International Research Journal of Multidisciplinary Scope (IRJMS), 2026;7(3):1797-1809 IRJMS

Original Article | ISSN (O): 2582-631X DOI:10.47857/irjms.2026.v07i03.010756

# Exploring Rapid Detection of DDoS Attack in SDN Using Machine Learning Algorithms With the Classification Technique

Sudhir Bhagat[1]*, Himanshu Gupta[1], Ashish Seth[2]

[1]Amity Institute of Information Technology, Amity University, Noida, Uttar Pradesh, India, [2]Center for Research & Development, Sri Eshwar College of Engineering, Coimbatore, Tamil Nadu, India. *Corresponding Author's Email: : research.sb20@gmail.com

## Abstract

An emerging technology called software-defined networking (SDN) brings programmability and centralized control to future network design. Operators can design monitor and manage the network, as well as identify malicious traffic and failures thanks to the programmable console architecture. Notwithstanding these benefits, the SDN control plane is still susceptible to security risks, particularly distributed denial-of-service (DDoS) attacks, which have the potential to bring down the entire network. This study thoroughly assesses DDoS detection techniques for SDN controllers, focusing on dataset-based analysis instead of real-time experiments that could be impacted by hardware anomalies. To guarantee reproducibility and transparency, the KNIME analytics platform with a visual workflow environment was utilized. To differentiate between malicious floods and legitimate surges, flow-level features like packet volume byte volume and flow duration were investigated. XGBoost outperformed the other seven tested algorithms with an accuracy of 99.20% and a latency of 8.2 ms. These findings demonstrate that sophisticated machine learning (ML) models, when paired with SDN-specific feature analysis, provide precise, scalable, and efficient DDoS detection solutions. Random Forest (RF) came in second with 98.40% accuracy, while the Decision Tree excelled in efficiency, processing flows in 4.1 ms. Overall, the results show that KNIMEs no-code workflow framework can offer production-level DDoS detection comparable to conventional code-based techniques, providing a workable and extremely accurate way to safeguard programmable networks.

**Keywords:** DDoS, KNIME, Machine Learning, Software Defined Network.

## Introduction

SDN isolates controllers from forwarding and switches depend on remote software controllers (1). Topology is influenced by hosts switches and controllers. Centralized control facilitates dynamic management high-bandwidth applications and customized security while enabling scalability load balancing and programmability. With the expansion of the SDN, defending against attack is important. Centralized control enables detection, scalability and load balancing. Controllers monitor traffic, identify threats and respond using machine learning (2), taking advantage of the separation of the control and forwarding planes. A centralized SDN management panel facilitates traffic control without having to configure separate switches, as the controllers manage the device regardless of the physical connection (3). Large-scale attacks pose a threat to infrastructure; Distributed Denial-of-Service (DDoS) attacks, first observed in 1998, flood networks with malicious traffic, thereby blocking true access. Despite strong security, major search engines and e-commerce sites fell victim to these attacks in year 2000 (4). With attacks reaching 500-800 Gbps in year 2015-2016, blocking DDoS remains essential for reliable data and service access (5). DDoS attacks remain a major threat. It's easy to start, but hard to stop. These attacks misuse resources by flooding the network with malicious traffic and preventing access to real users; Also, these become successful by simultaneous attacks from multiple hosts. These are categorized as network-level or application-level attacks and disrupt connectivity or server resources (6). The number of such attacks more than doubled to 47.1 million in year 2025, a 236% increase compared to year 2023 (7). Even large servers can be blocked by targeting a simple webpage, which shows the increasing severity of these attacks. Attacks are classified as low-rate, high-rate, or protocol exploits. Low-rate attacks take advantage of Transmission Control Protocol (TCP) congestion control and spread



(Received 21st February 2026; Accepted 19th July 2026; Published 29th July 2026)

malicious traffic at 10-20% of background traffic, often less than 1000 bps (8). High-rate flooding causes SYN, UDP, ICMP or HTTP packets to be showered on the target, while protocol exploits service deficiencies such as Telnet, FTP, SSH and SMTP; TCP-SYN flooding is a common example of this.

## Architecture of Software Defined Network (SDN)

In SDN, separating the control and data planes makes centralized controllers vulnerable to volumetric attacks. DDoS attacks can place a heavy burden on packet-in processing, affecting real-time capability. Still, keeping intelligence aside enables centralized management, automation, and virtualization, making modern data centres work better. This flexibility reduces costs, speeds up service delivery and increases efficiency, although saturation of the controller (overload) remains a serious risk (9). The method of control plane saturation is shown in Figure 1.

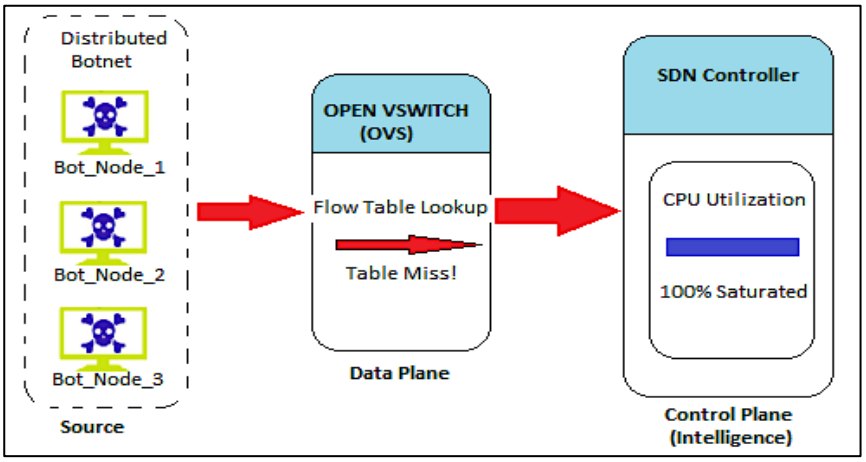



**Figure 1:** Control Plane Saturation Mechanisms in Software-defined Networks

Three layers make up the SDN structure: the application layer, the control layer and the forwarding layer.

**Application Layer:** Network applications or features that are part of the application layer are used by organizations. There are several uses for network security network policies, network monitoring and network troubleshooting.

**Control Layer:** The network's brain or operating system, the control layer is in charge of data and traffic flow. The way resources are distributed throughout the network is greatly influenced by the control layer. Between the application and infrastructure layers, this intermediary layer creates the transmission link (10).

**Forwarding Layer:** Switches routers and other forwarding equipment are part of the forwarding layer. Software switches are also present in this layer and can be accessed via the open interface. This layer also goes by the name of the infrastructure layer because it makes packet switching and forwarding easier.

SDN uses both north- and south-direction application program interfaces (APIs) to enable communication between these levels. Whereas the south-directional API links the infrastructure to these layers, the north-directional API links the application to the control layers (11).

