

Evaluation of distributed denial of service attacks detection in software defined networks

Neethu S.¹, H. V. Ravish Aradhya²

¹Department of Computer Science Engineering, RV College of Engineering, Visvesvaraya Technological University, Karnataka, India

²Department of Electronics and Communication Engineering, RV College of Engineering, Visvesvaraya Technological University, Karnataka, India

Article Info

Article history:

Received Jan 13, 2024

Revised Mar 22, 2024

Accepted Apr 17, 2024

Keywords:

Distributed denial of service attacks

Machine learning

OpenFlow protocol

Software-defined networks

Synthetic minority

oversampling technique

ABSTRACT

Software-defined networking (SDN) revolutionizes networking by separating control logic and data forwarding, enhancing security against threats like distributed denial of service (DDoS) attacks. These attacks flood control plane bandwidth, causing SDN network failures. Recent studies emphasize the efficacy of machine learning (ML) and statistical approaches in identifying and mitigating these security risks. However, there has been a lack of focus on employing ensembling techniques, amalgamating diverse ML models, selecting pertinent features, and utilizing oversampling techniques to balance categorical data. Our study evaluates 20 machine-learning models, emphasizing feature engineering and addressing class imbalance using synthetic minority oversampling technique (SMOTE). The results indicate that ensemble methods such as light gradient boosting machine (LGBM) classifier, random forest classifier, XGB classifier, decision tree classifier obtained near-perfect scores (almost 100%) across all metrics, suggesting potential overfitting. Conversely, models like AdaBoost classifier, k-neighbors classifier, and support vector classifier (SVC) exhibited slightly lower (99%) but realistic performance, underscoring the intricacy of accurate prediction in cybersecurity. Simpler models, including logistic regression, linear discriminant analysis, and Gaussian naive Bayes, demonstrated moderate to low accuracy, approximately around 70%. These findings stress the imperative need for a nuanced approach in the selection and fine-tuning of ML models to ensure effective DDoS detection in SDN environments.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Neethu S.

Department of Computer Science Engineering, RV College of Engineering

Visvesvaraya Technological University

Mysore Road, Bengaluru 560059, India

Email: neethus@rvce.edu.in

1. INTRODUCTION

The landscape of software-driven networks undergoes a revolutionary transformation through software-defined networking (SDN), a novel paradigm that effectively separates the control and data planes, addressing issues inherent in traditional network design. SDN has emerged as a paradigmatic shift, addressing limitations present in conventional IP-based networks. The architecture of SDN facilitates the decoupling of the control plane and data plane, enabling a centralized controller to abstract the underlying network framework from applications. This abstraction ensures simplified resource provisioning and enhanced programmability of network characteristics. Compared to traditional networks, the logical centralization of controllers in SDN provides improved visibility of the entire network, facilitating efficient management and optimization of

network resources [1]. Additionally, SDN allows for tailored quality of service (QoS) provisioning to meet the requirements of various applications. As a result, SDN has found widespread adoption in large-scale networks and data centers. OpenFlow technology [2] empowers OpenFlow controllers to manage multiple hardware or software entities.

The SDN architecture is vertically divided into three planes: data plane, controller plane, and application plane. The control plane orchestrates network configuration and policies. In traditional application-specific integrated circuit (ASIC) switches, where the data plane and control plane coexist, accommodating flow table entries and buffer space becomes nearly impossible. The controller issues instructions to the forwarding elements through the Southbound interface. However, the same characteristics that make SDN networks appealing also expose them to new security risks. One such risk is distributed denial of service (DDoS) attacks, which can have a catastrophic impact on an SDN network. If the network is inadequately protected, DDoS attacks can overwhelm the controller or OpenFlow switch by directing packets to various points. Numerous researchers have contributed to advancing strategies for detecting and mitigating DDoS attacks, employing diverse methodologies and frameworks [3]. Extensive literature exists that delves into research challenges and potential future directions for DDoS attack detection and mitigation in SDN [1], [4]–[7]. Several researchers have proposed innovative approaches to DDoS attack detection and mitigation within an SDN architecture. For instance, Ribeiro *et al.* [8] introduced a time-based moving target defense (MTD) technique utilizing machine learning (ML) sensors for real-time DDoS attack diagnosis and controller migration. Bhayo *et al.* [9] explored naive Bayes (NB), decision tree (DT), and support vector machine (SVM) algorithms, achieving remarkable accuracy in DDoS attack detection simulations. CoWatch [10] presented a framework for cooperative prediction and detection of DDoS attacks in edge computing (EC) scenarios, leveraging SDN's distributed architecture and employing the long short-term memory (LSTM) algorithm. The DOCUS model [11] was proposed for distinguishing genuine DDoS attacks from harmless flash traffic, reducing average detection time significantly in SDN-based networks. Anyanwu *et al.* [12] developed an intrusion detection system using the radial basis function kernel (RBF) of SVM for detecting DDoS attacks in vehicular ad hoc networks (VANETs) with high accuracy. Naula *et al.* [13] proposed a scalable SDN-based security framework employing deep reinforcement learning (DRL) for effective DDoS detection and mitigation. Numerous studies explore the application of deep learning techniques, such as recurrent neural networks (RNN), for DDoS detection [14], [15]. These methods find prevalence in data centers [16], [17] and cloud computing environments [18]. Long and Jinsong [19] proposed a dual approach, combining information entropy and a ML framework using an stacked sparse auto-encoder (SSAE)-SVM architecture for improved DDoS detection. A framework incorporating a composite multilayer perceptron and an effective feature extraction technique for detecting and identifying DDoS assaults was presented [20].

