A Review on Software Defined Networking-Based Network Traffic Classification Using Machine Learning Techniques

Kratika Halankar[1], Pradeep Nayak[2], Khushi R[3], Kiran Kumar S[4], Kiran Kumar[5]

Assistant Professor, Information Science and Engineering[2]

Students, Information Science and Engineering[1,3,4,5]

Alvas Institute of Engineering and Technology, Moodbidri, Karnataka.

Abstract:

With the rise in internet usage, accurate network traffic classification has become essential for secure and efficient communication. Traditional methods like port-based and payload inspection are no longer reliable due to dynamic traffic patterns and growing encryption. To address these challenges, this study applies Machine Learning (ML) models within a Software Defined Networking (SDN) environment to classify DNS, Telnet, Ping, and Voice traffic. Using D-ITG for traffic generation and Mininet for SDN simulation, multiple ML algorithms including Logistic Regression, Decision Tree, Random Forest, AdaBoost, SVM, and K-means were evaluated after thorough preprocessing and feature extraction. Among them, the Decision Tree achieved the highest accuracy of 99.81%. Overall, the findings show that ML integrated with SDN significantly enhances offline and real-time traffic classification, even for encrypted traffic, while improving QoS.

Keywords: Software defined networking, Machine learning, Traffic classification, Quality of service

Abbreviations:

This study uses several technical abbreviations, including ANN (Artificial Neural Network), CNN (Convolutional Neural Network), D-ITG (Distributed Internet Traffic Generator), DL (Deep Learning), DNN (Deep Neural Network), DoS (Denial of Service), DPI (Deep Packet Inspection), DT (Decision Tree), LR (Logistic Regression), ML (Machine Learning), MLP (Multilayer Perceptron), NB (Naïve Bayes), NC (Nearest Centroid), PCA (Principal Component Analysis), QoS (Quality of Service), RF (Random Forest), SDN (Software Defined Networking), SVM (Support Vector Machine), and TC (Traffic Classification).

Introduction

The rapid growth of internet usage and diverse online applications has made accurate traffic classification crucial for modern networks. Traditional methods such as port-based checks, protocol signatures, and payload inspection are becoming ineffective due to dynamic traffic patterns, encryption, and non-standard port usage. This has driven interest in Software Defined Networking (SDN) and Machine Learning (ML) for more adaptive and intelligent traffic analysis.

SDN provides centralized control and flexible network management, making it suitable for ML-based classification, especially for tasks like QoS management, congestion control, and security. Flow-based ML approaches using statistical features instead of payloads are well suited for encrypted traffic and reduce processing overhead. However, previous studies often rely on static datasets, lack real-time validation, and struggle with encrypted and evolving traffic behaviours.

This work addresses these limitations by integrating ML models into an SDN framework to classify Ping, Telnet, Voice, and DNS traffic. Realistic traffic generated using Mininet and D-ITG is analysed with algorithms such as Logistic Regression, Decision Tree, Random Forest, AdaBoost, SVM, and K-means. Results show that SDNassisted ML classification achieves high accuracy, improves encrypted traffic handling, and enhances real-time network performance.

Overall, the study demonstrates that combining ML with SDN significantly strengthens traffic classification and network management in modern, dynamic, and encrypted environments.

Literature Review

Efficient traffic management is vital for modern networks, and many studies have explored ML-based classification in SDN environments. Ref.[6] achieved 99.4% accuracy using K-NN for real-time DNS, Telnet, Ping, and Voice traffic, while Ref.[4] showed that Simulated Annealing–based clustering enhances SDN network lifetime. Surveys such as Ref.[11] and Ref.[12] highlight the growing role of ML and GNNs in programmable networks, though scalability and evaluation gaps remain.

Deep learning approaches (Ref.[13]) have reached ~87% accuracy, and ByteSGAN (Ref.[14]) introduced a semi-supervised GAN model but did not address data imbalance. With encrypted traffic exceeding 80%, Ref.[15] proposed new ML features achieving up to 93% accuracy. Security-focused works, including Ref.[8] and Ref.[5], emphasize challenges in QoS-aware and scalable SDN–ML integration. Real-time studies like Ref.[16] and Sherif et al. (Ref.[9]) achieved high accuracy, including 99.8% using a Decision Tree. IoT-based SDN research (Ref.[18]) also showed strong performance with lightweight flow features.

As payload-based methods struggle with encrypted traffic, ML approaches leveraging flow-level features have become essential. Building on these insights, this study integrates ML within an SDN framework to deliver scalable, accurate, and encryption-resilient real-time traffic classification.

Methodology

The proposed model follows a structured methodology that begins with collecting a comprehensive dataset for training and validation. The acquired traffic data is then pre-processed to remove noise, normalize features, and improve overall data quality. After preprocessing, the dataset is divided into training and testing subsets, and various supervised and unsupervised ML algorithms are applied to learn and evaluate classification patterns. This approach is designed to maximize the accuracy of classifying different traffic types such as Ping, Telnet, Voice, and DNS, while ensuring reliable performance in realistic network conditions. Ultimately, the methodology supports improved network management, enhanced Quality of Service (QoS), and strengthened security through more precise and intelligent traffic analysis.

System architecture

The proposed work develops an SDN-based model that detects and classifies network traffic using machine learning techniques. As illustrated in Fig. 1, the system architecture is organized into three key stages: preprocessing, feature extraction, and classification. The preprocessing phase involves preparing the raw traffic data by standardizing formats, filtering noise, and handling any missing values to ensure clean and consistent input. Once preprocessing is complete, relevant flow-level features are extracted and passed to the classification module, where ML algorithms are applied to accurately categorize the traffic types.

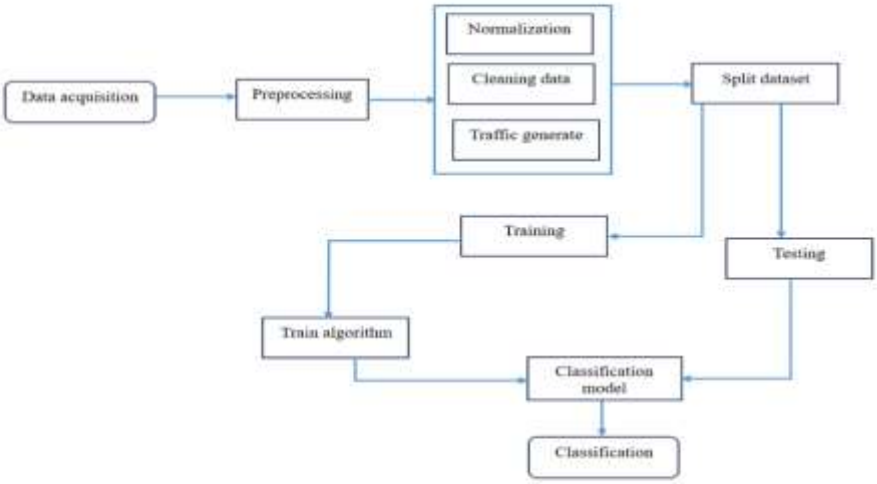


Fig. 1. System architecture of the proposed model.

