ResSDN: A Machine Learning-based Tool for Joint QoS-Security Optimization in Software-Defined Networking

by Jingting Ye

B.A. in International Security and Conflict Resolution, August 2018, San Diego State University
M.S. in Cybersecurity, May 2020, Fordham University

A Praxis submitted to

The Faculty of
The School of Engineering and Applied Science of
The George Washington University
in partial fulfilment of the requirements for
the Degree of Doctor of Engineering

May 18, 2025

Praxis directed by

Buddha Nepal
Professorial Lecturer of Engineering Management and Systems Engineering

The School of Engineering and Applied Sciences of The George Washington University certifies that Jingting Ye has passed the Final Examination for the degree of Doctor of Engineering as of March 22nd, 2025. This is the final and approved form of the Praxis.

ResSDN: A Machine Learning-based Tool for Joint QoS-Security Optimization in Software-Defined Networking

Jingting Ye

Praxis Research Committee:

Amir Etemadi, Associate Professor of Engineering and Applied Science, Committee Chair

Buddha Nepal, Professorial Lecturer in Engineering and Applied Science, Committee Advisor

Joseph Blackford, Associate Professor of Engineering and Applied Science, Committee Member

© Copyright 2025 by Jingting Ye
All rights reserved.

Dedication

The author would like to deeply thank her parents, Sam and Lily, for being an unshakeable pillar of support during her doctoral experience, to whom she lovingly dedicates this praxis. In addition, she dedicates this work to Steelo.Z, whose dedication to his musical career is evidently proved through his commitment and perseverance, encouraging her with his bravery in fighting against adversity and his relentless pursuit of excellence.

Acknowledgments

The author owes deep gratitude to her practice mentor, Dr. Buddha Nepal, for his productive recommendations, unrelenting support, and patient persistence, all being the basic requirements for the development process of this study. In addition, his mentorship has been the key source of illumination throughout her academic life.

Abstract of Praxis

ResSDN: A Machine Learning-based Tool for Joint QoS-Security Optimization in Software-Defined Networking

The Quality of Service (QoS) paradigm plays a vital role within the context of Software-Defined Networking (SDN), as it facilitates the efficient allocation of network resources to satisfy the diverse performance demands of various applications and services. Concurrently, it is imperative to address security concerns within the SDN framework, given that the distinctive architecture of SDN introduces novel vulnerabilities and associated risks, including control plane vulnerabilities, data plane vulnerabilities, and threats such as Distributed Denial-of-Service attacks. Nevertheless, the trade-off between security and QoS in SDN is daunting because the traditional QoS management methods, which rely primarily on static configuration and heuristics, cannot manage to optimize the allocation of resources to SDN traffic patterns with dynamic security controls, influencing the overall user experience (Bagaa et al., 2022). The incapacity to properly distribute resources for the implementation of security mechanisms under different network situations is a serious risk, which can leave the network vulnerable to intrusion and security breach.

This paper introduces ResSDN, a framework that incorporates Machine Learning into SDN network traffic routing and management to enhance both QoS and security in the scenario of complicated SDN. Through the joint use of novel Machine Learning algorithms, ResSDN preemptively blocks the malicious traffic and offers instructions on routing decisions on the benign traffic, thus protecting against the cyber threats while ensuring a smooth user experience.

Table of Contents

Acknowledgments ..... **v**

Abstract of Praxis ..... **vi**

Table of Contents ..... **vii**

List of Figures ..... **x**

List of Tables ..... **xi**

List of Acronyms ..... **xii**

Glossary of Terms ..... **xiii**

Chapter 1: Introduction **1**

1.1 Background 1

1.2 Research Motivation 2

1.3 Problem Statement 3

1.4 Thesis Statement 4

1.5 Research Objectives 4

1.6 Research Questions and Hypotheses 5

1.7 Scope of Research 6

1.8 Research Limitations 7

1.9 Organization of Praxis 7

Chapter 2: Literature Review **9**

2.1 Introduction 9

2.2 Related Work 1 10

2.3 Related Work 2 13

2.4 Related Work 3 15

2.5 Related Work 4 18

Chapter 3: Methodology....22

3.1 Introduction....22

3.1.1 Justification for the Two-Stage Approach 24

3.1.2 Challenges Addressed by the Framework....24

3.1.3 High-Level Overview of Data Flow 25

3.2 Stage 1 Model for Intrusion Detection....27

3.2.1 Data Collection 28

3.2.2 Data Preprocessing....29

3.2.3 Model Training 31

3.2.4 Model Evaluation and Validation 34

3.3 Stage 2 Model for QoS Routing....36

3.3.1 Data Collection 37

3.3.2 Data Preprocessing....38

3.3.3 Model Training 38

3.3.4 Model Evaluation and Validation 41

Chapter 4: Results....43

4.1 Model Selection for Stage 1....43

4.1.1 Model Evaluations in Each Class 44

4.1.2 Overall Performance 47

4.1.3 Recap of Research Question#1 and Hypothesis#1 49

4.2 Model Selection for Stage 2 51
4.2.1 Evaluation for Each Class 53
4.2.2 Overall Performance 57
4.2.3 Recap of Research Question#2 and Hypothesis#2 59
4.3 Tool Assessment 62
4.3.1 Simulation Setup 63
4.3.2 Data Preprocessing 65
4.3.3 ResSDN Model Training 68
4.3.4 ResSDN Model Evaluation and Validation 70
4.3.5 Recap of Research Question#3 and Hypothesis#3 76
Chapter 5: Conclusion 79
5.1 Results Summary and Analysis 79
5.2 Contributions to the Body of Knowledge 79
5.3 Recommendations for Future Research 80
5.4 Conclusion 81
References 82
Appendix A 92

List of Figures

Figure 1 – Workflow of ResSDN 27

Figure 2 - AUC-ROC Curves of Four Trained Models in Stage 1 35

Figure 3 - AUC-ROC Curves of Four Trained Models in Stage 2 42

Figure 4 - Simulated SDN Topology in GNS3 65

List of Tables

Table 1 - Stage 1 Attack Types Classification 29

Table 2 - Stage 2 Application Categories Classification 38

Table 3 - Precision, Recall, and F1 scores of Each Evaluated Machine Learning Models in Each Class in Stage 1 47

Table 4 - Accuracy, CV Scores, and ROC-AUC Scores of All Machine Learning Models Trained in Stage 1 48

Table 5 - Precision, Recall, and F1 scores of Each Evaluated Machine Learning Models in Each Class in Stage 2 56

Table 6 - Accuracy, CV Scores, and ROC-AUC Scores of Each Evaluated Machine Learning Models in Stage 2 57

Table 7 - Evaluation Scores of Each Class in Stage 1 72

Table 8 - Performance Summary of Stage 1 72

Table 9 – Evaluation Scores of Each Class in Stage 2 74

Table 10 - Performance Summary of Stage 2 74

Table 11 - ResSDN Security-QoS Trade-Off Analysis 76

List of Acronyms

Adam	Adaptive Moment Estimation
AI	Artificial Intelligence
AUC	Area Under the Curve
CV	Cross-Validation
DDoS	Distributed Denial of Service
DoS	Denial of Service
DNN	Deep Neural Network
FTP	File Transfer Protocol
GNS3	Graphical Network Simulator-3
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection Systems
KNN	K-Nearest Neighbor
LightGBM	Light Gradient Boosting Machine
QoS	Quality of Service
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristic
SDN	Software-Defined Networking
SMOTE	Synthetic Minority Over-sampling Technique
SVM	Support Vector Machine
XGBoost	eXtreme Gradient Boosting
XSS	Cross-site scripting

Glossary of Terms

Software-Defined Networking	A networking paradigm that separates the control plane from the data plane, enabling centralized management, programmability, and agility in network operations.
Hyperparameter Tuning	The process of optimizing the settings of an AI model to improve its accuracy and efficiency.
Machine Learning	A subset of AI that enables systems to learn from data and improve performance over time.
Model Training	The process of teaching an AI model by exposing it to data and adjusting its parameters.
OpenFlow	A protocol widely used in SDN architectures for communication between the control plane and network devices, allowing the controller to program forwarding rules and policies.
Overfitting	A modeling error where an AI model learns patterns too specific to the training data, making it ineffective on new data.
Reinforcement Learning	A machine learning system where the agent acquires skills by interacting and receiving feedback.
Supervised Learning	A machine learning approach where a model is created using data that is labeled.

Chapter 1: Introduction

1.1 Background

Software-Defined Networking (SDN) is a new networking paradigm that supports the programmability and centralization of networking infrastructure, thus enabling dynamic, flexible, and scalable management of network resources (Elsayed et al., 2020). However, the dynamic nature of traffic patterns coupled with the ever-changing nature of threats poses significant hurdles in optimizing Quality of Service (QoS) and Security simultaneously (Bagaa et al., 2022). Previously, these have been treated in isolation, and compromises have been made, resulting in inefficiencies. The growing integration of the two dimensions in today's networked environments emphasizes the need to optimize them in unison, making the method both holistic and much more important.

The advent of Machine Learning methods has profoundly impacted the fields of network management and security (Fadlullah et al., 2017). Machine Learning has the ability to identify intricate patterns in datasets, learn from evolving circumstances, and enforce data-informed decisions via predictive analytics. The spectrum of these methods encompasses both traditional algorithms and deep learning architectures, which show phenomenal efficiency in process automation and performance optimization (Hammad et al., 2023). By applying Machine Learning algorithms to data analysis, relationships can be revealed between QoS parameters and security threats. This will facilitate the creation of adaptive and context-aware solutions, e.g., runtime adjustments of QoS parameters, while at the same time enhancing security measures by scrutinizing network traffic patterns and detecting anomalies.

The convergence of Machine Learning and SDN holds great potential for revolutionizing network management structures, security measures, and optimization models. In SDN, Machine Learning enables proactive traffic flow management, instant threat detection, and dynamic resource allocation, leading to increased operational efficacy and resilience. Utilizing Machine Learning algorithms makes it possible to develop Artificial Intelligence-based applications supporting real-time decision-making that maximizes both service delivery and security (Rui et al., 2023). It is possible to use this approach to maximize network performance and resilience while simultaneously strengthening protection against the ever-changing threat environment.

1.2 Research Motivation

SDN has become a revolutionary framework that promotes better control and management of network structures (Kaliyamurthy et al., 2021). Nevertheless, the convergence of QoS and security into SDN is fraught with complex challenges. The need for this research arises from the need to overcome the rapidly changing and complex nature of contemporary network environments. Increased applications of multiple applications in addition to increased security threats necessitate the need for comprehensive optimization of QoS and security in SDN (Aleem & Ahmed, 2023). The application of Machine Learning approaches provides a solid ground for designing adaptive systems that would adapt to real-time changes in network conditions and related security issues. Forming a Machine Learning-based framework for simultaneous optimization of QoS and security in SDN provides the foundation for boosting the performance and robustness against security threats in the network (Fadlullah et al., 2022).