An anomaly tree proposed [21] is tailored to detect and trace the specific route taken by a DDoS attack within the SDN architecture, responding to observed fluctuations in network traffic. Cui *et al.* [22] illustrated a DDoS assault detection and defense system based on cognitive-inspired computing, which uses dual address entropy and suggests suitable defense and recovery procedures upon attack detection. Imran *et al.* [23] provides an extensive analysis of several mitigation strategies developed to combat malicious traffic in the SDN environment, systematically providing solutions based on how they manage such malicious traffic. According to Sahay *et al.* [24], an autonomic DDoS defense framework named analytics readiness and optimal maturity advancement (ArOMA), encompassing business monitoring, anomaly discovery, and mitigation, was proposed. ArOMA demonstrated the capability to facilitate collaborations between internet service providers (ISPs) and their guests in the realm of DDoS mitigation. Cui *et al.* [25] introduced a comprehensive system comprising four crucial modules-attack discovery detector, attack discovery, attack traceback, and attack mitigation. The existing literature only focuses on very few ML algorithms, while our work considers performance analysis of all ML algorithms and also uses synthetic minority over-sampling technique (SMOTE) technique for feature engineering and addressing class imbalance. Additionally, the paper implements a DDoS attack discovery system based on neural networks for direct attack identification and introduces an attack traceback system strategically leveraging the distinctive characteristics of SDN. In this work, the main focus is DDoS attack prediction in SDN environments. In this endeavor, this work makes contributions on various fronts:

Exploratory data analysis (EDA): a thorough inspection of the dataset consisting of a range of features pertinent to network traffic. Each feature is examined for its data type, range, distribution, and presence of missing or null values, and anomalous points.

- Data cleaning: depending on the nature and importance of the missing values, techniques such as imputation or exclusion are employed to address them.
- Statistical measures such as central tendency (mean, median) and dispersion (standard deviation, interquartile range) are calculated to provide a high-level understanding of the distribution and spread of the data. For instance, if the mean packet count is notably higher than the median, it suggests a skewed distribution, potentially due to the presence of outliers or heavy traffic episodes.

- Visualization is carried out using the following plots: i) boxplots to depict data distribution, aiding outlier detection, potentially indicating DDoS attacks in this study; ii) scatter plots to visualize relationships between continuous variables, hinting at correlated features, indicative of specific network behavior; and bar charts to display categorical data distributions, crucial for understanding the frequency of variables such as protocol types or port numbers.
- Given the typical class imbalance in DDoS datasets, the SMOTE was used to generate synthetic samples of the minority class. This balanced the dataset, thus preventing the models from being biased towards the majority class.
- We also utilized correlation analysis to identify linear relationships between features, while the random forest algorithm assessed feature importance, enabling a selection of the most relevant features for the models.
- A diverse suite of ML algorithms was selected to classify whether the traffic is normal or attacked. Algorithms such as gradient boosting and ensemble models: as light gradient boosting machine (LGBM) classifier, random forest classifier, XGB classifier, extra trees classifier, and AdaBoost classifier. DT: DT classifier and extra tree classifier. Proximity-based algorithms: K-neighbors classifier. SVM: support vector classifier (SVC). Stochastic gradient descent: SGD classifier. Linear models: logistic regression, linear discriminant analysis, ridge classifier CV, and ridge classifier. Bayesian methods: Gaussian NB. Other classifiers: perceptron, nearest centroid, passive aggressive classifier, quadratic discriminant analysis, and dummy classifier.
- The key findings of the research are to tune each model meticulously and validate using techniques such as cross-validation to mitigate overfitting and ensure the robustness of the predictions.

The remaining sections of the paper cover the following topics: section 2 elaborates on the datasets utilized, sourced from various databases. Section 3 outlines the schematic diagram detailing the proposed methodology for predicting DDoS attacks in the SDN environment. Section 4 delves into significant performance measures, results, and comparisons with state-of-the-art approaches. Lastly, section 5 concludes the paper and outlines directions for future research.

2. METHOD

2.1. Input dataset

For the DDoS attack detection, at first, SDN dataset is collected from [26] and preprocessed. The dataset contains 104346 rows and 23 columns, which are a mix of features and labels, a substantial and valuable resource for researching and developing DDoS detection methods. These columns could be divided into:

- Features: these are the attributes used to describe network traffic patterns and characteristics. These features help ML models learn to distinguish between normal and DDoS traffic. Some common features for DDoS detection might include packet sizes, packet rates, byte rates, source IP addresses, destination IP addresses, and various statistical metrics related to network traffic.
- Labels: this column indicates whether each network traffic recordf represents normal traffic or a DDoS attack. The label column is essential for supervised learning, where models learn from labeled data as shown in Figure 1.
- Data split: when using this dataset for ML, it's common to split it into training, validation, and test sets to evaluate model performance properly. Here 75% is used for training and 25% of the dataset is used for testing and validation.

	dt	switch	src	dst	pktscount	bytecount	dur	dur_nsec	tot_dur	flows	...	pktrate	Pairflow	Protocol	port_no	tx_bytes	rx_bytes	tx_kbps	rx_kbps	tot_kbps	label
0	11425	1	10.0.0.1	10.0.0.8	45304.00	48294064.00	100.00	716000000.00	101000000000.00	3.00	...	451.00	0.00	UDP	3.00	143928631.00	3917.00	0.00	0.00	0.00	0.00
1	11605	1	10.0.0.1	10.0.0.8	126395.00	134737070.00	280.00	734000000.00	281000000000.00	2.00	...	451.00	0.00	UDP	4.00	3842.00	3520.00	0.00	0.00	0.00	0.00
2	11425	1	10.0.0.2	10.0.0.8	90333.00	96294978.00	200.00	744000000.00	201000000000.00	3.00	...	451.00	0.00	UDP	1.00	3795.00	1242.00	0.00	0.00	0.00	0.00
3	11425	1	10.0.0.2	10.0.0.8	90333.00	96294978.00	200.00	744000000.00	201000000000.00	3.00	...	451.00	0.00	UDP	2.00	3688.00	1492.00	0.00	0.00	0.00	0.00
4	11425	1	10.0.0.2	10.0.0.8	90333.00	96294978.00	200.00	744000000.00	201000000000.00	3.00	...	451.00	0.00	UDP	3.00	3413.00	3665.00	0.00	0.00	0.00	0.00
...
8710	10056	3	10.0.0.10	10.0.0.7	54209.00	56485778.00	173.00	363000000.00	173000000000.00	4.00	...	307.00	0.00	UDP	1.00	3719.00	1156.00	0.00	0.00	0.00	1.00
8711	10056	3	10.0.0.10	10.0.0.7	54209.00	56485778.00	173.00	363000000.00	173000000000.00	4.00	...	307.00	0.00	UDP	4.00	307807519.00	4253.00	7259.00	0.00	7259.00	1.00
8712	10056	3	10.0.0.10	10.0.0.7	54209.00	56485778.00	173.00	363000000.00	173000000000.00	4.00	...	307.00	0.00	UDP	1.00	4139.00	200572874.00	0.00	3421.00	3421.00	1.00
8713	10056	3	10.0.0.10	10.0.0.7	54209.00	56485778.00	173.00	363000000.00	173000000000.00	4.00	...	307.00	0.00	UDP	3.00	3413.00	3539.00	0.00	0.00	0.00	1.00

Figure 1. Dataset with various features having labeled information

2.2. Framework

This study delineates a comprehensive approach to predict SDN-DDoS attacks by employing a single phased methodological framework that integrates various ML algorithms into lazy classifiers. The methodology aims to leverage the strengths of each algorithmic category to enhance predictive accuracy and gain deeper insights into the characteristics of network traffic data that could potentially signal the presence of DDoS attacks. Figure 2 depicts the overall stages in the proposed work. The subsequent sections provide concise descriptions of the elements constituting the models.