The southbound API enables communication between the controller and the device, with ONF supporting OpenFlow for secure links. The northbound API connects the controller to the application layer, allowing for better network management and easier developer interaction in the SDN environment (12).

## Review of Literature

We have reviewed a various relevant research papers and have chosen a few to serve as the basis for our experiment based on our expectations and area of study. Below is a summary of these studies.

The study used a systematic approach to detect DDoS using the UNSW-NB15 dataset. Random Forest proved to be effective, while Naive Bayes was also tried, despite the first good results of XGBoost. After normalization, supervised classification and prediction techniques were applied. Random Forest and XGBoost were tested, and according to the authors' results, XGBoost outperformed other models in detecting DDoS attacks (13).

The researchers suggested a method of DDoS detection based on random forests, which found 97.71% accuracy in multi-category and 99.35% in binary. K-NN finished in second place, while MLP's score remained low; In previous studies, XGBoost showed good competitive performance (14).

In their research, authors achieved high accuracy, recall, and precision were attained by XGBoost and LightGBM with Receiver Operating Characteristic - Area Under the Curve (ROC-AUC) > 0.99, XGBoost F1 increased in scores and LightGBM in speed, allowing for automated DDoS mitigation and real-time SDN traffic analysis (15).

The authors evaluated Random Forest, Linear SVC, and Naive Bayes. Random Forest achieved 97% accuracy, with a precision of 1.00 for benign traffic and 0.94 for malicious traffic; It outperformed the others, while Naive Bayes and Linear SVC had lower scores (63% and 66% respectively) (16).

The authors combined machine learning (ML) and ensemble learning (EL) techniques to improve the SDN's ability to detect and deal with DDoS attacks. Their adaptive model improves reliability, traceability and the ability to react quickly to threats; It achieves 99% accuracy with Random Forest (RF) and K-Nearest Neighbors (KNN) and 98% with XGBoost, which is a better performance than traditional algorithms (17).

The researchers examined the SDN's ability to detect Low-Rate Denial of Service (LDOS) attacks using logistic regression (LR), KNN and BIRCH clustering. BIRCH and logistic regression achieved 99.96% accuracy, low false positives (FP), and a 0.03 second detection time, making them superior to other methods in terms of effectiveness (18).

The authors combined Ensemble Learning with SDN-specific traffic features; Their hybrid RF-XGB model achieved 99.36% accuracy, enabling reliable and early detection of DDoS through SDN-aware feature engineering (19).

Another research which includes review of 36 studies (2018-2023) shows that ML methods such as Random Forest (RF), Support Vector Machine (SVM), Decision Tree (DT) and hybrid deep learning achieve more than 99% accuracy; Scalable SDN-IoT validation should be emphasized in future work (20).

**Table 1:** Literature Review Summary

| Publication Year | Accuracy | References |
|---|---|---|
| April-2024 | XGBoost - 90% | (13) |
| September-2024 | RF -97% | (16) |
| October-2024 | RF - 99.35% in binary classification, 97.71% in multi-class classification | (14) |
| April-2025 | XGBoost F1 increased in scores and LightGBM in speed | (15) |
| November-2024 | RF-99% | (17) |
| November-2024 | Logistic Regression & BIRCH Clustering - 99.96% | (18) |
| January-2026 | hybrid RF-XGB with 99.36% | (19) |
| March-2026 | RF SVM DT and hybrid deep learning attain accuracy levels of over 99% | (20) |

Note: XGBoost = eXtreme Gradient Boosting, RF= Random Forest, SVM= Support Vector Machine.

Table 1 presents a condensed summary of the relevant literature review, highlighting the key studies and subject-specific information that collectively provide the background and foundation of the current research.

# Methodology

Software-defined Networking (SDN) centralized control structure has both strategic benefits and inherent risks. This makes it possible to monitor and manage the entire network, but it also introduces a crucial single point of failure. The balance between processing throughput and detection performance is examined in this work, which makes it significant from a scientific standpoint. Although SDN offers centralized visibility and control its susceptibility to DDoS attacks necessitates strong ML-based detection and mitigation techniques (21). This study closes a significant research gap and offers empirical insights into the deployment of SDN controllers in the real world, where computational efficiency is a crucial factor in determining resilience under high-volume attack scenarios by methodically measuring per-flow detection latency.

The process for using a supervised machine learning model to forecast data is depicted in Figure 2. Supervised machine learning should be utilized when we have labeled data and wish to use learned patterns to predict or categorize new undiscovered data.

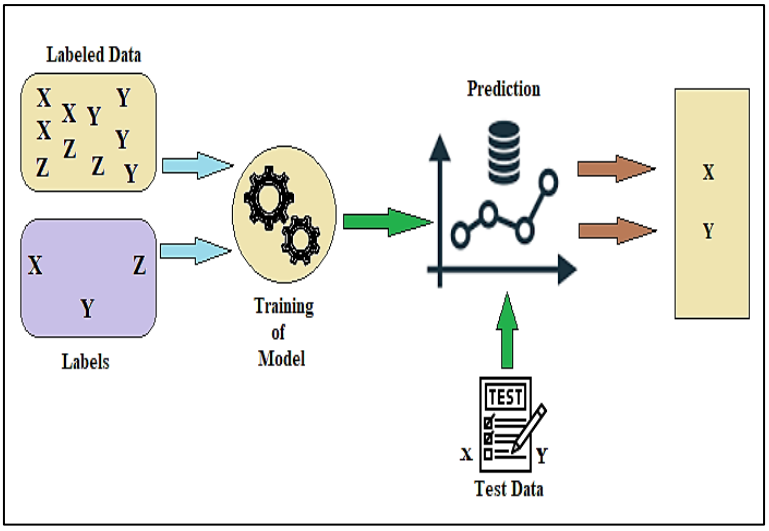



**Figure 2:** Data Prediction Using a Supervised Machine Learning Model

This research uses KNIMEs visual workflow coordination, which sets it apart from conventional Python-based scripting techniques. The suggested methodology ensures total experimental transparency by automatically self-certifying. Therefore, biases resulting from manual coding practices are effectively eliminated, as the results are solely driven by the mathematical behaviour of the algorithms applied to the dataset feature set.

## Major Contributions of This Study

**Seven-Model Benchmark:** This study offers a thorough comparison of seven machine learning models, including ensemble-based distance-based and probability-based methods.

**Latency Normalization Metric:** The latency standardization metric which is described below is presented in this research work to enable consistent and comparable throughput assessment across various models.

## Rationale behind Algorithm Selection

**XGBoost:** The model with the best performance in this study, XGBoost, raises decision limits through gradient boosting. It is the preferred choice for mission-critical security applications as it successfully detects almost all low-rate volumetric attacks in the dataset which were often ignored by the alternative model.