The development in this area keeps advancing, and the successful development of a system that utilizes Machine Learning to simultaneously improve both QoS and Security in SDN would have important implications in different fields, such as telecommunications, cloud computing, and critical infrastructure. For each of these fields, the combined management of QoS and Security is critical in providing stable and secure network operations (Aleem & Ahmed, 2023).

1.3 Problem Statement

Traditional solutions to QoS management that ensure sufficient cybersecurity readiness within SDN are hampered by the complex and dynamic nature of current networks. Thus, this challenge amounts to an annual loss of more than $700,000 for big business (Netround, 2019).

Traditional approaches to managing QoS generally depend on fixed configurations and heuristic-based approaches (Khamosh et al., 2020), which can insufficiently optimize the allocation of resources for the adaptive security demands of SDN traffic flows. This can potentially negatively affect the quality of the overall user experience. In addition, the problem of optimally allocating resources to enact security measures under dynamic network conditions presents a key threat, leaving the network exposed to attacks and intrusions (Tu et al., 2019). In order to overcome these challenges, there is a need to employ QoS management practices that entail more adaptive methodologies.

1.4 Thesis Statement

Suggesting the approach for simultaneous optimization of both SDN cyber maturity and QoS based on Machine Learning models that minimize the threats against the system and optimize the resource allocation.

Proposed tool, ResSDN, offers adaptive proactive network resource management and security strategy. Based on two-stage Machine Learning approach, the ResSDN first identifies and removes the malicious traffic and maintains the integrity and security of the network. Subsequently, the remaining legitimate traffic gets classified by application type and priority levels are allotted for better QoS management. Not only is the process eliminating threats to the security in real-time, the network performance also gets optimized by dynamically adaptive routing plans for the traffic. Models also leverage the history-based network records for optimization of the classification accuracy and dynamically update the QoS parameters accordingly, thereby maintaining better resilience against new-age cyber threats.

1.5 Research Objectives

The main purpose of this research is to provide an in-depth and systematic analysis of the security issues related to SDN structures, and in particular the optimization and management of SDN traffic. Throughout this research work, the authors propose a two-stage Machine Learning model: the first stage includes the identification and elimination of malicious traffic to enhance the protection of the network; the second stage includes the classification of legitimate traffic by application and the assignment of priority levels for the improvement of QoS. Based on the foundation of Machine Learning algorithms, the research aims to provide intelligent systems that are capable of

learning changes in the network and dynamically modifying QoS parameters in addition to real-time reaction towards new threats. By using Machine Learning techniques, adaptive and reactive models will be created and the overall performance and protection of the SDN system will improve (Gelenbe et al., 2020). Based on empirical research and new approaches, the research work plans to empower SDN network administrators in order to strengthen operational performance and protection indicators. Finally, the research plans to build a more robust and protected SDN structure that is capable of acting against new threats and needs while maintaining stable operation and dependability.

1.6 Research Questions and Hypotheses

Against the backdrop of the exponentially changing cyber environment ubiquitous in the modern world, innovations in SDN management call for novel approaches beyond conventional structures (Coman & Dobrota, 2022). As such, this chapter seeks to develop a unified framework to examine the research questions and hypotheses that will propel the study towards a holistic understanding of the integration of Machine Learning approaches for the simultaneous optimization of QoS and security in SDN.

RQ1: Among Random Forest, Decision Tree, Support Vector Machine (SVM), and Deep Neural Network (DNN), which is the best approach for evaluating and enhancing the cyber maturity of SDN environments through the detection of intrusions?

RQ2: Among Random Forest, Light Gradient Boosting Machine (LightGBM), eXtreme Gradient Boosting (XGBoost), and K-Nearest Neighbor (KNN), which is able to assess and improve the QoS of the SDN environments using priority allocation?

RQ3: Can the proposed joint QoS-Security Optimization tool, ResSDN, effectively enhance cyber maturity and QoS in SDN networks?

H1: Out of Random Forest, Decision Tree, SVM, and DNN, Random Forest model can best evaluate and enhance the cyber maturity of the SDN environments by predicting intrusions.

H2: Among Random Forest, LightGBM, XGBoost, and KNN, XGBoost is able to assess and improve the QoS of the SDN environments most effectively by assigning priorities.

H3: The proposed joint QoS-Security Optimization tool, ResSDN, is able to enhance cyber maturity and QoS in SDN environments effectively by resolving security threats while optimizing resource allocation.

1.7 Scope of Research

The scope of the research involves a comprehensive assessment of security challenges in SDN infrastructures with specific focus on the extended SDN management plane. Specifically, the research aims at examining the evolving threat landscape and QoS parameter optimization in SDN networks. The research will employ a set of feature selection, hyperparameter search, supervised learning, reinforcement learning, and Cross-Validation (CV) to quantify solution performance. The objective is to present a whole research paper, ranging from literature reviews of related work through industry discussion, the proposal of a two-stage SDN management model, and indicating recommendations for future work. The proposed framework leverages Machine Learning first for identifying and dropping malicious traffic to enhance security before benign traffic classification and prioritization on a per-application basis to realize optimum QoS. Furthermore, the paper will include relevant appendices such as a glossary, infrastructure

diagram, threat model, and evaluation criteria. The goal here is to make the SDN systems more resilient towards new threats and also optimize the network performance by incorporating the careful application of Machine Learning techniques.

1.8 Research Limitations

Development of machine learning-based tools for simultaneous improvement of QoS and security in SDN environments has faced several critical roadblocks. One main roadblock relates to the availability and quality of training and test data. Lack of diversity and relevance in existing datasets that poorly capture the complex interdependencies among QoS measures and the dynamic nature of the threats typical of SDNs pose threats often to the integrity and usability of the resultant models (Ahmad & Mir, 2020). In addition, computational capacity restraints inherent in the SDN system negatively influence the deployment and use of machine learning models for simultaneous optimization of QoS and security measures within the system because of real-time processing capacity limitations, energy expenditure, and latency allowances (Wakili et al., 2024).

1.9 Organization of Praxis

A comprehensive literature review of four appropriate research papers examining the application of Machine Learning algorithms for addressing security and QoS problems in SDN settings or other environments such as cloud service and 5G is conducted in Chapter 2. The review aims to identify and critique common gaps in the current literature and where solutions are lacking and how this proposed approach can contribute to the field. Chapter 3 outlines the model training process details used in developing the ResSDN solution with an end-to-end data collection process from

publicly available sources and offers a comprehensive discussion on methodologies, parameters tuning, and model evaluation metrics applicable to both stages. The architecture of the chosen Machine Learning models, as defined in the ResSDN solution's design, is expressed in explicit clarity. Through comparing the model validation and evaluation results for each model experimented on in Chapter 4, the best model for stage 1 and the best model for stage 2 are identified. An evaluation of the tool is carried out with the use of the dataset collected from a simulated SDN. Finally, Chapter 5 summarizes the paper by highlighting the key findings, contributions, and future work recommendations.

Chapter 2: Literature Review

2.1 Introduction

The objective of this chapter is to present a detailed review of the pertinent literature of ResSDN, which is a Machine Learning-based tool for the concurrent optimization of Quality of Service (QoS) and security in Software-Defined Networking (SDN). Growing complexity and requirements that accompany SDN environments have made the concurrent management of QoS and security daunting in today's networking (Zhao et al., 2018). Machine Learning has been a powerful solution to such issues; however, literature tends to concentrate on enhancing QoS or security in isolation. In this chapter, we discuss four seminal studies that have influenced state-of-the-art approaches towards Machine Learning in SDN and investigate how they have influenced the development of ResSDN.

This review of the literature combines current research, pointing out the strengths and weaknesses of the methodologies used in each study and the gaps that ResSDN will address. While each work makes significant contributions to the fields of network security, QoS optimization, or cloud service performance, none fully addresses the need for a comprehensive, integrated approach that dynamically balances both security and QoS in SDN environments. By analyzing these gaps, this chapter lays the foundation for the novel contributions of ResSDN.

The first section reviews a study focused on using Machine Learning to enhance network security in SDN. The second section presents research that leverages Machine Learning techniques to improve QoS routing decisions in SDN environments. The third section analyzes a study that combines QoS and security issues in cloud service

environments, focusing on the performance-security trade-offs. The fourth work presents an optimization framework for IDS that balances cybersecurity and QoS in the context of 5G networks. These works are critically evaluated to demonstrate how ResSDN builds upon and advances existing literature by proposing a unified solution for QoS-security optimization in SDNs.

By the end of this chapter, it will have established a clear understanding of the current state of the art in QoS and security optimization in SDN, identifying key gaps in the literature that ResSDN seeks to address. This review will guide the design and implementation of ResSDN, ensuring it effectively balances QoS and security requirements, contributing to the broader fields of network management, Machine Learning, and SDN security.

2.2 Related Work 1

The more integration of SDN with Machine Learning techniques provides significant enhancement in network management, especially in QoS and security. However, challenges persist in balancing these two critical components simultaneously (Zhao et al., 2018). This review focuses on identifying gaps in the existing literature, methodologies used, and limitations that remain in the current body of work.

Although various research works have explored the integration of Machine Learning with SDN to enhance security or improve QoS, fewer studies address the joint optimization of these two components. Most existing research tends to focus on one aspect, either using Machine Learning for real-time security threat detection or using Machine Learning for improving QoS through adaptive network management. The paper by Mohammadi, et al. (2022) titled *HTTPScout: A Machine Learning-based*

Countermeasure for HTTP Flood Attacks in SDN highlights the efficacy of Machine Learning in addressing network security vulnerabilities like HTTP flood attacks, but they seldom address QoS in conjunction with security concerns. This indicates a gap where further research is required to develop comprehensive solutions that optimize both QoS and security simultaneously. For example, the study focuses primarily on detecting HTTP flood attacks in SDN environments using Machine Learning. While it effectively demonstrates that SDN can improve the detection of network attacks, the study does not address the trade-offs or the combined influence of security enhancements on QoS, leaving a potential research gap for ResSDN to explore.

The methodologies in the literature include *Feature Selection, Supervised Learning Models*, and *Cross-Validation (CV)*. To analyze network traffic data to extract features that can distinguish between normal and malicious behavior, HTTPScout uses flow-level traffic data to train Machine Learning models in identifying HTTP flood attacks. Besides, it employs supervised learning models like Random Forest and Support Vector Machine (SVM) to classify traffic as benign or malicious based on labeled datasets. The study also relies on simulation environments like Mininet and real-world traffic datasets to validate the effectiveness of their proposed Machine Learning solutions. Despite the promising outcomes, the methodologies employed often focus on individual aspects of network management, either QoS or security, but seldom the integration of both. In some cases, the focus on security using Machine Learning leads to an increase in computational overhead, which may degrade QoS, illustrating a methodological limitation that the proposed ResSDN could address.