Figure 2. A schematic diagram of the methodology

2.2.1. Exploratory data analysis

The statistical summary of the dataset is shown in Tables 1 and 2. The dataset comprises 8,714 observations across various variables, shedding light on network traffic dynamics. The 'switch' variable, likely representing network switch identifiers, has a mean value of 11,089.83, suggesting potential encoding or scaling, as switch identifiers are typically categorical.

Table 1. Dataset

	pktcount	bytecount	dur	dur_nsec	tot_dur	packetins	pktperflow
count	14.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00
mean	7.47E+04	7.91E+07	194.01	5.48E+08	1.95E+11	1,920.42	1.05E+04
std	4.07E+04	4.31E+07	117.35	2.32E+08	1.17E+11	254.13	4.05E+03
min	284.00	3.03E+05	-	7.90E+07	8.37E+08	558.00	0.00E+00
25%	3.73E+04	3.96E+07	100.00	3.91E+08	1.01E+11	1,931.00	8.64E+03
50%	7.66E+04	8.17E+07	190.00	5.56E+08	1.91E+11	1,943.00	1.34E+04
75%	1.13E+05	1.20E+08	280.00	7.26E+08	2.81E+11	1,943.00	1.35E+04
max	1.35E+05	1.44E+08	473.00	9.14E+08	4.73E+11	2,242.00	1.37E+04

Table 2. Feature selection in dataset

	byteperflow	pktrate	tx_bytes	rx_bytes	tx_kbps	rx_kbps	tot_kbps
count	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00
mean	1.11E+07	349.86	4.82E+07	4.81E+07	873.43	873.11	1,746.54
std	4.35E+06	134.91	1.51E+08	1.10E+08	2,848.97	2,269.07	3,426.36
min	0.00E+00	-	2.85E+03	9.26E+02	-	-	-
25%	9.21E+06	288.00	3.59E+03	1.47E+03	-	-	-
50%	1.43E+07	446.00	3.84E+03	3.54E+03	-	-	-
75%	1.44E+07	451.00	4.25E+03	6.16E+06	-	-	2.57E+03
max	1.46E+07	456.00	1.27E+09	9.91E+08	2.06E+04	1.66E+04	2.06E+04

'Pktcount' and 'bytecount' signify packet and byte counts per network event, with 'pktcount' averaging at 2.32, indicating consistency across observations. Meanwhile, 'bytecount' averages 79,060,776.54 bytes, reflecting varying event sizes. Duration variables 'dur', 'dur_nsec', and 'tot_dur' depict event lengths in seconds, nanoseconds, and total duration, with mean values indicating a wide range of event durations. 'Flows' average 3.32 per event, indicating simultaneous communication sessions, with consistent flow numbers. 'Packetins', possibly indicating incoming packets, averages 1,920.42, hinting at variability in incoming packet frequency. 'Pktperflow' and 'byteperflow' show packet and byte counts per flow, with diverse averages and high standard deviations, crucial for traffic profiling. 'Pktrate' averages 349.86, with a significant standard deviation, suggesting a wide range in packet transmission rates, potentially reflecting idle periods or keep-alive connections.

2.2.2. Visualization

The following visualizations were observed here. The chart is a pairs plot matrix that visualizes the relationships between several categorical variables: 'switch', 'flows', 'pairflow', and 'port_no', along with their distribution with respect to two labels (0 and 1, which could represent different classes such as 'normal' and 'attack' in the context of network traffic). Looking at the diagonal, we see density plots for each variable,

colored differently for each label, indicating the distribution of each variable within the two classes. It appears that the variables have discrete values which might represent categorical or binned numerical data. For instance, 'switch' shows a bimodal distribution for one label and a more uniform distribution for the other, suggesting that certain switch values are more prevalent in one class than in the other.

In the off-diagonal plots, which show the relationship between two variables, we see distinct groupings of points. These scatter plots reveal how the categories of each variable are distributed concerning each other. For example, the 'switch' vs. 'flows' plot shows that certain switches are associated with specific flow counts, and this association differs by label, which could be indicative of certain patterns of network behavior that are characteristic of normal versus attack traffic. The plots with 'pairflow' against other variables exhibit a relatively tight clustering of points at lower values of 'pairflow', especially for one label, perhaps indicating that low 'pairflow' values are strongly associated with a particular class as depicted in Figure 3.

Finally, the 'port_no' variable shows multiple peaks in distribution, suggesting that there are common ports that are frequently used in both classes, but there's a visible distinction in the distribution of 'port_no' between the labels. The 'switch' value count indicates the number of network events or connections handled by each switch identifier. With switch 3 handling the most events (4,077), followed by switch 2 (3,358), and switch 1 (1,280), there's a clear indication that some switches are more heavily utilized than others. This could be due to the network topology, where certain switches serve as major hubs for traffic. The differences in counts might also reflect the strategic positioning of switches within the network, or varying levels of protection against DDoS attacks, assuming the switches are part of an SDN environment being monitored for security purposes. The 'flows' count provides insights into how many concurrent network flows are common within the events captured. The majority of events have between 2 and 4 flows, with 2.00 flows being the most frequent (2,569 times). This suggests that dual-flow communication is a common pattern, possibly indicative of bidirectional traffic between hosts. The presence of a single not a number (NaN) value indicates an event where the flow count was not recorded or is missing, which may require additional data cleaning or imputation.