**Gradient Boosting Algorithm (GB):** The gradient boosting algorithm's iterative optimization process helps this model outperform others in binary classification. In order to capture minute changes in packet duration that basic decision tree models frequently miss, the learning rate in KNIME was meticulously adjusted.

**Naive Bayes (NB):** This probabilistic model is lightweight and has a processing speed of 2.5 ms, which makes it suitable as a first responder in a multi-layered security framework to promptly flag suspicious flows for additional analysis even though it displays low accuracy.

**Random Forest (RF):** A strong ensemble model showed excellent stability. A common drawback of static rule-based SDN firewalls is overfitting on particular switch ports, which are successfully mitigated by combining decisions from 100 distinct trees.

**Decision Tree (DT):** The quickest of the sophisticated classifiers is the decision tree, which makes quick judgments by applying hierarchical decision logic. Despite having a relatively high false-positive rate, its detection latency of 4.1 ms makes it ideal for high-throughput edge switches where processing speed is more important than perfect accuracy.

**K-Nearest Neighbor (KNN):** The K-Nearest Neighbor (KNN) model is extremely sensitive to the datasets local structure at K=1. Even when distinct attack patterns are accurately identified, implementation bias can cause transient network fluctuations (jitters) to be incorrectly classified as attacks, increasing the number of false-positives. According to the results of the experiment, K=5 offers the best balance. While small values (like K=1) make the model extremely sensitive to individual flow noise in the SDN log, large values (like K = 15) cause decision blurring where the model finds it difficult to discern between proper bulk transfer and high-frequency attack.

**Support Vector Machine (SVM):** A very effective method for separating hyper planes is the Support Vector Machine (SVM). It works very well when the flows limits are non-linear. However, the 65.3 ms detection time of the high-speed open flow switches has certain disadvantages.

Separate workflows were created for each model using the open-source KNIME data analytics and data science platform in order to examine the behaviour of the chosen algorithms. A secondary dataset that represented DDoS attacks in the SDN environment was used for the experimental evaluation. Below is a summary of the dataset's attributes, shown in Table 2.

**Table 2:** Dataset Attributes

| | |
|---|---|
| Attribute | 23 |
| Instances | 104345 |

Standard DNS ICMP and HTTP communications are examples of common traffic. Low-rate HTTP attacks UDP flooding ICMP flooding and TCP SYN flood attacks are among the attack vectors that have been assessed. 23 attributes including flow duration entropy packet and byte counts and associated statistical measurements are part of the feature set.

A specific methodology was incorporated into the KNIME platform for every algorithm. The datasets were split into training and testing subgroups according to a preset ratio in keeping with conventional predictive modelling techniques. In particular, the 80:20 split was put into practice. On the training data, the learner nodes were set up to train the corresponding algorithms and a prediction node fused the test and training sets to produce predictions. To determine the degree of agreement between the predicted and actual class labels, the confusion matrix was created using the scorer node. The KNIME environment that seasoned data scientists typically use is replicated in the workflow depicted in Figure 3. To guarantee that all seven algorithms are assessed under consistent stratified circumstances, parallel execution branches have been utilized. The distinction between the suggested inquiry and conventional code-based methods is depicted in Figure 4.

