scientific reports

Check for updates

OPEN

Software defined networking based network traffic classification using machine learning techniques

Ayodeji Olalekan Salau[1,3]✉ & Melesew Mossie Beyene[2]

The classification of network traffic has become increasingly crucial due to the rapid growth in the number of internet users. Conventional approaches, such as identifying traffic based on port numbers and payload inspection are becoming ineffective due to the dynamic and encrypted nature of modern network traffic. A number of researchers have implemented Software Defined Networking (SDN) based traffic classification using Machine Learning (ML) and Deep Learning (DL) models. However, the studies had various limitations such as encrypted traffic detection, payload inspection, poor detection accuracy, and challenges with testing models both in offline and real-time traffic modes. ML models together with SDN are adopted nowadays to enhance classification performance. In this paper, both supervised (Logistic Regression, Decision Tree, Random Forest, AdaBoost, and Support Vector Machine) and unsupervised (K-means clustering) ML models were used to classify Domain Name System (DNS), Telnet, Ping, and Voice traffic flows simulated using the Distributed Internet Traffic Generator (D-ITG) tool. The use of this tool effectively manages and classifies traffic types based on their application. The study discussed the dataset used, model selection, implementation of the model, and implementation techniques (such as pre-processing, feature extraction, ML algorithm, and model evaluation metrics). The proposed model in SDN was implemented in Mininet for designing the network architecture and generating network traffic. Anaconda Python environment was utilized for traffic classification using various ML techniques. Among the models tested, the Decision Tree supervised learning achieved the highest accuracy of 99.81%, outperforming other supervised and unsupervised learning algorithms. These results indicate that the integration of ML with SDN provides an efficient classification method for identifying and accurately classifying both offline and real-time network traffic, enhanced quality of service (QoS), detection of encrypted packets, deep packet inspection and management.

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

Abbreviations

ANN	Artificial neural network
CNN	Convolutional neural network
DITG	Distributed internet traffic generator
DL	Deep learning
DNN	Deep neural network
DoS	Denial of services
DPI	Deep packet inspection
DT	Decision tree
LR	Logistic regression
ML	Machine learning
MLP	Multilayer perceptron
NB	Naive Bayes
NC	Nearest centroid
PCA	Principal component analysis

[1]Department of Electrical/Electronics and Computer Engineering, Afe Babalola University, Ado-Ekiti, Nigeria. [2]Department of Computer Science, Institute of Technology, Debre Markos University, Debre Markos, Ethiopia. [3]Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, Tamil Nadu, India. ✉email: ayodejisalau98@gmail.com

QoS	Quality of service
RF	Random forest
SDN	Software defined networking
SVM	Support vector machine
TC	Traffic classification

The classification of traffic flow in today's Internet Protocol (IP) network has become an important research area, especially with the recent adoption of Machine Learning (ML) techniques and Software Defined Networking (SDN) principles. SDN redefines network architecture by separating the control plane from the data plane[1,2], thereby departing from the OSI model hierarchy[3–5]. SDN integrates controllers, switches and hosts, enabling centralized controllers to optimize network management and packet routing, thereby overcoming conventional network limitations. Classification methods enhance network service quality by efficiently allocating resources and detecting anomalies, though conventional rule-based approaches struggle with dynamic ports and unknown applications[6]. Properly classifying network traffic using advanced classification metrics has become increasingly important as the number of internet users continues to grow rapidly. Effective traffic classification (TC) is essential for managing and optimizing network performance, ensuring QoS, and enhancing cybersecurity[7]. With the dynamic and encrypted nature of modern network traffic, conventional methods such as port-based and payload inspection[8], are no longer efficient. Deep Packet Inspection (DPI) is effective at identifying traffic patterns but faces significant challenges when dealing with encrypted content due to high costs and practical limitations. As encryption becomes more prevalent, DPI effectiveness decreases, prompting the use of ML techniques for more accurate and efficient traffic classification[4,6]. The existing studies have many limitations: such as traffic detection with encrypted traffic, payload inspection in dynamic traffic data, poor detection, testing various offline and real-time traffic data. In addition, most of the existing studies have not implemented traffic detection in both offline and real-time networks, provides inaccurate traffic classification for real-time data, and poor QoS delivery. In this paper, we compared the proposed model with other models having different datasets and its performance using real-time traffic data to mimic real-world scenarios.

This paper integrates ML techniques with SDNs to classify Ping, Telnet, Voice, and DNS traffic data using ML-based flow classification. Processing large datasets from extensive internet traffic poses significant challenges, even for experienced professionals with advanced tools. Employing ML for TC and clustering is crucial for pinpointing network hotspots and bottlenecks. By using bandwidth and QoS metrics for classification, Traffic Engineering (TE) can adjust flow paths and allocate virtual resources in the network. This helps to boost performance and enhances security. Unlike DPI, ML flow classification focuses on extracting flow characteristics such as packet size history, source IP addresses, protocol types and flow arrival times, without the need to inspect packet payloads. This approach renders ML particularly suitable for encrypted traffic analysis while offering advantages such as computational efficiency over conventional SDN techniques[1]. ML algorithms are pivotal in minimizing packet loss and achieving highly precise classification outcomes[9]. The current study utilizes a combination of supervised and unsupervised ML algorithms, leveraging tools like Scikit-learn to enhance classification accuracy. These enhancements support a variety of applications including QoS optimization, dynamic access control, lawful inspection, and streamlined network management practices[10]. The proposed system integrates ML and SDN to mitigate critical challenges in accurately classifying dynamic and encrypted traffic, which is fundamental for advancing network reliability and performance in contemporary IP networks[11].

Literature review

Efficient traffic management ensures precise allocation of resources, which is crucial for demanding tasks like server-based code compilation. The authors of[6] presented a K-NN approach which achieved a 99.4% accuracy in classifying DNS, Telnet, Ping, and Voice traffic with real-time data within SDNs. Additionally, SDN clustering methods such as Simulated Annealing have proven effective in optimizing network lifespan and managing heterogeneous sensor data traffic efficiently[4]. According to Ref.[11], the integration of SDN with ML is crucial to analyze classical ML models and classify SDN-based network applications. This integration explores upcoming ML advancements within SDN, bridging artificial intelligence (AI), big data and networking disciplines. Graph-based deep learning (DL) and graph neural network (GNN) in Ref.[12] survey were used and provided a state-of-the-art performance in addressing diverse challenges across communication networks. However, the survey did not mention the use of graph-based DL and GNN on larger networks and also their performance was not evaluated with performance metrics.

The authors in Ref.[13] presented the application of MLP, CNN and SAE models within an SDN framework for traffic classification. The models achieved competitive accuracies of 87.167%, 87.208%, and 87.079% respectively. Performance metrics such as accuracy, precision, recall, and F1-score were comprehensively employed to gauge their efficacy in traffic classification, which inturn demonstrated promising results for practical SDN deployments. ByteSGAN[14], is a Generative Adversarial Network (GAN)-based Semi-Supervised Learning method integrated into the SDN Edge Gateway for detailed traffic classification, which aims to optimize network resource utilization. It leverages a small set of labeled traffic samples and a larger pool of unlabeled samples by adjusting the structure and loss function of the GAN discriminator network for semi-supervised learning. However, the current implementation of GAN does not specifically address data imbalance issues in traffic classification. User privacy breaches can occur highly (through direct data theft) or passively (via DPI for profiling), now complicated by widespread encryption (HTTPS and QUIC, covering > 80% of internet traffic since 2017). The increasing diversity of network applications necessitates precise traffic identification[15]. The study in Ref.[15] introduced new ML features, evaluated on 28,000 time-frame samples, and achieved robust classification of mobile and desktop traffic (81%) and application actions (94% and 93% accuracy respectively).

