JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 11 Issue 02
2026
Research Article
https://doi.org/10.56979/1102/2026/1523

Towards Trustworthy Software-Defined Network Security: An Explainable and Computationally Efficient Machine Learning Framework for Intrusion Detection

Suhail Ashfaq Butt[1,2], Shakir M. Usman[3], Muhammad Ahsan Raza[4], Qasem M. Kharma[5], Sagheer Abbas[6], and Taher M. Ghazal[7,8,9]

[1]Department of Computer Science, National College of Business Administration and Economics, Lahore, 54000,Pakistan.
[2]Department of Information Sciences, Division of Science and Technology, University of Education, Lahore, 54000, Pakistan.
[3]Center for Cognitive Intelligence and AI Systems, College of Sciences and Human Studies, Prince Mohammad Bin Fahd University, Saudi Arabia.
[4]Department of Information Sciences, University of Education, Lahore, Multan Campus 60000, Pakistan.
[5]Software Engineering Department, Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman, Jordan.
[6]Department of Computer Science, Prince Mohammad Bin Fahd University, Alkhobar, 31952, KSA.
[7]Department of Networks and Cybersecurity, Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman, Jordan.
[8]Faculty of Computing and IT, Sohar University, Oman.
[9]Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), 43600 Bangi, Selangor, Malaysia.
*Corresponding Author: Sagheer Abbas. Email: jamsagheer@gmail.com

Received: April 10, 2026 **Accepted:** August 01, 2026

Abstract: Software-defined networking (SDN) offers flexibility and scalability but also introduces new cybersecurity challenges. We propose a scalable and transparent intrusion detection system (IDS) for SDNs using machine learning models that balance accuracy and computational efficiency. Our results show that the decision tree model achieves 99.6% accuracy with minimal missed attacks and lower computational costs compared with random forests. The training time increased with the dataset size, but the prediction time remained stable, even with 24,000 samples. Our simulation results showed that the training complexity of Random Forest increases with the number of samples, whereas the calculation time for inference is comparatively low–a finding that makes Random Forest particularly suitable for real-time detections in Software-Defined Networking (SDN) environments. By using the SHAP-based explainability analysis, we explain the most important traffic characteristics, such as the length of the packets and the destination port, which are used to decide the decision process of the model. This methodological approach ensures high recall detection without compromising performance, thus providing a high-performance and efficient solution for SDN security.

Keywords: Software-Defined Networking (SDN); Intrusion Detection System (IDS); Machine Learning; Explainable Artificial Intelligence (XAI); Computational Complexity

1. Introduction

Software-defined networking has marked a major shift in networking today, offering organizations much more flexibility, control and scalability. SDNs offer seemingly more control over networks by separating the control and data planes in a purposeful way and moving approval activities to more rapidly evolving patterns of demands. This new paradigm for architecture, however, raises novel security concerns since the flexibility/ organicity that is the cornerstone of SDNs can be taken advantage of by malicious users. With increasingly complex traffic patterns in the SDN traffic, successful detection of intrusions

becomes increasingly challenging and demands complex methods that balance the thoroughness and efficiency of the detection.

Traditional intrusion detection systems (IDS) tend to fall behind as software defined networks (SDN) rapidly evolve and are inherently dynamic. To overcome these challenges, this study proposes a machine learning-based IDS that not only identifies an intrusion with a high level of accuracy but also provides transparency in decision making. The methodology uses tree-based models (decision trees and random forests) that model complicated traffic patterns in SDNs with low computational overhead. This makes it easier to detect real-time with a minimum of resources, which is a prerequisite for maintaining the performance and reactivity of SDN networks.

Despite major progress in IDS technology, there is still a large gap between the achievements of simultaneously achieving a high detection accuracy and computation efficiency with transparency guaranteed. Many extant systems compromise between one of the two metrics, leading to performance versus explainability tradeoffs. Furthermore, there is still a lack of research that explicitly examines the computational tradeoffs found in SDN environments, especially when trying to achieve a high-recall transparent detection system. Accordingly, our work is focused on making a leap by presenting a path to a scalable and efficient IDS to meet both security and transparency requirements, enabling both demand and practical implementation in a next-generation SDN infrastructure.

Despite significant progress in machine learning-based intrusion detection for SDN environments, existing approaches often face challenges in achieving a balance between detection accuracy, computational efficiency, and interpretability. Some models, like complex ensemble and deep learning models, can be very powerful but can be very hard to compute and may be opaque in terms of how they make decisions. This is a drawback to their use in real-time SDN, where quick response and reliable security decisions are crucial.

Thus, this study has a motivation of developing an efficient and transparent intrusion detection framework which can offer correct intrusion detection with minimal computational overhead and explainable decision-making mechanism. To resolve the challenge, we suggest a framework for IDS in SDN environments that is based on machine learning, is explainable, and is scalable to computation. To address the challenge, we propose an explainable machine learning-based IDS framework that incorporates the following three key capabilities: predictive behavior estimation, interpretation using SHAP, and scalability with respect to computation.

1.1. Research Contributions

This research has the following contributions:

- Decision Tree and Random Forest classifiers-based machine-learning IDS framework for SDN environments.
- A mathematical description of mechanisms for feature processing, classification, and explainability.
- A computational complexity analysis of the training time, inference latency, memory requirements.
- An explainable AI approach to find influential features of network traffic.
- Comparative evaluation shows the compromise between the accuracy of the detection and the efficiency of the computation.

2. Literature Review

There is a growing intrusion detection domain in Software-Defined Networking (SDN) to defend these highly flexible but sensitive network environments. By separating the control and data planes, SDN provides centralized control but simultaneously creates new security issues and offers a broader attack surface [15]. Initial studies on SDN security mainly focused on anomaly- and signature-based intrusion detection systems (IDSs) enhanced by standard machine learning techniques. Empirical studies using Random Forest and logistic regression classifiers have shown promising results in detecting Distributed Denial of Service (DDoS) and other types of attacks in SDN traffic, thus highlighting the impact of feature selection on classification and controller performance [1,3]. Nonetheless, feature selection remains a challenging problem, as numerous and extraneous features can degrade model performance and increase computational costs.