Software-Defined Network Environment and Dataset Generation

Figure 2 shows the simulated network topology built in the VirtualBox environment, consisting of five virtual machines: an SDN controller, a Layer-2 OVS switch, and three hosts. This setup uses Mininet to emulate realistic network behaviour with greater flexibility and scalability than a single-VM simulation. By modelling multiple interconnected virtual devices, Mininet more accurately reproduces real-world traffic interactions and provides a reliable platform for validating SDN-based traffic classification models.

In the SDN architecture, the controller continuously monitors flow activities across all hosts to maintain complete visibility of packet forwarding through the switch. Flow-level statistics including timestamps, Datapath IDs, MAC addresses, ports, packet counts, and byte totals were captured at one-second intervals, as summarized in Table 1. A dedicated Python script retrieved and updated these flow statistics in real time, forming a dynamic and accurate dataset for machine learning analysis.

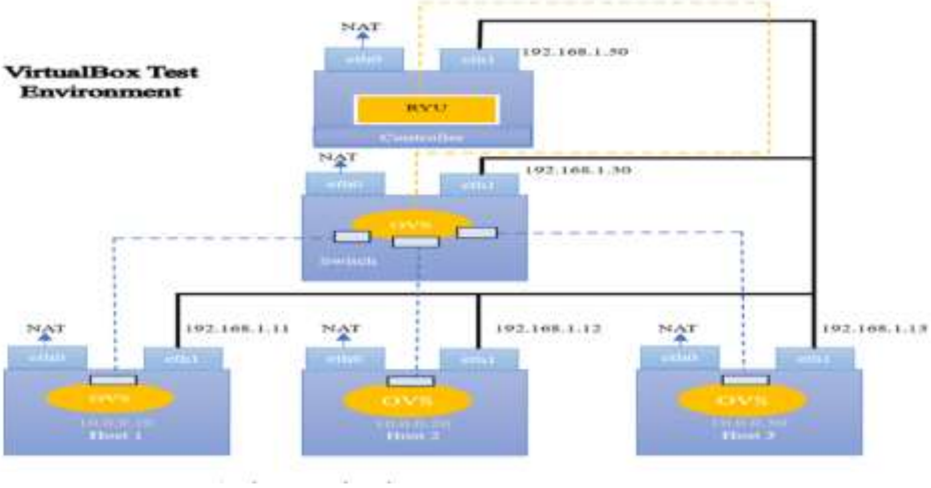
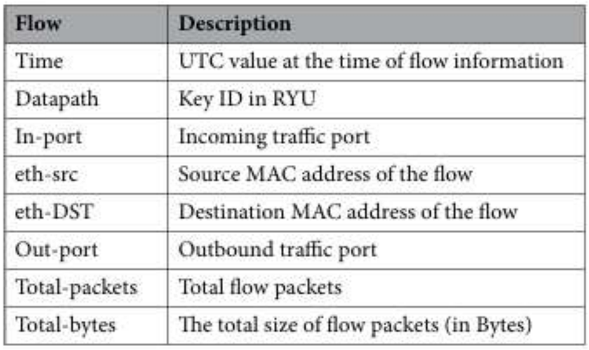


Fig. 2. Simulation network topology.



Flow	Description
Time	UTC value at the time of flow information
Datapath	Key ID in RYU
In-port	Incoming traffic port
eth-src	Source MAC address of the flow
eth-DST	Destination MAC address of the flow
Out-port	Outbound traffic port
Total-packets	Total flow packets
Total-bytes	The total size of flow packets (in Bytes)

Table 1. Flow data features.

Data acquisition

To generate realistic traffic flows for ML training, D-ITG was used to produce encrypted IPv4/IPv6 traffic for Ping, Telnet, DNS, and Voice (G.711) applications. Because encrypted communication is increasingly common in modern networks, all traffic in this study was collected in encrypted form.

Each application was executed sequentially in a controlled environment to ensure accurate labelling. Traffic flows were initiated between client hosts, and a classification script integrated with the RYU controller and a monitoring Python program captured flow-level information (Fig. 3).

The monitoring script extracted key features and stored them in separate CSV files for each traffic type, which were later merged into a unified dataset for model training and evaluation. Table 2 presents the number of instances generated for each traffic class used across the ML experiments.

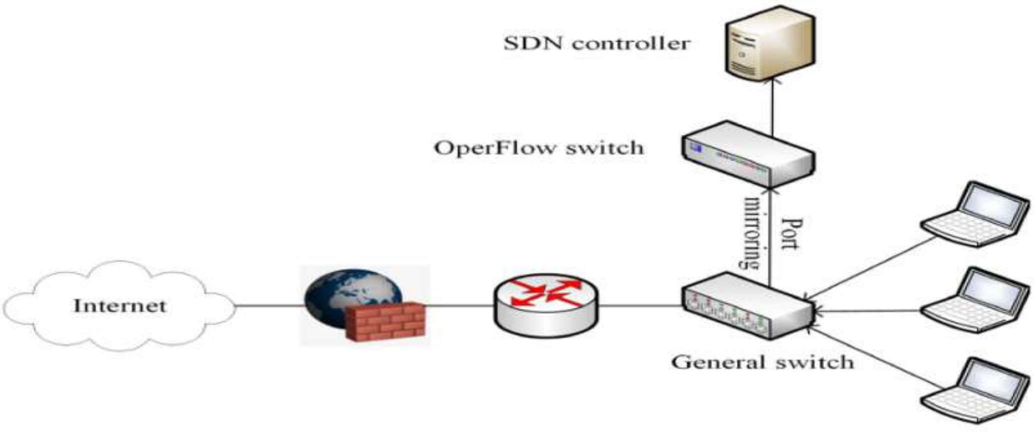
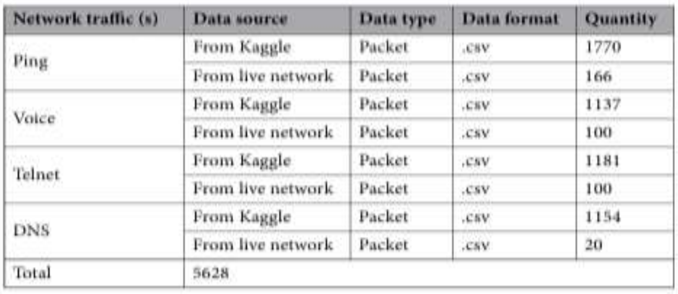


Fig. 3. Illustration of traffic collection.



Network traffic (s)	Data source	Data type	Data format	Quantity
Ping	From Kaggle	Packet	.csv	1770
	From live network	Packet	.csv	166
Voice	From Kaggle	Packet	.csv	1137
	From live network	Packet	.csv	100
Telnet	From Kaggle	Packet	.csv	1181
	From live network	Packet	.csv	100
DNS	From Kaggle	Packet	.csv	1154
	From live network	Packet	.csv	20
Total	5628			

Table 2. Number of network traffic data taken from each category.

