



An Enhanced Network Traffic Classification in Software-Defined Networking Using Softmax Regression

Prabu. U^{1,*}, Gorli Uday Kumar¹, Yadlapalli Leelakrishna¹, Pagadam Siddu¹, Geetha. V²

¹ Department of Computer Science and Engineering, Velagapudi Ramakrishna Siddhartha School of Engineering, Siddhartha Academy of Higher Education (Deemed to be University), Vijayawada, – 520 007 India.

² Department of Information Technology, Puducherry Technological University, Puducherry, India – 605 014.

Article's Information	Abstract
Received: 01.08.2025 Accepted: 17.03.2026 Published: 15.09.2026	Network traffic classification in Software-Defined Networking is an essential aspect of SDN and plays an important role in the efficient management and security of modern networks. The classical algorithm for traffic classification based on port numbers cannot handle the increasing encrypted traffic and dynamic port allocation. At the same time, algorithms based on machine learning, for example, flow-based and statistics-based, fail to classify accurately because of the complexity of the multi-class problem. A multi-class traffic classification framework based on the Softmax Regression algorithm is proposed to efficiently identify DNS, Telnet, Ping, and Voice flows in an SDN environment. The algorithm provides a probabilistic method for multi-class classification maintaining both interpretability and high accuracy. Normalization and parameter tuning are carried out to enhance performance of classification. The results of the experiments show that the classification accuracy is 99.68% for the Softmax regression algorithm, while it is 98.53% for the logistic regression algorithm. This illustrates the significance of Softmax regression for multi-class classification and highlights its scalability and reliability for optimizing traffic management in dynamic SDN systems.

Keywords:

Software-Defined Networking,
Network Traffic Classification,
Traffic Management,
Softmax Regression.

<http://doi.org/10.22401/ANJS.29.3.16>

*Corresponding author: uprabu28@gmail.com



This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/)

1. Introduction

The classification of traffic flows has become a very important field of research in current IP-based networks, especially due to the increasing incorporation of machine learning (ML) and SDN architecture into the network [1]. SDN has completely transformed networking by decoupling the control plane (the decision-making process) from the data plane (the forwarding process). This enables centralized controllers to program and regulate the network dynamically for improved routing and resource allocation. Traffic classification (TC) using traditional approaches, for instance, port-based methods, involves matching port numbers to traffic belonging to applications [2]. Although this is easy and effective in early networking systems, it poses great difficulties in today's environment because of

changing ports and encryption of traffic. This has led to the need for more sophisticated systems to classify traffic effectively. ML provides an effective means of addressing these problems. Using mathematical, statistical, and algorithmic concepts, ML allows traffic analysis and traffic classification based on behavior and patterns rather than predefined rules [3]. ML can effectively identify complicated and changing traffic behavior characteristics, classify various applications, and identify any anomalies. It is particularly useful in dealing with dynamically changing traffic on the Internet due to its learning capabilities. The centralized nature of SDN facilitates the gathering of all data and its analysis using ML techniques that then classify the traffic, optimize the usage of resources, and enhance the performance of the network. Thus, SDN combined

with ML techniques provides an effective environment for network engineering. It allows for effective traffic control, increased Quality of Service (QoS), and proactive security strategies.

1.1. SDN Architecture

SDN is a revolutionary network architecture that solves the problems encountered in managing the current generation of complex networks with many

different hardware vendors and their specific configurations. Current networks depend on very tight integration between control plane (CP) and data plane (DP), where the distributed CP operates independently of the DP in each device such as routers and switches. However, SDN solves this problem by separating CP, which makes decisions in the network, from DP [4]. Figure 1 shows the architecture of SDN.

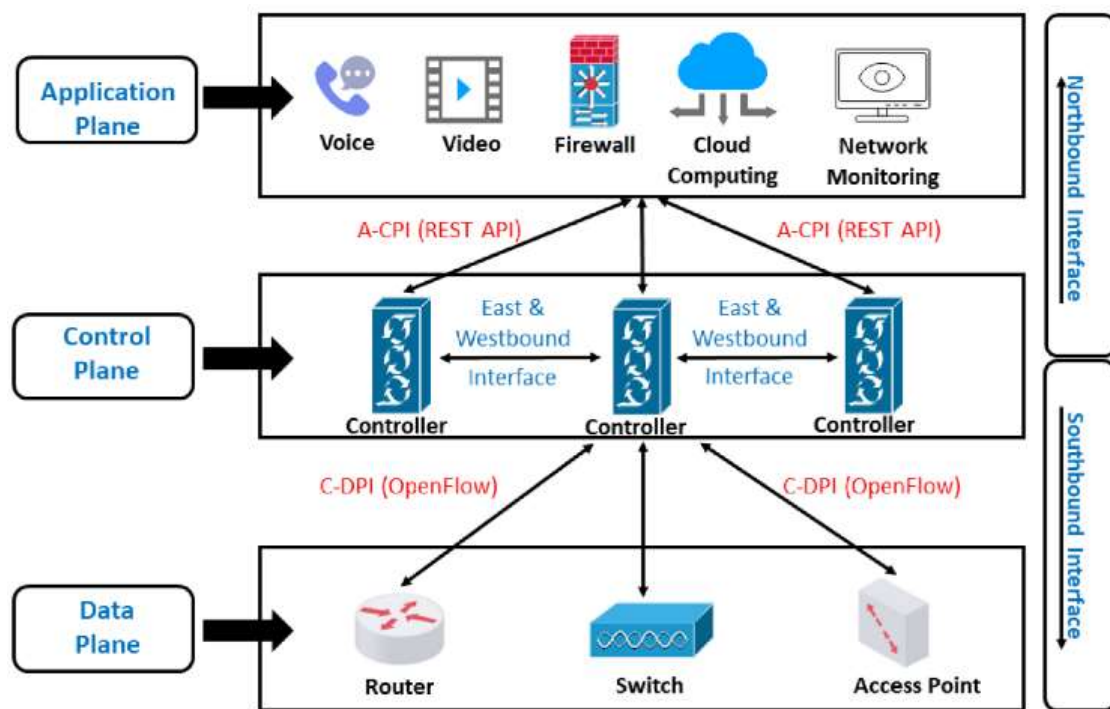


Figure 1. Architecture of SDN [4]

The SDN architecture comprises three primary planes, namely CP, DP, and application plane. CP is defined as a center point that gives one or multiple controllers for the network, and these controllers then control the network operation. CP uses a standardized protocol such as OpenFlow for communicating with DP. Network devices such as routers and switches are available at the DP. Applications along with their interaction with CP through open interfaces for the application controller plane, such as REST API [5] are considered part of the application plane. This abstraction provides applications with a unified, simplified view of the network to allocate resources, optimize traffic and enforce security.