The application of deep learning to improve intrusion detection system performance in software-defined networking environments has attracted considerable scholarly attention. Early explorations demonstrated the efficacy of neural networks and deep learning architectures for intrusion detection tasks [8,10]. In the context of SDN, deep neural networks and recurrent neural networks have shown a clear ability to detect complex and previously unseen attacks, thanks to their inherent ability to learn traffic features hierarchically and automatically [6,11]. Hybrid paradigms, which combine deep autoencoders with random forest classifiers, have depicted accurate anomaly detection rates of more than 98 and nine native SDN environments, which have proposed the capability for deep learning integration with classic machine learning techniques to enhance detection accuracy without losing manageable computational complexity [2]. Other investigations suggest the hybrid feature selection mechanisms, for example, the combination of correlation-based feature selection and recursive feature elimination and gradient boosting models such as LightGBM, improves the detection accuracy and decreases the dimensions of the dataset [3]. These studies show that model ensembling can yield better performance; however, there is a tendency to focus on classification accuracy rather than computational demands, and, as a result, a thorough analysis of complexity remains comparatively unexplored.

Contemporary scholarship emphasizes the importance of explainable artificial intelligence (XAI) in the framework of intrusion detection systems (IDS). As machine learning models become more complex, they become less transparent in their decision-making, making it very difficult for security analysts to identify malicious traffic flows and understand why a particular flow is considered malicious. Explainable methodologies, such as SHAP [SHapley Additive ex-"Planations] and LIME [Local Interpretable Model-agnostic Explanations] have become staple tools for providing interpretability in IDS models [13,14]. These are techniques that enable analysts to trace the contribution of each feature in a prediction by the model. For example, research integrating SHAP with ensemble methods of trees and classifiers revealed how specific network traffic characteristics affect intrusion detection decisions, thereby increasing trust and transparency in the detection process [4]. Empirical evidence demonstrates that feature importance attribution can identify subtle patterns suggesting the presence of malicious activities and, as such, can be useful in both verification and debugging efforts. Furthermore, explainable deep learning frameworks in the software-defined networking (SDN)-based DDoS detection model have been proposed to provide explanations on model predictions while ensuring high detection accuracy [7]. These frameworks attempt to compromise between the depth of representation given by deep networks and the need for interpretability in operational situations. Despite these advances, most explainability studies focus on interpretability results rather than a rigorous evaluation of the computational cost of generating explanations, which, in real-time SDN environments, may prove to be a crucial factor.

Beyond the design of models and their interpretability, it is incumbent upon researchers to make exhaustive evaluations and systematic benchmarking of intrusion detection systems in software-defined networking contexts. A comprehensive meta-analysis of machine learning-based systems for DDoS detection shows that, while many systems claim to report very high false-positive rates, there are still significant barriers to scalability, dataset heterogeneity, and the practicality of employing such systems in real life [5]. Many implementations of IDS are based on a curated set of data or artificial testbeds, and could ultimately become a poor substitute for the constantly changing nature of real traffic. Furthermore, the necessity of putting detection logic directly in SDN controllers makes it important to examine the computational costs and latency or at least to reduce overall throughput of the network.

One other aspect of the study of an intrusion detection system is security techniques that run outside the detection system, such as authentication protocols and secure communication between network components. In connected systems, methods like cryptographic and authentication schemes, such as elliptic-curve cryptography (ECC), have been developed to enhance the security of the devices and to prevent unauthorized access [9]. The ideas behind these methods can be transferred to the software-defined networking infrastructure to enhance the trust between controllers and the periphery of the network as it is deployed in the Internet of Knowledge of Things. These security measures, when used in conjunction with new forms of intrusion detection, can help provide a more holistic defense against both prevention and detection of malicious activity.

New developments of AI have further enriched the intrusion detection with deep learning and distributed learning. Deep neural networks, convolutional neural networks (CNNs), and recurrent neural

networks (RNNs) have shown good potential in extracting complicated temporal and spatial patterns from network traffic, which can be used to enhance the detection ability of sophisticated and unknown attacks. But deep Learning models can impose significant computational demands and are difficult to deploy in latency sensitive SDN environments. In recent years, hybrid architecture consists of deep learning feature extractors and lightweight machine learning classifiers have been investigated to enhance the accuracy while minimizing computational consumption.

The federated learning (FL) paradigm has been recently introduced as a potential solution for intrusion detection in-distributed networks that maintain privacy. Compared to traditional centralized learning approaches, federated IDS can facilitate multiple network entities to collaboratively build detection models while preserving the raw traffic within local entities. A number of these solutions mitigate privacy concerns of sharing sensitive cybersecurity information; however, there are still major research challenges such as communication overhead, model aggregation efficiency and resource constraints.

Moreover, lightweight cybersecurity has become a focal point of interest as the number of edge computing, IoT and SDN infrastructures come into operation. There has been recent research that highlights the need for building efficient IDS models that can be able to detect malicious activities with high detection accuracy, low inference latency and with low computational complexity while consuming less memory. While these methods help to make the model scalable and easier to deploy, most of the present work is predominantly focused on the predictive performance without deep analysis on the transparency of the models and the computational trade-offs.

Explainable Artificial Intelligence (XAI) is thus a research focus of great relevance for a reliable cybersecurity system. Recent approaches to XAI-based IDS, e.g., based on SHAP and LIME, give important insights into the contribution of features and the process by which the attack is decided. Yet, the computational expenses of generating explanations and the effects on real-time SDN deployment is still not enough explored. To this end, this study aims to fill the gap by defining an explainable machine learning framework that comprehensively considers the performance of intrusion detection systems, interpretability and computational efficiency.

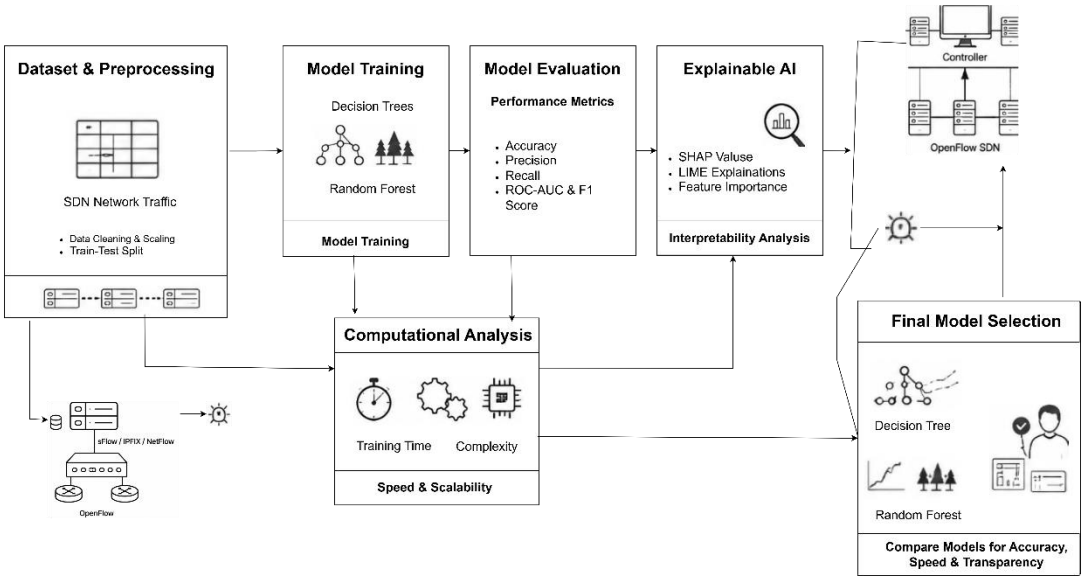

Figure 1. Scalable and Transparent Intrusion Detection for SDNs

The proposed explainable and computationally efficient system architecture of intrusion detection in SDN environment is shown in figure 1. The framework is composed of four steps: SDN traffic acquisition, data preprocessing, machine learning based detection, explanation and computational analysis.