Several limitations emerge from the current literature that ResSDN aims to overcome. First, the datasets used to train Machine Learning models are often static, which limits the adaptability of the systems in real-world dynamic environments. The research often does not account for evolving network conditions or sophisticated, zero-day attacks. Additionally, the study is limited by the scope of their experimental setups, which may not fully replicate real-world network conditions, leaving the findings constrained in terms of applicability to large-scale networks. For instance, HTTPScout successfully detects and mitigates HTTP flood attacks but focuses on traffic analysis for security, neglecting the QoS implications of its deployment in a real-time SDN environment. Another limitation is the potential for increased computational demands that could impair network performance, an issue that ResSDN should consider in its joint optimization of QoS and security.

The reviewed literature provides valuable insights into the use of Machine Learning for security and QoS optimization in SDN. There still exist significant gaps, especially in optimizing both these two aspects together. ResSDN bridges the gaps by proposing a Machine Learning -based framework that optimizes security and QoS simultaneously using a two-stage framework for SDN networks. The methodological weaknesses of previous studies, such as reliance on static databases and the inability to reconcile QoS and security, will be mitigated by ResSDN's comprehensive, dynamic methodology. This invention will enable the growth in network management by offering better, more dynamic, and more robust alternatives for new SDN technologies.

2.3 Related Work 2

Another contribution that is applicable in the review herein includes *Application-aware QoS Routing in SDNs using Machine Learning Techniques* by Zheng (2022) that addresses the QoS optimization problem in SDNs that has been rated as difficult. The contribution in the work brings the full picture on what machine learning and the more specific Deep Neural Network performs in traffic classification and the optimization of routing behavior based on the QoS parameters such as latency and bandwidth. The contribution from the work is the understanding into the potential offered by dynamic real-time QoS management within SDN environments that form one component of the introduced ResSDN tool.

This study employs the supervised machine learning approach using the DNN framework for classifying traffic based on past records. The considered dataset includes over 75 varied applications and about 3.57 million transmission records that have 87 varied traffic attributes. The method proposed has several important stages, i.e., Data Dimension Reduction, Traffic Classification, and Quality of Service-aware Routing (QAR) Algorithm. In order to optimize computational complexity and ensure the fine-tuning of the dataset, the authors propose the use of an Integer Linear Programming-based method for reducing the dimension. The authors' recommended technique determines the most significant features by computing the correlation coefficients among the points and then performs better than the conventional approaches like Principal Component Analysis (PCA) and Least Absolute Shrinkage and Selection Operator (LASSO) that may provide less optimal results for highly non-linear-based scenarios.

DNN classifies the traffic into QoS requirements with latency sensitivity in special consideration. There are three QoS classes for the applications based on the latency requirements: not sensitive, weakly sensitive, and strongly sensitive. The performance of DNN is superior to the other class models like Random Forest and Naive Bayes and thus extremely appropriate for large-scale traffic classification. Following traffic classification, the paper presents a heuristic QoS-aware routing protocol that dynamically selects the most appropriate path based on the QoS requirements of each traffic flow. The protocol picks the best route by either minimizing the mean link occupied time or maximizing the remaining path bandwidth.

While this work makes useful contributions to QoS routing in SDNs, its primary focus is traffic classification and routing optimization but not security concerns. The lack of intertwinement between QoS and security optimization is a clear gap. In actual SDN installations, QoS must also take into account the considerations for security because quality maximization of services in the absence of the protection afforded by the network would result in weaknesses that degrade overall network performance. All this imbalance necessitates the use of a combined QoS-security optimization approach such as the one offered in ResSDN that tries to solve both QoS and security issues in parallel. Experimentation also draws heavily on existing static datasets that would not necessarily capture the dynamism present in actual traffic based on the variability in network conditions. All this necessitates the use of more adaptive Machine Learning models that learn and respond based on dynamic traffic characteristics in real-time.

One of the biggest flaws of this study is the offline training of the DNN model, which may not be suitable for real-time, high-dynamic traffic in SDNs. Although the

model performs well in terms of classification accuracy, the training process and the ability of the model to learn new or unknown traffic patterns would render the model impractical if implemented in real-world scenarios. The scalability issue of the proposed ILP-based feature reduction approach is another issue because the computational complexity would be very high for larger datasets. The study goes a long way towards QoS optimization in SDNs using machine learning for traffic classification and routing.

However, the gaps in managing the security capacity and adapting to the dynamic traffic conditions necessitate further efforts. While the proposed ResSDN tool tries to leverage the strengths of the existing work and build upon the weaknesses, the new tool proposes the cleaner method that optimizes both QoS and security concurrently in dynamic SDN networks. Combining QoS and security optimization will not only keep the network running smoothly but also secure against potential threats in the future and therefore makes the ResSDN an even more robust and adaptable solution to networking issues at hand.

2.4 Related Work 3

The third piece of work that is relevant here is the work on *A QoS-Driven Approach to the Cloud Service Addressing Attributes of Security* by Xu et al. (2018), the work which discusses the interplay between QoS and security in cloud computing systems. The research work suggests a complete framework for modeling intricate dependencies between measures taken for security and the quality of the services in the cloud systems based on the hybridization of Markov models and queue theory. The work is well applicable in the development of ResSDN because the work discusses the trade-offs in providing security and ensuring the high levels of the concept of service

performance that can directly map into the goals of the maximization of both QoS and the security in the SDN networks.

The authors propose the adoption of a hierarchical modeling framework to study the interaction between QoS and security. The methods that support this framework include Markov Modeling to analyze the process, Queue Theory to analyze performance, and the Bayesian Approach to study the interaction between QoS and security. This study deals with the security of cloud services by analyzing the probability of availability with respect to compromised or attacked virtual machines. The Markov model efficiently characterizes the stochastic nature of security events, such as Denial of Service (DoS) attacks, and the security mechanisms applied, such as encryption and firewalls.

It enables in-depth analysis of the effect that security incidents have on the availability and performance of cloud services. The performance parameters of cloud services are represented by an M/M/S queuing model, which simulates both arrival and completion of service requests. This method explains how security mechanisms relate to the performance of services, i.e., task completion rates. The model takes into account the overhead introduced by security mechanisms, such as the computational time needed for encryption, that might impact the overall effectiveness of the service. In addition, a novel Bayesian approach is introduced to align the output of the security and performance models for a better estimation of the security factors' impact on service performance. The method calculates a correlation factor measuring the extent of the relationship between security intensity and service performance, which allows dynamic tuning based on security and QoS demands.

Although this research offers useful information on the interplay between QoS and security, it is focused in cloud environments and does not directly address the special requirements of SDN environments. The cloud service model it outlines relies strongly on virtual machine security, which might not be entirely relevant to SDN's unique architecture, where the central controller and network flows are the key features. This limitation points to a gap in research that is specific to SDN and its specific challenge in consolidating QoS and security management. Second, while the research addresses the impact of security mechanisms on performance, it does not examine in-depth how to dynamically adjust these mechanisms based on live traffic flows or evolving threat levels. This inequality highlights the necessity for more adaptive strategies, such as those put forward in ResSDN, which aims to utilize machine learning to enhance QoS and security in real-time environments.

One significant limitation inherent in this research concerns the static properties of the security and performance models employed. As useful as Markov and queue models are as tools to assess settled security threats and service demands, these models can capture difficulties in extending to more dynamic settings in which security threats and network states evolve with speed. This constraint echoes the necessity for machine learning-based mechanisms such as ResSDN that continuously learn new things and possess the capability to tune security and QoS policies. The other constraint is the consideration of a very limited subset of security mechanisms such as CPU-intensive ones such as encryption. The study also fails to account for the wider variety of security controls such as intrusion detection systems that are required for identification and prevention of real-time threats in SDN networks. The models that are proposed are also

computationally intensive and can hinder their deployment in large-scale systems where computational cost and thus the cost related to the operation must also be minimized in order to achieve maximum performance for the services.

This work provides us with the effective framework for describing QoS and security interactions in cloud systems and the trade-offs that must be traded-off for the deployment of the controls. It does include the static models and the focus on cloud systems that provide limitations for the mapping of the results into SDN. These are the limitations that are overcome by the use of Machine Learning in order to build the more adaptive and scalable joint QoS and security optimization. Modulating security controls based on real-time input in real-time will provide the more robust and dynamic approach for handling the problematic trade-offs between QoS and security in SDN networks.

2.5 Related Work 4

The fourth piece that has been taken into consideration here is *Novel Modeling and Optimization for Joint Cybersecurity-vs-QoS Intrusion Detection Mechanisms in 5G Networks* by Bozorgchenani et al. (2023), which provides an optimization approach for IDS that balances cybersecurity and QoS in the 5G networking environment. As the need for high-performance and high-security services increases in the case of the 5G network environment, this work finds particular interest in the development of ResSDN because it provides insight into the optimization of both the security and QoS parameters in high-requirement network environments.

This work offers an optimization approach that solves both cybersecurity issues and QoS issues in 5G networks concurrently using the Cobb-Douglas production utility function. The main goal of this utility function is the balancing between the inherently

conflicting needs created by the introduction of the many intrusion detection methods across the multiple tiers of network intrusion detection. Main methodologies utilized in the work include Security Agents (SA), Intrusion Detection Mechanisms (IDM), and the Cobb-Douglas Production Utility Function. SAs are placed strategically all around the network to detect and monitor cyber threats while the detection and allocation of resources are segmented into three hierarchal levels. Level 1 utilizes signature-based detection that excels in the identification of known attacks at low resource cost. Level 2 uses anomaly-based detection that applies machine learning algorithms for the identification of new attack patterns despite the large amount of resources that must be allocated. Level 3 characterizes Hybrid detection, which combines signature and anomaly-based detection mechanisms, requiring a significant amount of resources while providing high accuracy in detecting threats. In addition, this study outlines the security utility and QoS utility relationship, where security utility is measured as a function of detection effectiveness for intrusion detection and QoS utility is measured based on resource consumption levels through bandwidth, computational overhead, and monetary expenditure. The Cobb-Douglas function helps in finding the optimal values of elasticity parameters, thus bringing about a balance between security and QoS to maximize both objectives concurrently.