1.2. D-ITG

Distributed Internet Traffic Generator (D-ITG) is an advanced software program that allows generation of

network traffic using IPv4 and IPv6 protocols, such as TCP, UDP, and ICMP. It offers useful performance measurements like one way delay, round trip time, jitter, loss ratio, and throughput. Therefore, D-ITG is indispensable for network traffic analysis. Its distributed structure consisting of ITGSend and ITGRecv components provides remote management, logging capabilities, and customization of packet properties. This software is highly useful in generating traffic data in a controlled environment, which can be exported into CSV format for ML application purposes [6]. For this study, a simulated network setup with one controller, one L2 switch, and three client machines was created to generate network traffic data for applications like Ping, Voice, Telnet, and DNS.

1.3. Network Traffic Classification

Traffic Classification (TC) is an important component of present-day network management since it is required for dealing with growing complexities of modern-day networks and the amount of information transmitted through them. It enables network administrators to manage the resources effectively and ensure the QoS required for various types of applications. Appropriate classification and categorization of the traffic help in allocating bandwidth and prioritize traffic in order to avoid congestion.

Several TC systems are used, ranging from conventional approaches such as the port-based approach, where traffic matching is performed by matching the application to port numbers. Nevertheless, the dynamicity of contemporary applications and encryption of traffic have made these systems inefficient, thereby necessitating more enhanced ones. TC is the fundamental component of several network functions, among them security surveillance, traffic engineering, troubleshooting, network usage billing, and management. Hence, it helps the user access customized services with unique QoS metrics according to the type of network services that they choose. Due to the increasing number of internet users, TC has become even more crucial to efficient network functioning and management.

1.4. Softmax Regression

Softmax regression is an extension of logistic regression [7] to address the limitations of multi-class classification, where the target variable can belong to two or more categories. The softmax activation function, which generalizes the sigmoid function from binary logistic regression, computes a score for each class from the given input instance. After computing the scores for all classes, the softmax function converts them into probabilities, estimating the likelihood that an instance belongs to each class. The model's prediction is the category with the highest probability. Although the Softmax function is commonly used as the final output layer in neural networks, in this work it is applied as an independent multinomial logistic regression model, which is mathematically equivalent but simpler and interpretable. Equation (1) is used to calculate the Softmax score $s_k(x)$ for a specific class k .

$$s_k(x) = x^T \theta^{(k)} \dots (1)$$

After calculating the scores for all classes for the instance x , the probability p_k that the instance belongs to the class k is determined by applying the Softmax function to the scores, as shown in Equation (2).

$$\hat{p}_k = \sigma(s(x))_k = \frac{\exp(s_k(x))}{\sum_{j=1}^K \exp(s_j(x))} \dots (2)$$

where k is the number of classes, $s(x)$ is a vector containing the scores of each class for the instance x . $\sigma(s(x))_k$ is the estimated probability that the instance x belongs to class k for the scores of each class for that instance.

The major contributions of this article are as follows:

- The limitations of the traditional classification method are addressed.
- A Softmax regression algorithm is proposed for classifying network traffic into multiple categories (DNS, Telnet, Ping, and Voice).
- An experimental evaluation is conducted on D-ITG-simulated traffic flows, and a comparative analysis is performed between logistic regression and the proposed Softmax regression to demonstrate the superior performance of Softmax regression for network traffic classification.

The article is organized as follows. The related work on NTC in SDN using ML is discussed in Section 2. Section 3 presents the proposed work. Section 4 describes the implementation and results. Section 5 concludes the article by summarizing key findings and suggesting directions for future research.

2. Related Works

Salau A. O. et al. [8] proposed a model for NTC in SDN using ML. The model classifies DNS, Telnet, Ping, and Voice traffic using supervised algorithms such as decision trees (DT), random forests (RF), and SVMs, as well as unsupervised K-means clustering. Using Mininet for simulation and D-ITG for dataset generation, the process includes data preprocessing, feature extraction, and model evaluation. Decision Trees showed the best accuracy of 99.81% which demonstrates that the model is excellent for real time traffic classification. Adil et al. [9] proposed a study on traffic data classification through ML algorithms in SDN with an objective to improve network service management and security by means of efficient NTC. In the proposed study, various ML algorithms such as k-NN, RF, and CNN are used for traffic data sets such as CICIDS-2017, through feature selection and optimization techniques. Classification accuracy has been found to be high, particularly for RF and CNN models, exceeding 99%. Serag et al. [4] proposed a study on effectively classifying network traffic in SDN through the application of ML algorithms to enhance network performance, quality of service (QoS) and security. The study involves both supervised and unsupervised ML algorithms like RF,

DT and K-Means for NTC. Preprocessing involves extracting features in the form of traffic flow characteristics such as packet sizes and types of protocols. The data sets used in conducting this study are CIC-IDS2017 and other SDN traffic data sets. The results indicate considerable increases in the classification accuracy, scalability and adaptive QoS provisioning, indicating that ML can be used to optimize SDN operation and performance. Ashour M. M. et al. [10] presented the NTC in SDN by employing ML algorithms. The proposed methodology employs ML algorithms including DT, RF, SVM, k-NN, Naive Bayes, and Logistic Regression for classifying network traffic. The dataset of SDN-traffic was used to train and test the ML algorithms. Steps of data preprocessing include data cleaning, normalization, and feature selection to prepare the dataset for better classification. The performance of the proposed algorithms is determined by parameters including accuracy, F1-score, precision, recall, and processing time. The DT model had the highest accuracy of 99.8%. A NTC scheme based on federated learning was presented by Wang et al. [11] for smart home networks. This scheme consists of deep packet inspection (DPI) traffic labeling, semi-supervised model with autoencoder (AE) and CNN, as well as federated learning with gradient aggregation method. Experiments were carried out on public ISCX VPN2016 dataset and private dataset with real-world app traffic after performing pre-processing operations like five-tuple flow extraction and vector generation. The experiments revealed that the proposed scheme achieves higher accuracy (up to 98.6%) compared to other schemes, especially with the lack of labeled data. Moreover, this scheme ensures privacy of the data and model interpretability. Eldhai et al. [12] proposed NTC in SDNs using stream learning (SL) techniques and Boruta-based feature selection (FS). The methodology introduced Hoeffding Adaptive Trees (HAT), Adaptive Random Forest (ARF), and k-NN with ADWIN (KNN-ADWIN) to address concept drift, memory usage, and time consumption, thereby reducing SDN CP overhead. Boruta FS achieved 95% accuracy, outperforming in precision, recall, and F-score. SL methods demonstrated 85% accuracy, with HAT performing best, requiring only 15 seconds and 105 KB of memory.

