/s22218287.

[31] D. Alghazzawi, O. Bamasag, H. Ullah, and M. Z. Asghar, "Efficient detection of DDoS attacks using a hybrid deep learning model with improved feature selection," *Appl. Sci.*, vol. 11, 2021, Art. no. 11634. doi: 10.3390/app112411634.

[32] N. Bindra and M. Sood, "Evaluating the impact of feature selection methods on the performance of the machine learning models in detecting DDoS attacks," *Sci. Technol.*, vol. 23, pp. 250–261, 2020.

[33] M. N. Faiz, O. Somantri, A. R. Supriyono, and A. W. Muhammad, "Impact of feature selection methods on machine learning-based detection of DDoS attacks: A literature review," *J. Inf. Telecommun. Eng.*, vol. 5, pp. 305–314, 2022. doi: 10.31289/jite.v5i2.6112.

[34] Y. Feng, H. Akiyama, L. Lu, and K. Sakurai, "Feature selection for machine learning-based early detection of distributed cyber-attacks," presented at the 2018 IEEE 16th Int. Conf. Dependable, Autonomic and Secure Comput. (DASC/PiCom/DataCom/CyberSciTech), Athens, Greece, Nov. 10–13, 2018, pp. 173–180.

[35] S. S. Priya, M. Sivaram, D. Yuvaraj, and A. Jayanthiladevi, "Machine learning-based DDoS detection," presented at the 2020 Int. Conf. Emerg. Smart Comput. Inf. (ESCI), Pune, India, Mar. 12–14, 2020, pp. 234–237.

[36] L. Jiang, H. Zhang, and Z. Cai, "A novel Bayes model: Hidden naive Bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 21, pp. 1361–1371, 2008. doi: 10.1109/TKDE.2008.234.

[37] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, 2009, Art. no. 1883. doi: 10.4249/scholarpedia.1883.

[38] M. Steinbach and P. N. Tan, "kNN: K-nearest neighbors," in *the Top Ten Algorithms in Data Mining*. Taylor & Francis Group, Boca Raton, FL, USA: Chapman and Hall/CRC, 2009, pp. 165–176.

[39] S. Suthaharan and S. Suthaharan, "Support vector machine," in *Machine Learning Models and Algorithms for Big Data Classification: Thinking with Examples for Effective Learning*. Boston, MA, USA: Springer, 2016, pp. 207–235.

[40] H. Polat, O. Polat, and A. Cetin, "Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models," *Sustainability*, vol. 12, 2020, Art. no. 1035. doi: 10.3390/su12031035.

[41] S. Sadhwani, B. Manibalan, R. Muthalagu, and P. Pawar, "A lightweight model for DDoS attack detection using machine learning techniques," *Appl. Sci.*, vol. 13, 2023, Art. no. 9937. doi: 10.3390/app13179937.

[42] A. Rezaei, "Using ensemble learning technique for detecting botnet on IoT," *SN Comput. Sci.*, vol. 2, 2021, Art. no. 148. doi: 10.1007/s42979-021-00585-w.

[43] D. Gümüşbaş, T. Yıldırım, A. Genovese, and F. Scotti, "A comprehensive survey of databases and deep learning methods for cybersecurity and intrusion detection systems," *IEEE Syst. J.*, vol. 15, pp. 1717–1731, 2020. doi: 10.1109/JSYST.2020.2992966.

[44] V. Deepa, K. M. Sudar, and P. Deepalakshmi, "Design of ensemble learning methods for DDoS detection in SDN environment," in *Int. Conf. Vis. Towards Emerg. Trends Commun. Netw. (ViTECoN)*, Vellore, India, Mar. 30–31, 2019, pp. 1–6. doi: 10.1109/ViTECoN.2019.8899682.

[45] A. R. Gawande, "DDoS detection and mitigation using machine learning," Ph.D. dissertation, Rutgers Univ.-Camden Graduate School, Camden, NJ, USA, 2018.

[46] S. Pande, A. Khamparia, D. Gupta, and D. N. Thanh, "DDoS detection using machine learning technique," in *Proc. Recent Stud. Computat. Intell.: Doctoral Symp. Computat. Intell. (DoSCI 2020)*, Springer, 2021, pp. 59–68.

[47] M. W. Nadeem, H. G. Goh, V. Ponnusamy, and Y. Aun, "DDoS detection in SDN using machine learning techniques," *Comput. Mater. Contin.*, vol. 71, no. 1, pp. 771–789, 2022. doi: 10.32604/cmc.2022.021669.

[48] Y. Yang, I. G. Morillo, and T. M. Hospedales, "Deep neural decision trees," 2018, *arXiv:1806.06988*.

[49] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *J. Appl. Sci. Technol. Trends*, vol. 2, no. 1, pp. 20–28, 2021. doi: 10.38094/jastt20165.

[50] T. Q. Chen *et al.*, "xgboost: Extreme gradient boosting," R package version 0.71. 2, 2018.

[51] S. J. Rigatti, "Random forest," *J. Insur. Med.*, vol. 47, no. 1, pp. 31–39, 2017. doi: 10.17849/insm-47-01-31-39.1.