Data Preprocessing

The preprocessing stage involved improving data quality and optimizing the dataset for accurate classification. Missing values were handled using mean imputation, and features showing linear monotonic trends were removed due to their low analytical value. Principal Component Analysis (PCA) was then applied to reduce feature redundancy and highlight the most informative components.

After preprocessing, the dataset contained 5628 instances and 13 attributes, with 12 features used as inputs and one as the class label. The data was well balanced across four traffic types: Ping (1936), Telnet (1281), Voice (1237), and DNS (1174) ensuring unbiased model training. QoS-based feature grouping was used, as these attributes are common across many applications and improve the adaptability and effectiveness of traffic classification.

Loading and cleaning data

CSV files were imported into Jupyter Notebook for preprocessing. Records containing NaN values, often due to abrupt script interruptions, were removed. Attributes like Forward/Reverse Packets and Bytes were excluded because their cumulative trends over time contributed little to classification. Data was collected from live network captures and Kaggle sources, categorized into Ping, Voice, Telnet, and DNS datasets, then combined and split 80/20 for training and testing, ensuring robust model evaluation.

Feature Selection

Redundant identifiers such as source/destination IPs, ports, and uniform protocol fields were removed, leaving primarily time-based and flow-statistical features. While the dataset size did not strictly require feature selection, this step reduces computational overhead and helps assess the impact of different feature combinations on classification accuracy. PCA-based evaluation for each model is discussed later.

Distributed Internet Traffic Generator

D-ITG simulated realistic traffic for Telnet, Voice (G.711), Ping, and DNS, supporting stochastic modelling of packet size and inter-departure time with multiple unidirectional flows. Ping traffic was generated via ICMP echo requests. Traffic flows were captured using a Python classifier integrated with the RYU controller and a monitoring script that updated Flow objects and exported CSV files. These CSVs were combined into a unified Pandas Data Frame, forming the final dataset for ML model training and testing.

Tools Used

Experiments were conducted on a Linux system with an Intel® Core™ i5-6100U, 4 GB RAM, and 500 GB HDD. Mininet emulated the SDN network, while Anaconda Python 3.7 and Jupyter Notebook were used for data preprocessing and analysis. ML models were developed with TensorFlow and Keras for efficient training and evaluation.

Framework Design

The overall system framework, illustrated in Fig. 4, integrates key SDN components such as the controller, Open vSwitch (OVS), and multiple end hosts. After the machine learning models are trained and validated on the prepared datasets, the framework is capable of classifying newly generated traffic by analyzing its flow characteristics. This design enables the system to recognize and categorize different traffic patterns effectively, ensuring accurate and efficient traffic identification within the SDN environment.

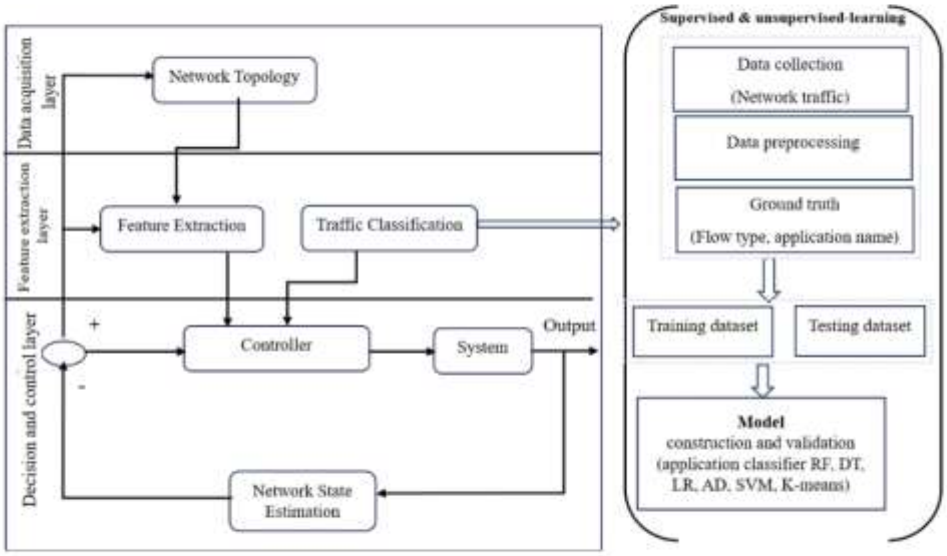

Data acquisition layer
Network Topology
Feature extraction layer
Feature Extraction
Traffic Classification
Decision and control layer
+
-
Controller
System
Output
Network State Estimation
Supervised & unsupervised learning
Data collection (Network traffic)
Data preprocessing
Ground truth (Flow type, application name)
Training dataset
Testing dataset
Model
construction and validation (application classifier RF, DT, LR, AD, SVM, K-means)

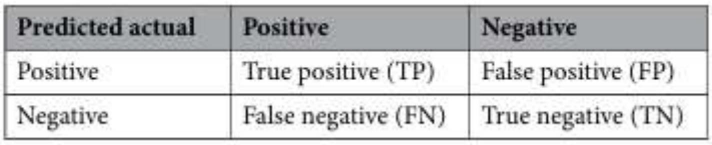
Fig. 4. Framework for traffic classification in software defined networks.

Performance Evaluation

To assess the effectiveness of the proposed model, several standard evaluation metrics were employed. Metrics such as **Accuracy, Precision, Recall, and F1-score** are commonly used in classification tasks and provide a comprehensive understanding of how well the model distinguishes between different traffic types. These measures help evaluate not just the correctness of predictions, but also the model's ability to identify each class reliably.

Confusion Matrix

A confusion matrix offers a clear visual representation of a classifier's prediction outcomes by comparing actual class labels with predicted ones. It summarizes the number of correct and incorrect predictions across all classes, making it an essential tool for evaluating model performance. Table 3 presents the structure of the confusion matrix.



Predicted actual	Positive	Negative
Positive	True positive (TP)	False positive (FP)
Negative	False negative (FN)	True negative (TN)

Table 3. Truth table of confusion matrix

The performance metrics are calculated as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

Results and Discussion

After training the proposed model, it was extensively tested to evaluate its ability to classify different types of network traffic. The evaluation included comparisons with other ML models, tests on multiple datasets, and real-time traffic experiments to mimic real operational conditions. Six algorithms: **RF, AdaBoost, SVM, DT, LR, and K-means** were used for benchmarking due to their effectiveness in identifying diverse traffic patterns. The results show that the proposed approach performs consistently well across all scenarios, accurately classifying complex traffic and demonstrating strong potential for practical SDN deployments with improved traffic management and decision-making.