Yoo et al. [13] proposed Elixir, a ML-based framework for predicting control traffic in SDN systems, to overcome the limitations of descriptive modeling, which suffers from poor accuracy (up to 85% error) and low adaptability. The framework

generates self-created datasets, trains prediction models tailored to specific SDN instances, and selects robust models that maintain accuracy across varied network topologies. The proposed framework is evaluated on nine heterogeneous SDN systems; Elixir achieves a significant reduction in prediction errors, delivering up to a 10.6× improvement in control traffic throughput prediction for the OpenDayLight controller compared to existing models. Belkadi. et al. [14] proposed the ML-based NTC in an SDN and cloud environment to improve network management, security, and QoS. The proposed work uses ML algorithms such as Naive Bayes, Support Vector Machines (SVM), RF, and C4.5 to classify traffic patterns from applications such as Facebook and YouTube using selected features. The supervised algorithms namely Naive Bayes, SVM (SMO), RF, and C4.5 (J48) showed good performance, achieving up to 97% accuracy with studied features and over 95% with generated features. Tonkal. et al. [15] proposed a study to classify network traffic types in SDN using ML to enhance network management, QoS, and security. The proposed study involves generating real-time traffic using the D-ITG tool, extracting features from flows such as packet counts and byte rates, etc., and employing ML algorithms for classification. Preprocessing steps include handling missing values, removing redundant data, and employing Principal Component Analysis (PCA) to reduce feature correlation. The performance of k-NN, SVM, MLP, DT, and NB classifiers was evaluated, and the k-NN achieved the highest accuracy of 99.4%. These results demonstrate the effectiveness of the proposed model in real-time TC. Raikar et al. [16] used supervised machine learning techniques like Naive Bayes, SVM, and k-NN for TC in SDN. Capturing was done with the help of TCPdump, while packet size, duration, and inter-arrival time were used as features. Classifiers were trained on labeled data and were tested with metrics like accuracy, precision, recall, and F-score. Naive Bayes had the highest accuracy of 96.79%, while SVM and k-NN had accuracies of 92.3% and 91.02%, respectively. Margariti et al. [17] proposed a study on TC in the SDN environment employing open SDN traffic traces datasets. Classifier performance is measured based on three aspects: accuracy, detection rate, and precision by applying different classifiers such as MLP (BFGS), FC2 (RBF), FC2 (MLP), DT, SVM, and GenClass. However, among all of these classifiers, the GenClass classifier exhibited excellent performance due to its genetic programming approach in classification rule generation and feature relationship discovery.

A summary of the related works is presented in Table 1. While there have been many successful attempts in achieving high accuracy through the use of complicated classifiers like Random Forests, deep neural networks, and ensemble approaches, these classifiers usually demand high computational power. Furthermore, such classifiers are difficult to interpret in real-time SDN scenarios. On the other hand, this study explores a relatively simple classifier named Softmax Regression Classifier that delivers good performance along with efficiency and transparency in decision-making. This provides a trade-off between accuracy and interpretability, and computational simplicity, making it more feasible for SDN traffic classification.

3. Proposed Work

This research aims at designing a NTC framework through the use of Softmax Regression technique to classify various traffic types like DNS, Telnet, Ping, and Voice. Traffic data is collected using the D-ITG tool, while the performance of the suggested framework is analyzed in terms of Accuracy, Precision, Recall, and F1 Score.

3.1. Problem Formulation

The NTC problem involves identifying and categorizing traffic types flowing through a network. The traffic is classified into four categories such as ping, voice, telnet, and DNS. Each instance contains the feature vector $X = \{x_1, x_2, \dots, x_{12}\}$, comprising features such as Delta Forward Packets, Delta Forward Bytes, etc. The action corresponds to classifying the traffic into one of the target categories $T = \{\text{ping}, \text{telnet}, \text{voice}, \text{DNS}\}$.

The Softmax function is then used to compute the probability of each traffic type.

$$p(T_j/S_i) = \frac{e^{w_j T x + b_j}}{\sum_{k=1}^4 e^{w_k T x_i + b_k}} \dots (3)$$

Here w, b are the weights and biases for the i^{th} traffic type. Traditional methods like One-vs-Rest train separate binary classifiers for each class, ignoring class relationships, while Softmax regression solves this by directly computing a unified probability distribution over all classes and selecting the one with the highest probability in a single step. The Softmax function minimizes entropy loss to optimize w (weight), thereby improving classification by selecting the class with the highest likelihood and ensuring effective, accurate NTC in SDN. The list of notations is depicted in Table 2.

Table 1. A Summary of Related Works

Research Work	Addressed Issue	Algorithm Used	Dataset	Advantages	Disadvantages
[8]	NTC in SDN using ML techniques	LR, DT, RF, AdaBoost, SVM, K-means clustering	Traffic data is generated through the D-ITG tool	Works effectively for both offline and real-time network data.	Applying the model to large-scale SDN environments may face scalability and latency issues.
[9]	TC in SDN	RF, CNN, SVM, DT, k-NN	Moore, USTC-TFC2016, and KDD'99	ML methods outperform traditional approaches by analyzing statistical features.	Most models are tested in controlled environments, and real-world implementation remains challenging.
[4]	ML-based TC in SDN	DT, RF, SVM, NB, k-NN	SDN Dataset	ML-based classification, particularly RF and DT, achieved high accuracy.	While SDN supports ML, handling large-scale network traffic remains challenging.
[10]	NTC in SDN using ML algorithms	DT, RF, SVM, k-NN, NB, LR	SDN-Traffic	Reducing unnecessary features improves classification speed and accuracy.	Some categories (e.g., DNS and FTP) had fewer instances, affecting classification accuracy.