The authors in Ref.[8] deployed an architecture for an enterprise network to gather traffic data. The authors explored available datasets and applied ML techniques for TC, achieving high accuracy with supervised learning methods. Rehab et al.[5] presented the integration of SDN with ML to improve network security through proactive threat detection. Challenges in SDN-ML QoS integration and scalability in large-scale implementations were also addressed. The authors in Ref.[16] deployed traffic classifiers in a live network for DDoS detection[17] using ML algorithms. Their findings underscore the challenges of adapting ML algorithms to dynamic SDN environments, where network interactions between switches and controllers play a crucial role in performance outcomes. Sherif et al.[9] presented ML algorithms for detecting ad classifying network traffic using 15 features of simulated traffic flows like (WWW, DNS, FTP, ICMP, P2P, and VOIP) on a SDN and created a real-time dataset. The DT model achieved the highest accuracy at 99.8%, making it effective for SDN traffic classification with minimal features. As described in[18], the authors presented a securing smart home IoT device which prevents DDoS attacks using VLAN-based network isolation managed by an SDN controller. It uses lightweight flow-based features for efficient data collection, including ICMP, TCP, and UDP protocol percentages, packet count and size, and IP diversity ratio. ML models such as KNN, RF, and SVM were employed to classify IoT devices and detect DDoS attacks based on TCP-SYN, UDP, and ICMP protocols. Evaluation on an OpenVSwitch and Faucet SDN controller setup using flow traces from IoT devices shows an average accuracy of 97% for device classification and 98% for DDoS detection, with an average latency of 1.18 ms. The authors in Ref.[19] presented a traffic model which is used to categorize traffic in the network path, while monte carlo method was used to assign traffic routes to packet. Payload-based methods compare packet payloads against predefined patterns but are ineffective against encrypted traffic[14]. Machine learning and statistical approaches, as discussed in[14], offer robust solutions by extracting features from data to train models effectively. Previous researches have extensively leveraged DL and ML methodologies[29] to detect and categorize traffic based on application types, with a primary focus on enhancing QoS, performance improvement, packet inspection for encrypted data, port-based detection, routing optimization, attack detection and mitigating security risks. This paper proposed integrating SDNs with ML techniques to enhance TC for efficient real-time traffic analysis and detection.

Methodology

The proposed model adopts a systematic approach which starts with a comprehensive data collection for training and validation. Preprocessing was used to enhance data quality, followed by structured training and testing phases using supervised and unsupervised ML techniques. This is aimed at optimizing the classification accuracy for various traffic types (e.g., ping, telnet, voice, DNS) and to ensure robust performance in real-world scenarios. This further aims to improve network management, enhance Quality of Service (QoS), and bolster network security through precise traffic analysis.

System architecture

This paper presents a model designed to detect and classify network traffic using ML algorithms in conjunction with SDNs. The proposed system architecture, depicted in Fig. 1, consists of three primary components: preprocessing, feature extraction, and classification. Data preprocessing encompasses procedures such as standardizing

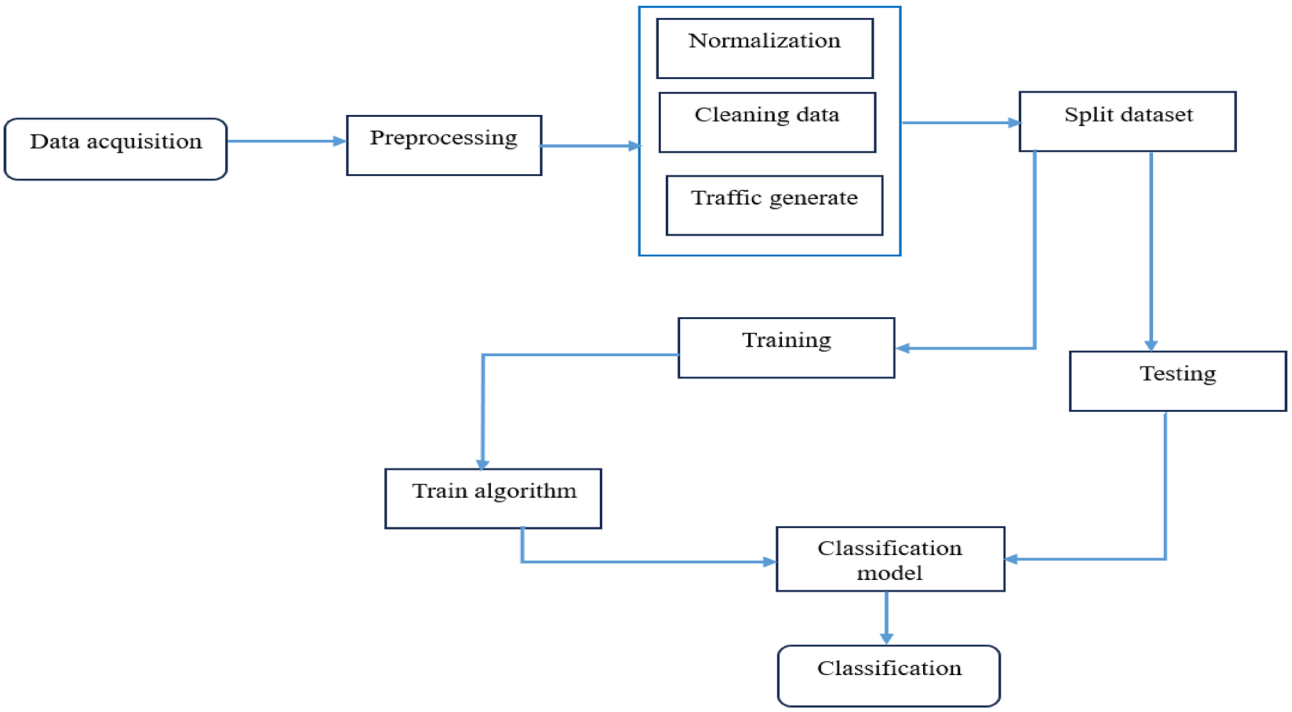

Fig. 1. System architecture of the proposed model.

data formats, handling missing values and applying data filtering techniques. Thereafter, feature extraction and classification stages were performed.

Software-defined network environment and dataset generation

Figure 2 illustrates the basic network topology in VirtualBox-VM environment with five virtual machines: a controller, a Layer-2 switch, and three hosts. The study highlights Mininet's superiority in accurately simulating real-world network conditions. Unlike Mininet, which supports multiple virtual network elements and interactions, single VM setups lack scalability and realism for effectively simulating large, interconnected networks. This preference ensures more reliable testing and validation of network configurations and models.

Upon enabling SDN, it becomes imperative for the controller to maintain real-time visibility of packet movements among client computers via the switch. This necessitates continuous monitoring of flow details, as outlined in Table 1, which was updated every second through a dedicated Python script.

Data acquisition

We have used the D-ITG application to generate traffic flow data for training machine learning models[30]. D-ITG accurately simulates both IPv4 and IPv6 traffic, mimicking various internet applications like Ping, Telnet, DNS, and Voice (using the G.711 codec)[31]. Due to the sensitive nature of network traffic, we believe most network traffic data will be encrypted in the future for security purposes. Hence, the dataset employed for TC was encrypted. Since the data capturing was done in a controlled environment, the applications were executed one after another, so the traffic was labeled. We simulated specific traffic flows between client machines and then employed a traffic classifier script with the RYU controller and a monitoring Python file as depicted in Fig. 3 to collect and classify the traffic data.

Data collection for the flow object is managed by a monitoring script that systematically gathers and processes traffic features, exporting them into CSV files. These files contain essential characteristics of traffic types such as Ping, Telnet, Voice (G.711), and DNS. Aggregated into a unified dataset, these CSV files serve as the core data for training and testing the machine learning models. Table 2 provides the instance counts for each traffic class—Ping, Telnet, Voice, and DNS used in the classification tasks employing different ML algorithms.