At first, the traffic in the network that is created inside the SDN environment is gathered and depicted as flow level features. The features are then processed through various preprocessing steps, such as cleaning, encoding, and normalization, to enhance the quality and consistency of the input data. The feature vectors are processed in the detection module for the detection of malicious traffic and malicious traffic pattern by using the classifiers Decision Tree Classifier and Random Forest Classifier.

The explainability module then performs SHAP analysis after the classification to understand the contribution of each feature of the network to the prediction result. This way, security operators can understand the model's decisions and recognize key attributes of important attacks. Finally, the computational analysis module measures the efficiency of the proposed framework by computing the training complexity, inference latency, and scalability with the growing amount of traffic.

This integrated workflow ensures the proposed framework has a tradeoff between detection accuracy, interpretability and computational efficiency to be suitable for practical SDN security applications.

Table 1. Key studies from previous work

Study	Methodology	Key Findings	Gap Identified
DDoS Detection in SDNs [1]	Random Forest, Logistic Regression	High accuracy in DDoS detection with feature selection impacting performance	Feature selection and computational cost trade-offs not discussed
Hybrid Deep Learning Models [2]	Autoencoders + Random Forest	Detection rates >98%, hybrid models improve accuracy while managing complexity	No detailed computational cost analysis
Feature Selection + LightGBM [3]	Correlation-based and recursive feature elimination	High accuracy with reduced dimensionality in SDN IDS	Lack of comprehensive computational complexity analysis
Explainable IDS with SHAP [4]	Random Forest, SHAP	SHAP provides transparency into decision-making, highlighting key features like destination port	XAI's impact on performance and computational cost remains under-explored
End-to-End Explainability [5]	Various models (black-box, glass-box)	Demonstrates minimal overhead from explainability while improving trust	No clear computational trade-off analysis between explainability and speed

Two key gaps remain in the literature.

- Computational Complexity Analysis Although many IDS proposals claim to have high accuracy, few studies carefully analyze the training and inference complexity in terms of the number of samples and feature dimensions.
- Explainability vs. Performance Trade-offs Despite the venerable role of XAI in improving interpretability, systematic studies of trade-offs between explanation generation and performance/scalability for SDN as well as performance conatus and coauthors, 2020 are conspicuously sparse.

To overcome the gaps, the above-mentioned computational trade-offs should be properly analyzed and the above-mentioned explainability and efficiency should be considered in an integrated manner in the development of intrusion detection systems in practical deployments of SDN.

3. Proposed Methodology

The proposed model for scalable and transparent intrusion detection in software defined networks (SDNs) is split into a series of important phases to guarantee the high performance, efficiency and interpretability of the network. The process starts with Dataset & Preprocessing, in which the data of the

SDN network traffic is put through certain steps, such as cleaning the data, scaling features, and creating the train-test split for training the model further. In this phase, the features of the SDN traffic are refined to improve learning the model. Specifically, the data set contains samples of benign and malicious network traffic, and is especially focused on ensuring that the data is of high quality and consistent before being fed into the machine learning models.

Next, the model training phase uses two popular machine learning algorithms–decision trees and random forests–to train the IDS models. These models are known for their ability to handle complex and large datasets, making them suitable candidates for SDN traffic analysis. Once the models are trained, it is important to evaluate them using a few performance metrics, such as accuracy, precision, recall, F1-score, and ROC-AUC, to ensure that the models are not only accurate but also reduce false positives and false negatives. Following model evaluation using metrics, the Explainable AI phase employs approaches such as SHAP values, LIME explanations, and feature importance to provide insights into the model's decision-making process. This transparency is important in SDN security because it is important for network administrators to understand why a model classifies some network behaviour as benign or malicious.

In the final computational analysis phase, we assessed the efficiency of the trained models by measuring training time and resource consumption. This is the goal of assessments like this, to ensure that the models scale to meet the growing volumes of network traffic without costing too much in a computational context. The overall goal is to achieve a perfect balance between speed, scalability, and transparency, thereby realizing real-time intrusion detection in Software-Defined Networking scenarios. Subsequently, the final model was selected from the decision tree and random forest methodologies, choosing the model that best balances accuracy, latency, and interpretability, ensuring a resilient and efficient intrusion detection solution for SDN systems.

The proposed framework was implemented using a Python-based machine learning environment. The experiments included data preprocessing, model training, performance assessment, explainability analysis using SHAP and computational scalability assessment. The models evaluated were Logistic Regression, Decision Tree and Random Forest. Model performance was evaluated based on accuracy, precision, recall, F1-score and ROC-AUC, and computational efficiency was compared with respect to the training time, prediction latency, and scalability analysis with the increase in the size of the data set. The SHAP framework was used to identify the contributions of each individual feature and to generate the explanations for which features are important for the intrusion detection decisions.

3.1. Mathematical Formulation of the Proposed IDS Framework

Mathematical Model: Let the SDN traffic dataset be represented as:

$$D = \{(X_i, y_i)\}_{i=1}^{N} \quad (1)$$

where:

- Nrepresents the total number of network traffic samples.
- $X_i = \{x_1, x_2, \ldots, x_m\}$represents the feature vector of the i^{th}traffic flow.
- $m$represents the number of extracted network features.
- $y_i$represents the traffic class label.

The classification objective is defined as:

$$f(X_i) \rightarrow y_i \quad (2)$$

where the classifier predicts whether a network flow belongs to a normal or malicious class.

For multi-class SDN intrusion detection:

$$y_i \in \{0,1,2,\ldots,C-1\} \quad (3)$$

where:

- $C$is the number of traffic categories.
- 0 represents benign traffic.
- Remaining values represent different attack categories.

Feature Normalization: Before model training, input features are normalized using Min-Max scaling:

$$X_i' = \frac{X_i - X_{min}}{X_{max} - X_{min}} \quad (4)$$

where:

- $X_{min}$is the minimum feature value.