[11]	NTC using Federated Learning and Semi-Supervised Learning	Federated Learning and Semi-Supervised Learning	ISCX VPN2016	Semi-supervised learning enables high performance with fewer labeled samples.	While results on datasets are promising, real-world implementation in large-scale networks remains a challenge.
[12]	TC in SDN with feature selection and stream traffic classification	HAT, ARF, and k-NN	ISCX VPN2016	Feature reduction techniques decreased processing time by 40%, making real-time classification feasible.	Some traffic types (e.g., FTP and DNS) had fewer samples, which affected classification precision.
[13]	ML-based prediction models for traffic in SDN	LR, RF, LightGBM, XGBoost	The dataset is generated using the SDN-DAG framework.	Elixir reduced control traffic prediction errors by 10.6× compared to traditional models.	Selecting the best input features for prediction requires extensive preprocessing.
[14]	NTC in SDN-enabled cloud	NB, SVM, RF	Data is collected through the tcpdump tool.	Random Forest achieved 97.44% accuracy, making it the best classifier.	Real-world implementation in large-scale SDN/cloud networks needs further optimization.
[15]	NTC in SDN using ML algorithms	k-NN, SVM, MLP, DT, NB	Data is generated using the D-ITG tool.	k-NN achieved 99.4% accuracy, the best among tested models.	The dataset focused on only four traffic types, reducing generalizability.
[16]	NTC in SDN using supervised learning techniques	SVM, k-NN, NB	Data is collected using the tcpdump tool.	The study was conducted on live network data, making it suitable for real-world deployment.	The dataset focused only on Browsing, Email, and Streaming, which limits generalizability.
[17]	TC in SDN using Genetic Programming Tools	MLP, DT, SVM, GenClass	SDN-Traffic Dataset.	GenClass achieved 99.42% classification accuracy, surpassing traditional ML models.	Genetic Programming requires significantly more processing time compared to traditional ML models.

Table 2. List of Notations

x_i	Features vector of the i^{th} training sample in the dataset
y_i	True class label corresponding to x_i
X	The entire dataset contains all feature vectors.
Y	The set of all class labels corresponding to the dataset X .
W	Weight matrix of the model
w	Weights for the i^{th} traffic type
b	Bias vector of the model.
η	The learning rate.
T	The number of training iterations (epochs).
Z_i	The logits (raw predictions) for the i^{th} sample, computed as $Z_i = w \times x_i + b$.

$\hat{Y}_i$	The predicted probability distribution over all classes for the i^{th} sample, obtained by applying the Softmax function to Z_i .
L_i	The loss for the i^{th} sample.
$\frac{\partial L_i}{\partial W}$	Weight gradient for the i^{th} sample.
$\frac{\partial L_i}{\partial b}$	Bias gradient for the i^{th} sample.
$Softmax(Z)$	The Softmax function converts the logits z into a probability distribution.
$Argmax(\hat{Y}_i)$	The class with the highest predicted probability is used to determine the predicted class label.
n	Number of samples in the dataset.
d	Number of features of each sample.
C	Number of Classes.

3.2. An Enhanced NTC in SDN using Softmax Regression

A Softmax Regression algorithm is proposed to enhance NTC in SDN. The goal is to classify traffic into predefined classes, such as DNS, Telnet, Ping, and Voice, using Softmax Regression. The workflow comprises extracting key features that influence traffic behavior and calculating Z_i values using a weighted sum of features, normalizing these values to stabilize computation, and applying the Softmax function to generate class probabilities. The final prediction is made by selecting the class with the highest probability. The algorithm introduces a unique method by combining domain-specific feature extraction, robust normalization techniques, and Softmax-based classification, enabling accurate and efficient classification of diverse traffic types in SDN environments. The workflow diagram is shown in Figure 2.

3.3. Workflow of Softmax Regression

In Softmax regression, the dataset is partitioned into features (X) and labels (Y), where each feature vector maps to a class label as shown in Algorithm 1. The model initializes weights (w) and biases (b) and uses the Softmax function to compute probabilities for each class. During training, parameters are iteratively updated to minimize cross-entropy loss, thereby refining decision boundaries for accurate classification. In line 1, the algorithm begins marking the initialization of the process. In line 2, it initializes the parameters W (weights) and b (bias) randomly which are required for the regression model. In line 3, a loop starts from $t = 1$ to T (the total number of iterations) to iteratively optimize the model. In line 4, the linear combination $Z_i = w \times x_i + b$ is computed for each data point x_i , where z_i represents the unnormalized log probabilities. In line 5, the predicted probabilities $\hat{Y}_i$ are calculated using the SoftMax function, which normalizes Z_i to a probability distribution. In line 6, the loss function $L_i = -\log(\hat{Y}_i | y_i|)$ is computed, where $L_i = -\log(\hat{Y}_i)$ measures the negative log-likelihood of the true label y_i . In line 7, the loss function, $\frac{\partial L_i}{\partial w_i} = (\hat{Y}_i - y_i) \times x_i$ is calculated to determine the direction to adjust w . In line 8, the gradient with respect to the bias is computed as $\frac{\partial L_i}{\partial b} = \hat{Y}_i - y_i$, which reflects the bias adjustment needed to minimize the loss. In line 9, the weights w are updated using the learning rate η and the computed gradient $W \leftarrow w - \eta \times \frac{\partial L_i}{\partial w}$. In line 10, the bias b is updated similarly, $b \leftarrow b - \eta \times \frac{\partial L_i}{\partial b}$. In line 11, the loop completes the updates for all iterations, and in line 12, the outer loop concludes. Finally, the algorithm outputs the optimized parameters w and b , which can now be used for predictions. The Time and Space complexities of Algorithm 1 are $O(n \times d \times C)$ and $O(n \times (d + C) + d \times C)$. In the testing phase of Softmax regression, the trained weights (W) and biases (b) are used to compute logits. $Z_{test} = w \times X_{test} + b$ for the given test sample, as shown in Algorithm 2. The Softmax function transforms these logits into class probabilities. The class with the higher probability is selected as the predicted label for the test sample. The Time and Space complexities of Algorithm 2 are $O(d \times C)$ and $O(d \times C)$.