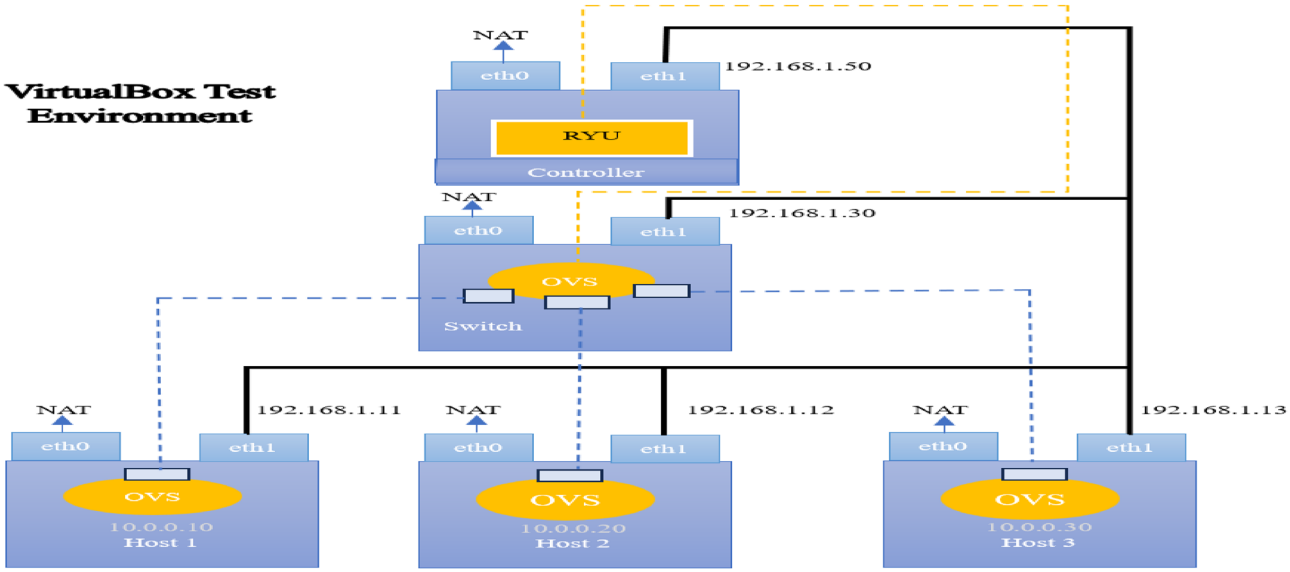

Fig. 2. Simulation network topology.

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.

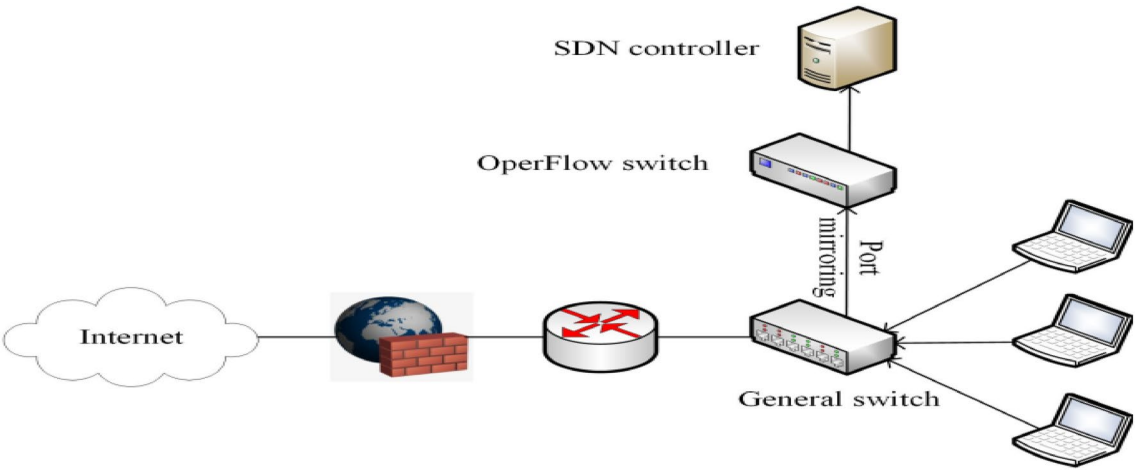

Fig. 3. Illustration of traffic collection.

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 of the acquired dataset involved several critical steps to ensure data integrity and optimal classification performance. Missing data rows were handled using mean value imputation to maintain dataset completeness. Features showing linearly increasing trends, which could potentially bias analysis, were removed to enhance the dataset suitability for ML algorithms. Principal Component Analysis (PCA) was subsequently applied to reduce feature correlation, thereby improving the dataset's efficiency in capturing essential patterns without redundancy. As a result, the dataset was refined to consist of 5628 instances and 13 attributes, where 12 features were used for classification and 1 for the class label.

The dataset exhibited a balanced distribution across four distinct traffic types: Ping (1936), Telnet (1281), Voice (1237), and DNS (1174) packets. This balance facilitated robust training and evaluation of models. QoS class-based approaches were prioritized due to their adaptability across various internet applications and their ability to leverage common characteristics within QoS categories, ultimately aimed at optimizing network performance and reliability.

Loading and cleaning data

The collected CSV data was imported into a Jupyter notebook for preprocessing. Rows containing NaN values, often resulting from abrupt script termination, are dropped to ensure data integrity. Features such as Forward Packets, Forward Bytes, Reverse Packets, and Reverse Bytes, which exhibit linear increases and lack meaningful contributions, are excluded from the dataset. Extensive literature on network traffic detection and classification methods informs the approach, highlighting various techniques and identifying research gaps. Data is sourced from Kaggle and live networks, categorized into distinct datasets for Ping, Voice, Telnet, and DNS traffic. These datasets are partitioned into 80% for training and a 20% is for validation and testing to assess and optimize model performance.

Feature selection

Often traffic datasets have the same source and destination IPs, same source and destination ports, and the same protocols, we have dropped these features leaving us time-based features. It is extremely important to select the subset of features that have the most impact on the classification task. Our feature size is not that large to warrant feature selection but one of the goals of this session of our work is to investigate how different feature

combinations contribute to the classification task. Also using feature selection will result in a reduced number of needed features hence reducing computation time which is very vital for real-time use. The PCA used in this paper are discussed below in each model discussion.

Distributed internet traffic generator

We used D-ITG to simulate traffic for Telnet, Voice (G.711 codec), Ping and DNS with stochastic models for packet size (PS) and inter-departure time (IDT). It is able to generate multiple unidirectional flows from many senders toward many receivers having many features[31]. For Ping traffic, we ran a simple ping command to the destination host. The process involved several steps: first, we simulated traffic flows between hosts using D-ITG. Next, we ran a traffic classifier Python script configured with the RYU controller and a simple monitor to track and log traffic data. The simple monitor script collected data and updated the attributes of a Flow object, which were then periodically outputted to CSV files. Finally, these CSV files for each traffic type were combined into a single Pandas Data Frame for model training and testing, ensuring the dataset accurately represented realistic and comprehensive traffic data for reliable model evaluation.

Tool used

We implemented our model on a Linux system with an Intel® Core™ i5-6100U CPU, 4 GB RAM, and a 500 GB hard drive. Our setup included Mininet for SDN simulation, Anaconda Python 3.7 with Jupyter for data analysis, and TensorFlow/Keras for machine learning.

Framework design

The proposed system, depicted in Fig. 4, integrates SDN components including a controller, Open Virtual Switch (OVS) and end devices. Once trained and validated on specific datasets, it effectively classifies new traffic data by evaluating how well it identifies and categorizes diverse traffic patterns.

Performance evaluation

To evaluate the performance of the proposed model different performance metrics were used. The effectiveness of classification can be evaluated in numerous ways. Accuracy, Recall, Precision and F1-score are widely used metrics to evaluate the classification performance.