One of the significant gaps identified in this work is the focus on 5G-specific environments. While the study effectively balances security and QoS for 5G networks, it does not directly address SDN-specific challenges. SDNs, by nature, offer centralized control over network traffic, which may offer additional opportunities for optimizing both security and QoS at a more granular level. The study also focuses primarily on resource

allocation between SAs in 5G networks but does not explore how real-time traffic conditions or rapidly evolving attack vectors could influence the optimal deployment of IDS. Furthermore, while the Cobb-Douglas utility function offers a powerful approach to balancing security and QoS, the study does not consider the full spectrum of network threats that SDNs face, such as DDoS attacks or network-level anomalies that are prevalent in SDN-based infrastructures. This indicates a gap in generalizing the findings to SDN, which ResSDN aims to address by developing a more adaptable and robust solution for joint QoS and security optimization in SDN environments.

The inherent limitation of this research lies with its assumption of static consumption across different detection levels. With real-world applications involving 5G or SDN networks, however, both threats as well as patterns of traffic have dynamic behavior, meaning that the consumption of security utility and QoS also remain subject to fluctuation all the time. The limitation acts as a setback for the real-time adaptability of the proposed solution. Additionally, reliance on pre-defined SAs might limit the framework's scalability for larger, more distributed network settings where real-time dynamic reallocation of detection strategies as well as resources are vital. Lastly, the framework in this work does not account for potential bottlenecks or degradation effects due to high-security detection levels, which may cause detrimental effects on QoS at peak network loads.

This work presents a sophisticated model for balancing cybersecurity and QoS in 5G networks, offering valuable insights into the trade-offs required to optimize network performance and security. However, its limitations in adapting to real-time traffic changes and its focus on 5G environments leave room for further research. ResSDN aims

to build upon these findings by applying a similar joint optimization model to SDN environments, where real-time traffic analysis and dynamic threat detection are crucial. By incorporating Machine Learning, ResSDN will address the challenge of dynamically optimizing both QoS and security in SDN environments, ensuring higher adaptability and resilience in modern network infrastructures.

Chapter 3: Methodology

3.1 Introduction

The Two-Stage Machine Learning Framework proposed here boosts security as well as performance on Software-Defined Networks (SDN) with integrated threat mitigation and Quality of Service (QoS)-focused traffic classification. The research here focused on constructing a holistic Security-QoS framework through a two-step approach. The reason for a two-step strategy is because combining security enforcement with QoS improvement within a common step would bring with it significant challenges concerning correlation of features, interpretability of models, and real-time response. The two-step strategy allows for prioritization through detection and elimination of bad traffic, followed by QoS classification on clean, legitimate traffic alone, eventually at the expense of precision and reduced computational burdens. While implementation follows a linear two-step approach, each step occurs within a unified SDN system, addressing security as well as QoS improvement at the same time. Future work may look into constructing real-time model switching capability to better balance security and QoS.

Random Forest, Decision Tree, Support Vector Machine (SVM), and Deep Neural Network (DNN) algorithms are utilized in Stage 1 in the elimination and identification of malicious traffic and hence make a great reduction in vulnerability to attacks viable as well as avoid effects on QoS policies (Zhang et al., 2020). Supervised learning methods in the algorithms are applied on labeled sets of data in the improvement of the efficiency of network intrusion detection systems (NIDS) by identifying attacks and legitimate traffic very accurately (Shafiq et al., 2021).

Stage 2 removes malicious traffic and classifies benign traffic into application types and QoS priority levels. Random Forest, Light Gradient Boosting Machine (LightGBM), eXtreme Gradient Boosting (XGBoost), and K-Nearest Neighbor (KNN) classifiers classify application-specific priority levels and are employed for priority level tagging. They maximize network performance for delay-sensitive applications such as Voice over Internet Protocol (VoIP), video streaming, and cloud services (Tariq et al., 2023). It offers multi-output classification where the application type and related QoS priority level are classified at the same time, minimizing computational overhead and maximizing the level of effort.

The Machine Learning-based model that has been proposed improves scalability and performance in contrast to rule-based systems in managing constantly dynamic high-capacity network traffic (Huang et al., 2020). Through Supervised Learning-based operation, the system adapts QoS policies dynamically in real-time and therefore provides adaptive bandwidth allocation and adaptive network traffic prioritization (Mnih et al., 2015). Additionally, using sophisticated machine learning methods, the likelihood of threats being misinterpreted as threats by the system decreases significantly, therefore providing improved robustness in security operations while minimizing the invasion into legitimate traffic flows (Osman et al., 2024). The self-learning nature of the models also enables the prediction of traffic and security incidents from analysis thereof and therefore allows SDN controllers to influence data-driven autonomous modifications in network policies (Salau & Beyene, 2024). The method encourages resilience in the network where optimal performance is achieved even in the presence of congestion as well as adaptability in the constantly dynamic nature of cyber threats. Additionally, the diversity

in networking environments that the framework supports—ranging from enterprise networks, ISPs, and SDN-based cloud-based topologies—means the framework makes the optimal choice as the optimal candidate for optimization and protection of today's networks (Kreutz et al., 2015).

3.1.1 Justification for the Two-Stage Approach

A one-stage machine learning approach might not meet the proper balance between performance and security within SDN networks. Either the one-stage model would compromise QoS performance for the sake of security or would optimize QoS at the cost of potential adversarial influence. The proposed two-stage approach brings the layered approach where one model would be dedicated to guaranteeing the security by filtering malicious traffic and another model dedicated to the classification of benign traffic for the purpose of effective QoS management (Salman et al., 2022). Separating these two models decreases the likelihood that the mechanism for security would misclassify legitimate traffic, providing both increased network security and better quality of services.

3.1.2 Challenges Addressed by the Framework

This solution solves certain principal challenges faced by SDN environments: **Dynamic threats on the network:** While rule-based systems cannot keep up with the constantly evolving nature of threats, machine learning-based systems learn and improve constantly and enhance the capabilities related to threat detection (Zhang et al., 2020). **Dynamic traffic flow:** Based on classification models that analyze traffic types, the system dynamically adjusts QoS policies for bandwidth optimization in real-time.

Scalability Issues: Rule-based systems become inefficient as the volume of data increases; machine learning-based classification methods have the quality-preserved one that resists spike demand (Huang et al., 2020).

False Positives in the Detection: Supervised learning algorithms combined with new feature selection algorithms greatly reduce the frequency of false positives and prevent excessive packet loss and services loss (Shafiq et al., 2021).

3.1.3 High-Level Overview of Data Flow

The proposed framework follows systematic data pipeline processing, outlined in subsequent sections:

Ingress Traffic: All data traffic goes into SDN controller.

Stage 1: Detection and Elimination of Malicious Traffic

The classification model utilizes methodologies like Random Forest, Decision Tree, SVM, and DNN for identifying and blocking malicious network traffic. These methodologies have been proven previously, through research, as being very effective for anomaly-based intrusion detection systems, especially if supervised learning algorithms are used with labeled data sets (Zhang et al., 2020).

Process:

1. Training Phase: The model is trained on labeled datasets comprising benign as well as malicious traffic patterns, e.g., the CIC-IDS2018 dataset (Ring et al., 2019).
2. Data Preprocessing: Different data preprocessing techniques, such as feature importance evaluation and class imbalance handling, are applied to increase classification accuracy (Zhou et al., 2022).

3. Detection & Filtering: Incoming traffic is classified in real-time, and any malicious traffic is immediately dropped to prevent security threats.

Stage 2: Classification of Benign Traffic Based on QoS Priority

For benign traffic, a different classification model, such as Random Forest, LightGBM, XGBoost, or KNN may, however, be used for classification depending on the type of intended application, where QoS priority level is allocated based on the detected class of application.

QoS Routing Priority Levels:

Level 4 - Latency Strongly Sensitive, like video conferencing and collaborative tools

Level 3 - Latency Weakly Sensitive, like web browsing, video streaming, and cloud

Level 2 - Latency Not Sensitive, like email and software update

Level 1 - Non-Business Relevant, like social media and some others that are not defined in business usage

Process:

1. Traffic Classification: Once malicious traffic is filtered out, the benign traffic is categorized into application types using Machine Learning-based classifiers.
2. Output Prediction: Both the application type and the QoS priority level will be predicted at the same time by an output classifier to increase computational efficiency.
3. QoS Optimization: Based on the result of the classification, the corresponding traffic control and prioritization processes are enabled within the SDN controller and therefore ensure the improvement in the network performance and user satisfaction (Wang et al., 2021).

QoS Enforcement: The SDN controller makes the network operate smoothly by regulating the traffic according to the classification outcome, prioritizing the essential applications for increased efficiency.

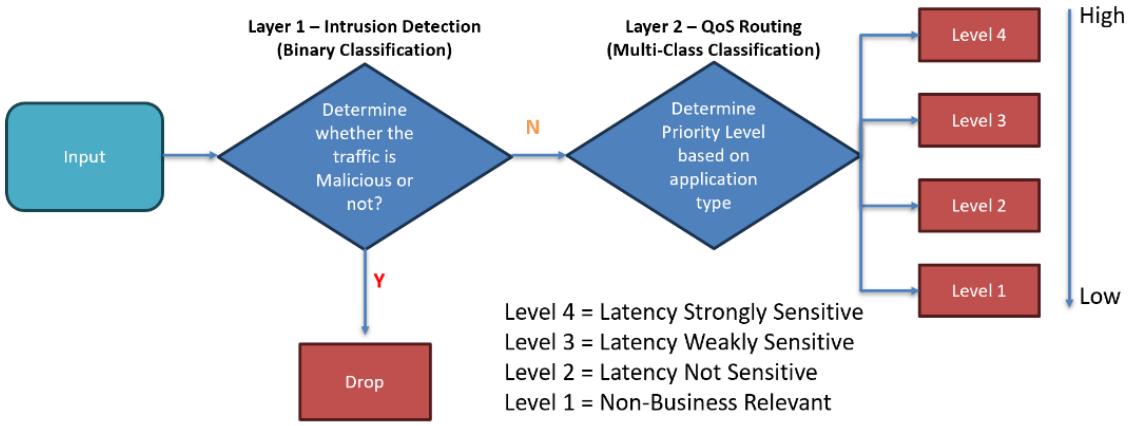

Figure 1 – Workflow of ResSDN

The two-stage machine learning-based mechanism depicted in *Figure 1* ensures that network security is optimized by having an adaptive and optimized QoS policy that provides the SDN environment with robust contemporary environment-friendly solution.

3.2 Stage 1 Model for Intrusion Detection

The primary function of the Stage 1 Model is the identification and removal of malicious traffic within the SDN platform before reaching the point where they affect the network throughput and security measures. The primary function of the operation includes the identification of cyber threats that include Distributed Denials-of-Service (DDoS) attacks, unauthorized access, and the dissemination of malware (Kumar & Rose, 2023). Applying classification models that rely on machine learning, the system effectively detects threats proactively with a low rate of false positives (Wassie et al., 2023).