4. Implementation and Results

The results are evaluated across four traffic types: DNS, Telnet, Ping, and Voice. The proposed ML-based classification method is compared with existing algorithms. The performance metrics such as accuracy, precision, recall, and F1-score are considered to evaluate the proposed work.

4.1. Environmental Setup

The experiments are carried out on a Jupyter notebook using a Windows PC equipped with an Intel i5-1135G7 2.40GHz processor and 16 GB RAM. The platform utilizes Python 3.12. The hardware and software specifications for experimentation are listed in Table 3.

4.2. Dataset

D-ITG traffic flows are employed as a dataset [8]. The dataset is augmented with encrypted traffic flows, considering the current security challenges in the real world, and is considered the base for classification. It is comprised of traffic classes, such as Ping, Telnet, DNS, and Voice traffic collected in a systematic manner in a controlled environment. The CSV files containing various traffic features collected in the process of data collection were very helpful in analyzing and classifying the traffic. The dataset included 16 different features: Forward Packets, Forward Bytes, Delta Forward Packets, Delta Forward Bytes, Forward Instantaneous Packets per Second, Forward Average Packets per Second, Forward Instantaneous Bytes per Second, Forward Average Bytes per Second, Reverse Packets, Reverse Bytes, Delta Reverse Packets, Delta Reverse Bytes, Reverse Instantaneous Packets per Second, Reverse Average Packets per Second, Reverse Instantaneous Bytes per Second, and Reverse Average Bytes per Second, representing important attributes of the traffic. Based on feature selection through correlation analysis and variance filter, only 12 features have been selected. Traffic classes have been classified into four types, such as Ping, Telnet, DNS, and Voice traffic, as presented in Table 4.

4.3. Results and Discussions

The proposed Softmax regression algorithm is compared with the logistic regression algorithm. Both algorithms are implemented in the same environmental setup. They are evaluated based on performance metrics such as accuracy, precision, recall, and F1 score.

4.3.1. Confusion Matrix of Softmax Regression

The confusion matrix for Softmax Regression demonstrates outstanding performance, with the majority of predictions correctly classified along the diagonal for DNS, Ping, Telnet, and Voice classes. This indicates the algorithm's effectiveness in distinguishing between the four traffic types. The confusion matrix is presented in Figure 3.

Algorithm 1: Softmax Regression

Input: X, Y, T	
1:	Start
2:	Initialize w and b randomly
3:	for $t = 1$ to T do
4:	$Z_i = w \times x_i + b$
5:	$\hat{Y}_i = \text{Softmax}(Z_i)$
6:	$L_i = -\log(\hat{Y}_i[y_i])$
7:	$\frac{\partial L_i}{\partial w_i} = (\hat{y}_i - y_i) \times x_i$ // Compute the gradients
8:	$\frac{\partial L_i}{\partial b} = \hat{y}_i - y_i$ // Compute the gradients
9:	$W \leftarrow w - \eta \times \frac{\partial L_i}{\partial w}$ // Update Parameters
10:	$b \leftarrow w - \eta \times \frac{\partial L_i}{\partial b}$ // Update Parameters
11:	end for
12:	end for
Output: Learned W and b	

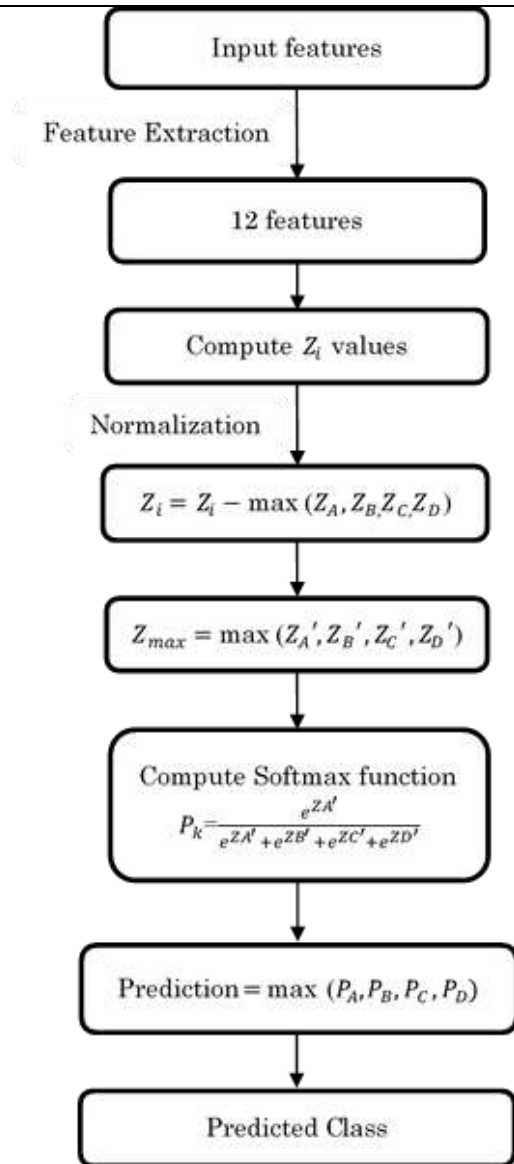


Figure 2. Workflow Diagram

Algorithm 2: Testing a sample using Softmax regression

Input: X_{test}, w, b					
1:	Start				
2:		$Z_{test} = w \times x_{test} + b$			
3:			$\widehat{y}_{test} = \text{softmax}(Z_{test})$		
4:				$y_{pred} = \text{argmax}(\widehat{y}_{test})$	
	Output: y_{pred}				

Table 3. Hardware and Software Specifications for Experimentation

Component	Specification
Processor	Intel i5-1135G7, 2.40GHz
RAM	16 GB
Environment	Windows
Programming Language	Python 3.12
Platform	Jupyter Notebook
Storage Device	256 GB SSD