- $X_{max}$is the maximum feature value.
- X_i'represents normalized input features.

This transformation ensures that high-scale network attributes do not dominate the learning process.

3.2. Machine Learning Classification Model

Decision Tree Mathematical Representation: A decision tree divides the feature space into subsets using decision rules:

$$R_j = \{X \mid X \leq t_j\} \quad (5)$$

where:

- $t_j$represents the decision threshold.

The optimal split is selected by minimizing impurity:

$$Gini(D) = 1 - \sum_{k=1}^{C} p_k^2 \quad (6)$$

where:

- $p_k$represents the probability of class k.

Random Forest Mathematical Formulation: The Random Forest classifier consists of multiple decision trees:

$$RF(X) = \frac{1}{T}\sum_{t=1}^{T} h_t\,(X) \quad (7)$$

where:

- Trepresents the total number of trees.
- $h_t(X)$represents prediction from the t^{th}tree.

The final class decision is obtained using majority voting:

$$\hat{y} = argmax_c \sum_{t=1}^{T} I(\,h_t(X) = c) \quad (8)$$

where:

- $I(.)$is an Indicator function.

3.3. Performance Evaluation Mathematical Model

The proposed IDS performance is evaluated using:

Accuracy $$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (9)$$

Precision $$Precision = \frac{TP}{TP+FP} \quad (10)$$

Recall $$Recall = \frac{TP}{TP+FN} \quad (11)$$

F1-score $$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (12)$$

where:

- TP = True Positive
- TN = True Negative
- FP = False Positive
- FN = False Negative

3.4. Explainable AI Mathematical Formulation

The SHAP explanation model can be represented as:

$$f(x) = \phi_0 + \sum_{i=1}^{m} \phi_i \quad (13)$$

where:

- $f(x)$represents model output.
- $\phi_0$represents the base prediction.
- $\phi_i$represents the contribution of feature i.

The SHAP value for a feature is calculated as:

$$\phi_i = \sum_{S \subseteq F \setminus i} \frac{|S|!(M-|S|-1)!}{M!}[f(S \cup i) - f(S)] \quad (14)$$

where:

- Frepresents all features.

- Srepresents a subset of features.
- Mrepresents total features.

This formulation explains how features such as:

- Destination Port
- Packet Length
- TCP Window Size

influence intrusion decisions. This directly supports your SHAP analysis section.

3.5. Proposed Explainable SDN Intrusion Detection Framework

Algorithm 1: Explainable Machine Learning Based IDS for SDN

```
Input:
    SDN traffic dataset D
    Feature vector X
    Class labels Y
Output:
    Predicted attack class and explanation
Step 1:
    Load SDN network traffic dataset
Step 2:
    Perform data preprocessing:
        - Remove missing values
        - Encode labels
        - Normalize features
Step 3:
    Split dataset into:
        Training set Dtrain
        Testing set Dtest
Step 4:
    Train Decision Tree classifier
Step 5:
    Train Random Forest classifier:
        For t = 1 to T:
            Generate bootstrap sample
            Train decision tree ht
        End
Step 6:
    Evaluate models using:
        Accuracy
        Precision
        Recall
        F1-score
        ROC-AUC
Step 7:
    Select optimal classifier based on:
        Performance + computational cost
Step 8:
    Generate SHAP explanations:
        Calculate feature contribution values
        Rank important traffic features
Step 9:
    Perform complexity analysis:
        Measure training time
        Measure prediction latency
        Measure memory consumption
Step 10:
    Return:
        Intrusion prediction
```

Explanation report
End Algorithm

4. Experimental Results

Figure 2 shows two complementary visualizations: the class distribution plot reveals a highly unbalanced dataset with a high representation of the BENIGN class, and the feature correlation heatmap reveals strong interrelationships among the top 20 features, helping to inform the feature selection process for model development.