Supervised learning

Logistic Regression

Logistic Regression (LR) was applied to predict categorical traffic classes based on the relationship between input features and class-dependent outcomes. The resulting decision boundaries, illustrated in Fig. 5, show that LR forms clear separations between categories. While Voice and Telnet traffic are easily distinguishable, Ping and DNS traffic exhibit greater overlap due to their similar characteristics. A confusion matrix (Fig. 6) was used to evaluate misclassification patterns by comparing predicted and actual class labels. Overall, the LR model demonstrated strong reliability, achieving an accuracy of **99.68%** on 5628 data instances.

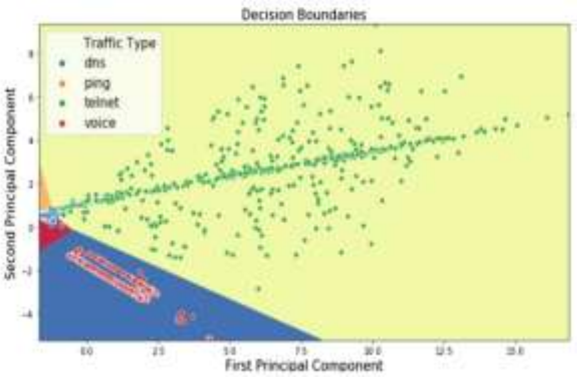


Fig. 5. Principal component analysis for logistic regression.

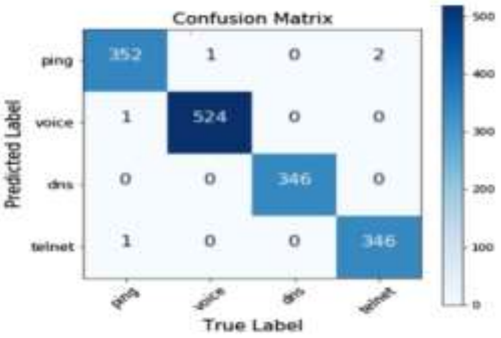
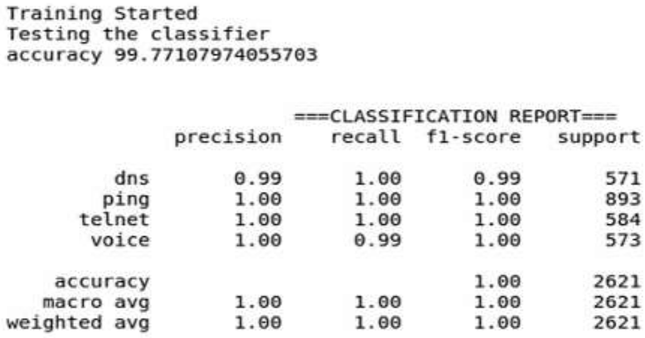


Fig. 6. Confusion matrix for logistic regression.

AdaBoost Classifier

AdaBoost, a boosting technique that combines multiple weak learners into a single powerful classifier, also performed exceptionally well. In our experiments, the algorithm achieved an accuracy exceeding **99.77%** (Fig. 7). Typically using shallow decision trees as base learners, AdaBoost iteratively adjusts instance weights to focus more on difficult-to-classify samples. PCA was used to transform correlated features into orthogonal components by extracting eigenvectors via singular value decomposition, as shown in Fig. 8. Although Telnet and Voice traffic were clearly separated in this PCA space, differentiating Ping from DNS remained challenging with only two components. The AdaBoost confusion matrix (Fig. 9) further highlights the model's minimal misclassifications, reflecting its overall effectiveness.



```
Training Started
Testing the classifier
accuracy 99.77107974055703

                    ===CLASSIFICATION REPORT===
              precision    recall  f1-score   support

         dns       0.99      1.00      0.99       571
        ping       1.00      1.00      1.00       893
      telnet       1.00      1.00      1.00       584
       voice       1.00      0.99      1.00       573

    accuracy                           1.00      2621
   macro avg       1.00      1.00      1.00      2621
weighted avg       1.00      1.00      1.00      2621
```

Fig. 7. AdaBoost classifier with different evaluation metrics.

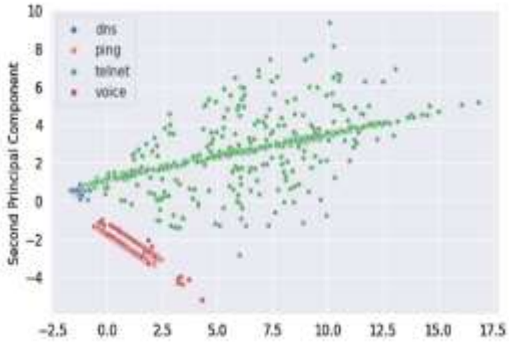


Fig. 8. Principal component analysis using AdaBoost classifier.

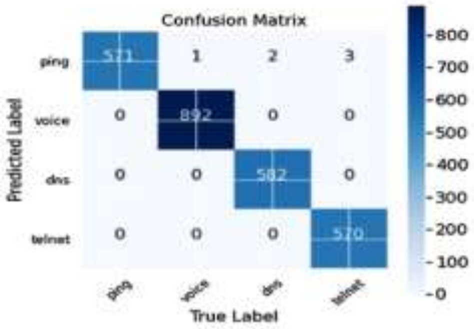
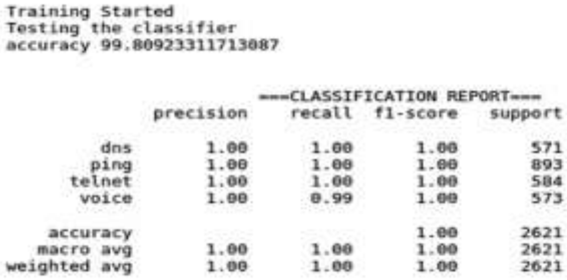


Fig. 9. Confusion matrix for AdaBoost classifier.

Decision Tree Classifier

The Decision Tree (DT) classifier categorizes traffic by recursively splitting data according to feature values. Categorical attributes are divided by distinct values, while continuous features are separated by threshold-based splits. The classifier's evaluation metrics including accuracy, precision, recall, and F1-score are shown in Fig. 10. Applying PCA prior to training (Fig. 11) helped reduce redundancy and emphasize directions of highest variance, improving both efficiency and classification quality. Although the DT model effectively distinguished Telnet, Voice, and DNS traffic, it encountered difficulty classifying Ping traffic when limited to a single component. The confusion matrix in Fig. 12 provides further insight into these misclassifications. The DT classifier achieved the highest performance among all models, with an accuracy of **99.81%** on the dataset.



```
Training Started
Testing the classifier
accuracy 99.80923311713087

                    ===CLASSIFICATION REPORT===
              precision    recall  f1-score   support

         dns       1.00      1.00      1.00       571
        ping       1.00      1.00      1.00       893
      telnet       1.00      1.00      1.00       584
       voice       1.00      0.99      1.00       573

    accuracy                           1.00      2621
   macro avg       1.00      1.00      1.00      2621
weighted avg       1.00      1.00      1.00      2621
```

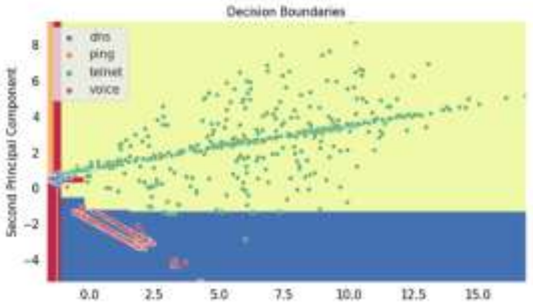


Fig. 10. Decision tree classifier with different evaluation metrics.