Table 4. Network traffic data

Traffic Type	Instances
Ping	1770
DNS	1181
Voice	1154
Telnet	1137

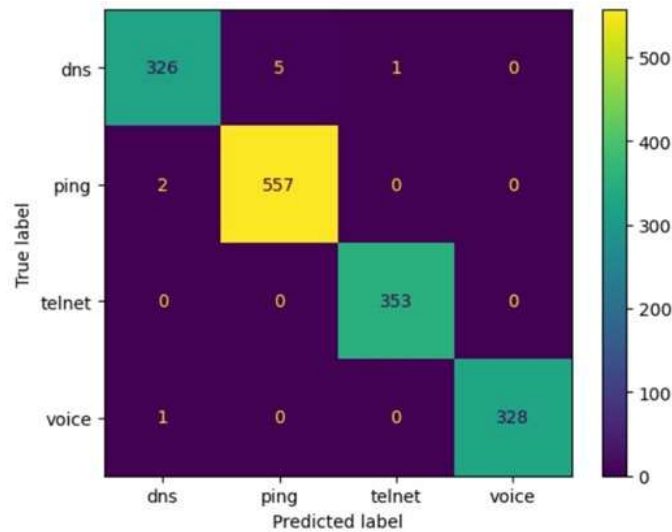


Figure 3. Confusion matrix

4.3.2. Performance analysis of logistic regression and softmax regression

To assess the NTC performance, evaluation metrics such as accuracy, precision, recall, and F1 Score is used. This will provide information about the effectiveness of the model in classifying various traffic types like Ping, Telnet, DNS, and Voice under distinct scenarios.

A. Accuracy: The accuracy metric evaluates the frequency at which the model makes predictions that are right. This metric is obtained through the ratio between the number of correct predictions and the total number of predictions.

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Instances}}$$

The graph for the accuracy comparison shows how well these two regression models performed throughout various epochs. The accuracy of Softmax Regression is high during the early stages of the training process, which proves that it performs better compared to Logistic Regression in terms of learning and generalization. The accuracy of Logistic Regression is increasing continuously but still trails that of Softmax Regression. Softmax Regression performs better than Logistic Regression with an accuracy of 99.68% compared to 98.53%.

Figure 4 depicts accuracy comparison between Logistic Regression and Softmax Regression. The increasing accuracy means Softmax regression has stronger ability to separate traffic kinds such as DNS, Telnet, Ping and Voice, by predicting the class label accurately in more cases.

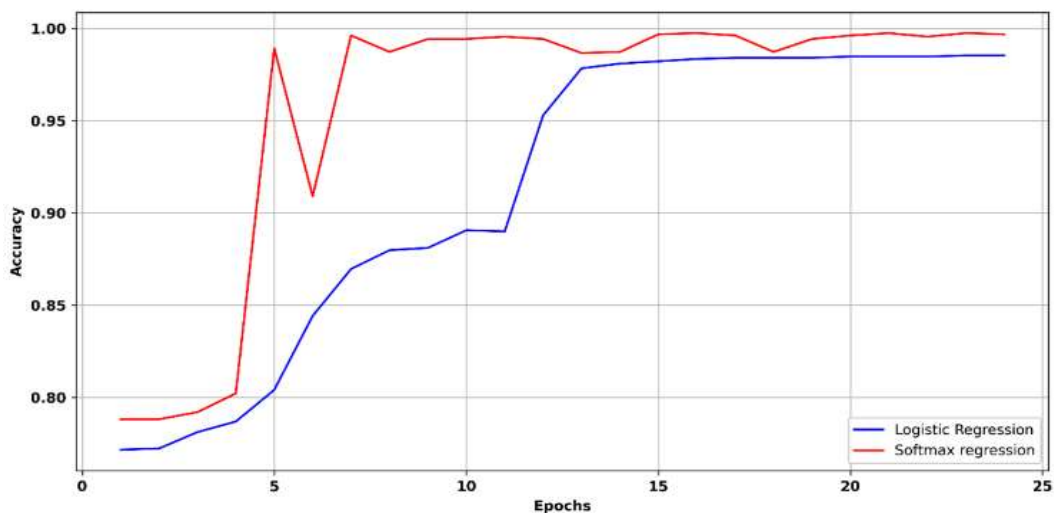


Figure 4. Logistic Regression vs Softmax Regression

- B. Precision: Precision is a measure of correct positive estimates. The emphasis is on the model's ability to avoid false positives.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

The bar graph in Figure 5 shows the precision performance of Softmax Regression and Logistic

Regression on four different network traffic categories; DNS, Ping, Telnet, and Voice. There are similarities in the precision of both models since they perform extremely well with little difference among themselves. The improvement in precision means that the Softmax regression is a good classifier for various types of network traffic such as DNS, Telnet, Ping, and Voice.