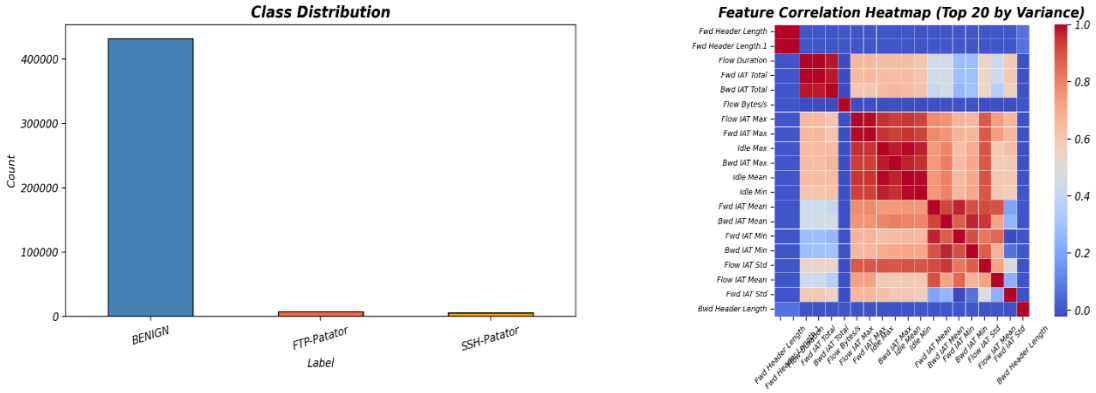

Figure 2. Class Distribution of the Network Traffic Dataset

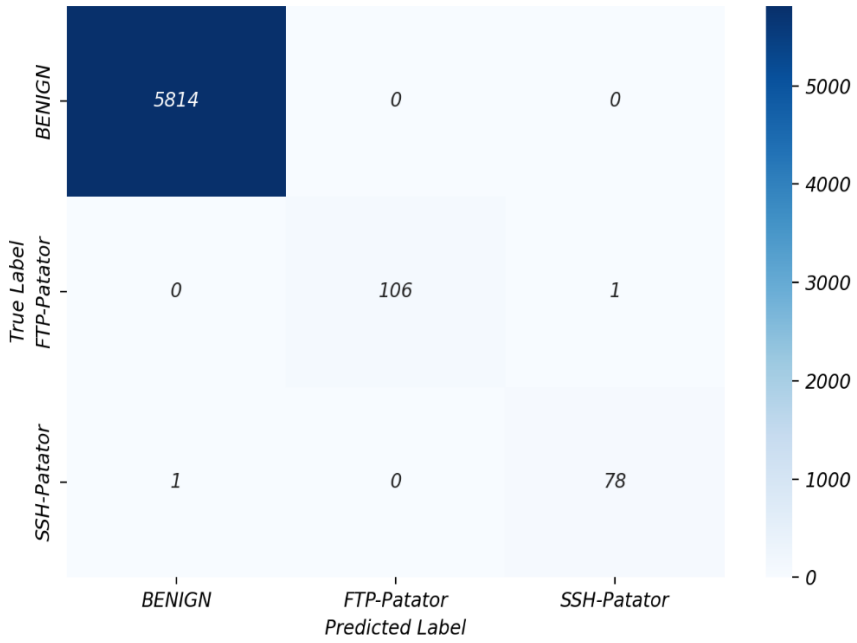

Figure 3. Confusion Matrix of the Intrusion Detection Model

As shown in figure 3, the Confusion Matrix Heatmap demonstrations the performance of the model in classifying the network traffic. It shows that the model successfully separates most of the innocent traffic, accounting for 5814 true positives, while misclassifying a small fraction of FTP-Pator and SSH-Pator samples. Notably, the number of false positives for FTP-Pator and SSH-Pator is negligible, with numbers one and zero, respectively.

Figure 4 shows the receiver operating characteristic (ROC) curves (one-versus-rest) for the classifiers logistic regression, decision tree, and random forest classifier. Each plot shows area under the curve values coming closer to unity for all three target classes (BENIGN, FTP - Pator, and SSH - Pator), providing evidence of strong discriminative performance. The curves show that all the models investigated reliably segregate instances across the defined classes, with the random forest achieving the highest AUC across all labels. As a result, the visualized data support an excellent detection capability, which is inherent to all three modelling approaches.

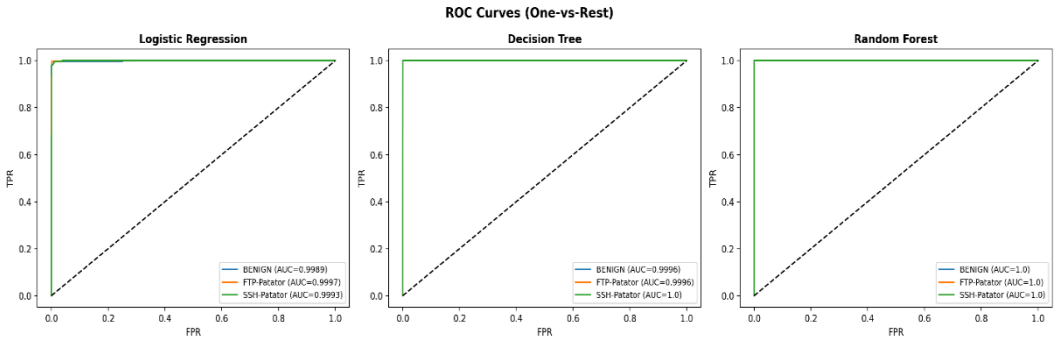

Figure 4. ROC Curves for Logistic Regression, Decision Tree, and Random Forest Models

Figure 5 displays a bar graph comparing the performance of various classification models; in the graph, the Random Forest algorithm outperforms both Logistic Regression and Decision Tree across all metrics considered: accuracy, precision, recall, F1 score, and ROC AUC. Although all models perform excellently with scores close to 1, Random Forest achieves the highest consistency. Decision Tree and Logistic Regression also demonstrate strong performance but slightly behind.

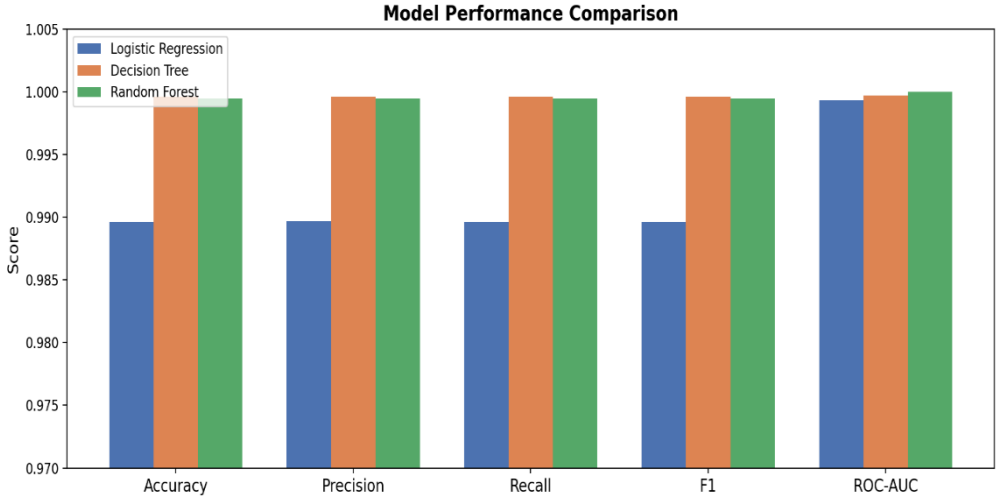

Figure 5. Performance Comparison of Machine Learning Models

Table 2. Performance Evaluation of the Proposed Machine Learning Models

Model	Accuracy	Precision	Recall
Model 1	0.9896	0.9897	0.9896
Model 2	0.9996	0.9996	0.9996
Model 3	0.9994	0.9994	0.9994

Table 2 presents the performance comparison of the evaluated models using accuracy, precision, and recall metrics. Model 2 achieved the highest performance with accuracy, precision, and recall of **99.96%**, indicating superior classification capability. The results demonstrate that all models achieved high predictive performance, with minimal differences among evaluation metrics, reflecting reliable and consistent classification results.

5. SHAP Analysis

The SHAP analysis provides insight into how individual network traffic features influence the intrusion detection decisions of the proposed model. The results show that destination port, packet length statistics, TCP window related characteristics and packet variability features are the most influential features for the results. These features are significant to network communication actions and give sensible explanations for classification results.