Fig. 11. Principal component analysis for decision tree

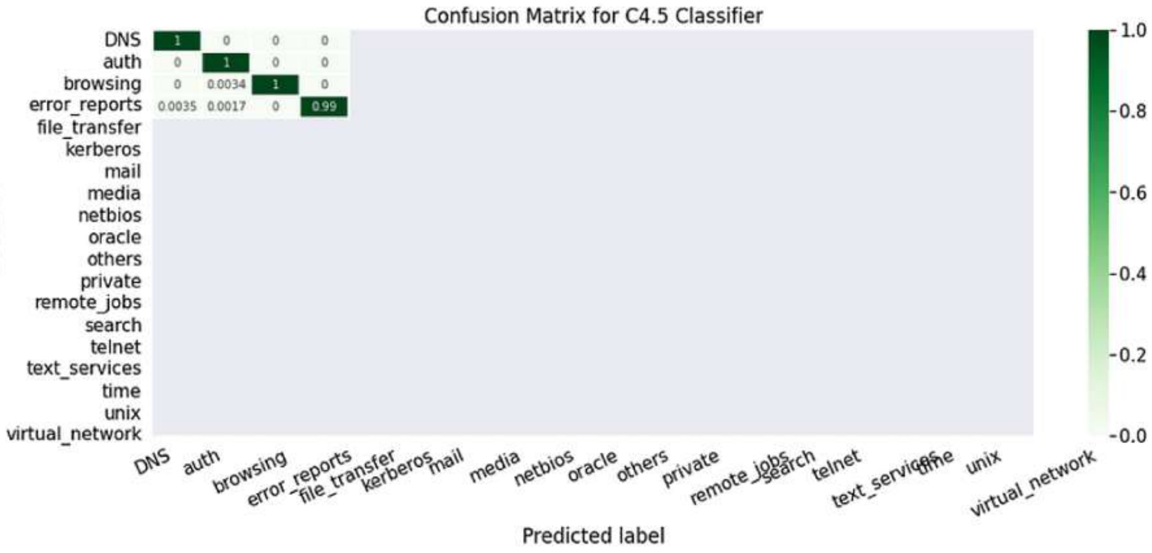
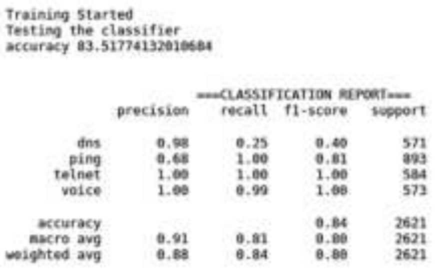


Fig. 12. Confusion matrix for decision tree classifier.

Support Vector Machine

SVM classifies network traffic by finding an optimal separating hyperplane, with its performance shown in the confusion matrix (Fig. 13). After normalization and PCA (Fig. 14), Telnet, Voice, and DNS traffic appear clearly separable, while Ping overlaps with other classes, making it harder to classify. The final confusion matrix (Fig. 15) confirms that most errors occur in identifying Ping traffic.



```
Training Started
Testing the classifier
accuracy 83.51774132010684

                 ===CLASSIFICATION REPORT===
              precision    recall  f1-score   support

         dns       0.98      0.25      0.40       571
        ping       0.68      1.00      0.81       893
      telnet       1.00      1.00      1.00       584
       voice       1.00      0.99      1.00       573

    accuracy                           0.84      2621
   macro avg       0.91      0.81      0.80      2621
weighted avg       0.88      0.84      0.80      2621
```

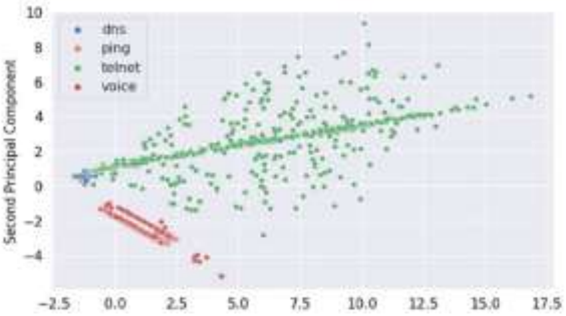


Fig. 13. Support vector machine classifier with different evaluation metrics.

Fig. 14. Support vector machine principal component analysis.

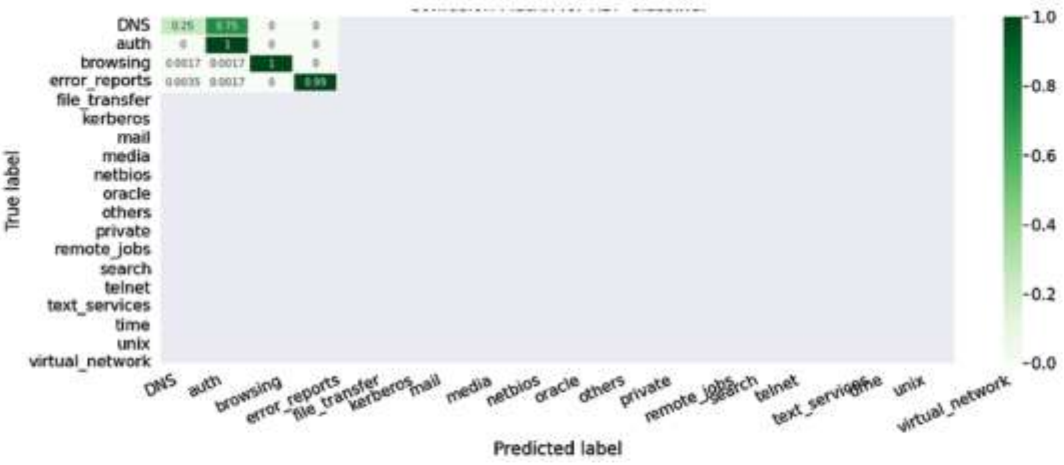


Fig. 15. Confusion matrix for support vector machine.

The SVM-based model achieved 83.51% accuracy with a dataset comprising 5628 instances.

Random Forest Classifier

The Random Forest (RF) classifier operates by aggregating predictions from multiple decision trees, each trained on randomly sampled subsets of the dataset and feature space. This ensemble approach enhances robustness and reduces overfitting. The confusion matrix for the RF model is shown in Fig. 16, and the PCA visualization in Fig. 17 highlights component-based separability. The classifier easily identified Telnet, Voice, and Ping traffic, though DNS traffic remained relatively difficult to separate with minimal PCA components. The RF model achieved a high accuracy of **99.74%**, demonstrating strong classification capability comparable to DT and AdaBoost.