Confusion matrix

It is used as a clean way to present the prediction results of a classifier. It consists of true class labels and predicted class labels. Table 3 presents the truth table of confusion matrix.

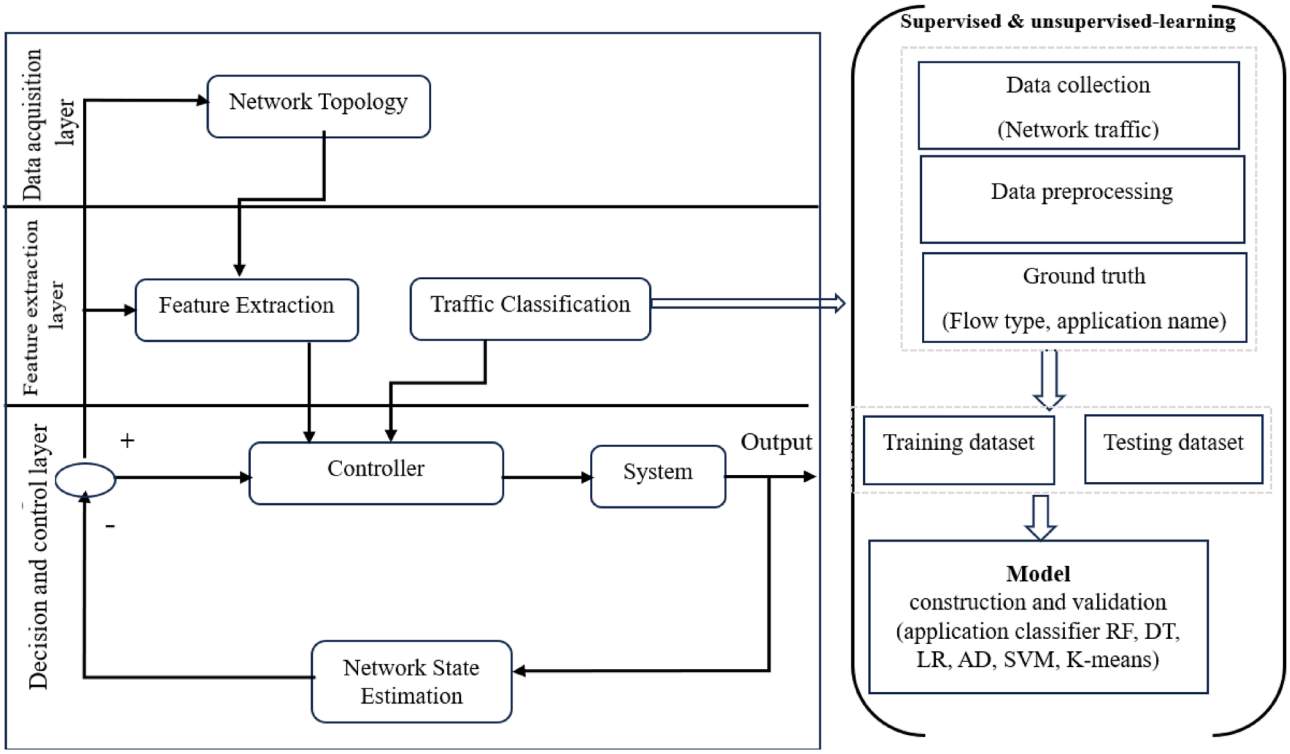

Fig. 4. Framework for traffic classification in software defined networks.

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.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{4}$$

Results and discussion

After training our model, we carefully tested how well it could classify network traffic using standard metrics. We compared it with other models, datasets and checked its performance using real-time traffic data to mimic real-world scenarios. We used six different ML methods: RF, AdaBoost, SVM, DT, LR, and K-means Clustering. These were chosen because they are good at sorting different types of network traffic accurately. Our thorough testing helped confirm that our model performed well and can handle complex network situations effectively.

Supervised learning

Logistic regression

We employed Logistic Regression (LR) to predict categorical target variables. This statistical method analyzes datasets, where outcomes depend on independent variables, generating decision boundaries (see Fig. 5) that illustrate its high accuracy in classification tasks.

As we observed, the decision boundaries are well-defined. Classifying Voice and Telnet traffic is straightforward, while distinguishing between Ping and DNS poses challenges due to their similarities. To evaluate the model accuracy in handling these distinctions, a confusion matrix was used. This matrix aligns actual traffic labels on the X-axis with predicted labels on the Y-axis, highlighting any tendencies of the model to misclassify specific traffic types. Figure 6 exemplifies the minimal errors observed, particularly when employing LR.

The proposed model with LR achieved a classification accuracy of 99.68% with a dataset size of 5628 instances.

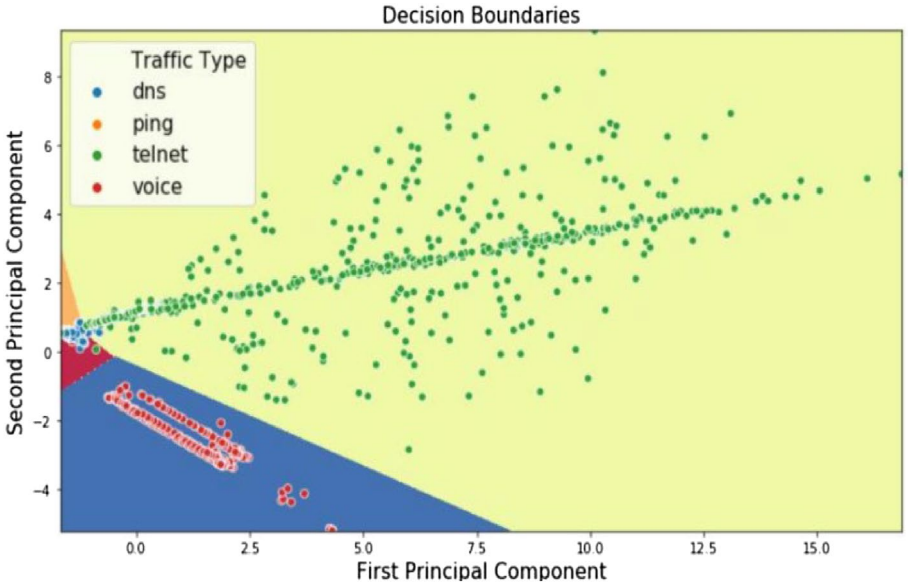

Fig. 5. Principal component analysis for logistic regression.

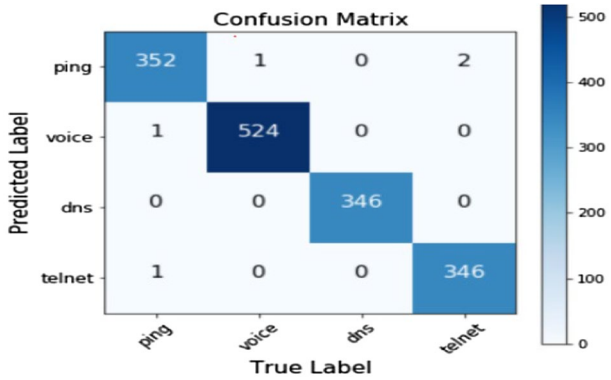

Fig. 6. Confusion matrix for logistic regression.

AdaBoost classifier

AdaBoost, a widely popular boosting algorithm used for binary classification, combines multiple weak classifiers into a robust single classifier. In our experiment, the AdaBoost classifier achieved a good accuracy of over 99.77%, as illustrated in Fig. 7.