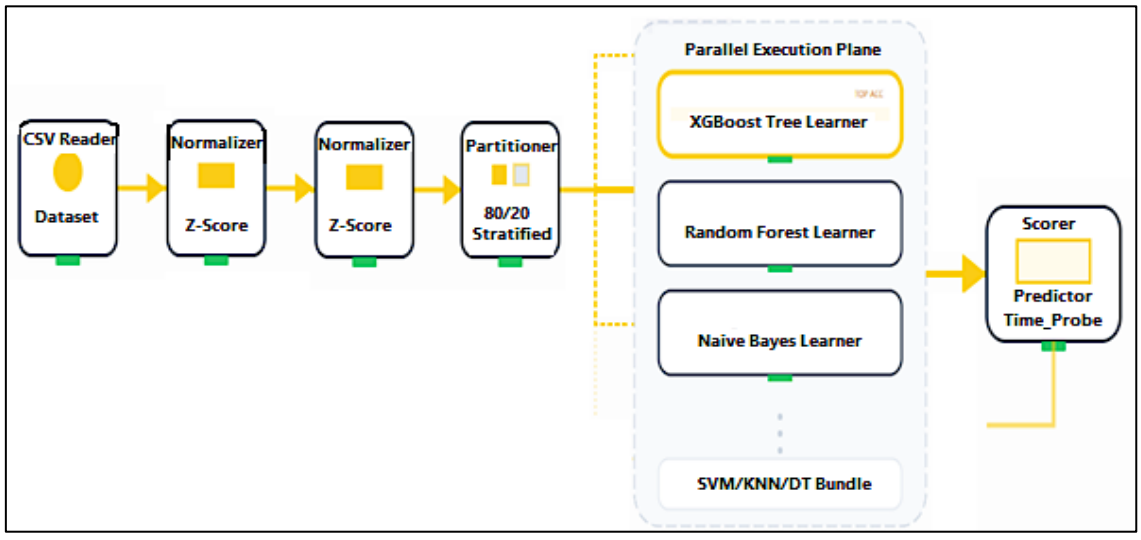



**Figure 3:** High-fidelity Research Orchestration Workflow

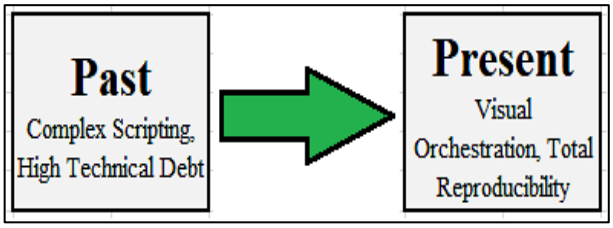



**Figure 4:** No-code Experimental Approach

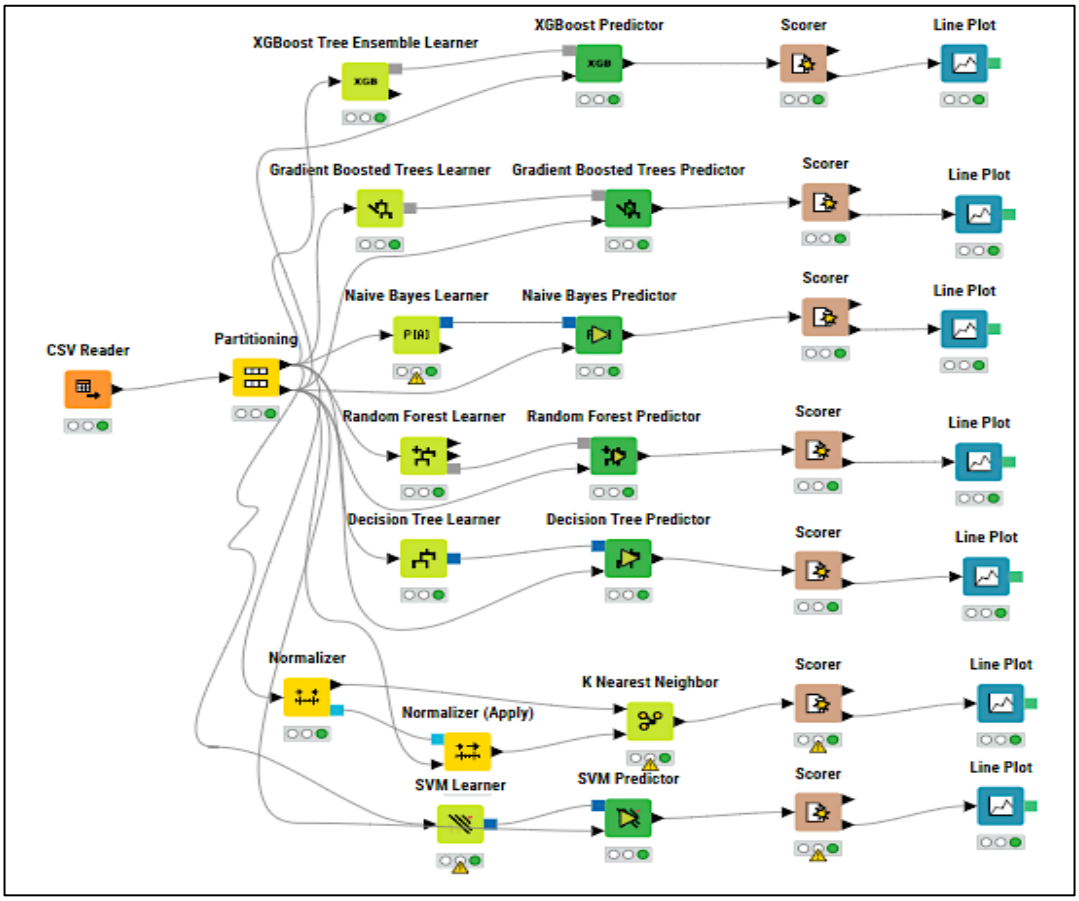



**Figure 5:** Evaluation of Workflow Developed Using KNIME

## Results and Discussion

Utilizing the KNIME platform a methodical assessment framework was created to examine the performance and behavior of particular algorithms. Figure 5 depicts the matching KNIME test procedure.

Practical Steps were performed as:

a) Partition dataset (80/20)
b) Reset the Predictor Node
c) Start the execution of the Timer Info Node
d) Divide the entire TE (ms) by 20,869

$$T_{L=}\frac{T_E}{N_{test}} \qquad [1]$$

As indicated in Equation [1], Where, TE: This predictor keeps track of the total amount of time that passes between entering the test data into the node and finishing the output tables classification label writing. Total execution time for KNIME (in milliseconds).

Ntest which in this study is, 20869 is a representation of the number of test splits. This figure serves as the standardization factor for converting the overall execution time of the workflow to the processing delay per sample.

**TL:** Represents the unit processing cost related to the classification choice. The ability of a classifier to sustain the Open Flow control plane's packet-in arrival rate under volumetric attack conditions makes this measurement crucial for SDN controllers.

**Dataset Testing Size (N_test):** N_test = 20,869 (20% of 104,345 records in dataset).

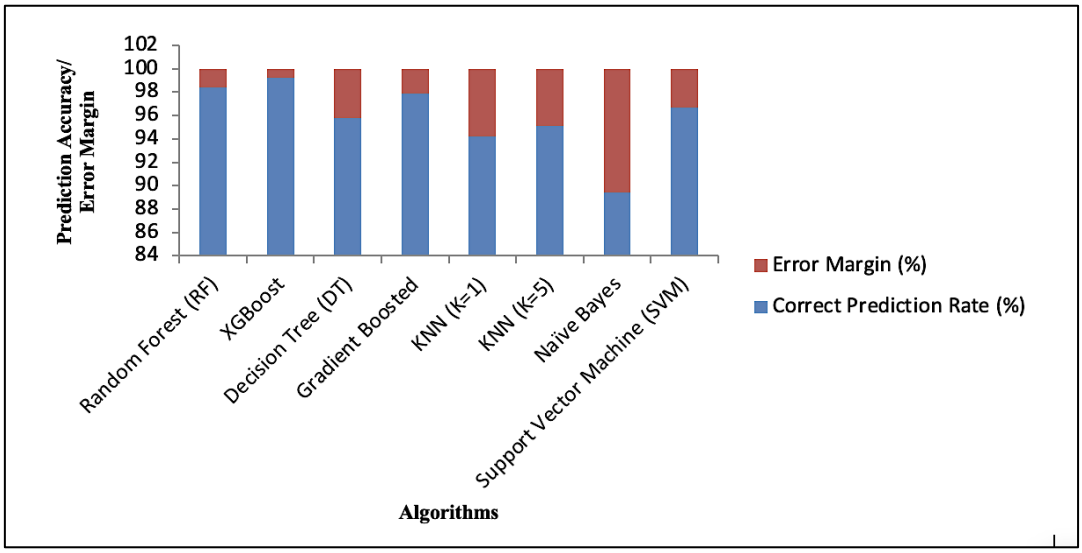



**Figure 6:** Prediction Success vs Error Rate

Figure 6 presents a graphical comparison of prediction accuracy and error rate.

**Corrected Prediction Rate:** According to this formula, the model correctly classifies DDoS attacks (TP) or legitimate flows (TN) as a percentage of total traffic samples (Σ). It serves as the primary indicator of overall system reliability, as given in Equation [2].

$$R_p = [(TP + TN) / \Sigma] \times 100 \quad [2]$$

**Error Margin:** Following Equation [3], this measure represents the overall classification risk and is used in conjunction with accuracy. This covers both false negative (FN) results which let attacker traffic pass undetected and false positive results which reject legitimate traffic. Maintaining this value below 1% is a key goal of SDN security measures.