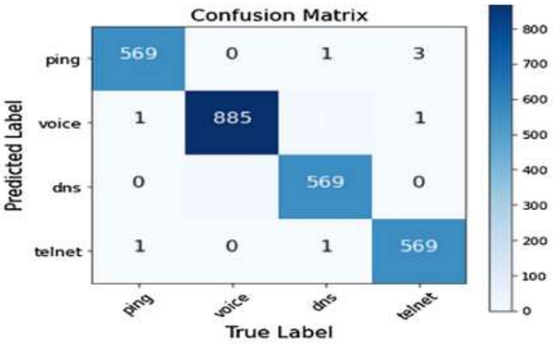


Fig. 16. Confusion matrix for random forest.

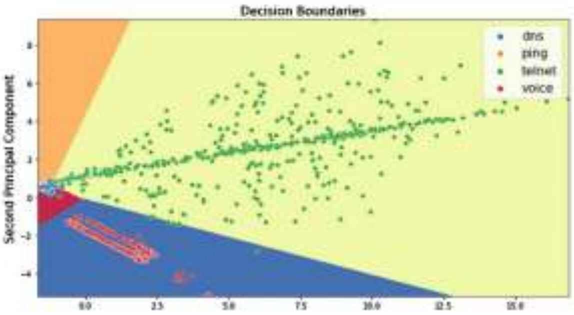


Fig. 17. Random forest principal component analysis.

Unsupervised Learning: K-Means Clustering

K-Means clustering, an unsupervised ML technique, groups data points by assigning each to the nearest centroid. In this study, it formed four clusters for the traffic types. Figure 18 visualizes the centroids in a 12-dimensional feature space, with shading indicating feature magnitude. PCA reduced the dataset to 2 dimensions for clearer visualization (Figure 19), revealing significant cluster overlap and explaining the low accuracy (~30%). The confusion matrix (Figure 20) shows mislabelled Telnet flows and fragmented Voice traffic, reflecting K-Means' limitations with non-spherical, linearly structured data.

Among all algorithms, the Decision Tree (DT) achieved the best results in accuracy (99.81%) and processing efficiency, outperforming LR, SVM, and ensemble methods like AdaBoost and RF. Table 4 summarizes model performance, and Table 5 compares our results with existing studies, demonstrating superior classification accuracy.

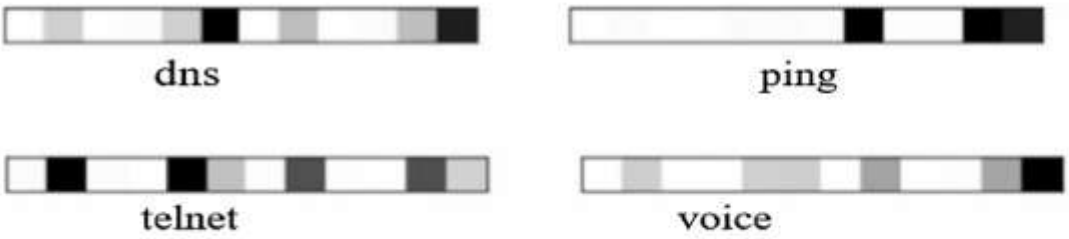


Fig. 18. Visualization of cluster centroids.

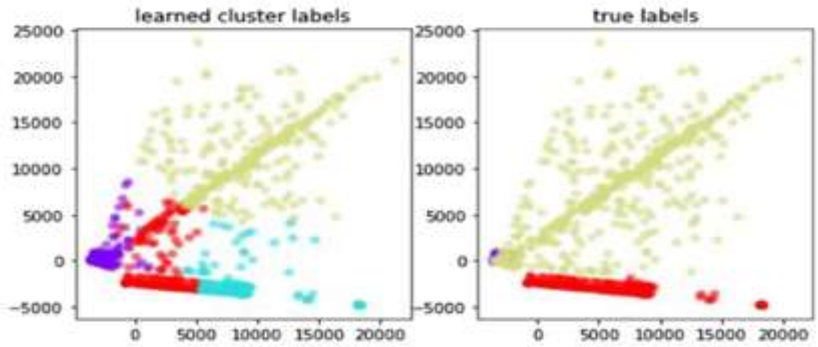


Fig. 19. Visualizing K-means clustering performance.