Traditional security measures, such as rule-based and signature-based, lack suitability for coping with constantly emerging threats in cyber-attacks. These static approaches often come up short against zero-day vulnerabilities as well as complex adversarial strategies (Prasanth & Uma, 2024). However, machine learning has a more dynamic approach as it learns from past attack patterns and gets insight from new threats. However, challenges concerning unbalanced datasets, high-dimensional spaces, and real-time classification accuracy have to be addressed for improving detection accuracy as well as computational efficiency (Krishnan et al., 2019).

Stage 1 Model works sequentially for enabling effective threats detection, ensuring that only non-threatening traffic is allowed to reach Stage 2 for QoS classification.

Data Collection: Creating labeled datasets containing benign and malicious traffic behaviors.

Data Preprocessing: Applying feature selection, normalization, and handling class imbalances to improve model performance.

Model Training: Using supervised learning methods with classifiers such as Random Forest, Decision Tree, SVM, and DNN.

Model Validation and Evaluation: Checking for accuracy, precision, recall, and F1 scores for ensuring dependability for real-world implementation.

3.2.1 Data Collection

The CIC-IDS-2018 dataset is especially suitable for the first step of creating an intrusion detection model because of its extensive range of traffic types, realistic depiction of network activity, rich feature set, balanced class distribution, and low

preprocessing needs. This allows for better discrimination between malicious and normal traffic. Since the dataset includes more than 80 unique traffic features, which are given and stored as separate CSV files for each computer, there is a need to merge these varied CSV files into a single data frame for further analysis. After the files are merged, classification is done based on the attack types, as described in the dataset documentation provided on the official websites of UNB (University of New Brunswick, Fredericton) and CIC (Canadian Institute for Cybersecurity), thus categorizing them into six distinct attack types in addition to normal traffic, which is counted separately to show traffic distribution, as shown in *Table 1* (Sharma et al., 2018).

Class	Attack Type	Count
0	Benign	13,484,708
1	Web attack	928
2	Infiltration attack	161,934
3	Botnet attack	286,191
4	Brute-force attack	380,949
5	DoS attack	654,300
6	DDoS+PortScan	1,263,933

Table 1 - Stage 1 Attack Types Classification

3.2.2 Data Preprocessing

Data preprocessing is an important step in preparing the CIC-IDS-2018 dataset for training a machine learning model. To ensure the dataset was correctly sanitized, balanced, and ready for classification purposes, various operations were performed.

Label Encoding: The "AttackType" attribute of the data set was converted into numerical values for better compatibility with machine learning algorithms. Different attack types were given unique integer values, with "Benign" traffic given a value 0, and remaining attack types, such as "Web attack," "Infiltration attack," and "DDoS+PortScan," given specific numerical values as depicted through *Table 1*. This

step converts data into a form that supports easier classification through simpler handling of categorical data.

Removal of Non-Numeric Columns: Some non-numeric columns, for example, 'Timestamp,' 'Flow ID,' 'Src IP,' 'Src Port,' and 'Dst IP,' were discarded from the data. These columns were not relevant for classification because they do not provide relevant features for attack type predictions; thus, this made data purging simpler and also helped with dimension reduction.

Separation of Features and Target Variable: The data was split into features, labeled as x, and a dependent variable, labeled as y. The dependent variable "Attack_encoded" shows whether the traffic type is classed as attack or benign, and all the other columns act as attributes for training. Segmentation of the data into dependent variable and attributes ensures that the dependent variable and attributes are separate, allowing for a focus on relevant features for making accurate predictions.

Data Splitting: The data, with respect to model development and assessment, is split into training and testing subsets according to an 80/20 split. The split ensures that a significant majority of data is allocated for training a model, with the test subset remaining inaccessible for training, thus allowing a determination of how well a model will perform on novel data.

Handling Infinite and Missing Values: Infinite values within the dataset were replaced by NaN (Not a Number) based on infinite and missing value analysis. Missing values were filled by the mean of the corresponding feature columns. These processes ensure that the dataset does not have any inconsistencies that would negatively influence the performance of the model.

Feature selection: To identify the most contributing features to the classification problem, the data were trained on an XGBoost classifier, and the contribution of each feature was evaluated. The top 50 most contributing features were selected based on the contribution to the decision-making by the model. Feature selection minimizes the dimensionality while focusing the attention of the model towards the most contributing features, thereby improving the efficiency as well as the interpretability of the model. Class Imbalance Handling: Class imbalance was addressed by employing the Synthetic Minority Over-sampling Technique (SMOTE) that generates synthetic samples of the minority class and thereby assists in the more balanced distribution of classes and removal of the threat of the model being biased towards the majority class. Through the resampling process being conducted on the dataset, the model is trained on a represented dataset of benign traffic and malicious traffic.

Data Scaling: Feature scaling was eventually performed by using StandardScaler. The processing brings the features into the same scale numerically and therefore no opportunity exists for large, ranged features overwhelming the training process of the model. The approach also causes the process of training the model to converge earlier and also offers the same scale for inputting the features.

These preprocessing operations played the key roles towards preparing the data set in ordered, balanced, and training-model-ready format, and therefore creating sound foundation for the development of the successful intrusion detection system.

3.2.3 Model Training

In stage one of the Two-Stage Machine Learning approach, the traffic is either marked as malicious or benign. We selected Random Forest, Decision Tree, SVM, and

DNN because each one excelled in resolving the classification problem posed by complex datasets like CIC-IDS-2018. The purpose of the first stage is to identify the optimal model for classifying traffic into malicious or benign by comparing so that the second stage involving QoS classification would operate on trusted information.

Random Forest is one of the ensemble learning methodologies that employs several decision trees for the purpose of boosting the accuracy in classification and reducing the potential for overfitting (Liaw & Wiener, 2002). That the technique has the capacity to handle large numbers of input features, as the heterogeneous traffic content of the CIC-IDS-2018 dataset presents, makes Random Forest one strong candidate capable of doing the work. In addition, Random Forest has proven itself effective in the case of class imbalance and also able to detect complex feature-to-feature interactions significant in the separation between benign and malicious traffic.

Decision Trees are one of the simple yet robust classification methods that partition the dataset into smaller subsets iteratively based on the feature values. Decision Trees possess the significant strength that they provide understandable rules for classification (Quinlan, 1986). It is highly useful in the case of intrusion detection systems where the mechanism by which the traffic is being classified by the system must be understandable in order to build up the confidence and verification. Moreover, the fact that DTs are suitable both for numerical and categorical inputs and therefore highly adaptable for the variety in feature types in the CIC-IDS-2018 dataset has been mentioned.

SVM is also very fast and strong classification that has performed very well in high-dimensional spaces, which are typical in large feature sets such as CIC-IDS-2018. SVM maximizes margin during training and therefore does very well on the binary

classification task of identifying malicious traffic versus benign traffic. Support for non-linear decision boundaries using the application of kernel functions is another feature that contributes towards the versatility of the model by allowing the model to learn complex interactions within the dataset (Cortes & Vapnik, 1995).

DNN is defined by the layered structure. They are able to identify complex representations and structures in large datasets using these layers. DNNs' intrinsic ability to learn feature representations from the input by themselves makes them highly appropriate for complex high-dimensional problems such as network traffic classification (LeCun et al., 2015). DNN has also performed very well in the scenario where large datasets are present, such as in the case of CIC-IDS-2018 where they are able to effectively leverage large samples in order to achieve high classification accuracy.

To maximize the performance of the models, the RandomizedSearchCV method was utilized. RandomizedSearchCV was utilized for the optimization process because it effectively searches through enormous spaces of hyperparameters at the expense of computational cost. Random selection among hyperparameter settings significantly decreases the cost and computational cost associated with complete search methods like GridSearchCV (Bergstra & Bengio, 2012). RandomizedSearchCV also utilizes CV that prevents overfitting and ensures the trained model generalizes well beyond the training set and therefore makes the method highly suitable for the task of intrusion detection systems (Kuhn & Johnson, 2013). Capability to handle both continuous and categorical distributions makes the method highly suitable to fine-tune all types of models ranging from Random Forest and SVM, all the way up to DNNs (Hutter et al., 2011).

Some hyperparameters were tested using the RandomizedSearchCV that included the estimators count and the bootstrapping activation in the Random Forest; the various impurity measures and the choice between the best and the random splits in the Decision Trees; the various values of the parameter C in the SVM so that the trade-off between the training error minimization and test error would be activated; and the dense values and three other various alternative activation functions in the DNN.

There were run 50 cumulative iterations based on a 10-fold CV strategy so that the model would have aggressive tuning with the emphasis being on the optimization of the performance. The four models were trained based on the standardized training set. Models were trained based on the hyperparameters that produced the most optimal results. RandomizedSearchCV-based determination found the best among the models and then fitted the same into the resampled training set. The model was saved for the potential subsequent use in prediction.

3.2.4 Model Evaluation and Validation

After the process of training the model came to an end, the performance test based on the test dataset (x_test_scaled) was performed. The test included the predicted labels (y_pred) and predicted probabilities (y_prob), both being needed in the calculation of the most key performance measures, most importantly the Receiver Operating Characteristic (ROC) Area Under the Curve (AUC) score. It measures the ability of the model in class discrimination and one main marker of the performance by the model in the resolution of the problem of the imbalanced classification (Fawcett, 2006). Some more measures of performance were calculated in order to have further information concerning the accuracy by the classification by the model. Specifically, the accuracy was observed and

confusion matrix produced that explained the classification errors by the model. In addition, the classification report produced enabled the comparison between the precision, recall, and F1 score for each individual class and therefore offered full insight into the performance by the model by all the distinct classes. In addition, ROC curve for each class in terms of the True Positive Rate (TPR) against the False Positive Rate (FPR) useful for the calculation of the accuracy by the classification by the model at decision thresholds that vary (Bradley, 1997). Finally, the ROC curves of the four trained Machine Learning models are compared in *Figure 2*.