The AdaBoost classifier employs an ensemble approach, typically using one-level decision trees, which are highly effective within this algorithm. Initially, each data point is weighted equally, with weights adjusted in subsequent iterations to focus on incorrectly classified points. PCA transforms correlated variables into orthogonal components known as principal components, capturing maximum variance in the data. In our method, we computed eigenvectors from the covariance matrix using singular value decomposition, producing results similar to PCA due to the covariance matrix's properties, as shown in Fig. 8.

```
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.

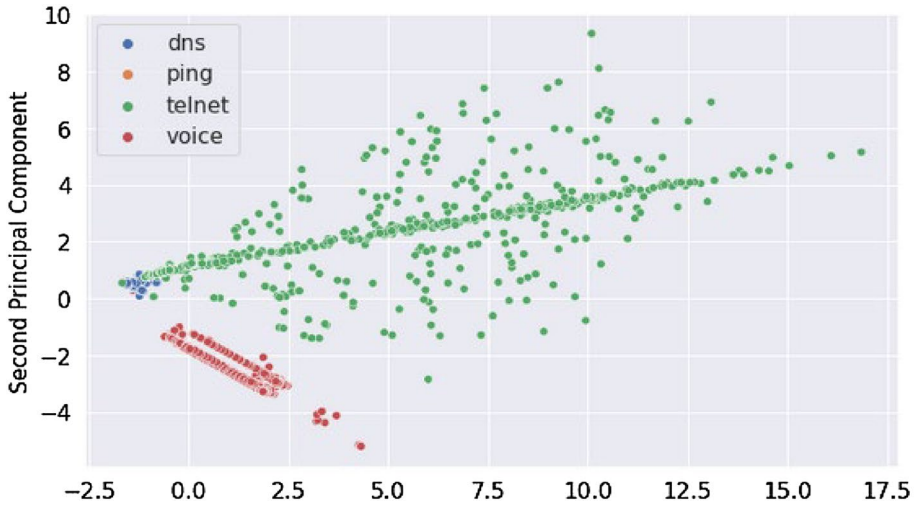

Fig. 8. Principal component analysis using AdaBoost classifier.

In Fig. 8, Telnet and Voice traffic are accurately categorized, but distinguishing between Ping and DNS traffic remains challenging with only two components. The AdaBoost classifier confusion matrix (Fig. 9) highlights any misclassifications. Our model achieved relatively good performance: 99.77% accuracy with 5,628 instances.

Decision tree classifier

The Decision Tree (DT) method classifies unseen cases by traversing a tree structure where nodes make decisions based on feature values. Categorical features split based on possible values, while continuous features divide by thresholds. Figure 10 presents outcomes of the DT classifier, showing metrics like accuracy, precision, recall, and F1-score.

Figure 11 depicts PCA applied to the DT, transforming correlated features into orthogonal components to boost classification efficiency and lower computational complexity. PCA is beneficial before training a DT because it transforms the dataset to emphasize high-variance directions. These directions typically correspond to those providing the most Information Gain for the DT. In high-dimensional datasets with correlated variables, PCA improves classification accuracy by reducing redundancy and focusing on the most informative components.

As illustrated, the method effectively separates Telnet, voice and DNS traffic with clear boundaries. However, it struggles to classify Ping traffic correctly using only one component. Figure 12 shows a confusion matrix for the DT classifier, comparing predicted labels with actual labels. This helps us see where the classifier may make mistakes and where it can improve in accurately classifying traffic types.

The proposed model using the DT classifier achieved 99.81% accuracy with a dataset of 5,628 instances.

Support vector machine

In Support Vector Machines (SVM), a hyperplane is crucial for separating different classes effectively. The goal is to find the optimal hyperplane that accurately divides the network traffic classes. Figure 13 displays the confusion matrix for the SVM classifier, providing insights into how well the model classifies traffic.

PCA and SVM are implemented by first splitting the dataset into training and testing sets. PCA reduces the dimensionality of features, ensuring they are orthogonal. Before PCA, features are normalized using Standard Scaler to subtract the mean and scale to unit variance. Figure 14 illustrates the application of PCA to SVM.

Telnet, voice, and DNS traffics are clearly identifiable, but distinguishing Ping traffic with only one component is difficult. Figure 15 displays the confusion matrix for the SVM classifier, with predicted labels along the Y-axis and true labels along the X-axis. This matrix highlights where the model misclassifies traffic types, giving a clear view of its performance.

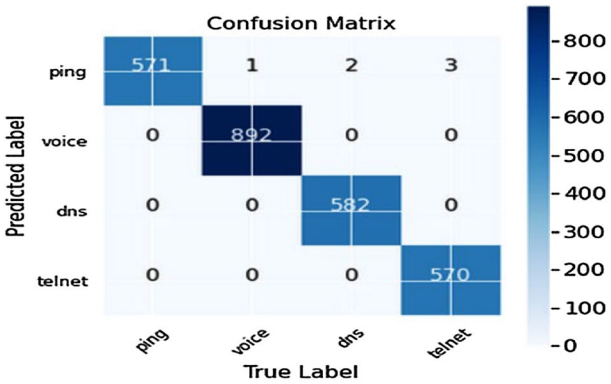

Fig. 9. Confusion matrix for AdaBoost classifier.

```
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.

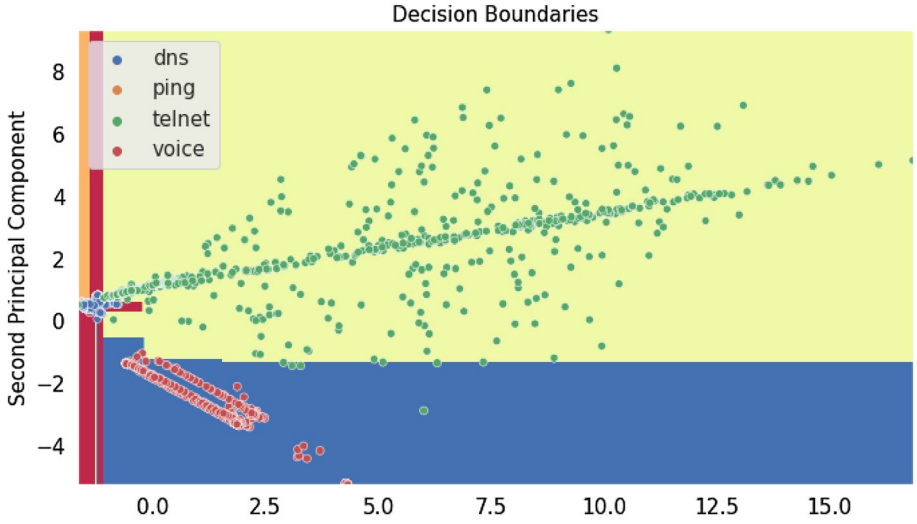

Fig. 11. Principal component analysis for decision tree.