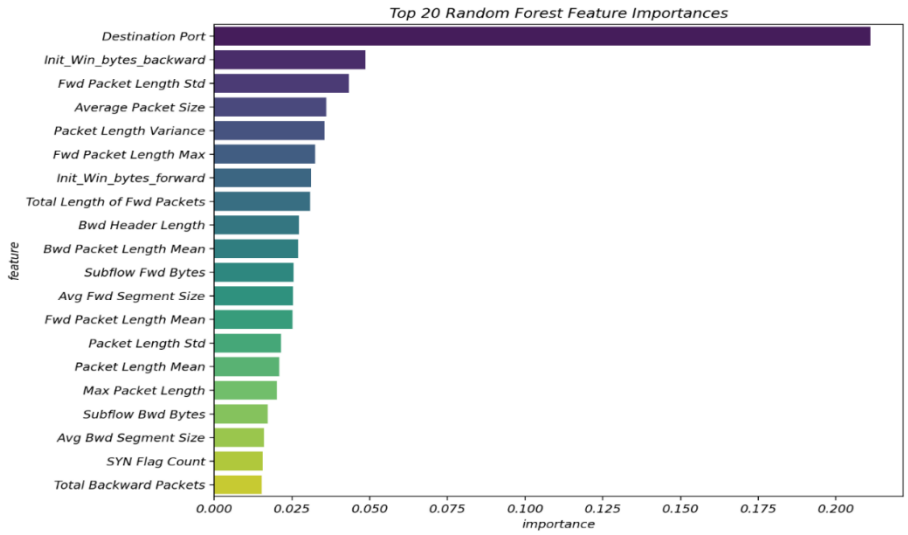

Figure 6. Top 20 Feature Importances from the Random Forest Model

Attackers may target specific services, or take advantage of vulnerable endpoints for communication, which is why Destination Port is such an important piece of the puzzle. An unusual access pattern to the ports might thus be a sign of scanning activity, unauthorized port access, or a targeted attack. Packet length statistics provide information about traffic behavior variations, where unusual packet distributions may reveal malicious communication patterns. Furthermore, TCP window size features capture transport-layer behavior and can assist in identifying abnormal flow characteristics associated with attack traffic. From a practical security perspective, SHAP explanations allow SDN administrators to move beyond simple attack alerts and understand the reasons behind each detection decision. This enables more effective investigation, prioritization of security incidents, and validation of automated IDS predictions. Therefore, the integration of SHAP-based explanations improves the trustworthiness and operational usability of the proposed SDN intrusion detection framework. This is a good sign, because the model is not relying on a large diffuse set of weak predictors. It is making decisions based on interpretable network-flow characteristics. Figure 6 shows the top 20 random forest feature importances bar graph, where Destination Port emerges as the most important feature, significantly higher than the others. Other influential features include Init_Win_bytes_backward and Fwd Packet Length Std, indicating their strong contribution to predicting network behaviour.

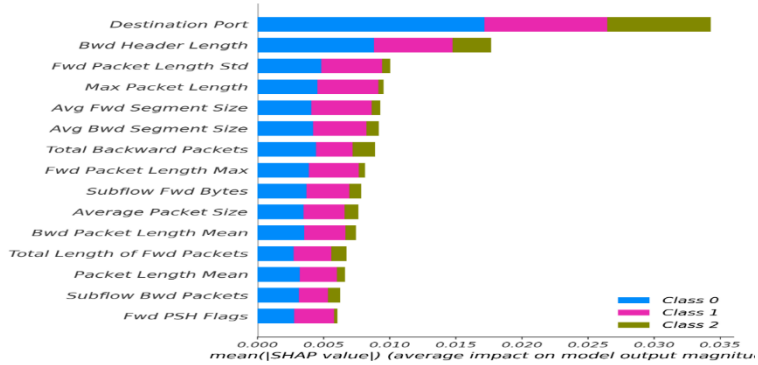

Figure 7. SHAP Feature Importance Analysis

The SHAP Feature Importance visualization is shown in Figure 7, where the average influence of each predictor variable on the probabilistic outputs of the model across three classes of the constituent is elaborated. The Destination Port variable was the most important predictor for all classes, whereas Bwd Header Length and Fwd Packet Length Std also had large impacts on the accuracy of the predictive model.

Computational complexity analysis

In empirical scaling studies, an identical random forest was used to train identical models incrementally.

Table 3. Scalability Analysis of the Machine Learning Model with Increasing Dataset Size

Samples	Train Time (s)	Predict Time (s)	Accuracy
3,000	0.353	0.0587	0.9983
6,000	0.516	0.0668	1.0000
12,000	0.602	0.0569	0.9992
24,000	1.017	0.0562	0.9996

The training time, prediction time, and accuracy of the models are analyzed to determine the model's scalability with respect to the varying sizes of the datasets shown in table 3. The training time grows slowly with the increase of samples, while the prediction time remains nearly unchanged, showing efficient inference. The model has consistently high accuracy ($R \geq 99.83\%$), demonstrating good scalability and performance stability as the data size grows.

Interpretation: From the computational analysis, the number of samples processed during the construction of trees increases with the size of the data set, and therefore the training time for the Random Forest model increases gradually as the size of the data set increases. The training time of Random Forest is roughly proportional to $T \cdot n \cdot m \cdot \log n$, where T is the number of trees, n is the number of training samples and m is the number of candidates features that are examined at each split. In contrast, prediction time remains relatively stable because inference requires traversing the already constructed trees, with an approximate complexity of $O(T \cdot d)$, where d represents the average tree depth. These results indicate that Random Forest provides scalable performance while maintaining efficient prediction capability for large-scale intrusion detection applications.

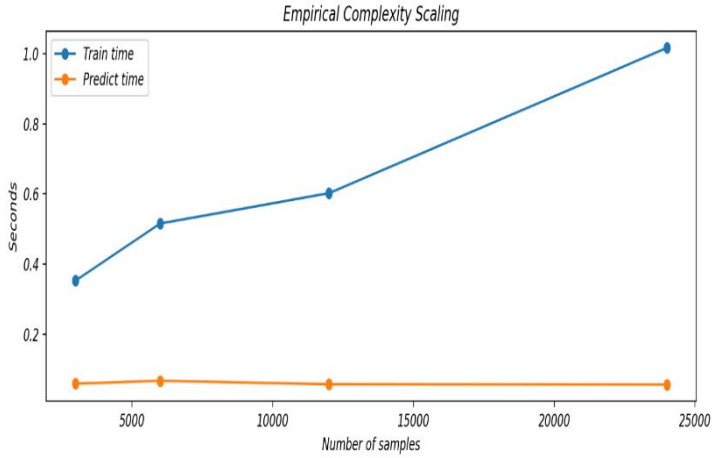

Figure 8. Empirical Computational Complexity Scaling

Figure 8 shows the graph of Empirical Complexity Scaling, where the amount of time to train the data unambiguously increases with the number of samples, while the latency of prediction changes little. As the size of the dataset progressively increases, the training costs increase considerably; however, the prediction times remain at low and constant values, thus emphasising the efficacy of the model for real-time inferencing even when faced with a voluminous amount of data.