The 'pairflow' count, with 0.00 occurring 8,714 times, suggests that all recorded events are of one type, potentially representing unpaired flows. This could signify unidirectional communication or non-responsive traffic, which might be characteristic of certain types of network behavior or applications. The single NaN value indicates one instance of missing data, which, depending on the context, might be negligible or could warrant further investigation to ensure data integrity. For the 'protocol' variable, every single counted event (8,714 in total) used the user datagram protocol (UDP) protocol. This uniformity implies that the dataset might be filtered to capture only UDP traffic, or that UDP is overwhelmingly the protocol of choice in the network environment under study. UDP's connectionless nature makes it a frequent choice for streaming, gaming, or certain types of DDoS attacks. The single NaN value suggests an instance where protocol information is missing.

The 'port_no' variable count indicates the number of network events associated with different port numbers. Ports 1 and 2 are nearly equally represented (2,397 and 2,392 events, respectively), followed by port 3 (2,180) and port 4 (1,745). This distribution could reflect the usage of these ports for different services or applications within the network. The singular NaN value indicates one event where the port number was not recorded, which may need to be addressed during data preprocessing. Overall, this pairs plot matrix is a powerful exploratory tool that allows us to see the relationships between different pairs of variables and how these relationships may differ by class. This can provide valuable insights for further analysis, such as feature selection or anomaly detection in the context of network security as depicted in Figure 3.

Data preprocessing played a critical role in maintaining dataset integrity by addressing missing or null values. This ensured that the predictive models received clean and complete data, a crucial factor for the reliability of subsequent analyses. One-hot encoding was applied to handle categorical variables. This transformation converted categorical data into a numerical format, facilitating better understanding and evaluation of input features by ML algorithms.

Feature engineering, a pivotal step, involved applying domain knowledge to create new features aimed at enhancing model performance. Techniques such as correlation analysis identified linear relationships between features, while the random forest algorithm assessed feature importance. This process allowed for the selection of the most relevant features, reducing dimensionality, and directing the models towards the most predictive attributes. To address class imbalance inherent in DDoS datasets, the SMOTE was employed. This technique generated synthetic samples of the minority class, effectively balancing the dataset and preventing bias towards the majority class. A diverse array of ML algorithms was selected for the subsequent phase, aiming to provide a comprehensive perspective on the dataset. The selection encompassed various algorithms, reflecting different approaches and methodologies in order to capture the complexity and nuances of the data. In summary, this comprehensive approach, from exploratory analysis through preprocessing and feature engineering to the selection of diverse ML models, was undertaken to ensure a robust and well-informed

foundation for subsequent phases of analysis and model development. Each step was strategically chosen to address specific challenges and enhance the overall reliability and predictive capability of the models.

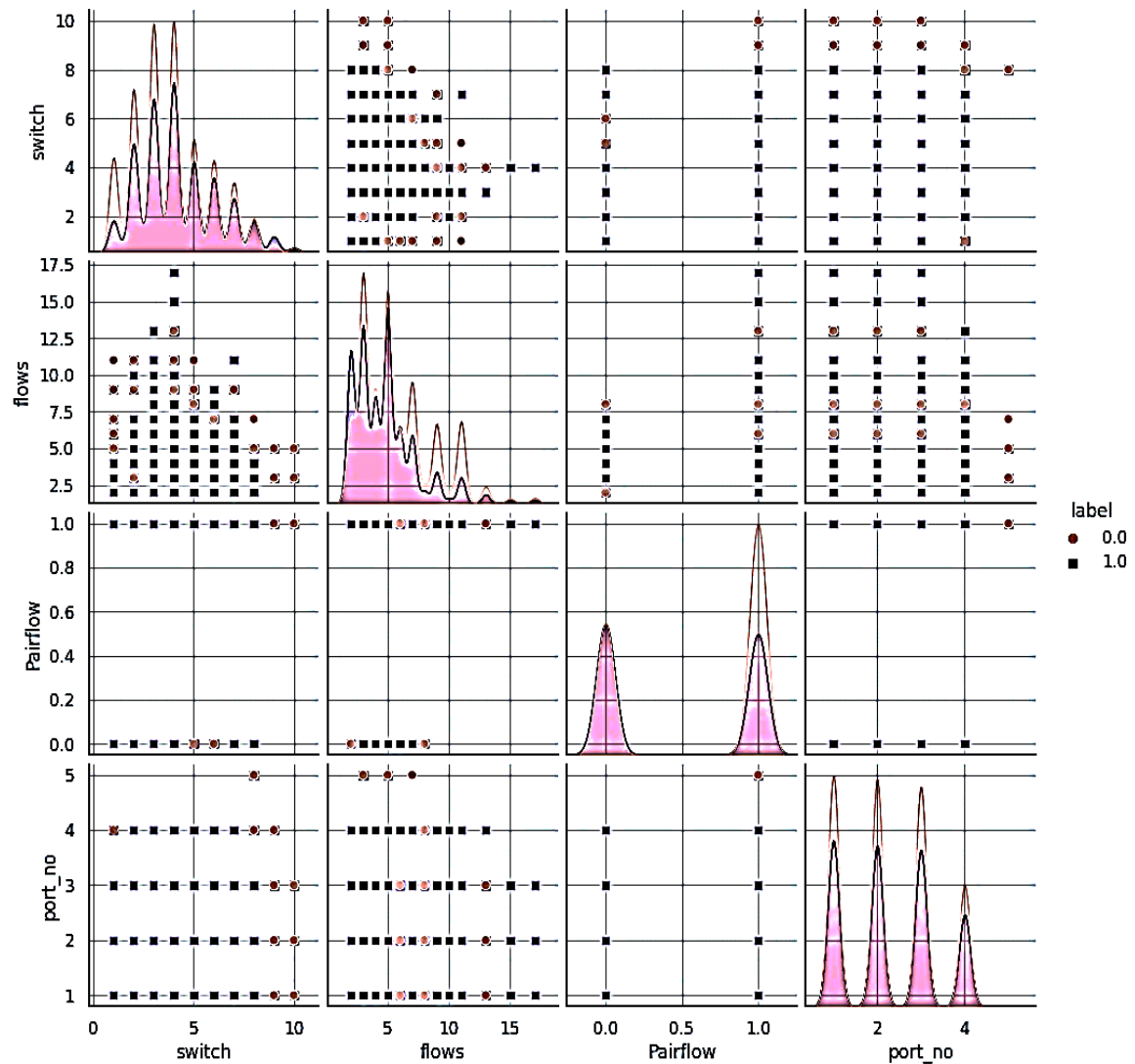


Figure 3. Plot matrix visualizing the relationships between several categorical variables: 'switch', 'flows', 'pairflow', and 'port_no', along with their distribution with respect to normal traffic and attack traffic

2.2.3. Model development

Various ensemble methods, including LGBM classifier, random forest classifier, XGB classifier, DT classifier, and extra trees classifier, as well as popular ML classifiers like AdaBoost classifier, K-neighbors classifier, and SVC, are being explored for training models to detect DDoS attacks. The dataset is divided into 75% for training and 25% for testing and validation. Ensemble techniques, pivotal in ML, involve gradient boosting, which elevates weak learners into robust prediction models.