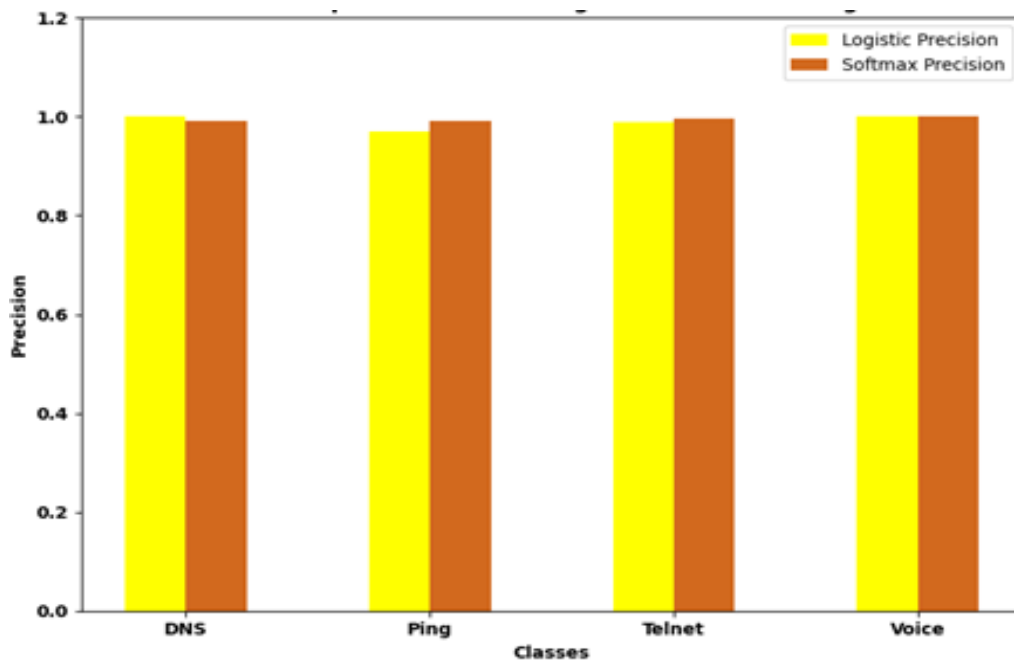


Figure 5. Precision comparison between Logistic and Softmax regression

- C. Recall: Recall measures the model's ability to identify all relevant positive instances. It indicates how many actual positives the model correctly identified.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Figure 6 shows the comparison of the recall scores of Logistic Regression and Softmax Regression for the

four network traffic classes: DNS, Ping, Telnet, and Voice. Both methods demonstrate a consistently high recall for all classes, which means that they are able to effectively identify real positive cases. There is a small difference observed between the two strategies. It can be seen that the improvement in recall shows that SoftMax regression is more efficient in identifying all types of network traffic including DNS, Telnet, Ping and Voice and therefore reducing the misclassification rates.

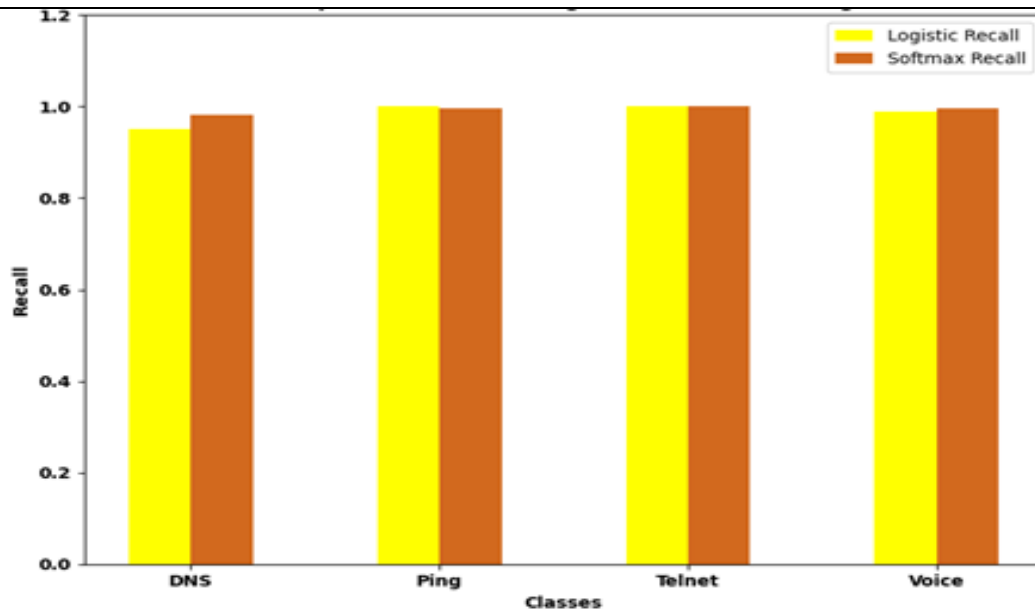


Figure 6. Recall comparison between Logistic and Softmax regression

D. F1-Score: The F1-score is an evaluation metric that combines Precision and Recall into a single value using their harmonic mean, making it effective for handling imbalanced datasets.

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

Figure 7 shows a comparison between the F1 Scores of both Logistic Regression and Softmax Regression using four categories of network traffic; these include DNS, Ping, Telnet, and Voice. The two models have almost equal F1 scores, which are fairly high as well. The rise in the value of the F1 score represents the improvement made by the Softmax Regression model in terms of Precision and Recall.