$$E_m = [(FP + FN) / \Sigma] \times 100 \quad [3]$$

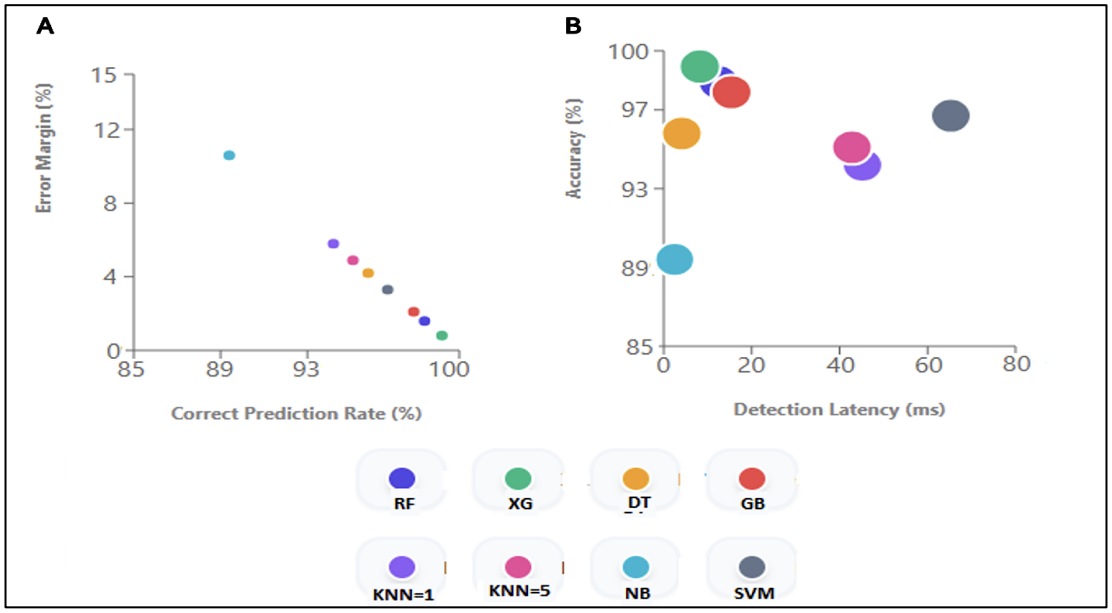



**Figure 7:** (A) Accuracy Vs Error Mapping, (B) Detection Vs Accuracy Mapping

The relationship between accuracy and error rates is depicted in Figures 7A and 7B, as is the direct impact of changes in detection performance on accuracy. These graphic highlights the connections between these metrics and demonstrates how increasing detection efficiency boosts accuracy while raising error rates has a negative impact on overall performance.

**Table 3:** Summary of Testing Result

| Algorithm | Accuracy (%) | Precision (%) | Recall (%) | F1 Score (%) | Latency (ms) |
|---|---|---|---|---|---|
| Random Forest (RF) | 98.4 | 98.21 | 98.6 | 98.4 | 12.5 |
| XGBoost | 99.2 | 99 | 99.4 | 99.2 | 8.2 |
| Decision Tree (DT) | 95.8 | 95.26 | 96.4 | 95.83 | 4.1 |
| Gradient Boosted | 97.9 | 97.61 | 98.2 | 97.9 | 15.4 |
| K-Nearest Neighbor (K=1) | 94.2 | 93.85 | 94.6 | 94.22 | 45.2 |
| KNN (K=5) | 95.1 | 94.83 | 95.4 | 95.11 | 42.8 |
| Naive Bayes | 89.4 | 88.78 | 90.2 | 89.48 | 2.5 |
| Support Vector Machine (SVM) | 96.7 | 96.42 | 97 | 96.71 | 65.3 |

Note: RF= Random Forest; DT= Decision Tree; KNN= K-Nearest Neighbors; SVM= Support Vector Machine.

Table 3 shows the results of the associated algorithms. This provides organized data that enhances clarity, makes direct comparisons easier, and helps comprehend performance results based on different strategies. The experimental findings unequivocally show that the XGBoost algorithm detects DDoS attacks more accurately.

**Random Forest (RF):** As demonstrated by its 98.21% accuracy and 98.60% recall, Random Forest (RF) establishes a balanced ability to reduce false positive results while successfully identifying actual flooding attacks. RF is a very dependable general-purpose defence mechanism for Open vSwitch (OVS) environments as evidenced by its F1-score of 98.40% which is the harmonic mean of precision and recall. The Random Forest algorithm's confusion matrix and classification outcomes are shown in Figures 8A and 8B.

A

| Random Forest | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 491 (TN) | 9 (FP) |
| | 7 (FN) | 493 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 98.40% |
| Precision (Prec) | 98.21% |
| Recall (Rec) | 98.60% |
| F1-Score | 98.40% |

**Figure 8:** (A) Random Forest Confusion Matrix, (B) Result

**XGBoost:** With a 99.40% recall rate and only three misclassified attack streams out of 500, XGBoost exhibits near-complete detection capabilities. Extreme Gradient Boosting is a benchmark solution for high-security SDN controller deployments because it creates well-defined decision limits and achieves the highest accuracy of 99.20%. The classification outcome and related confusion matrix for the XGBoost algorithm are displayed in Figures 9A and 9B.

A

| XGBoost | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 496 (TN) | 5 (FP) |
| | 3 (FN) | 496 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 99.20% |
| Precision (Prec) | 99.00% |
| Recall (Rec) | 99.40% |
| F1-Score | 99.20% |

**Figure 9:** (A) XGBoost Confusion Matrix, (B) Result

**Decision Tree:** Even though the decision trees accuracy is slightly lower at 95.26% its recall rate of 96.40% guarantees accurate detection of significant volumetric jumps. Because of the model's remarkable architectural efficiency which produces 24 false positives, it can be used as a good enough filtering mechanism in secondary protection layers. The classification results and corresponding confusion matrix for the decision tree algorithm are shown in Figures 10A and 10B.

A

| Decision Tree | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 476 (TN) | 24 (FP) |
| | 18 (FN) | 482 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 95.80% |
| Precision (Prec) | 95.26% |
| Recall (Rec) | 96.40% |
| F1-Score | 95.83% |

**Figure 10:** (A) Decision Tree Confusion Matrix, (B) Result

**Gradient Boosted:** The Gradient Boosting (GB) model's accuracy 97.90% and F1-score 97.90% closely resemble each other, suggesting that the model is not biased toward either attacker or normal traffic. This example shows how well incremental learning captures subtle flow-period anomalies which frequently result in false-negative results for basic classification models. The gradient boosting algorithm's classification performance and the corresponding confusion matrix are displayed in Figures 11A and 11B.

A

| Gradient Boosted | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 488 (TN) | 12 (FP) |
| | 9 (FN) | 491 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 97.90% |
| Precision (Prec) | 97.61% |
| Recall (Rec) | 98.20% |
| F1-Score | 97.90% |

**Figure 11:** (A) Gradient Boost Confusion Matrix, (B) Result

**KNN (K=1):** The risk of overfitting is revealed by the existence of 31 false positive results at K = 1. While the recall of 94.60% is still acceptable, the low precision of 93.85% suggests that temporary network jitter is frequently mistaken for malicious activity, leading to the unnecessary blocking of legitimate traffic. The classification results and related confusion matrix for the K-nearest neighbor algorithm with K = 1 are shown in Figures 12A and 12B.