The LGBM classifier, abbreviated from light gradient boosting machine classifier, utilizes DT algorithms for classification and ranking. It incorporates gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB) techniques to enhance efficiency with large-scale data, boosting speed and reducing memory usage. Random forest, another ensemble method, excels in regression and classification tasks by clustering DT into an ensemble, combating overfitting and lack of resilience. XGBoost, or extreme gradient boosting, is renowned for its high performance and strategies to mitigate overfitting, especially adept at handling large datasets and parallel processing, contributing to its wide adoption in ML. Extra trees classifier, akin to random forest, creates multiple trees and employs random feature subsets for node splitting. However, it diverges by forgoing bootstrapping and opting for random splits, emphasizing its unique approach of

introducing randomness into the tree-building process. AdaBoost collaborates multiple weak classifiers to form a robust ensemble, adjusting their roles incrementally to improve accuracy, particularly in challenging scenarios. DT classifiers extract decision-making insights from input data, serving as foundational tools in ML.

The K-neighbors classifier, a part of the k-nearest neighbors (KNN) family, predicts based on the majority class of its k-nearest neighbors in the feature set, offering flexibility in instance-based learning. Support vector machines (SVM) construct models to distinguish between different class instances, assuming linear separability. The stochastic gradient descent (SGD) classifier is a linear classifier optimized using SGD, suitable for scenarios with substantial data sizes. Logistic regression predicts binary outcomes using continuous independent variables, while linear discriminant analysis identifies linear feature combinations to effectively distinguish classes. The ridge classifier, tailored for multi-class classification tasks, incorporates concepts from traditional classification methods and ridge regression to minimize overfitting risks. Gaussian NB excels in classification tasks with normally distributed, continuous data, assuming conditional independence among features within each class. Quadratic discriminant analysis allows distinct covariance matrices for each class, enabling a flexible capture of non-linear decision boundaries. A dummy classifier serves as a basic baseline model, contrasting with more complex algorithms and establishing a fundamental reference point for performance evaluation.

2.2.4. Performance metrics

The evaluation of models involved the use of diverse metrics appropriate for classification tasks, encompassing accuracy, balanced accuracy, receiver operating characteristic (ROC)-area under the curve (AUC), and F1-score. These metrics collectively offered a comprehensive assessment of the performance of each model. Accuracy is determined as the ratio of correct predictions to the total number of predictions. It can be computed using (1). A metric commonly used in ML to assess the performance of a classification model, particularly in scenarios with imbalanced class distributions. Balanced accuracy addresses this issue by computing the average of sensitivity (true positive rate) across all classes as depicted in (2). The F1-score, depicted in (3) represents the harmonic average of recall and precision, taking into account both false positives and false negatives. This makes it particularly effective when dealing with imbalanced datasets.

$$Accuracy = \frac{(TP+TN)}{(TP+FP+TN+FN)} \quad (1)$$

$$Balanced\ Accuracy = \frac{sensitivity_1 + sensitivity_2 + \dots + sensitivity_k}{k} \quad (2)$$

$$F_1 = 2 \times \frac{(precision \times recall)}{(precision + recall)} \quad (3)$$

The AUC is a measure calculated from the ROC curve, offering a singular scalar value that encapsulates the performance of the classifier across various thresholds. An ideal classifier achieves an AUC of 1.0, whereas a random classifier yields an AUC of 0.5. Efficient models with brief training times are desirable but selecting them should not compromise predictive performance. This plot is essential for assessing the trade-offs between model accuracy and the computational resources needed for both training and inference.

3. RESULTS AND DISCUSSION

The assessment of ML models for DDoS attack detection in SDN reveals diverse performance characteristics. Table 3 depicts the comprehensive performance evaluation of various ML algorithms in the detection of DDoS Attacks. The LGBM classifier, DT classifier, random forest classifier, XGB classifier, extra trees classifier has achieved perfect scores across all metrics, including accuracy, balanced accuracy, ROC-AUC, and F1 score, all at 1.00 (100%). This suggests that the model has likely overfit to the training data, especially in the context of a complex problem like predicting DDoS attacks in SDN. The computation time of 1.68, 0.52, 6.38, 1.7, and 3.41 seconds respectively indicates high efficiency in model training, highlights simplicity, speed, and extremely fast training. Despite efficient training times, these models may not reflect real-world performance accurately. Summary: however, decision tree classifiers give the best performance with a fast computation time of 0.52 seconds.

Models such as extra tree classifier, AdaBoost classifier and K-neighbors classifier achieve commendable accuracy of 0.99 (99%) with varying computation times, such as 3.41, 0.18, and 4.66 seconds respectively, showcasing their efficacy and scalability. In contrast, models like SVC and SGD classifier present slightly lower scores, indicating a focus on generalization over memorization. Logistic regression strikes a balance between simplicity and efficiency, demonstrating moderate accuracy as shown in Table 3. Summary:

however, extratree classifiers give the best performance of 99% accuracy with a fast computation time of 0.18 seconds.

Linear discriminant analysis, ridge classifier CV, and ridge classifier exhibit lower accuracy around 72.33% suggesting potential limitations in capturing complex patterns. Gaussian NB, perceptron, nearest centroid, passive aggressive classifier, and quadratic discriminant analysis show varying degrees of accuracy of 63%, each with implications for their suitability in DDoS detection scenarios. The dummy classifier serves as a baseline with a minimal accuracy of 57%, reflecting its simplistic prediction approach based on simple rules. This model provides a benchmark for evaluating the performance of more sophisticated algorithms. The swift computation time of the dummy classifier aligns with its inherent simplicity. Each model's unique strengths and weaknesses underscore the critical importance of selecting an appropriate algorithm tailored to specific task requirements and dataset characteristics. Striking a balance between accuracy, efficiency, and generalization is crucial when implementing ML models for DDoS detection in the dynamic landscape of SDN. Further exploration, including techniques like cross-validation and testing on unseen data, is imperative to ensure the reliability and practical applicability of these models in real-world cybersecurity scenarios.