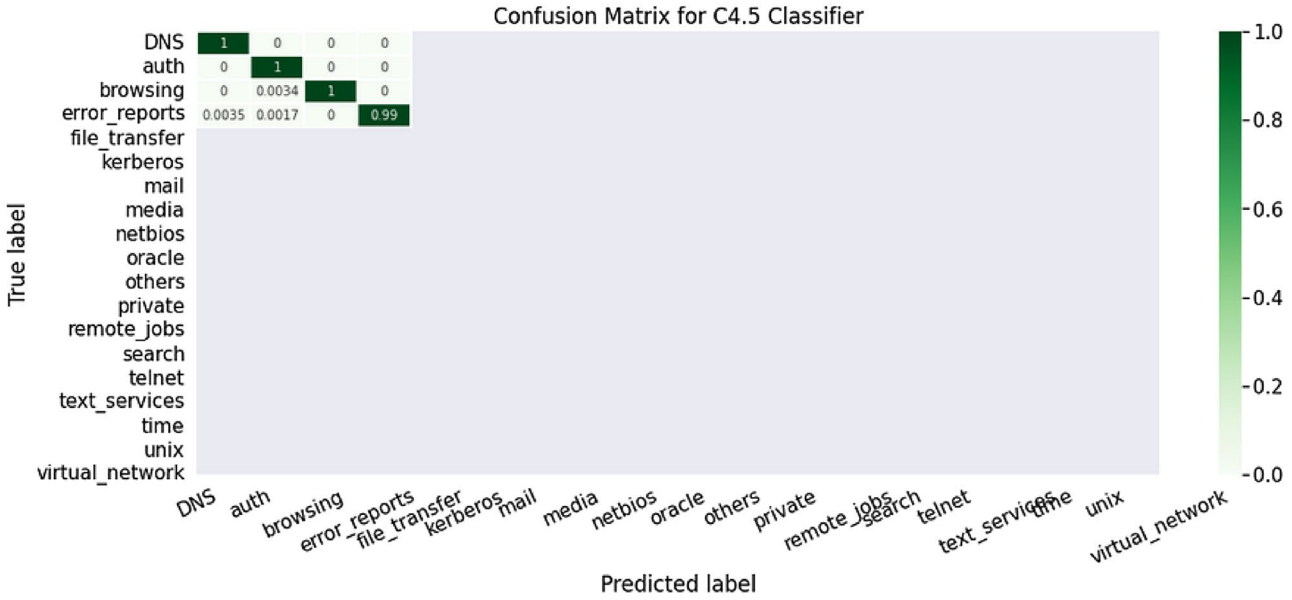

Fig. 12. Confusion matrix for decision tree classifier.

```
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.

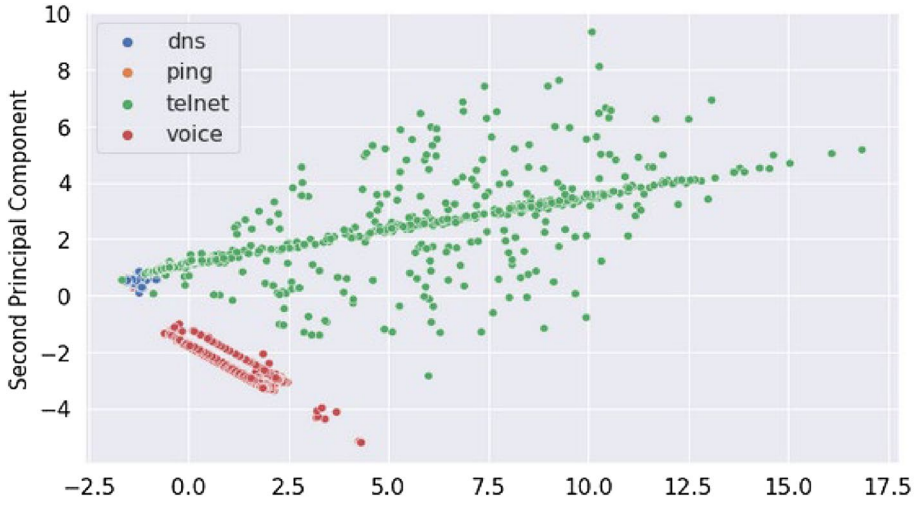

Fig. 14. Support vector machine principal component analysis.

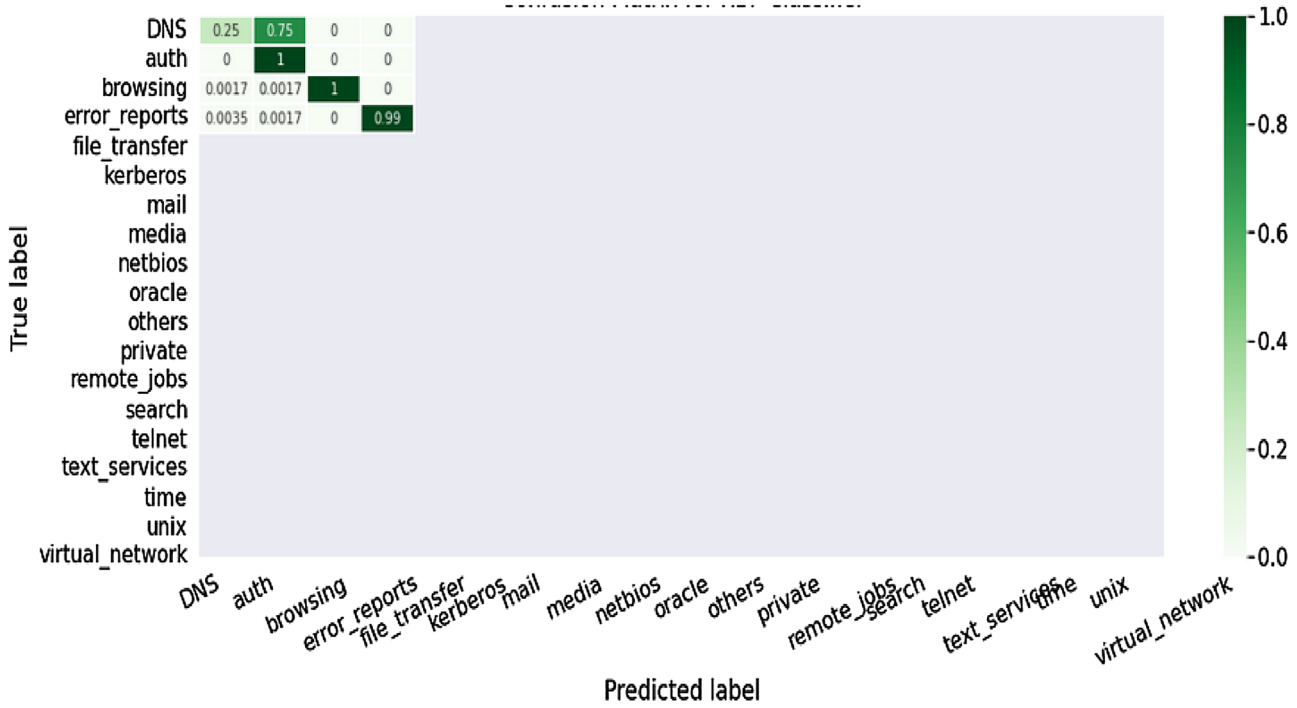

Fig. 15. Confusion matrix for support vector machine.

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

Random forest classifier

The Random Forest (RF) classifier in this study aggregates multiple tree classifiers, each trained on a randomly sampled subset of input vectors. Each tree then contributes a vote towards the most frequent class prediction. The RF classifier utilized in this paper employs randomly selected features or feature combinations at each node to construct each tree. Figure 16 presents the confusion matrix for random forest, the random forest principal component analysis in shown in Fig. 17.

Telnet, voice, and Ping traffics are easily categorized, while DNS remains challenging to identify with only one component. The proposed model using RF achieved a classification accuracy of 99.74% with a dataset comprising 5628 instances.

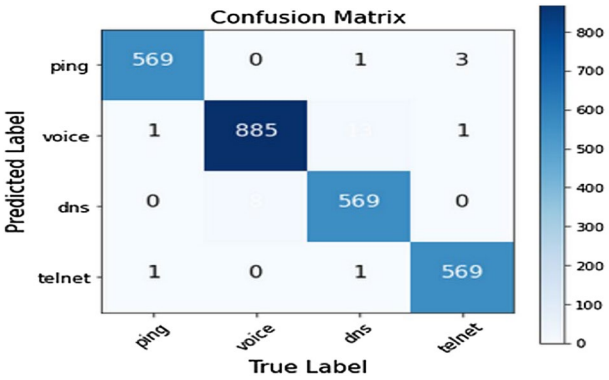

Fig. 16. Confusion matrix for random forest.

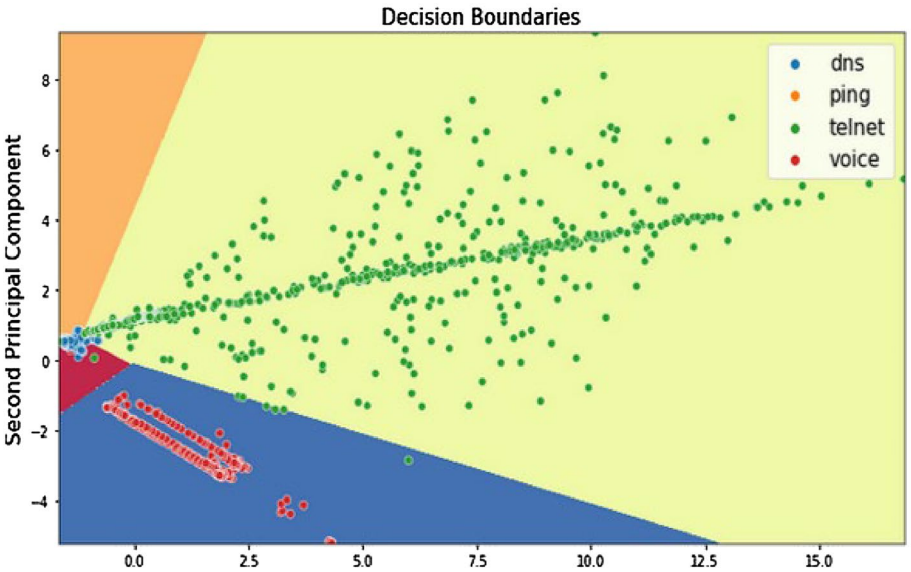

Fig. 17. Random forest principal component analysis.

Unsupervised learning

K-means clustering

K-Means Clustering, an unsupervised machine learning algorithm, groups data points in multidimensional space by assigning each point to the nearest cluster centroid. In our study, we identified four clusters representing distinct traffic types. Figure 18 illustrates the centroids of these clusters in a 12-dimensional vector space, with each square indicating a dimension and shading representing value magnitudes. This visualization provides insight into the cluster centroids' positions and the distinct characteristics of each traffic type.

As we observed in Fig. 18 that K-means prioritizes certain features to differentiate between traffic categories. To visualize the model's performance, Principal Component Analysis (PCA) is employed to reduce the dimensions from 12 to 2, facilitating plotting in a two-dimensional space. This visualization clarifies why the algorithm achieved approximately 30% accuracy; it struggles with clustering non-circular data groups effectively. Figure 19 illustrates the performance of K-means clustering.

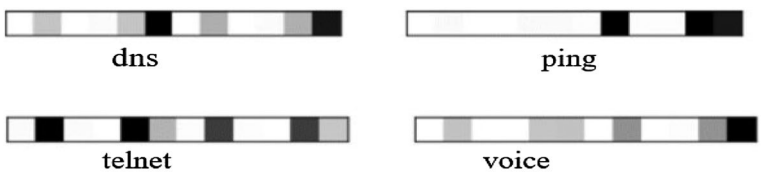

Fig. 18. Visualization of cluster centroids.

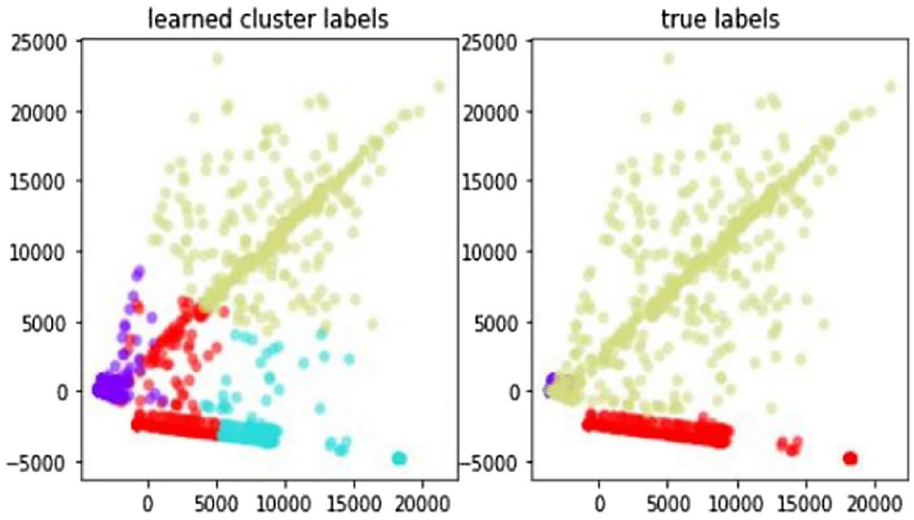

Fig. 19. Visualizing K-means clustering performance.

The labeling accuracy of the K-means model shows deficiencies: half of the Telnet flows are mislabeled, and the Voice category is split into two labels. This discrepancy arises because our traffic data exhibits a more linear structure, whereas K-means performs better with circular clusters. Figure 20, depicting the confusion matrix, confirms these performance issues.

In our evaluation of ML algorithms, the DT consistently showed superior performance in both accuracy and computational efficiency. It is used for real-time TC having fewer features and integrate decisions efficiently. Compared to LR and SVM, which excel with limited data, and AdaBoost and RF, which have their own strengths and limitations, the DT stood out with the highest average accuracy of 99.81%. As observed in Table 4, the results show that the accuracy of the proposed model with DT classifier is higher than others. In Table 5, a comparative analysis with existing works is presented. The results show a comparatively high accuracy compared with existing works.

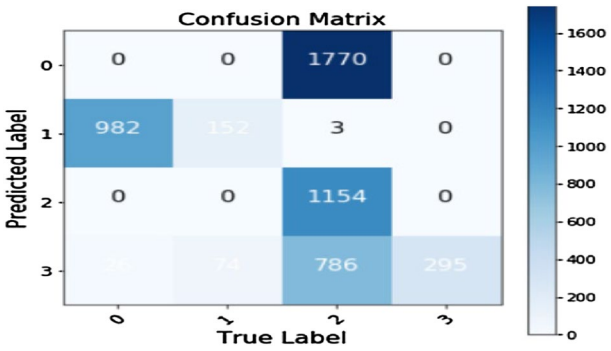

Fig. 20. Confusion matrix of K-means clustering.

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