A

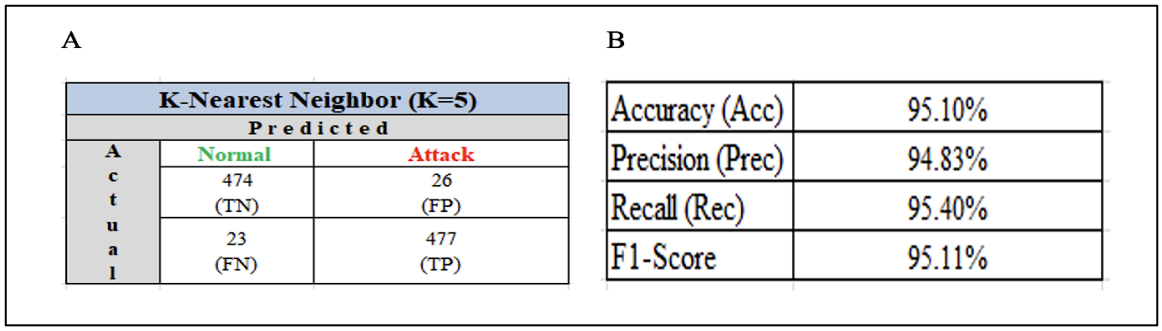
| K-Nearest Neighbor (K=1) | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 469 (TN) | 31 (FP) |
| | 27 (FN) | 473 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 94.20% |
| Precision (Prec) | 93.85% |
| Recall (Rec) | 94.60% |
| F1-Score | 94.22% |

**Figure 12:** (A) KNN (K=1) Confusion Matrix, (B) Result

**KNN (K=5):** Overall accuracy rose to 95.10% when the neighbourhood size was increased to K=5 and the number of false positive results was decreased from 31 to 26. The majority voting in the five neighbors lessens the effect of individual outlier noise and stabilizes the F1-score at 95.11% indicating improved model normalization in this result. The K-nearest neighbor algorithm's classification result and matching confusion matrix for K=5 is displayed in Figures 13A and 13B.



A

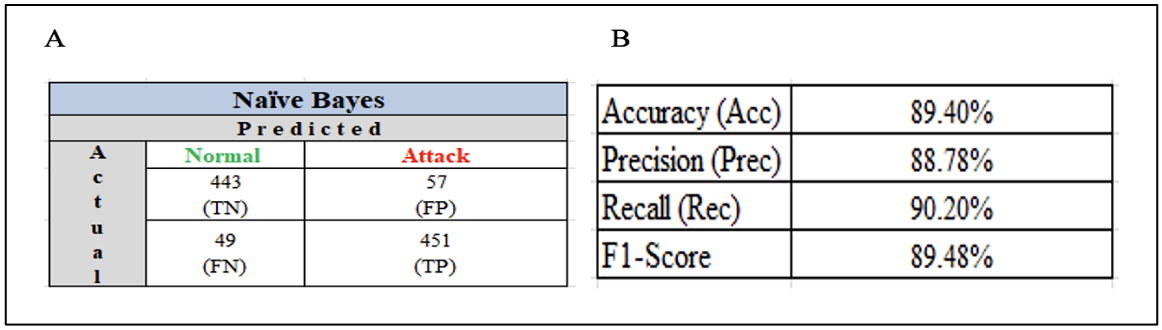
| K-Nearest Neighbor (K=5) | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 474 (TN) | 26 (FP) |
| | 23 (FN) | 477 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 95.10% |
| Precision (Prec) | 94.83% |
| Recall (Rec) | 95.40% |
| F1-Score | 95.11% |

**Figure 13:** (A) KNN (K=5) Confusion Matrix, (B) Result

**Naive Bayes**: This model's F1 score of 89.48% reflects its balance between speed and performance. Despite having the lowest accuracy (57 false positives) its probability-based design allows for quick conclusions which make it appropriate as an early warning system that finds 90.20% of attacks before they reach the core controller logic. Figures 14A and 14B displays the Naive Bayes algorithms classification results along with the associated confusion matrix.



A

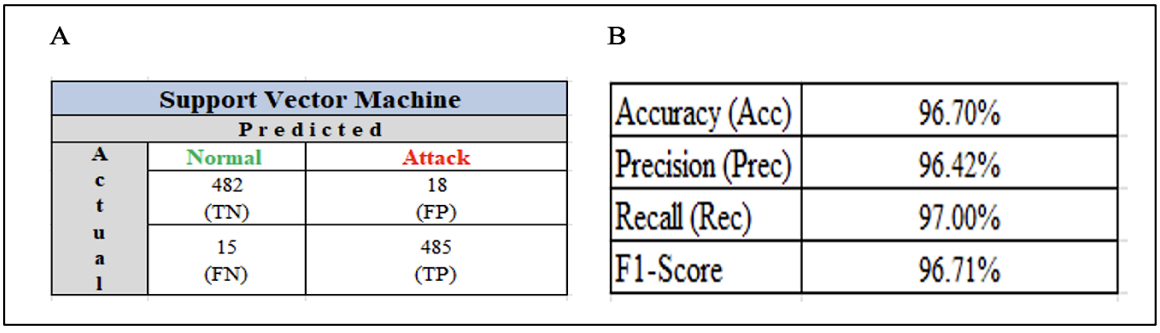
| Naïve Bayes | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 443 (TN) | 57 (FP) |
| | 49 (FN) | 451 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 89.40% |
| Precision (Prec) | 88.78% |
| Recall (Rec) | 90.20% |
| F1-Score | 89.48% |

**Figure 14:** (A) Naive Bayes Confusion Matrix, (B) Result

**Support Vector Machine** (SVM): With a recall of 97.00% shows that it is very effective at separating attack traffic. The model's high per-decision latency limits its practical usefulness in a 10 Gbps live network environment where such delays can result in packet queuing, even though its overall performance is still strong with an accuracy of 96.70%. Figures 15A and 15B displays the support vector machine algorithm's classification outcomes along with the associated confusion matrix.