From the Table 3 it is also observed that based on accuracy and time taken for computation shows that DT is the best (100% - 0.52 seconds), extra tree classifier (99% - 0.18 seconds) and Gaussian NB - low accuracy with low computation time of 0.12 seconds. The bar plot for accuracy as shown in Figure 4, likely shows that models such as LGBM classifier, DT classifier, random forest classifier, and XGB classifier are performing with high accuracy, potentially reaching 1.00. This can be indicative of very effective models, but in real-world scenarios, such perfection often flags a need for scrutiny, possibly hinting at overfitting.

Table 3. Performance comparison of ML algorithms in detection of DDoS attacks

Model	Accuracy	Balanced accuracy	ROC-AUC	F1-score	Time taken
LGBM classifier	1	1	1	1	1.68
DT classifier	1	1	1	1	0.52
Random forest classifier	1	1	1	1	6.38
XGB classifier	1	1	1	1	1.7
ExtraTrees classifier	1	1	1	1	3.41
ExtraTree classifier	0.99	0.99	0.99	0.99	0.18
AdaBoost classifier	0.99	0.99	0.99	0.99	4.66
KNeighbors Classifier	0.99	0.99	0.99	0.99	5.49
SVC	0.95	0.96	0.96	0.96	58.14
SGD classifier	0.81	0.81	0.81	0.81	0.51
Logistic regression	0.79	0.78	0.78	0.79	0.84
Linear discriminant analysis	0.73	0.73	0.73	0.73	0.32
Ridge classifier CV	0.72	0.72	0.72	0.72	0.3
Ridge classifier	0.72	0.72	0.72	0.72	0.15
Gaussian NB	0.68	0.69	0.69	0.68	0.12
Perceptron	0.66	0.68	0.68	0.66	0.31
Nearest centroid	0.65	0.65	0.65	0.65	0.17
Passive aggressive classifier	0.58	0.61	0.61	0.57	0.45
Quadratic discriminant analysis	0.58	0.51	0.51	0.44	0.28
Dummy classifier	0.57	0.5	0.5	0.41	0.12

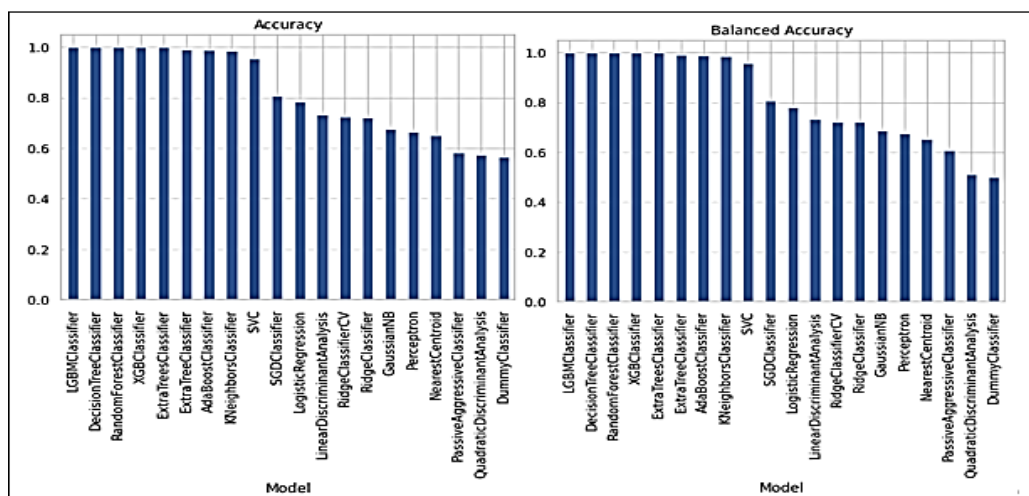


Figure 4. Performance comparison of various models with respect to the accuracy and balanced accuracy

Models with slightly less accuracy, such as AdaBoost classifier and K-neighbors classifier, may offer more realistic predictive performances and potentially better generalization to unseen data. Balanced accuracy as depicted in Figure 4, is crucial in datasets with imbalanced classes. A bar plot illustrating high balanced accuracy for the same top-performing models suggests that these models are effective across both majority and minority classes. However, the same caution regarding overfitting applies here. For models with lower balanced accuracy, it may indicate that these models struggle more with class imbalance, affecting their performance in a real-world setting. The ROC AUC bar in Figure 5 likely mirrors accuracy and balanced accuracy trends.

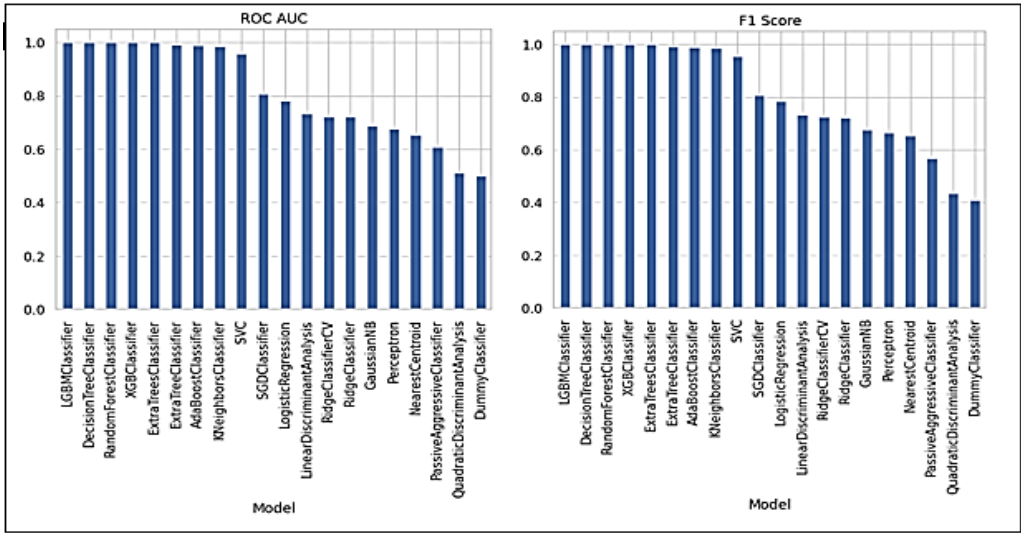


Figure 5. Performance comparison of various models with respect to the ROC-AUC and F1 score