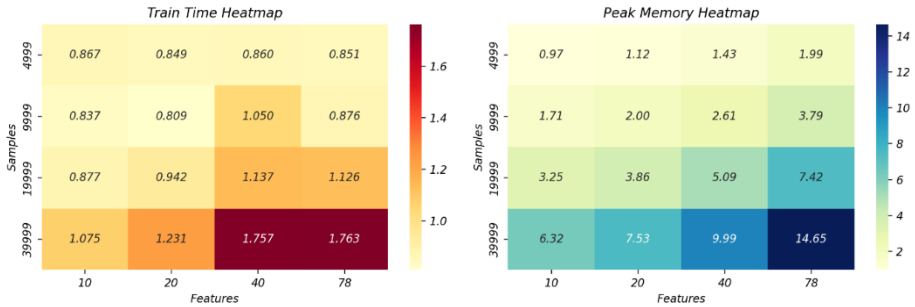

Figure 9. Training Time and Peak Memory Heatmaps for Different Dataset Sizes and Feature Counts

Figure 9 illustrates the training time heatmap and peak memory heatmap, which prescribe the complex interaction between the number of examples in a dataset, the number and dimensionality of features, and the computation requirements. We observe that as the number of samples and feature dimensions both increase, training time and maximum memory usage increase disproportionately, highlighting the increased resource requirements for voluminous datasets.

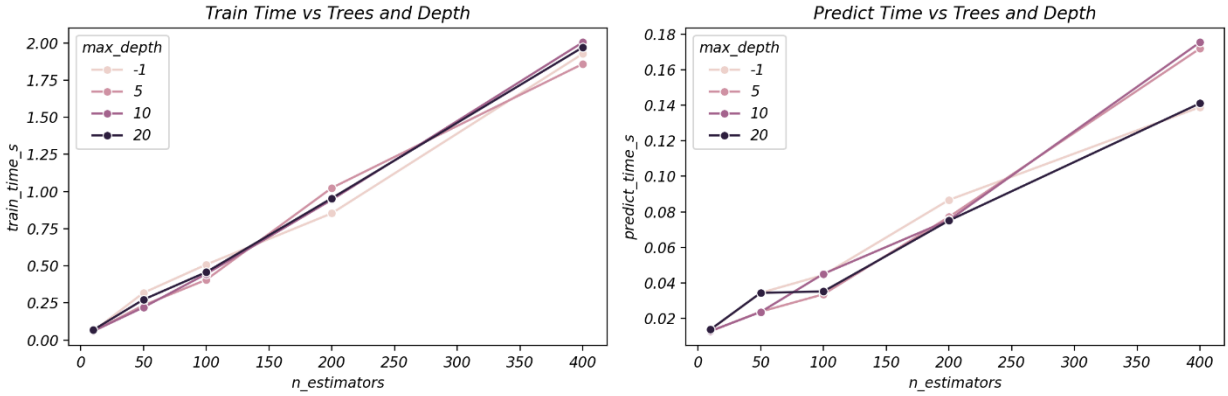

Figure 3. Training and Prediction Time vs Number of Trees and Tree Depth in the Random Forest Model

The impact of the estimator count (trees) and the depth of the model on both training and inference time is shown in the Train Time vs. Trees and Depth and the Predict Time vs. Trees and Depth plots in Figure 10. As the number of trees and the depth grow, the training time and the inference time grow accordingly, with the deepest trees (max depth = 20) giving the highest computational times.

Figure 11 shows the relationship between available computational resources and some crucial parameters, namely, train time vs. samples, train time vs. trees, and peak memory vs. features. The graphs show how these resources scale as the number of samples, trees, and features increase. It is worth noting that although the training time shows a strong effect of increasing sample size and the number of trees, peak memory steadily increases in proportion to the number of features.

The proposed framework was evaluated using various machine learning baselines, such as Logistic Regression, Decision Tree, and Random Forest classifiers, to ensure reliable evaluation. The overall performance was measured by using a well-composed set of performance metrics such as accuracy, precision, recall, F1-score, and ROC-AUC. Besides predictive performance, measures of computational efficiency were also taken, such as training time, inference latency, and scalability with growing datasets. The multi-dimensional evaluation methodology offers insights for both detection effectiveness and deployment feasibility in SDN environments. Additionally, in-corporation of SHAP-based explainability analysis was done to validate the decision making behavior of models by finding the most influential traffic feature that affects intrusion prediction.

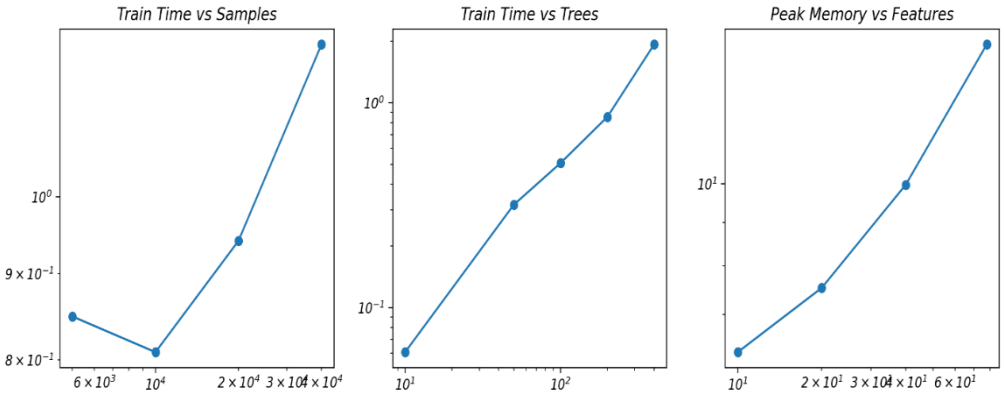

Figure 4. Computational Scaling of Training Time and Memory Usage with Samples, Trees, and Features

6. Discussion

The performance of the proposed framework was also evaluated and compared with some previous SDN intrusion detection methods. There are some existing studies that have been focused on the detection of malicious network activities using machine learning, deep learning, and hybrid architecture. Artificial intelligence techniques such as machine Learning based models (Random Forest and Logistic Regression) have shown potential in detecting attacks with a high detection accuracy, notably in the detection of DDoS and flow based attacks. The approaches, however, typically offer only partial assessment of the cost of computation and interpretability of the model.

Further, deep learning-based IDS approaches have enhanced the detection capability by automatically learning the complex traffic representations. However, their computational needs and lack of transparency can limit their use in real-time SDN networks. While hybrid methods (deep learning + traditional classifiers) boost the detection accuracy, they also need to be thoughtfully implemented with attention to scalability and resource usage.

The proposed framework is a balanced approach that leverages the strengths of all these existing methods, with both high detection capabilities and explainability and computational analysis. Experimental results show that the proposed tree-based models outperform the other models in terms of classification accuracy with an efficient inference behavior. Moreover, the analysis based on SHAP gives an explanation that is easy to understand, since it can identify important traffic features such as destination port, packet length statistics, and TCP window related features. Such a mix of accuracy, efficiency and interpretability is an important benefit over traditional methods that are based mainly on detection performance.