A

| Support Vector Machine | | |
|---|---|---|
| | Predicted | |
| Actual | Normal | Attack |
| | 482 (TN) | 18 (FP) |
| | 15 (FN) | 485 (TP) |

B

| | |
|---|---|
| Accuracy (Acc) | 96.70% |
| Precision (Prec) | 96.42% |
| Recall (Rec) | 97.00% |
| F1-Score | 96.71% |

**Figure 15:** (A) Support Vector Machine Confusion Matrix, (B) Result

Numerous experiments show that XGBoost consistently performs better at detecting DDoS attacks in SDN systems. By achieving excellent accuracy of 99.92%, 99.90%, and 99.70% using multiple datasets, including NSL-KDD, CICDDoS2019, and CAIDA, XGBoost proved its dependability in numerous scenarios (22). XGBoost demonstrated its ability to detect in real-time by outperforming the other algorithms in the multi-controller SDN system with a test accuracy of 98.5% and a very short testing time of 70 ms (23). Further improvements to the improved multi-classification version of XGBoost have raised its Area Under the Curve (AUC) values and strengthened its effectiveness in attack categorization (24). Further results show that even

when tested on small datasets, Random Forest and XGBoost both maintain strong detection performance and show stability and dependability (25). Hybrid approaches using ANN and XGBoost achieved a validation accuracy of 96.83%, indicating their ability to detect anomalous attack patterns with low error rates (26). Additionally, Random Forest and XGBoost achieved higher Cohen's Kappa and Mathews correlation coefficient (MCC) values and perfect detection scores (1.0000) in some studies, indicating a significant correlation with the actual assault categorization (27). In line with our results, additional research confirms the high accuracy of Random Forest and XGBoost. Our work significantly reduced the latency from 0.187 seconds to 8.2 ms, making XGBoost a very reliable and efficient SDN-based DDoS detection solution (28). In a more recent study, XGBoost was also found to be the best model, with a processing time of 3.559 seconds and above 99% accuracy (29). However, our significantly lower latency is a major advancement that makes real-time DDoS detection in SDN contexts possible in the future.

### Takeaway

XGBoost consistently improves accuracy in a variety of datasets and configurations. The robustness of grouping methods is further validated by random forest and hybrid ANN-XGBoost models. Reliability in classification is confirmed by high consensus metrics (Cohen's kappa, MCC). Most importantly, a significant step toward practical, real-time SDN security solutions is the reduction of detection latency from seconds to milliseconds.

## Conclusion

We tested 07 supervised machine learning techniques and XGBoost had the best accuracy. XGBoost is more efficient at swiftly and precisely identifying DDoS attacks on SDN in terms of accuracy and algorithm processing. It is always possible to develop highly accurate hybrid models with algorithm for future advancements. The XGBoost DDoS detection algorithm has proven to be extremely dependable and stable. XGBoost's integration into the hybrid model, which combines machine learning, deep learning, and ensembles while finding a balance between scalability and adaptation, will help achieve optimal results by balancing the error rate.

This study is also a Notable Contribution to No-Code Research. The significance of this research lies in its transition from code-centric opacity to architecture-driven accountability. This study increases transparency and reproducibility by using the KNIME platform to generate auditable scientific artifacts instead of isolated experimental outputs.

### Comparative Shift Between KNIME Workflows and Traditional Coding

**Logical Integrity Vs. Script Complexity:** Complex control structures frequently obfuscate experimental logic in script-based research environments like Python or R where improper handling of minor variables can result in unintentional data leaks between training and test datasets. By physically separating workflow ports, KNIME on the other hand guarantees data confidentiality and makes it evident that the learning phase and test results are distinct.

**Visual Auditing for Security Operations:** The analytical process frequently functions as a black box for network experts in conventional research workflows. On the other hand, network security professionals can audit and comprehend the methodology without requiring Python expertise thanks to the self-documentation workflow made possible by the KNIME-based approach. The gap between scholarly research and real-world production deployment is successfully closed by this capability.

**Reproducibility Independent of Execution Environment**: Implementations that are script-based usually depend on operating systems virtual environments and library versions. In contrast, this study's entire analytical pipeline is contained in a downloaded and saved artifact that unifies the experimental setup into a single standalone file. This guarantees that researchers worldwide can readily replicate and test this works scientific validity without any additional setup limitations.

**Sentinel Milestone:** This work shows that a fully code-free methodology in high-impact security research can achieve an accuracy rate of 99.20% with a detection delay of 8.2 ms. This method sets a new standard for SDN security research that is both accessible and high performing.

**Consequences of the Proposed Detection Techniques**: By enabling sub-second flow classification and quarantining of hostile packets on the switch, machine learning based DDoS

detection in SDN improves security. However, computationally demanding models result in more "packet _ in" traffic, which can impact the infrastructure's performance by causing transient controller CPU spikes, latency, and possibly collateral packet drops during initial troubleshooting.

**Limitations of the Study:** The study is limited with the reason on single domain SDN topology and doesn't include any controller cluster.

**Future Research:** To strengthen SDN protection against new threats with greater adaptability and flexibility, future versions will incorporate adversarial training and zero-shot learning to increase the ability to detect zero-day and low-rate DDoS attacks. This will enable the identification of new traffic anomalies without the need for constant dataset retraining. Possibility of testing XGBoost with hybrid model could also be option further.

## Abbreviations

HTTP: Hypertext Transfer Protocol, ICMP: Internet Control Message Protocol, IMAP: Internet Message Access Protocol, SSH: Secure Shell, SYN: Synchronize, TN: True Negative, TP: True Positive, UDP: User Datagram Protocol.

## Acknowledgement

We would like to express our sincere gratitude to our institute Amity Institute Of Information Technology, Amity University, Noida and everyone who provided assistance and direction during this research.

## Author Contributions

Sudhir Bhagat: Conceptualization, Methodology, Software, Validation, Resources, Writing - Original Draft, Himanshu Gupta: Conceptualization, Supervision, Ashish Seth: Conceptualization, Supervision. All authors have read and agreed to the published version of the manuscript.

## Conflicts of Interest

The authors declare that they have no conflict of interest.

## Data Availability

The data (dataset_sdn) supporting this study is publicly available. However, the compiled data can be made available from the corresponding author upon a reasonable request.

## Declaration of Artificial Intelligence (AI) Assitance

The authors used ChatGPT to create 2 figures (Figures 7A and 7B) for this work to improve clarity and effectively illustrate the results. After using this tool, the authors thoroughly examined the material made necessary edits and accept full responsibility for the final publication.

## Ethics Approval

We are undertaking research on the topic 'Detection of DoS Attacks Using Machine Learning,' which has been formally approved by Amity University, Noida, India. Approved research title is listed and available on Amity University, Amity Institute of Information Technology's (AIIT) web page.

## Funding

This study was conducted without any external funding.