High ROC AUC values near 1.00 suggest effective class distinction, though excessive values may imply overfitting. Lower values indicate less discriminative power, improvable with feature engineering or model tuning. The F1 Score bar combines precision and recall, revealing a model's true-positive identification balance. High F1 scores are crucial for DDoS detection, while lower scores require precision-recall balance refinement. The time taken barplot, as depicted in Figure 6, likely shows significant variation across models. Models like SVC that take longer to train may not be suitable for environments where quick retraining is necessary. Conversely, models with short training times are desirable for their efficiency but should not be chosen at the cost of predictive performance. This plot is crucial for understanding the trade-offs between model accuracy and the computational resources required for model training and inference.

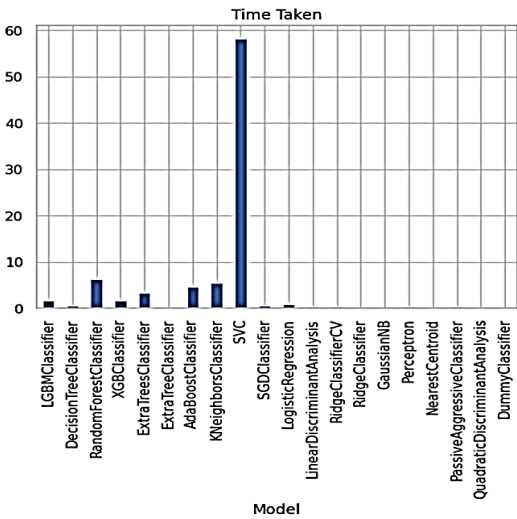


Figure 6. Performance comparison of various models with respect to the computation time

As it can be seen from Table 4, the proposed work performs better than most of the state-of-the-art methods. The present study has provided best results in terms of accuracy, balanced accuracy, ROC-AUC and F1 score by using SMOTE for feature engineering and all popular ensembling techniques available. This performance spectrum emphasizes the challenge of accurately predicting DDoS attacks and underscores the importance of selecting a suitable model aligned with the specific characteristics of cybersecurity data. The varying outcomes stress the significance of comprehending the underlying data and problem context when choosing and fine-tuning ML algorithms for cybersecurity applications.

Table 4. Comparison of proposed work with state of art

Author (year)	Dataset used	Methodology		Performance metrics		
		No. of features considered	Classifiers	Accuracy (%)	ROC-AUC (%)	F1-score (%)
Alghoson. and Abbas (2021) [3]	CICDDoS 2019 dataset	20	RF, LGB, Catboost, CNN	99	unknown	unknown
Long and Jinsong (2022) [19]	DARPAR dataset	18	SSAE-SVM	98	unknown	unknown
Polat <i>et al.</i> [14]	SCADA dataset	10	RNN, LSTM, and GRU	97.62	unknown	unknown
Vadhil <i>et al.</i> [6]	CICIDS-2017)	variable number of features	DT, RF, LR, GNB, AB & Ensemble	99.50	unknown	99.44
Bhayo <i>et al.</i> (2023) [9]	Simulated traffic	05	NB, SVM, and DT	98.1	unknown	unknown
Present study	[27]	16	Ensemble techniques, ML with SMOTE	100	100	100

4. CONCLUSION

In this work, we applied various widely used ML techniques to identify DDoS attacks within SDN. Our study extensively evaluates the performance of these popular ML techniques, distinguishing itself from existing works that focus on only a subset of these methods. Notably, we observed that although certain models exhibit outstanding performance across multiple metrics, consistent high scores may indicate potential overfitting, particularly in intricate domains like cybersecurity. The ML models, including LGBM classifier, DT classifier, random forest classifier, XGB classifier, extra trees classifier, AdaBoost classifier, and K-neighbors classifier, demonstrated noteworthy effectiveness in various aspects of DDoS detection and mitigation. To ensure practical applicability, it is crucial to subject these models to further scrutiny through techniques such as cross-validation and testing on unseen data. Future endeavors should explore enhancing models with lower performance using advanced feature selection, hyperparameter tuning, or ensemble methods. Implementing ML models for DDoS detection and mitigation transcends being an option; it stands as a necessity in the contemporary cybersecurity landscape.




REFERENCES

- [1] B. Alhijawi, S. Almajali, H. Elgala, H. B. Salameh, and M. Ayyash, "A survey on DoS/DDoS mitigation techniques in SDNs: classification, comparison, solutions, testing tools and datasets," *Computers and Electrical Engineering*, vol. 99, Apr. 2022, doi: 10.1016/j.compeleceng.2022.107706.
- [2] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments," *Computer Networks*, vol. 62, pp. 122–136, 2014, doi: 10.1016/j.bjp.2013.10.014.
- [3] E. S. Alghoson and O. Abbass, "Detecting distributed denial of service attacks using machine learning models," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 12, pp. 616–622, 2021, doi: 10.14569/IJACSA.2021.0121277.
- [4] Y. Cui *et al.*, "Towards DDoS detection mechanisms in software-defined networking," *Journal of Network and Computer Applications*, vol. 190, 2021, doi: 10.1016/j.jnca.2021.103156.
- [5] J. Singh and S. Behal, "Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions," *Computer Science Review*, vol. 37, 2020, doi: 10.1016/j.cosrev.2020.100279.
- [6] F. A. Vadhil, M. L. Salihi, and M. F. Nanne, "Machine learning-based intrusion detection system for detecting web attacks," *IAES International Journal of Artificial Intelligence*, vol. 13, no. 1, pp. 711–721, 2024, doi: 10.11591/ijai.v13.i1.pp711-721.
- [7] B. Wang, Y. Zheng, W. Lou, and Y. T. Hou, "DDoS attack protection in the era of cloud computing and software-defined networking," *Computer Networks*, vol. 81, pp. 308–319, 2015, doi: 10.1016/j.comnet.2015.02.026.
- [8] M. A. Ribeiro, M. S. P. Fonseca, and J. D. Santi, "Detecting and mitigating DDoS attacks with moving target defense approach based on automated flow classification in SDN networks," *Computers and Security*, vol. 134, 2023, doi: 10.1016/j.cose.2023.103462.
- [9] J. Bhayo, S. A. Shah, S. Hameed, A. Ahmed, J. Nasir, and D. Draheim, "Towards a machine learning-based framework for DDOS attack detection in software-defined IoT (SD-IoT) networks," *Engineering Applications of Artificial Intelligence*, vol. 123, 2023, doi: 10.1016/j.engappai.2023.106432.
- [10] H. Zhou, Y. Zheng, X. Jia, and J. Shu, "Collaborative prediction and detection of DDoS attacks in edge computing: A deep learning-based approach with distributed SDN," *Computer Networks*, vol. 225, 2023, doi: 10.1016/j.comnet.2023.109642.