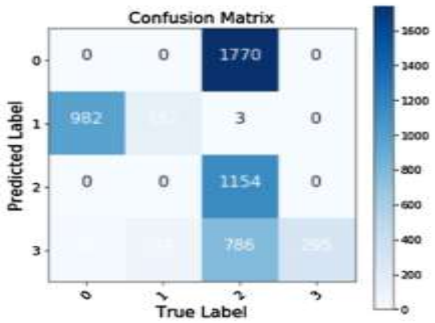
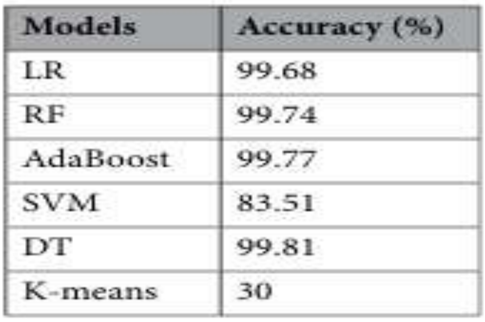
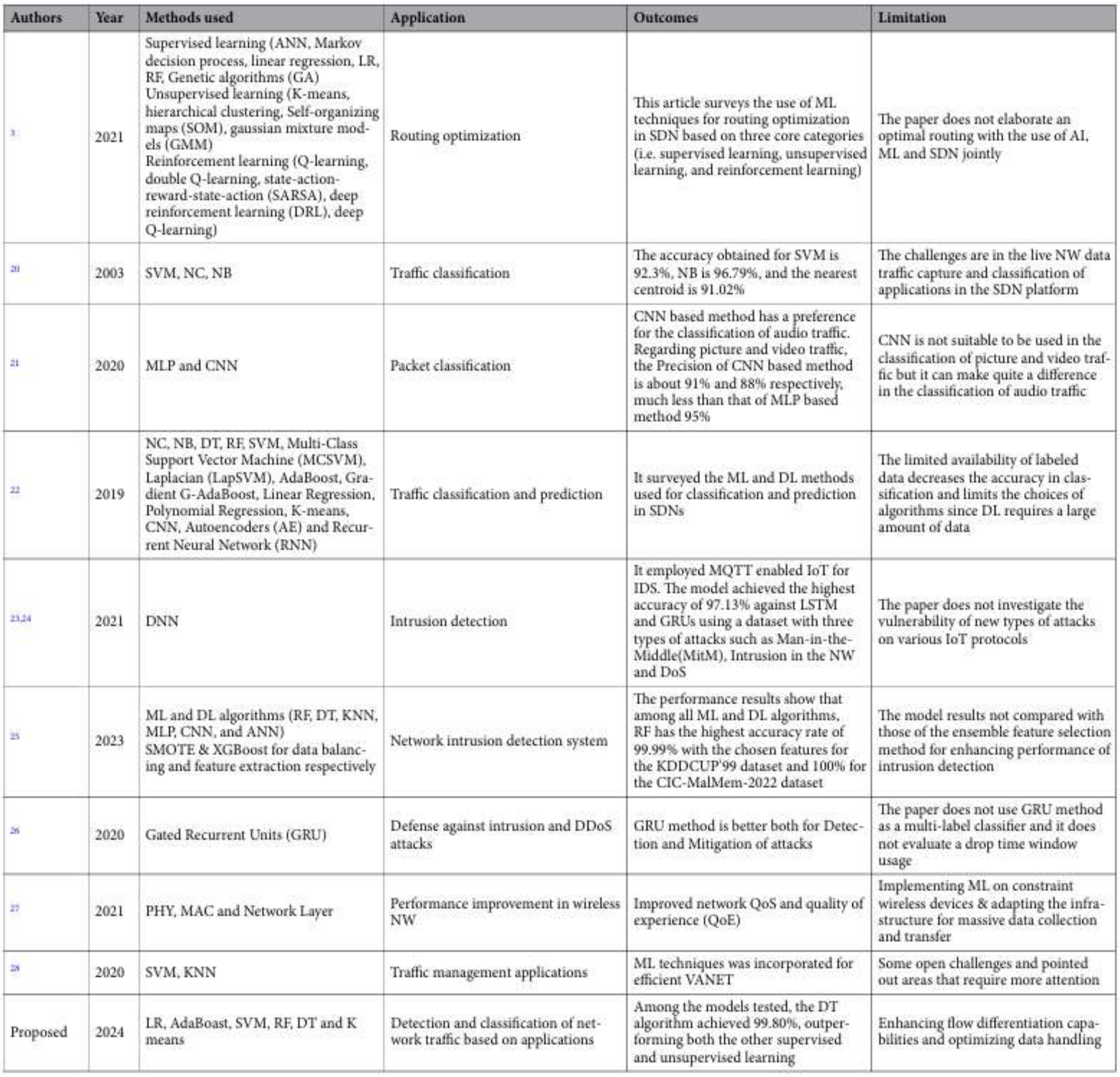


Fig. 20. Confusion matrix of K-means clustering.



Models	Accuracy (%)
LR	99.68
RF	99.74
AdaBoost	99.77
SVM	83.51
DT	99.81
K-means	30

Table 4. Performance evaluation of the machine learning models.



Authors	Year	Methods used	Application	Outcomes	Limitation
3	2021	Supervised learning (ANN, Markov decision process, linear regression, LR, RF, Genetic algorithms (GA) Unsupervised learning (K-means, hierarchical clustering, Self-organizing maps (SOM), gaussian mixture models (GMM) Reinforcement learning (Q-learning, double Q-learning, state-action-reward-state-action (SARSA), deep reinforcement learning (DRL), deep Q-learning)	Routing optimization	This article surveys the use of ML techniques for routing optimization in SDN based on three core categories (i.e. supervised learning, unsupervised learning, and reinforcement learning)	The paper does not elaborate an optimal routing with the use of AI, ML and SDN jointly
20	2003	SVM, NC, NB	Traffic classification	The accuracy obtained for SVM is 92.3%, NB is 96.79%, and the nearest centroid is 91.02%	The challenges are in the live NW data traffic capture and classification of applications in the SDN platform
21	2020	MLP and CNN	Packet classification	CNN based method has a preference for the classification of audio traffic. Regarding picture and video traffic, the Precision of CNN based method is about 91% and 88% respectively, much less than that of MLP based method 95%	CNN is not suitable to be used in the classification of picture and video traffic but it can make quite a difference in the classification of audio traffic
22	2019	NC, NB, DT, RF, SVM, Multi-Class Support Vector Machine (MCSVM), Laplacian (LapSVM), AdaBoost, Gradient G-AdaBoost, Linear Regression, Polynomial Regression, K-means, CNN, Autoencoders (AE) and Recurrent Neural Network (RNN)	Traffic classification and prediction	It surveyed the ML and DL methods used for classification and prediction in SDNs	The limited availability of labeled data decreases the accuracy in classification and limits the choices of algorithms since DL requires a large amount of data
23,24	2021	DNN	Intrusion detection	It employed MQTT enabled IoT for IDS. The model achieved the highest accuracy of 97.13% against LSTM and GRUs using a dataset with three types of attacks such as Man-in-the-Middle(MitM), Intrusion in the NW and DoS	The paper does not investigate the vulnerability of new types of attacks on various IoT protocols
25	2023	ML and DL algorithms (RF, DT, KNN, MLP, CNN, and ANN) SMOTE & XGBoost for data balancing and feature extraction respectively	Network intrusion detection system	The performance results show that among all ML and DL algorithms, RF has the highest accuracy rate of 99.99% with the chosen features for the KDDCUP'99 dataset and 100% for the CIC-MalMem-2022 dataset	The model results not compared with those of the ensemble feature selection method for enhancing performance of intrusion detection
26	2020	Gated Recurrent Units (GRU)	Defense against intrusion and DDoS attacks	GRU method is better both for Detection and Mitigation of attacks	The paper does not use GRU method as a multi-label classifier and it does not evaluate a drop time window usage
27	2021	PHY, MAC and Network Layer	Performance improvement in wireless NW	Improved network QoS and quality of experience (QoE)	Implementing ML on constraint wireless devices & adapting the infrastructure for massive data collection and transfer
28	2020	SVM, KNN	Traffic management applications	ML techniques was incorporated for efficient VANET	Some open challenges and pointed out areas that require more attention
Proposed	2024	LR, AdaBoast, SVM, RF, DT and K means	Detection and classification of network traffic based on applications	Among the models tested, the DT algorithm achieved 99.80%, outperforming both the other supervised and unsupervised learning	Enhancing flow differentiation capabilities and optimizing data handling

Table 5. Comparison of proposed approach with related works.

Conclusion

Traditional traffic classification methods, such as port-based identification and payload inspection, are increasingly unreliable due to dynamic behaviours and encrypted traffic. Accurate classification of both offline and real-time traffic, including Ping, Telnet, DNS, and Voice, is essential for effective network management and QoS.

This study combines Software-Defined Networking (SDN) with Machine Learning models Decision Tree, Random Forest, SVM, AdaBoost, Logistic Regression, and K-Means to classify application-level traffic. Using Mininet, network topologies were emulated, flow data generated, and models evaluated across multiple performance metrics. The Decision Tree classifier achieved the highest accuracy, demonstrating strong performance in handling diverse traffic patterns in real-time. Integrating ML with SDN enhances traffic classification, network efficiency, and adaptive traffic management.