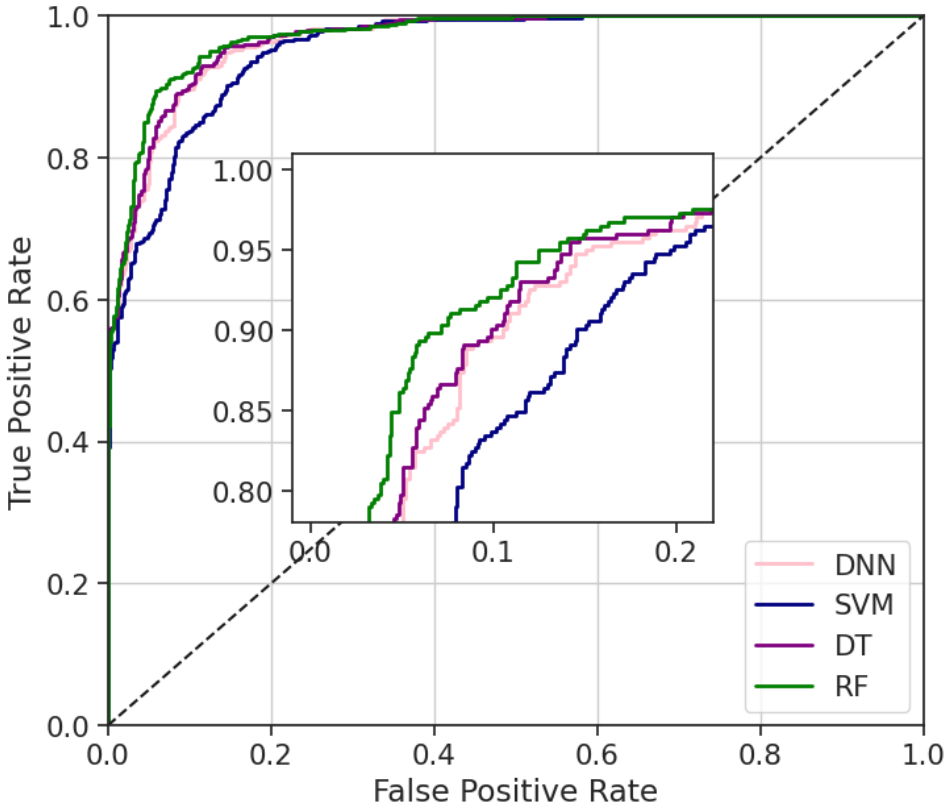

Figure 2 - AUC-ROC Curves of Four Trained Models in Stage 1

A ten-fold Stratified K-Fold Cross-Validation strategy was employed so that the strength and robustness of the model would enhance. The method guarantees that all folds have the class distribution of the entire dataset and thereby the creation of balanced

training and validating subsets, something that in the case of imbalanced classification is highly essential (Kohavi, 1995). Model generalization capacity was better expressed by the average performance on the folds and by reducing the risk of the model being overlearned (Bengio, 2012). These models were performance-tuned on the basis of highly balanced practice in terms of hyperparameter tuning, validating, and testing.

3.3 Stage 2 Model for QoS Routing

Stage 2 Model's primary purpose is the classification of the benign traffic that has already overcome Stage 1 in order to perform QoS analysis. It is one giant leap towards the right assignment of the network resources based on the nature of traffic and the corresponding QoS requirements that have direct effects on the performance of programs demanding real-time information or low-delay connections, i.e., Voice over Internet Protocol (VoIP) (Chaudhry & Soni, 2022). Based on machine learning algorithms, the model acquires the capacity to learn from the new developing traffic pattern changes and therefore adaptive QoS policy changes based on developing situations (Wang et al., 2021).

Compared to the traditional static methods, the machine learning-based networks have the potential to learn and improve the performance of the network based on the history and empower the network with a better scalable and efficient traffic management system (Rajput et al., 2020). Phase 2 targets training models for the purpose of the effective classification of benign traffic in order to provide the better optimization of network resources.

Data Gathering: Gathering benign traffic samples that have labels and QoS-relevant feature information.

Data Preprocessing: Subsumes class imbalance handling and feature selection optimization for the purpose of bettering the model.

Model Training: Using machine learning models including Random Forest, LightGBM, XGBoost, KNN, for QoS classification.

Model Evaluation and Verification: Verification against the performance using measures such as accuracy, precision, recall, and F1 score for robust classification.

3.3.1 Data Collection

The “IP Network Traffic Flows Labeled by 75 Apps” dataset lends itself very well for training the QoS routing model because the applications are well-represented by the wide variety covered in the application classes and the full traffic flow characteristics and the multi-target scope for the model that makes concurrent classification of the application traffic and corresponding priorities. Labeled traffic flows for the traffic of 75 applications are included in the dataset that makes the work easy for the model to predict traffic behavior and allocate the network resources accordingly based on the allocated priorities. The traffic features, before processing, require preprocessing steps, including handling missing values, normalizing the data, and choosing relevant features before applying machine learning algorithms toward multi-output classification (Chaudhry & Soni, 2022).

The mapping of "Protocol Name" to "AppCat" was performed by referring to the application category listed in ntop website (ntop. n.d). A total of twenty-one application categories are defined and reducing the application categories from twenty-one to the top nine to improve model accuracy and efficiency by focusing on the most prevalent traffic

types, thus enhancing classification performance. These nine application categories are counted separately for presenting the traffic distribution in *Table 2*.

Class	Application Type	Count
0	Others	421,057
1	Web	2,718,207
2	Streaming	182,251
3	Cloud	81,809
4	SocialMedia	49,707
5	Email	40,269
6	SoftwareUpdate	34,471
7	VoIP	30,699
8	Collaborative	5,947

Table 2 - Stage 2 Application Categories Classification

3.3.2 Data Preprocessing

Like Data Preprocessing for intrusion detection in Stage 1, several steps were performed to ensure data consistency, feature relevance, and computational efficiency for the QoS classification dataset in Stage 2.

The dataset underwent preprocessing steps including categorical encoding of application labels (as shown in *Table 2*), removal of non-numerical attributes, and an 80/20 train-test split for balanced evaluation. Feature selection was performed using an XGBoost classifier, retaining the top 50 most important features to reduce dimensionality while preserving critical information. Finally, standardization was applied to normalize feature distributions, ensuring consistent model performance.

3.3.3 Model Training

In the Two-Stage Machine Learning approach towards the classification of benign network traffic into application types and the assignment of corresponding priority levels towards the optimization of QoS, four models—Random Forest, LightGBM, XGBoost,

and KNN—were selected for the comparison. These models possess certain properties in handling multi-output classification problem scenarios within network traffic datasets as complicated as "IP Network Traffic Flows Labeled with 75 Apps." The focus in the present research work based on the comparison among these four models is the identification of the optimal solution in QoS classification and the balance among the predictive accuracy, computational cost, and responsiveness within complicated networking.

Random Forest is utilized in both the Two-Stage Machine Learning's Stage 1 and Stage 2 because of its versatility, strength, and strong performance in the classification problem within numerous fields. In the QoS classification within Stage 2, Random Forest is utilized where the approach doubles up as the application type classification as well as the corresponding prioritization level. Its ability to efficiently process large traffic datasets, estimate feature importance as well as classifying QoS makes the approach most suitable for this purpose. Additionally, the approach's non-parametric nature allows for the addition of new traffic without the need for excessive parameter tuning and therefore makes the approach the one to beat within both stages (Breiman, 2001).

LightGBM is a decision tree-based gradient boosting framework that has proven more effective in network traffic classification because of the ability to handle large data in an efficient manner. LightGBM's leaves-based tree growth strategy equals higher accuracy but decreases the computational expense relative to the conventional boosting algorithms (Ke et al., 2017). Considering the fact that QoS classification entails the mapping of complex interactions between the different features, the ability in LightGBM

to handle high-dimensional data in an efficient manner makes it most appropriate in classifying network traffic based on application type and priority. Its use in traffic classification has proven promising both in terms of accuracy of classification and prediction speed.

XGBoost is characterized by its scalability, efficiency, and higher prediction accuracy. Feature selection and interactions are handled well by the algorithm using the allocation of importance scores that also largely contributes towards reducing the computational complexity without compromising the effectiveness of the classification (Chen & Guestrin, 2016). Since the network traffic data is structured, the ability of XGBoost to process sparse and high-dimensional data sets makes it very suitable for QoS classification.

KNN is also known as an instance-based and non-parametric algorithm that has been found to be very easy and effective for classifying network traffic. KNN classifies network flows based on the similarity between the characteristics and the existing instance, making the approach highly useful in situations where the traffic pattern is highly discernible (Cover & Hart, 1967). Its ability to learn new information without the need for significant retraining makes the classification process adaptable based on application type and QoS priority levels. KNN's effectiveness in processing network traffic has also proven capable of pattern and anomaly detection while having low computational requirements, therefore being a viable option for real-time classification tasks.

Same as in Stage 1, in Stage 2, hyperparameter tuning is carried out using RandomizedSearchCV. In RandomizedSearchCV, certain hyperparameters such as

fraction values of data used per tree (subsample) and feature fraction values (colsample_bytree) in the LightGBM model as well as the XGBoost model; the number of neighbors, weight function values, and the choice of distance metric in the KNN model. 50 iterations were run with 10-fold CV to thoroughly optimize the model's performance. Each of the four models was trained on the scaled training data (x_train_scaled and y_train_resampled) using the best hyperparameters found in the tuning process. The top model, chosen by RandomizedSearchCV, was then tuned on the resampled training data. The tuned model was ultimately saved for future use in QoS classification tasks.

3.3.4 Model Evaluation and Validation

Stage 2 validation and evaluation replicated Stage 1's stringent methodology to promote consistency and reliability between the two stages. The models were evaluated on the test dataset (x_test_scaled) following training, with predictions being tested against both predicted labels (y_pred) and predicted probabilities (y_prob). These outputs were essential to the calculation of important performance measures such as the ROC-AUC score, which explains the ability of the model in separating between classes of traffic. In consideration of the need to solve imbalanced classification, this measure was an important performance measure (Fawcett, 2006). Various evaluation metrics were also computed so as to allow for an overall determination of model performance.

Accuracy was calculated and a confusion matrix was constructed to examine classification errors. A classification report was also generated, which outlined precision, recall, and F1 score by class to estimate the performance of the model on various traffic classes. In addition, an ROC curve for every model (See *Figure 3*) was plotted, TPR

against FPR at different decision thresholds, providing additional classification performance details (Bradley, 1997). Stratified K-Fold CV with five folds was used to improve generalization and reliability. This method made sure that the folds all had the same class distribution as the entire dataset, balancing the training and the test sets.