- [11] P. V. Shalini, V. Radha, and S. G. Sanjeevi, "DOCUS-DDoS detection in SDN using modified CUSUM with flash traffic discrimination and mitigation," *Computer Networks*, vol. 217, 2022, doi: 10.1016/j.comnet.2022.109361.
- [12] G. O. Anyanwu, C. I. Nwakanma, J. M. Lee, and D. S. Kim, "RBF-SVM kernel-based model for detecting DDoS attacks in SDN integrated vehicular network," *Ad Hoc Networks*, vol. 140, 2023, doi: 10.1016/j.adhoc.2022.103026.
- [13] N. M. Y. -Naula, C. V. -Rosales, J. A. P. -Díaz, and D. F. Carrera, "A flexible SDN-based framework for slow-rate DDoS attack mitigation by using deep reinforcement learning," *Journal of Network and Computer Applications*, vol. 205, 2022, doi: 10.1016/j.jnca.2022.103444.
- [14] H. Polat, M. Türkoğlu, O. Polat, and A. Şengür, "A novel approach for accurate detection of the DDoS attacks in SDN-based SCADA systems based on deep recurrent neural networks," *Expert Systems with Applications*, vol. 197, 2022, doi: 10.1016/j.eswa.2022.116748.
- [15] V. R. Karna and K. V. V. Reddy, "1-dimensional convolutional neural networks for predicting sudden cardiac," *IAES International Journal of Artificial Intelligence*, vol. 13, no. 1, pp. 984–993, 2024, doi: 10.11591/ijai.v13.i1.pp984-993.
- [16] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. P. C. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics," *Future Generation Computer Systems*, vol. 89, pp. 685–697, 2018, doi: 10.1016/j.future.2018.07.017.
- [17] P. Xiao, W. Qu, H. Qi, and Z. Li, "Detecting DDoS attacks against data center with correlation analysis," *Computer Communications*, vol. 67, pp. 66–74, 2015, doi: 10.1016/j.comcom.2015.06.012.
- [18] O. Osanaiye, K. K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147–165, 2016, doi: 10.1016/j.jnca.2016.01.001.
- [19] Z. Long and W. Jinsong, "A hybrid method of entropy and SSAE-SVM based DDoS detection and mitigation mechanism in SDN," *Computers and Security*, vol. 115, 2022, doi: 10.1016/j.cose.2022.102604.
- [20] G. C. Amaizu, C. I. Nwakanma, S. Bhardwaj, J. M. Lee, and D. S. Kim, "Composite and efficient DDoS attack detection framework for B5G networks," *Computer Networks*, vol. 188, 2021, doi: 10.1016/j.comnet.2021.107871.
- [21] W. Chen, S. Xiao, L. Liu, X. Jiang, and Z. Tang, "A DDoS attacks traceback scheme for SDN-based smart city," *Computers and Electrical Engineering*, vol. 81, 2020, doi: 10.1016/j.compeleceng.2019.106503.
- [22] J. Cui, M. Wang, Y. Luo, and H. Zhong, "DDoS detection and defense mechanism based on cognitive-inspired computing in SDN," *Future Generation Computer Systems*, vol. 97, pp. 275–283, 2019, doi: 10.1016/j.future.2019.02.037.
- [23] M. Imran, M. H. Durad, F. A. Khan, and A. Derhab, "Toward an optimal solution against denial of service attacks in software defined networks," *Future Generation Computer Systems*, vol. 92, pp. 444–453, 2019, doi: 10.1016/j.future.2018.09.022.
- [24] R. Sahay, G. Blanc, Z. Zhang, and H. Debar, "ArOMA: An SDN based autonomic DDoS mitigation framework," *Computers and Security*, vol. 70, pp. 482–499, 2017, doi: 10.1016/j.cose.2017.07.008.
- [25] Y. Cui *et al.*, "SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks," *Journal of Network and Computer Applications*, vol. 68, pp. 65–79, 2016, doi: 10.1016/j.jnca.2016.04.005.
- [26] N. Ahuja, G. Singal, and D. Mukhopadhyay, "DDoS attack SDN dataset," *Mendeley Data*, V1, 2020, doi: 10.17632/jxpfjc64kr.1.

BIOGRAPHIES OF AUTHORS



Neethu S.    received the B.Tech. degree in Electronics and Communication Engineering from Calicut University, India, in 2009, and the M.Tech. degree in Embedded Systems from Amrita Vishwa Vidyapeetham University, India, in 2012. She is currently pursuing the Ph.D. degree at R. V. College of Engineering under Visvesvaraya Technological University, India. She is working as Assistant Professor at R. V. College of Engineering. Her research interests include network security, software defined networking, machine learning, and wireless communication. She can be contacted at email: neethus@rvce.edu.in.



Dr. H. V. Ravish Aradhya    is currently Head of the Department of Electronics and Communication Engineering at R. V. College of Engineering. He has academic and research experience of over 30 years. He has guided over 80 projects for undergraduate and postgraduate students. He has guided 2 Ph.D. scholars and is currently guiding 4 other scholars. He has over 47 Journal publications and 75 internal conference publications and good number of citations serving the research community; Scopus (400+), WoS (90+), Semantic Scholar (300+), and Google (650+) citations. He holds a B.E. (Electronics) and M.E. (Electronics) from Bangalore University and Ph.D. (Engineering) from VTU Belagavi. He can be contacted at email: ravisharadhya@rvce.edu.in.