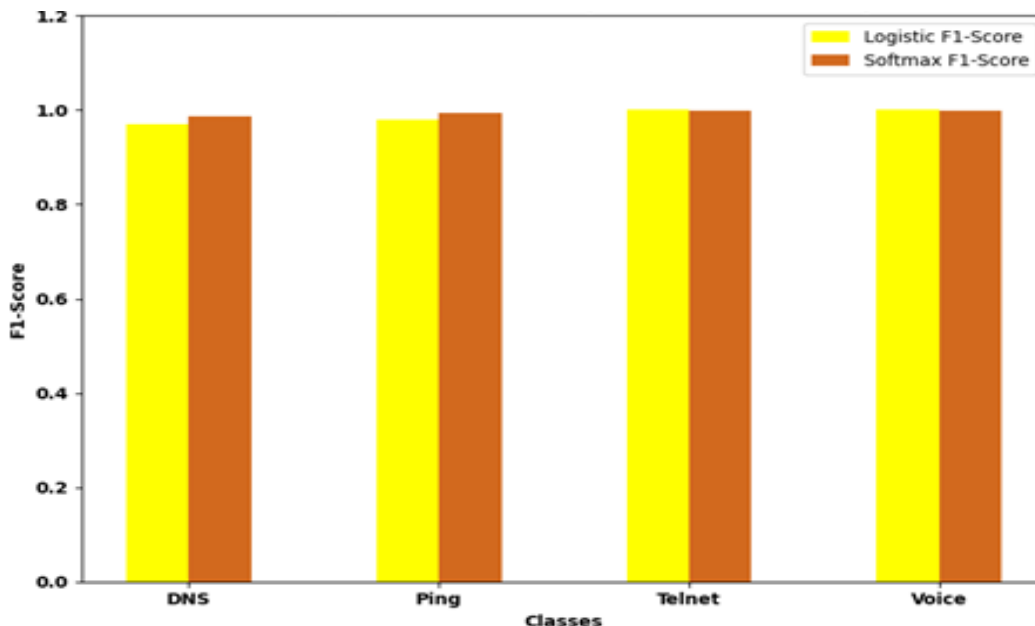


Figure 7. F1-Score comparison between Logistic and Softmax regression

5. Conclusions

Network Traffic Classification in Software-Defined Networking is an important task for managing and securing the network. This research demonstrates the efficacy of Softmax regression for classifying network traffic into four classes: DNS, Telnet, Ping, and Voice. Compared with Logistic regression, Softmax regression achieved higher results across all measures and achieved a 1.15% higher accuracy, improving predictive capacity. Precision improved by 0.99%, indicating that positive predictions became more reliable and less prone to error. Recall increased by 1.02%, indicating that Softmax Regression recognized more positive cases and improved on the previous result. Finally, the F1 score increased by 1.01%. It means that Softmax Regression improved both precision and recall. These results allow us to conclude that Softmax regression can be used to advance traffic classification in SDN and improve network performance. Future work will focus on enhancing the proposed network traffic classification framework by incorporating advanced optimization techniques for automated hyperparameter tuning, thereby improving classification accuracy and computational efficiency. Adaptive learning mechanisms will also be integrated to enable continuous model updates and real-time traffic classification in dynamic SDN environments. Furthermore, the model will be extended to support encrypted traffic and emerging Internet of Things (IoT) communication protocols, improving its capability to classify diverse and evolving network traffic patterns. These enhancements are expected to increase the robustness, scalability, and practical applicability of the proposed framework in next-generation SDN-based networks.

Conflicts of Interest: The authors declare no conflict of interest.

Funding: No funds, grants, or other support were received.

References

- [1] Aldaoud, M.; Al-Abri, D.; Awadalla, M.; Kausar, F.; "Leveraging ICN and SDN for Future Internet Architecture: A Survey". *Electronics*, 12 (7): 1723, 2023.
- [2] Ferreira, G.O.; Ravazzi, C.; Dabbene, F.; Calafiore, G.C.; Fiore, M.; "Forecasting Network Traffic: A Survey and Tutorial with Open-Source Comparative Evaluation". *IEEE Access*, 11 (1): 6018–6044, 2023.
- [3] Azab, A.; Khasawneh, M.; Alrabaei, S.; Choo, K.K.R.; Sarsour, M.; "Network Traffic Classification: Techniques, Datasets, and Challenges". *Digit. Commun. Netw.*, 10 (3): 676–692, 2024.
- [4] Serag, R.H.; Abdalzaher, M.S.; Elsayed, H.A.E.A.; Sobh, M.; Krichen, M.; Salim, M.M.; "Machine-Learning-Based Traffic Classification in Software-Defined Networks". *Electronics*, 13 (6): 1108, 2024.
- [5] Abdi, A.H.; Audah, L.; Salh, A.; Alhartomi, M.A.; Rasheed, H.; Ahmed, S.; Tahir, A.; et al.; "Security Control and Data Planes of SDN: A Comprehensive Review of Traditional, AI and MTD Approaches to Security Solutions". *IEEE Access*, 12: 69941–69980, 2024.
- [6] Sokolov, Y.A.; Pugach, A.V.; Baykalova, N.A.; Gaipov, K.E.; Knyazeva, I.O.; Patsel, D.E.; Lukevich, D.R.; et al.; "Computational Analysis of the Influence of Internet Traffic Flows in IP Networks". In: *AIP Conf. Proc.*, Melville, NY, USA, 12 January 2024; Igor Kovalev and Anna Voroshilova., Eds.; AIP Publishing; 2024; AIP Publishing: Melville, NY, USA, 2024.
- [7] Shoba, V.B.T.; Sam, I.S.; "Adaptive Deep Feature Learning-Based Softmax Regressive Classification for Aging Facial Recognition". *Multimed. Tools Appl.*, 82 (15): 22343–22371, 2023.
- [8] Salau, A.O.; Beyene, M.M.; "Software-Defined Networking-Based Network Traffic Classification Using Machine Learning Techniques". *Sci. Rep.*, 14 (1): 20060, 2024.
- [9] Adil, S.; Alfoudi, A.S.; "Traffic Data Classification in SDN Network Based on Machine Learning Algorithms". *Wasit J. Pure Sci.*, 3 (2): 161–171, 2024.
- [10] Ashour, M.M.; Yakout, M.A.; Abdelhalim, E.; "Traffic Classification in Software-Defined Networks Based on Machine Learning Algorithms". *Int. J. Telecommun.*, 4 (1): 1–19, 2024.
- [11] Wang, Z.; Li, Z.; Fu, M.; Ye, Y.; Wang, P.; "Network Traffic Classification Based on Federated Semi-Supervised Learning". *J. Syst. Archit.*, 149 (1): 103091, 2024.
- [12] Eldhai, A.M.; Hamdan, M.; Abdelaziz, A.; Hashem, I.; Marsono, M.N.; Hamzah, M.; Jhanjhi, N.Z.; "Improved Feature Selection and Stream Traffic Classification Based on Machine Learning in Software-Defined Networks". *IEEE Access*, 12: 34141–34159, 2024.
- [13] Yoo, Y.; Yang, G.; Shin, C.; Lee, J.; Yoo, C.; "Machine Learning-Based Prediction Models for Control Traffic in SDN Systems". *IEEE Trans. Serv. Comput.*, 16 (6): 4389–4403, 2023.

- [14] Belkadi, O.; Vulpe, A.; Laaziz, Y.; Halunga, S.; "ML-Based Traffic Classification in an SDN-Enabled Cloud Environment". *Electronics*, 12 (2): 269, 2023.
- [15] Tonkal, Ö.; Polat, H.; "Traffic Classification and Comparative Analysis with Machine Learning Algorithms in Software-Defined Networks". *Gazi Univ. J. Sci. Part C*, 9 (1): 71–83, 2021.
- [16] Raikar, M.M.; Meena, S.M.; Mulla, M.M.; Shetti, N.S.; Karanandi, M.; "Data Traffic Classification in Software-Defined Networks Using Supervised Learning". *Procedia Comput. Sci.*, 171 (1): 2750–2759, 2020.
- [17] Margariti, S.V.; Tsoulos, I.G.; Kiouisi, E.; Stergiou, E.; "Traffic Classification in Software-Defined Networking Using Genetic Programming Tools". *Future Internet*, 16 (9): 338, 2024.