Identifying traffic based on port numbers and payload inspection is becoming ineffective due to the dynamic and encrypted nature of modern network traffic. Accurately identifying both offline and real-time network traffic types such as ping, telnet, DNS, and voice is crucial for effective network management and ensuring quality service delivery. In this research work, we have integrated SDN with ML techniques including RF, DT, SVM, AdaBoost, LR, and K-means Clustering. This paper aimed to manage and classify network traffic based on applications within a SDN environment using a systematic approach to provide QoS. The Mininet tool in SDN was used to design network architecture, generate traffic, and evaluate its performance with various metrics. Among the models tested, the Decision Tree was the most effective method due to its robustness in handling complex data and its performance in accurately classifying traffic across different network scenarios, including real-time simulations that replicate diverse network conditions. This integration not only enhances traffic classification but also improves network efficiency and reliability. For future work, since most of internet traffic in recent times are videos, D-ITG can be used to generate flows for video games or other video traffic.

Data availability

The datasets generated during and/or analysed during the current study are not publicly available but are available from the corresponding author on reasonable request.

Code availability

The code for this research is available on Github via the following link: https://github.com/melesewmossie.

Received: 4 February 2024; Accepted: 22 August 2024
Published online: 29 August 2024

References

1. Haji, S. H. *et al.* Comparison of software defined networking with traditional networking. *Asian J. Res. Comput. Sci.* https://doi.org/10.9734/ajrcos/2021/v9i230216 (2021).
2. Kafetzis, D., Vassilaras, S., Vardoulias, G. & Koutsopoulos, I. Software-defined networking meets software-defined radio in mobile ad hoc networks: State of the art and future directions. *IEEE Access* **10**, 9989–10014. https://doi.org/10.1109/ACCESS.2022.3144072 (2022).
3. Amin, R. *et al.* A survey on machine learning techniques for routing optimization in SDN. *IEEE Access* **9**, 104582–104611. https://doi.org/10.1109/ACCESS.2021.3099092 (2021).
4. Nandhini, R. & Evangelin Sonia, S. V. A survey and comparison of SDN based traffic management techniques. *Asian J. Appl. Sci. Technol.* **04**(03), 10–18. https://doi.org/10.38177/ajast.2020.4302 (2020).
5. Serag, R. H. *et al.* Machine-learning-based traffic classification in software-defined networks. *Electronics* **13**(6), 1108. https://doi.org/10.3390/electronics13061108 (2024).
6. Tonkal, Ö. & Polat, H. Traffic classification and comparative analysis with machine learning algorithms in software defined networks. *Gazi Üniversitesi Fen Bilim. Derg. Part C Tasar. Ve Teknol.* **9**(1), 71–83. https://doi.org/10.29109/gujsc.869418 (2021).
7. Keshari, S. K., Kansal, V. & Kumar, S. A systematic review of quality of services (QoS) in software defined networking (SDN). *Wirel. Pers. Commun.* **116**(3), 2593–2614. https://doi.org/10.1007/s11277-020-07812-2 (2021).
8. Amaral, P., Dinis J., Pinto, P., Bernardo, L., Tavares, J., Mamede, H. S. Machine learning in software defined networks: Data collection and traffic classification. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)* 1–5 (IEEE, 2016). https://doi.org/10.1109/ICNP.2016.7785327.
9. Mahgoub, S., Ashour, M., Yakout, M. & AbdElhalim, E. Traffic classification in software defined networks based on machine learning algorithms. *Int. J. Telecommun.* **04**(01), 1–19. https://doi.org/10.21608/ijt.2024.340441 (2024).
10. Salau, A. O., Yesufu, T. K. A probabilistic approach to time allocation for intersecting traffic routes. In *Advances in Intelligent Systems and Computing*, vol. 1124, 151–164 (Springer, 2020). https://doi.org/10.1007/978-981-15-2740-1_11.
11. Zhao, Y. *et al.* A survey of networking applications applying the software defined networking concept based on machine learning. *IEEE Access* **7**, 95397–95417. https://doi.org/10.1109/ACCESS.2019.2928564 (2019).
12. Jiang, W. Graph-based deep learning for communication networks: A survey. *Comput. Commun.* **185**, 40–54. https://doi.org/10.1016/j.comcom.2021.12.015 (2022).
13. Chang, L.-H., Lee, T.-H., Chu, H.-C. & Su, C.-W. Application-based online traffic classification with deep learning models on SDN networks. *Adv. Technol. Innov.* https://doi.org/10.46604/aiti.2020.4286 (2020).
14. Wang, P., Wang, Z., Ye, F. & Chen, X. ByteSGAN: A semi-supervised Generative Adversarial Network for encrypted traffic classification in SDN Edge Gateway. *Comput. Netw.* **200**, 108535. https://doi.org/10.1016/j.comnet.2021.108535 (2021).