7. Limitations

The proposed explainable machine learning framework can effectively detect intrusions and has high computing efficiency, but there are still some challenges in the practical application of the framework in a real SDN environment. SDN infrastructures tend to have dynamic traffic patterns, dynamic network configurations and continually changing attack behaviors. Hence unseen traffic distributions are important challenges to ensure robustness of the models. Furthermore, adopting IDS solutions based on machine learning coupled with SDN controllers necessitates a trade-off on processing latency, communication overhead and processing resources to prevent IDS systems from having an adverse impact on network performance.

Scalability is another challenge that needs to be addressed. The proposed framework shows that the prediction accuracy is stable as the size of the datasets decreases, but in large-scale enterprise and multi-domain SDN environments, there is a possibility of higher amount of traffic and complexity of features. Moreover, explainability mechanisms, like SHAP, help to gain transparency, which can also lead to extra computational overhead when producing explanations for frequent network flows.

There are also a few limitations of this study. The evaluation was taken from a set of benchmarks SDN traffic data, and more validation should be done in different real-world network environments. Moreover, the existing approach is based on centralized machine learning models, which do not consider distributed learning scenarios in which cypher ability is a key concern. Future studies will focus on federated learning for intrusion detection, adaptive online learning, and lightweight deep learning architectures for increased scalability, privacy and resilience to new cyber threats. Moreover, incorporation with real SDN controllers and evaluation on the large-scale testbeds will be considered to evaluate the practical deployment feasibility.

8. Conclusion

Proposed framework shows good intrusion detection ability, explainability, and computation efficiency, but there are still some problems before its deployment at large-scale operational SDN environments. In real world SDN networks, traffic patterns and applications are constantly changing, network configurations vary in each network, and attack strategies are continually changing and adapting, all of which can impact model generalization. Additionally, scalability is another key factor to consider because as traffic grows and features become more complex, there may be more computational needs.

There are some limitations to this study. The evaluation was performed with a benchmark SDN traffic dataset, with future testing of real-life SDN infrastructures to be done. Furthermore, SHAP can offer insights into model decisions, but generating explanations can add extra computational overhead in high throughput settings.

Future research will focus on developing adaptive learning mechanisms that allow the IDS to continuously update its knowledge based on emerging threats. Moreover, integration with real-time SDN controllers, federated learning-based privacy-preserving detection, and lightweight deep learning approaches will be explored to improve scalability, robustness, and deployment feasibility. These extensions will support the development of intelligent, adaptive, and real-time SDN security solutions capable of responding effectively to future cyber threats.

Authors' Contribution

Suhail Ashfaq Butt and Shakir M. Usman led the design and development of the machine learning framework, including model implementation, evaluation, and computational complexity analysis; Muhammad Ahsan Raza and Qasem M. Kharma contributed to data preprocessing, feature engineering, experimental implementation, and performance evaluation; Sagheer Abbas and Taher M. Ghazal supervised the research, guided the explainable AI analysis, and finalized the manuscript.

Data Availability: https://www.kaggle.com/search?q=SDN+network+traffic+

Conflict of Interest: The authors declare no conflicts of interest regarding the publication of this paper.

Funding Statement: No funding was received for this study.

Declarations

Ethical approval: Not applicable

Consent to participate: Not applicable

Consent to publish: Not applicable

Clinical trial number: Not applicable.

Competing interests: The authors declare no conflicts of interest.

References

1. Ma, R., Wang, Q., Bu, X., & Chen, X. (2023). Real-Time Detection of DDoS Attacks Based on Random Forest in SDN. Applied Sciences, 13(13), 7872. https://doi.org/10.3390/app13137872
2. Mhamdi, L., & Isa, M.M., 2024. Securing SDN: Hybrid Autoencoder-Random Forest for Intrusion Detection and Attack Mitigation. DOI: https://doi.org/10.1016/j.jnca.2024.103868
3. Liu, Z., Wang, Y., Feng, F., Liu, Y., Li, Z., & Shan, Y. (2023). A DDoS Detection Method Based on Feature Engineering and Machine Learning in Software-Defined Networks. Sensors, 23(13), 6176. https://doi.org/10.3390/s23136176
4. Le, T.-T.-H., Kim, H., Kang, H., & Kim, H. (2022). Classification and Explanation for Intrusion Detection System Based on Ensemble Trees and SHAP Method. Sensors, 22(3), 1154. https://doi.org/10.3390/s22031154
5. Ganeshan, S., & Ramasamy, R. K. (2026). A Systematic Review of Machine-Learning-Based Detection of DDoS Attacks in Software-Defined Networks. Future Internet, 18(2), 109. https://doi.org/10.3390/fi18020109
6. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi and M. Ghogho, "Deep learning approach for Network Intrusion Detection in Software Defined Networking," 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, 2016, pp. 258-263, doi: 10.1109/WINCOM.2016.7777224.
7. M. Z. Raihan and M. S. Islam, "Deep Learning-Based DDoS Detection in SDN Networks with Explainable AI Transparency," 2024 27th International Conference on Computer and Information Technology (ICCIT), Cox's Bazar, Bangladesh, 2024, pp. 1328-1333, doi: 10.1109/ICCIT64611.2024.11022440.
8. Yin, C., Zhu, Y., Fei, J., & He, X. (2017) A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. IEEE Access, 5, 21954–21961. https://doi.org/10.1109/ACCESS.2017.2762418
9. Lohachab, A., & Karambir. (2019). ECC based inter-device authentication and authorisation scheme using MQTT for IoT networks. Journal of Information Security and Applications. https://doi.org/10.1016/j.jisa.2019.02.005
10. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A Deep Learning Approach to Network Intrusion Detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 41–50. https://doi.org/10.1109/TETCI.2017.2772792
11. Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2020). Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study. Journal of Information Security and Applications, 50, 102419. https://doi.org/10.1016/j.jisa.2019.102419
12. Saxe, J., & Berlin, K. (2015). Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features. 2015 10th International Conference on Malicious and Unwanted Software (MALWARE). https://doi.org/10.1109/MALWARE.2015.7413680
13. Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems (NeurIPS). https://doi.org/10.48550/arXiv.1705.07874
14. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why Should I Trust You?" Explaining the Predictions of Any Classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. https://doi.org/10.1145/2939672.2939778
15. Kreutz, D., Ramos, F. M. V., Veríssimo, P., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Survey. Proceedings of the IEEE, 103(1), 14–76. https://doi.org/10.1109/JPROC.2014.2371999