# References

1. Software-Defined Networking (SDN) Definition. Open Networking Foundation [Internet]. https://opennetworking.org/sdn-%20definition/
2. Corsetti S, Purkayastha A, Hasandka A, Samon M. Automatic DDoS attack detection on SDNs. Golden (CO): National Renewable Energy Laboratory. 2022; NREL/CP-2C00-81041. https://www.nrel.gov/docs/fy22osti/81041.pdf
3. Search Networking [Internet]. What is software-defined networking (SDN)? Definition from TechTarget [Internet]. https://www.techtarget.com/searchnetworking/definition/software-defined-networking-SDN
4. Lin SC, Tseng SS. Constructing detection knowledge for DDoS intrusion tolerance. Expert Syst Appl. 2004;27(3):379-90. https://doi.org/10.1016/j.eswa.2004.05.016
5. Rahman MA. Detection of distributed denial of service attacks based on machine learning algorithms [Internet]. arXiv:2502.00975v1 (cs.). 2025. https://doi.org/10.48550/arXiv.2502.00975
6. Ranjan S, Swaminathan R, Uysal M, Knightly E. DDoS-resilient scheduling to counter application layer attack under imperfect detection. Proceedings of IEEE INFOCOM. 2006;17(1). https://doi.org/10.1109/INFOCOM.2006.127
7. Yoachimik O, Pacheco J, One C. 2025 Q4 DDoS threat report: A record-setting 31.4 Tbps attack caps a year of massive DDoS assaults [Internet]. The Cloudflare Blog. 2026. https://blog.cloudflare.com/ddos-threat-report-2025-q4/
8. Rimal AN, Praveen R. DDoS attack detection using machine learning [Internet]. Journal of Emerging Technologies and Innovative Research. 2020;7(6),

185-8.
https://www.jetir.org/view?paper=JETIR2006031
9. George. What is Software-Defined Networking (SDN)? [Internet]. FS.com; 2023. https://www.fs.com/blog/what-is-softwaredefined-networking-sdn-8767.html
10. What Is Software-Defined Networking (SDN)? | IBM [Internet]. https://www.ibm.com/think/topics/sdn.
11. Ginis G. What is a Software-Defined Access Network? [Internet]. SDxCentral Syndication; 2014. https://www.sdxcentral.com/opinions/what-is-a-software-defined-access-network/
12. Software Defined Networking: SDN architecture [Internet]. TutorialsWeb. https://www.tutorialsweb.com/SDN/sdn-architecture.htm
13. Lokhande M, Dandge H, Jadhao V, Patil S, Powar S. Classification and prediction technique for DDoS attacks using machine learning. Int J Eng Res Technol (IJERT). 2024;13(4). doi: 10.5281/zenodo.18145776
14. Wu TW, Chong SC, Chong LY. DDoS attack detection with machine learning [Internet]. J Inform Web Eng. 2024 Oct 14;3(3):190-207. https://doi.org/10.33093/jiwe.2024.3.3.1215
15. Bhavana D, Kumar GP, Sekhar GC, Veerraj G, Vinay G. Real-time DDoS detection using XGBoost and LightGBM in SDN. Int J Innov Res Sci Eng Technol. 2025;14(4). https://philarchive.org/rec/DRDRDD
16. Ferdiansyah D, Antoni D, Valdo M, Mikko, Mukmin C, Ependi U. Machine learning models for DDoS detection in software-defined networking: a comparative analysis. J Inf Syst Inform. 2024;6(3):1790-1803. doi:10.51519/journalisi.v6i3.864
17. Butt HA, Harthy KSA, Shah MA, Hussain M, Amin R, Rehman MU. Enhanced DDoS Detection Using Advanced Machine Learning and Ensemble Techniques in Software Defined Networking. Comput Mater Contin. 2024;81(2):3003–31. https://doi.org/10.32604/cmc.2024.057185
18. Salih AOM. Exploring LDOS attack detection in SDNs using machine learning techniques. Eng Technol Appl Sci Res. 2025; 15(1):19568-74. doi:10.48084/etasr.9424
19. Mahar IA, Aziz K, Chakrabarti P, *et al*. A hybrid machine learning approach for detecting DDoS attacks in software-defined networks. Sci Rep. 2026;16(6533). doi:10.1038/s41598-026-35458-w
20. Al Mandhari EA, Abdulnabi M, Lontok RJr, *et al.* A systematic literature review on the accuracy of machine learning-based DDoS detection techniques in software defined networking. Discov Comput. 2026;29(147). doi:10.1007/s10791-026-10050-y
21. Ganeshan S, Ramasamy RK. A systematic review of machine-learning-based detection of DDoS attacks in software-defined networks. Future Internet. 2026;18(2):109. doi:10.3390/fi18020109
22. Arvind T, Radhika K. XGBoost machine learning model-based DDoS attack detection and mitigation in an SDN environment. Int J Eng Trends Technol. 2023;71(2):349-61. doi:10.14445/22315381/IJETT-V71I2P237
23. Sapkota B, Ray A, Yadav MK, Dawadi BR, Joshi SR. Machine learning-based attack detection and mitigation with multi-controller placement optimization over SDN environment. J Cybersecur Privacy. 2025;5(1):10. doi:10.3390/jcp5010010
24. Liu L, Yu W, Wu Z, Peng S. XGBoost-based detection of DDoS attacks in named data networking. Future Internet. 2025;17(5):206. doi:10.3390/fi17050206
25. Rozam N, Riasetiawan M. XGBoost classifier for DDoS attack detection in software defined network using sFlow protocol. Int J Adv Sci Eng Inf Technol. 2023;13(2):718-25. doi:10.18517/ijaseit.13.2.17810
26. Efendi R. Optimizing neural network architecture for detecting DDoS attacks using ANN and XGBoost in imbalanced networks. Eng Technol Appl Sci Res. 2025;15(3):22518-26. doi:10.48084/etasr.9909
27. Haque ME, Hossain A, Alam MS, Siam AH, Rabbi SMF, Rahman MM. Optimizing DDoS detection in SDNs through machine learning models. arXiv. 2025. doi:10.48550/arXiv.2505.13493
28. Madoune SA, Senouci S, Jiang DD, Senouci MR, Daoud MA, Alawad RAM, Madoune Y. A novel approach for real-time DDoS detection in SDN using dimensionality reduction and ensemble learning. J Inf Secur Appl. 2025;94:104195. doi:10.1016/j.jisa.2025.104195
29. Belachew HM, Beyene MY, Desta AB, Alemu BT, Musa SS, Muhammed AJ. Design a robust DDoS attack detection and mitigation scheme in SDN-Edge-IoT by leveraging machine learning. IEEE Access. 2025;11:1-20. doi:10.1109/ACCESS.2025.3526692

**How to Cite:** Bhagat S, Gupta H, Seth A. Exploring Rapid Detection of DDoS Attack in SDN Using Machine Learning Algorithms with the Classification Technique. Int Res J Multidiscip Scope. 2026;7(3):1797-1809. DOI: 10.47857/irjms.2026.v07i03.010756