For future work, D-ITG can be extended to simulate video and gaming traffic, reflecting the increasing share of multimedia applications.

Data Availability: Datasets generated in this study are available from the corresponding author upon reasonable request.
Code Availability: Source code is accessible on GitHub: https://github.com/melesewmossie

References

[1] A. O. Salau and M. M. Beyene, "Software defined networking-based network traffic classification using machine learning techniques," *Scientific Reports*, vol. 14, no. 20060, 2024.

[2] S. H. Haji *et al.*, "Comparison of software defined networking with traditional networking," *Asian Journal of Research in Computer Science*, 2021.

[3] D. Kafetzis, S. Vassilaras, G. Vardoulias, and I. Koutsopoulos, "Software-defined networking meets software-defined radio in mobile ad hoc networks: State of the art and future directions," *IEEE Access*, vol. 10, pp. 9989–10014, 2022.

[4] R. Amin *et al.*, "A survey on machine learning techniques for routing optimization in SDN," *IEEE Access*, vol. 9, pp. 104582–104611, 2021.

[5] R. Nandhini and S. V. E. Sonia, "A survey and comparison of SDN-based traffic management techniques," *Asian J. Appl. Sci. Technol.*, vol. 4, no. 3, pp. 10–18, 2020.

[6] R. H. Serag *et al.*, "Machine-learning-based traffic classification in software-defined networks," *Electronics*, vol. 13, no. 6, p. 1108, 2024.

[7] Ö. Tonkal and H. Polat, "Traffic classification and comparative analysis with machine learning algorithms in software defined networks," *Gazi Univ. J. Sci. Part C*, vol. 9, no. 1, pp. 71–83, 2021.

[8] S. K. Keshari, V. Kansal, and S. Kumar, "A systematic review of quality of services (QoS) in software defined networking (SDN)," *Wireless Personal Communications*, vol. 116, no. 3, pp. 2593–2614, 2021.

[9] P. Amaral *et al.*, "Machine learning in software defined networks: Data collection and traffic classification," in *IEEE Int. Conf. on Network Protocols (ICNP)*, 2016, pp. 1–5.

[10] S. Mahgoub, M. Ashour, M. Yakout, and E. AbdElhalim, "Traffic classification in software defined networks based on machine learning algorithms," *Int. J. Telecommunications*, vol. 4, no. 1, pp. 1–19, 2024.

[11] A. O. Salau and T. K. Yesufu, "A probabilistic approach to time allocation for intersecting traffic routes," in *Advances in Intelligent Systems and Computing*, vol. 1124, pp. 151–164, Springer, 2020.

[12] Y. Zhao *et al.*, "A survey of networking applications applying the software defined networking concept based on machine learning," *IEEE Access*, vol. 7, pp. 95397–95417, 2019.

[13] W. Jiang, "Graph-based deep learning for communication networks: A survey," *Computer Communications*, vol. 185, pp. 40–54, 2022.

[14] L.-H. Chang, T.-H. Lee, H.-C. Chu, and C.-W. Su, "Application-based online traffic classification with deep learning models on SDN networks," *Advances in Technology and Innovation*, 2020.

[15] P. Wang, Z. Wang, F. Ye, and X. Chen, "ByteSGAN: A semi-supervised GAN for encrypted traffic classification in SDN edge gateway," *Computer Networks*, vol. 200, p. 108535, 2021.

[16] Y. Zion, D. A. Dvir, and D. Ofir, "Classification and enrichment of encrypted traffic using machine learning algorithms," unpublished.

[17] B. Ng, J. Bakker, W. K. G. Seah, and A. Pekar, "Traffic classification with machine learning in a live network," 2019.

[18] N. M. Yungaicela-Naula, C. Vargas-Rosales, and J. A. Perez-Diaz, "SDN-based architecture for DDoS attack detection using machine and deep learning," *IEEE Access*, vol. 9, pp. 108495–108512, 2021.

[19] H. Gordon, C. Batula, B. Tushir, B. Dezfouli, and Y. Liu, "Securing smart homes via software-defined networking and low-cost traffic classification," *arXiv preprint*, arXiv:2104.00296, 2021.

[20] A. O. Salau, *Development of a Technique for Simulating Traffic Congestion*, M.Sc. Thesis, Obafemi Awolowo Univ., Nigeria, 2015.

[21] F. G. Deriba, A. O. Salau, S. H. Mohammed, T. M. Kassa, and W. B. Demilie, "A compressive framework using machine learning approaches for SQL injection attacks," *Przeglad Elektrotechniczny*, vol. 7, no. 1, pp. 181–187, 2022.

[22] R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno, "Machine learning algorithms to detect DDoS attacks in SDN," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 16, p. e5402, 2020.

[23] A. R. Mohammed, S. A. Mohammed, and S. Shirmohammadi, "Machine learning and deep learning based traffic classification and prediction in SDN," in *IEEE Int. Symp. on Measurements & Networking (M&N)*, 2019, pp. 1–6.

[24] M. A. Khan *et al.*, "A deep learning-based intrusion detection system for MQTT-enabled IoT," *Sensors*, vol. 21, no. 21, p. 7016, 2021.

[25] T. E. Ali, Y.-W. Chong, and S. Manickam, "Machine learning techniques to detect a DDoS attack in SDN: A systematic review," *Applied Sciences*, vol. 13, no. 5, p. 3183, 2023.

[26] M. A. Talukder *et al.*, "A dependable hybrid machine learning model for network intrusion detection," *Journal of Information Security and Applications*, vol. 72, p. 103405, 2023.

[27] M. V. O. Assis, L. F. Carvalho, J. Lloret, and M. L. Proença, "A GRU deep learning system against attacks in SDN," *Journal of Network and Computer Applications*, vol. 177, p. 102942, 2021.

[28] M. Kulin, T. Kazaz, E. De Poorter, and I. Moerman, "A survey on machine learning-based performance improvement of wireless networks: PHY, MAC and network layer," *Electronics*, vol. 10, no. 3, p. 318, 2021.

[29] S. Khatri *et al.*, "Machine learning models for VANET-based traffic management: Issues and challenges," *Peer-to-Peer Networking and Applications*, vol. 14, no. 3, pp. 1778–1805, 2021.

[30] A. Malik, R. De Frein, M. Al-Zeyadi, and J. Andreu-Perez, "Intelligent SDN traffic classification using deep learning: Deep-SDN," in *ICCCI*, 2020, pp. 184–189.

[31] M. P. J. Kuranage, K. Piamrat, and S. Hamma, "Network traffic classification using machine learning for SDN," in *Machine Learning for Networking (LNCS 12081)*, pp. 28–39, 2020.

[32] A. Y. Eshetu, E. A. Mohammed, and A. O. Salau, "Cybersecurity vulnerabilities and solutions in Ethiopian university websites," *Journal of Big Data*, vol. 11, p. 118, 2024.