15. Zion, Y., Dvir, D. A., Ofir, D. Classification and enrichment of encrypted traffic Using Machine Learning algorithms.
16. Ng, B., Bakker, J., Seah, W. K. G., Pekar, A. Traffic classification with machine learning in a live network (2019).
17. Yungaicela-Naula, N. M., Vargas-Rosales, C. & Perez-Diaz, J. A. SDN-based architecture for transport and application layer DDoS attack detection by using machine and deep learning. *IEEE Access* **9**, 108495–108512. https://doi.org/10.1109/ACCESS.2021.3101650 (2021).
18. Gordon, H., Batula, C., Tushir, B., Dezfouli, B., Liu, Y. Securing smart homes via software-defined networking and low-cost traffic classification. (2021). *arXiv*: arXiv:2104.00296. http://arxiv.org/abs/2104.00296 (accessed 14 Jul 2024).
19. Salau, A. O. Development of a Technique for Simulating Traffic Congestion. M.Sc. Thesis, Obafemi Awolowo University Ile-Ife, Nigeria (2015).
20. Deriba, F. G., Salau, A. O., Mohammed, S. H., Kassa, T. M. & Demilie, W. B. Development of a Compressive Framework Using Machine Learning Approaches for SQL Injection Attacks. *Przeglad Elektrotechniczny*, **7**(1), 181–187. https://doi.org/10.15199/48.2022.07.30 (2022).
21. Santos, R., Souza, D., Santo, W., Ribeiro, A. & Moreno, E. Machine learning algorithms to detect DDoS attacks in SDN. *Concurr. Comput. Pract. Exp.* **32**(16), e5402. https://doi.org/10.1002/cpe.5402 (2020).
22. Mohammed, A. R., Mohammed, S. A., Shirmohammadi, S. Machine learning and deep learning based traffic classification and prediction in software defined networking. In *2019 IEEE International Symposium on Measurements & Networking (M&N)* 1–6 (IEEE, 2019). https://doi.org/10.1109/IWMN.2019.8805044.
23. Khan, M. A. *et al.* A deep learning-based intrusion detection system for MQTT enabled IoT. *Sensors* **21**(21), 7016. https://doi.org/10.3390/s21217016 (2021).
24. Ali, T. E., Chong, Y.-W. & Manickam, S. Machine learning techniques to detect a DDoS attack in SDN: A systematic review. *Appl. Sci.* **13**(5), 3183. https://doi.org/10.3390/app13053183 (2023).
25. Talukder, M. A. *et al.* A dependable hybrid machine learning model for network intrusion detection. *J. Inf. Secur. Appl.* **72**, 103405. https://doi.org/10.1016/j.jisa.2022.103405 (2023).
26. Assis, M. V. O., Carvalho, L. F., Lloret, J. & Proença, M. L. A GRU deep learning system against attacks in software defined networks. *J. Netw. Comput. Appl.* **177**, 102942. https://doi.org/10.1016/j.jnca.2020.102942 (2021).
27. Kulin, M., Kazaz, T., De Poorter, E. & Moerman, I. A survey on machine learning-based performance improvement of wireless networks: PHY, MAC and network layer. *Electronics* **10**(3), 318. https://doi.org/10.3390/electronics10030318 (2021).
28. Khatri, S. *et al.* Machine learning models and techniques for VANET based traffic management: Implementation issues and challenges. *Peer-Peer Netw. Appl.* **14**(3), 1778–1805. https://doi.org/10.1007/s12083-020-00993-4 (2021).
29. Malik, A., De Frein, R., Al-Zeyadi, M., Andreu-Perez, J. Intelligent SDN traffic classification using deep learning: Deep-SDN. In *2020 2nd International Conference on Computer Communication and the Internet (ICCCI)* 184–189 (IEEE, 2020). https://doi.org/10.1109/ICCCI49374.2020.9145971.
30. Perera Jayasuriya Kuranage, M., Piamrat, K., Hamma, S. Network traffic classification using machine learning for software defined networks. In *Machine Learning for Networking*, vol. 12081. Lecture Notes in Computer Science, vol. 12081 (eds. Boumerdassi, S., Renault, É., Mühlethaler, P.) 28–39 (Springer International Publishing, 2020). https://doi.org/10.1007/978-3-030-45778-5_3.
31. Eshetu, A.Y., Mohammed, E.A. & Salau, A.O. Cybersecurity vulnerabilities and solutions in Ethiopian university websites. *Journal of Big Data*, **11**, 118. https://doi.org/10.1186/s40537-024-00980-z (2024).

Author contributions

A.O.S.: Data curation, visualization, investigation, methodology and writing-reviewing and editing, validation. M.M.B.: Conceptualization, methodology, software, visualization, writing-original draft preparation, validation.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to A.O.S.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.