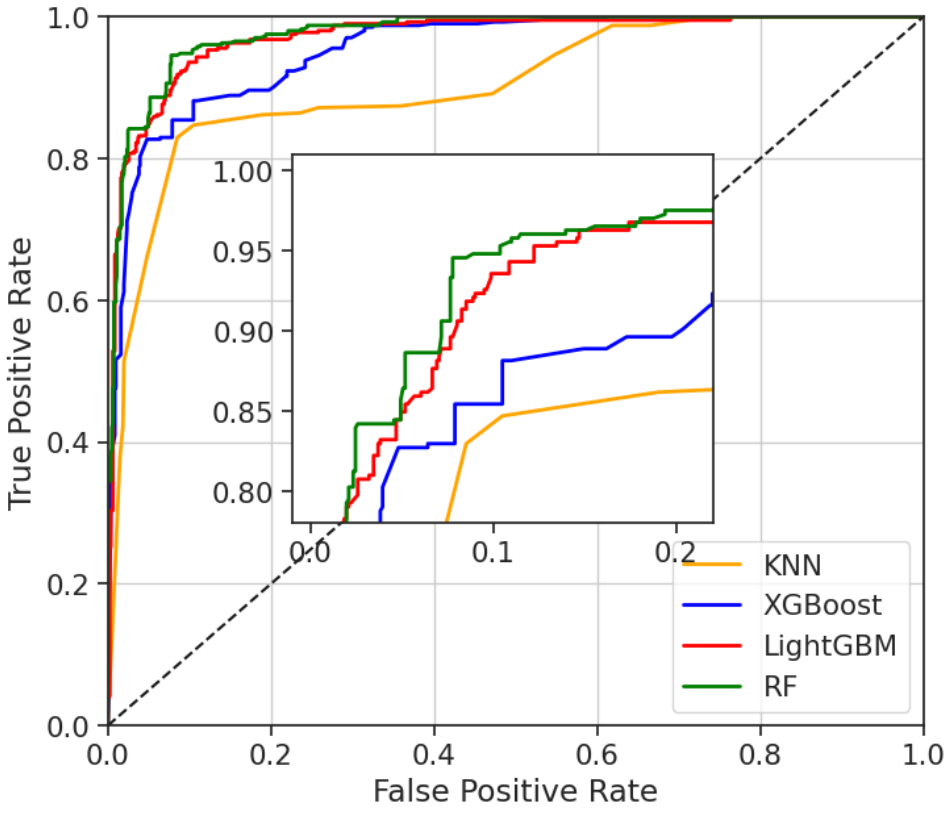

Figure 3 - AUC-ROC Curves of Four Trained Models in Stage 2

This method is beneficial for class imbalance problems only (Kohavi, 1995). The accuracy for each fold was averaged when trying to estimate the model's ability to generalize without subjecting the model to overfitting risk (Bengio, 2012). Models in Stage 2 were adequately tested with this stage to effectively classify benign network traffic by type of application and provide QoS priority levels.

Chapter 4: Results

4.1 Model Selection for Stage 1

The performance between four models selected for the first stage invasion detection is contrasted and compared based on the most appropriate Machine Learning model in distinguishing the malicious traffic from the normal traffic. Comparison has been made based on the set of performance measures that include accuracy, precision, recall, F1 score, and the Receiver Operating Characteristic Area Under the Curve (ROC-AUC) score in order to give general comparative analysis into the model's detection capacity.

During Stage 1, four machine learning models—Decision Tree, Random Forest, Support Vector Machine (SVM), and Deep Neural Networks (DNN)—were trained and evaluated on the CIC-IDS-2018 dataset that has labeled network traffic with more than one category of cyberattacks. The training dataset includes more than 16 million records comprising 13,484,708 benign samples and 3,213,705 attack samples divided into different types of attacks, as shown in *Table 1*. The class imbalance posed a challenge, necessitating appropriate resampling techniques to prevent bias towards the majority class. Given the large-scale nature of this dataset, training Machine Learning models requires substantial computational resources. The entire training process was conducted on Google Collaboratory Pro using an NVIDIA A100 GPU runtime, consuming over 220 compute units to complete training across all four models. The hyperparameter tuning process, particularly for DNN and SVM, was computationally intensive due to the iterative optimization required to achieve the best model performance.

To ensure robust and unbiased model evaluation, Stratified K-Fold Cross-Validation (CV), with ten splits, was employed. This approach ensured that every fold included examples from every class available in the dataset to avoid potential biases due to class imbalances. All the models were trained using RandomizedSearchCV to boost the hyperparameter tuning process, where the best model configurations were determined based on their performance on the validation set. After the training process, the models were tested on a separate holdout set, and their classification performance was analyzed using confusion matrices to visualize classification errors. The best performing model was determined by analyzing the balance between high detection rates (recall) for malicious traffic and low false positive rates to minimize disruptions to legitimate traffic. Additionally, ROC curves were plotted for every model to illustrate their capability in distinguishing between classes for different classification thresholds.

The following sections provide detailed quantitative analysis for each associated model along with their respective merits and demerits related to the accurate identification of threats. These analyses form the basis for the selection of the most suitable model for Stage 1 so that only good traffic gets forwarded towards Stage 2 for QoS classification and effectively denies the malicious traffic.

4.1.1 Model Evaluations in Each Class

Table 3 shows the fine-grained analysis between the four Machine Learning models— Random Forest, Decision Tree, SVM and DNN—based on classification accuracy for all types of attacks in the first stage. Precision, Recall, and F1 score are the measures taken into consideration. These provide insights about the strength of the models in the accurate classification between the malicious traffic and the benign traffic.

These measures are of prime importance towards the determination of the credibility of each model for the discrimination between the types of cyberattacks and the benign traffic. Higher the value for Precision means the less the number of the false positives are there, whereas the measure of the Recalls provides the accuracy about the identification by the model of the actual attacks. F1 score provides the balanced score between the Precision and Recalls. Comparison between the measures provides the determination about the overall highest performance by one among the models in the filtering between the malicious traffic based on the minimum classification errors.

Benign Traffic Classification: For benign traffic classification (Class 0), Random Forest and Decision Tree achieved similar performance, with Random Forest slightly outperforming Decision Tree across all metrics. Random Forest attained an F1 score of 0.73, indicating a balanced precision-recall tradeoff. DNN demonstrated the highest recall (0.81), suggesting a strong ability to detect benign traffic; however, its lower precision (0.60) implies a higher false positive rate. SVM performed the worst in this category, with a recall of only 0.42 and an F1 score of 0.51, indicating a significant number of misclassifications.

Web Attack Detection: For detecting web attacks (Class 1), Random Forest and Decision Tree exhibited excellent performance, with precision values of 0.93 and recall values above 0.98, leading to high F1 scores (0.96 and 0.95, respectively). In contrast, SVM failed to distinguish web attacks effectively, with an F1 score of only 0.27 due to an extremely low precision of 0.16. DNN performed moderately well with an F1 score of 0.71, but its recall (0.69) was lower than Random Forest and Decision Tree, indicating that it failed to detect some web attacks.

Detection of Infiltration Attacks: Detection of infiltration attacks (Class 2) was considerably more intricate. Random Forest and Decision Trees recorded F1 scores of 0.76 and 0.73, respectively; whereas SVM illustrated an F1 score of 0.64, reflecting difficulty in accurately identifying these kinds of attacks. The DNN performed the worst in this category, recording a recall rate of only 0.48 and an F1 score of 0.57, reflecting its weakness in correctly identifying infiltration attacks.

Botnet Attack Identification: All the four models performed well in identifying the botnet attacks (Class 3) using Random Forest, Decision Tree, and DNN achieving F1 scores of 0.98. SVM lagged behind by registering just 0.97 overall F1 score, yet variability between models was negligible.

Brute-force Attack Detection: For Brute-force attack detection (Class 4), Random Forest and Decision Tree demonstrated identical performance with F1 scores of 0.93. DNN achieved an F1 score of 0.92 due to slightly lower precision (0.88). SVM, while still competitive, had a lower recall (0.94) and F1 score (0.88), indicating that it missed some Brute-force attack instances.

DoS Attack Detection: DoS attack detection (Class 5) showed a similar trend, where Random Forest and Decision Tree attained the highest F1 scores (0.92), while DNN closely followed with 0.91. SVM, with an F1 score of 0.87, exhibited the weakest performance in this category due to a lower recall (0.81).

DDoS+PortScan Detection: For DDoS and Port Scan detection (Class 6), all models performed nearly identically, achieving F1 scores around 0.98. SVM was slightly weaker (0.97) but still highly effective.

Model	Random Forest	Decision Tree	SVM	DNN

Class 0 - Benign				
Precision	0.77	0.73	0.65	0.6
Recall	0.71	0.68	0.42	0.81
F1 Score	0.73	0.71	0.51	0.69
Class 1 - Web attack				
Precision	0.93	0.93	0.16	0.73
Recall	0.99	0.98	0.85	0.69
F1 Score	0.96	0.95	0.27	0.71
Class 2 - Infiltration attack				
Precision	0.73	0.71	0.59	0.73
Recall	0.79	0.75	0.71	0.48
F1 Score	0.76	0.73	0.64	0.57
Class 3 – Botnet attack				
Precision	0.98	0.98	0.96	0.98
Recall	0.98	0.97	0.98	0.98
F1 Score	0.98	0.98	0.97	0.98
Class 4 – Brute-force attack				
Precision	0.90	0.9	0.83	0.88
Recall	0.96	0.96	0.94	0.96
F1 Score	0.93	0.93	0.88	0.92
Class 5 - DoS attack				
Precision	0.96	0.96	0.93	0.95
Recall	0.89	0.89	0.81	0.87
F1 Score	0.92	0.92	0.87	0.91
Class 6 - DDoS+PortScan				
Precision	0.98	0.98	0.98	0.98
Recall	0.98	0.98	0.96	0.98
F1 Score	0.98	0.98	0.97	0.98

Table 3 - Precision, Recall, and F1 scores of Each Evaluated Machine Learning Models in Each Class in Stage 1

4.1.2 Overall Performance

To determine the most effective model for Stage 1, the Overall Accuracy, CV scores, and ROC-AUC scores of the four Machine Learning models as presented in *Table 4* were analyzed. The Random Forest model exhibited the highest accuracy, 0.89, followed closely by the Decision Tree model, 0.88. DNN model obtained an accuracy of 0.85 whereas the worst accuracy obtained by the SVM model was 0.81. From the results

we have the Random Forest and Decision Tree models having higher capacity for the classification between the malicious and benign network traffic. In addition, the CV scores gave useful information regarding the generalization capacity of the models. Once again, the Random Forest model proved the most effective by attaining a CV score of 0.90 closely followed by Decision Tree by attaining the score of 0.89, DNN attaining the score of 0.87 and SVM attaining the score of 0.82. It indicates that Random Forest and Decision Tree proved consistent performance on several training-validation splits and hence increased the dependability for the classification between Stage 1. Additionally, the ROC-AUC score that tests the models' ability to differentiate between the positive class and the negative class was analyzed. Random Forest model proved the maximum ROC-AUC score of 0.99 whereas Decision Tree and DNN proved the score of 0.98 whereas SVM proved the score less by 0.97. These measures prove that all the models proved high capacity for class separation whereas Random Forest proved higher discriminative capacity.

Model	Random Forest	Decision Tree	SVM	DNN
Accuracy	0.89	0.88	0.81	0.85
CV Score	0.90	0.89	0.82	0.87
ROC-AUC	0.99	0.98	0.97	0.98

Table 4 - Accuracy, CV Scores, and ROC-AUC Scores of All Machine Learning Models Trained in Stage 1

What's striking about one observation here is the relatively high scores for both models for Class 3 (Botnet attack) and Class 6 (DDoS+PortScan). Perhaps these kinds of attacks have very distinct traffic characteristics that the models have no problem distinguishing. Botnet attacks often generate characteristic network behaviors, such as repetitive communication patterns with command-and-control servers, which can be

effectively captured by machine learning models. Similarly, DDoS and Port Scan attacks produce high-volume, anomalous traffic that significantly deviates from normal network behavior, making them highly distinguishable. The well-defined nature of these attack patterns likely contributed to the consistently strong performance of all models in these two classes.

Overall, the Random Forest model demonstrated the most balanced performance in terms of accuracy, CV score, and ROC-AUC score, making it the optimal choice for Stage 1 classification. Although Decision Tree and DNN also reported competitive performance, the superior generalization ability and classification accuracy make Random Forest the preferred model to filter out the malicious traffic before classification in Stage 2.

4.1.3 Recap of Research Question#1 and Hypothesis#1

The first stage in the two-stage approach has the main purpose of efficiently distinguishing between malicious and benign traffic within the network so that the cyber maturity in the contexts of Software-Defined Networking (SDN) gets enhanced. Research Question #1 attempts to address the purpose by asking: Out of the classifies Random Forest, Decision Tree, SVM, and DNN, which classifier shows the most effectiveness in assessing and boosting the cyber maturity of SDN environments by the mechanism of intrusion detection?

The experimental results in this section give a detailed analysis of the four models being considered. The comparison of precision, recall, and F1-score for all classes of attacks (see *Table 3*), along with the overall accuracy, CV score, and ROC-AUC score, clarifies the relative strengths and weaknesses of each model. Of the four models

considered, the Random Forest model consistently performed the best, with the highest overall accuracy (0.89), the best CV score (0.90), and the highest ROC-AUC score (0.99). In addition, the Random Forest model showed strong classification behavior for all categories of attacks by maintaining an optimal balance between false positives and false negatives in comparison to the other models.

The best hyperparameter settings for each model, obtained through extensive tuning with RandomizedSearchCV, are as follows:

Random Forest: n_estimators: 500, min_samples_split: 2, min_samples_leaf: 1, max_features: 'sqrt', max_depth: 50, bootstrap: True

Decision Tree: criterion: 'gini', splitter: 'best', min_samples_split: 2, min_samples_leaf: 1, max_depth: 30

SVM: C: 1.0, kernel: 'rbf', gamma: 'scale'

DNN: units_1: 128, activation_1: tanh, dropout_1: 0.1, units_2: 192, activation_2: elu, dropout_2: 0.1

These findings provide strong empirical support for Hypothesis #1, which states: *Among Random Forest, Decision Tree, SVM, and DNN, Random Forest model can best assess and improve the cyber maturity of the SDN environments by predicting intrusions.* The superior performance of Random Forest in terms of classification metrics suggests that it is the most suitable model for Stage 1 of the framework, ensuring reliable intrusion detection in SDN environments. Therefore, Random Forest is selected as the optimal model for filtering out malicious traffic before forwarding benign traffic to Stage 2 for further classification. Although Random Forest was ideal, Decision Tree and DNN were both determined to have competing outcomes. They were not selected ultimately due to

Decision Tree being overfitting-prone, thus less generalization in classifying new attack types and DNN's excessive computational cost and training time, hence not ideal for real-time SDN intrusion detection. DNN also demonstrated a lower precision score, hence the system potentially with more false alarms. Thus, although Decision Tree and DNN were encouraging, Random Forest's balance between strength, efficiency, and interpretability was ideal for real-world applications.

4.2 Model Selection for Stage 2

Here, the performance of the four tested models for the classification task in Stage 2 QoS classification is analyzed and compared in order to identify the top performing Machine Learning model for classifying the benign traffic into specific application types for the subsequent assignment of priority levels. Comparison is based on multiple performance measures like accuracy, precision, recall, F1-score, and ROC-AUC to give a complete picture about the classification capacity of each respective model.

At Stage 2, four Machine Learning models—Random Forest, Light Gradient Boosting Machine (LightGBM), Extreme Gradient Boosting (XGBoost), and K-Nearest Neighbors (KNN)—were trained and tested using the IP Network Traffic Flows Labeled with 75 Apps dataset. The dataset contains over 3.1 million benign network flow records, distributed across multiple application categories. As shown in *Table 2*, the distribution of samples across categories is highly imbalanced, with Web traffic accounting for 2,718,207 records, whereas Collaborative applications have only 5,947 records. This class imbalance posed challenges in training models without bias toward the majority class, necessitating appropriate resampling techniques and weighted loss functions to improve minority class representation.

Given the complexity and size of this dataset, training the Machine Learning models required substantial computational resources. The entire training process was conducted on Google Colaboratory Pro using an NVIDIA A100 GPU runtime, consuming over 180 compute units to complete model training across all four models. The hyperparameter tuning process, particularly for LightGBM and XGBoost, was computationally intensive due to the iterative nature of boosting algorithms and the need for extensive hyperparameter optimization.

To ensure robust and unbiased model evaluation, Stratified K-Fold CV (with ten splits) was employed, preserving the proportion of each class within the dataset and mitigating potential biases from class imbalances. Each model was trained using RandomizedSearchCV to optimize hyperparameters, and the best model configurations were selected based on their performance on the validation set. After training, the models were tested on a separate holdout set, and their classification performance was analyzed using confusion matrices to assess classification errors.

Stage 2's main goal is classifying the benign flows based on specific types of applications that are associated with defined priority levels aimed at supporting QoS improvement. Prioritization will follow a methodical mapping approach where the applications that have high sensitivity towards delay, i.e., VoIP and collaborative tools, will receive the status of high-priority traffic and the non-business-critical traffic such as the traffic for social networking and other unknown traffic will receive the status of low-priority traffic.

Subsequent sections provide quantitative analysis of the performance of the models that emphasize both their strengths and weaknesses in identifying the application

categories with accuracy. Such analysis forms the basis for the determination of the most suitable model for Stage 2 and therefore effective QoS classification and network traffic prioritization in SDN scenarios. The following sections provide quantitative comparison of the performance of the models that include both their strengths and weaknesses in identifying application categories effectively. Evaluation forms the basis for the determination of the most suitable model for Stage 2 and therefore effective QoS classification and prioritization of network traffic in SDN scenarios.

4.2.1 Evaluation for Each Class

Table 5 illustrates the overall analysis based on the four Machine Learning models— Random Forest, LightGBM, XGB, and KNN—classification accuracy for each traffic class in Stage 2. Precision, Recall, and F1 score are the performance measures taken into consideration. These measures formulate the models' accuracy in classifying traffic types into respective types of applications providing optimal QoS. These measures are critical to identify the accuracy of each model in classifying network traffic into types of applications. Higher the value for Precision depicts the less misclassification and the value by Recall defines the capability of the model in classifying actual traffic instances. F1 score as the harmonic mean between Precision and Recall measures the balanced performance measure. Comparison between these measures defines the overall efficacy of the model in traffic classification having less misclassification.

Classification of Miscellaneous Traffic: Under general network traffic classification (Class 0 – Others), Random Forest achieved the best F1 score of 0.77, demonstrating a good balance between recall and precision. LightGBM and XGBoost (XGB) showed moderate performance with F1 scores of 0.68, indicating some limitations in their ability

to handle this class. The KNN algorithm was the least performing, with an F1 score of 0.52, thus struggling to distinguish between these traffic types effectively.

Web Traffic Classification: For web traffic classification (Class 1 – Web), Random Forest led the performance with an F1 score of 0.67, showing robust classification capability. LightGBM and XGB followed closely with F1 scores of 0.62 and 0.61, respectively. KNN exhibited the lowest performance, with an F1 score of 0.50, indicating a high misclassification rate.

Streaming Traffic Classification: Streaming traffic classification (Class 2 – Streaming) showed competitive performance across Random Forest, LightGBM, and XGBoost, with F1 scores ranging from 0.70 to 0.75. Random Forest attained the best performance with an F1 score of 0.75, while LightGBM and XGBoost closely followed at 0.71 and 0.70, respectively. KNN again lagged behind, achieving an F1 score of only 0.55.

Cloud Traffic Classification: Cloud-based traffic (Class 3 – Cloud) was best classified by Random Forest, which attained an F1 score of 0.81. LightGBM and XGBoost followed with 0.74 and 0.72, respectively. KNN, with an F1 score of 0.61, struggled in differentiating cloud-based applications from other traffic types.

Social Media Traffic Classification: Social media traffic classification (Class 4 – Social Media) exhibited strong results across Random Forest, LightGBM, and XGB, with F1 scores of 0.86, 0.81, and 0.79, respectively. Random Forest was the top performer in this category. KNN trailed significantly with an F1 score of 0.70, indicating challenges in distinguishing social media traffic accurately.

Email Traffic Classification: Email traffic classification (Class 5 – Email) was best handled by LightGBM, which achieved the highest F1 score of 0.78, followed closely by

XGB and Random Forest, both around 0.77 and 0.76. KNN performed the weakest in this category, with an F1 score of 0.61.

Software Update Traffic Classification: Software update traffic (Class 6 – Software Update) was classified with exceptional accuracy by Random Forest and LightGBM, both achieving an F1 score of 0.93. XGB also performed well with an F1 score of 0.92. KNN, while still effective, demonstrated a lower performance with an F1 score of 0.83.

VoIP Traffic Classification: VoIP traffic classification (Class 7 – VoIP) was dominated by LightGBM, with an F1 score of 0.80, followed closely by Random Forest and XGB at 0.79 and 0.78, respectively. KNN struggled in this category, attaining an F1 score of 0.63.

Collaborative Traffic Classification: For collaborative application traffic (Class 8 – Collaborative), LightGBM and XGB exhibited the strongest performance, both achieving an F1 score of 0.91. Random Forest followed closely at 0.85. KNN, with an F1 score of 0.81, performed decently but was outmatched by the tree-based models.

The evaluation results in Stage 2 indicate that Random Forest consistently demonstrated strong classification performance across all traffic types, making it a reliable choice for benign traffic classification. LightGBM and XGB also performed well, particularly in traffic types requiring fine-grained differentiation, such as social media, VoIP, and Collaborative applications. KNN, however, lagged behind in nearly all categories, suggesting that it is less suited for high-dimensional network traffic classification. These findings support the selection of Random Forest and LightGBM as the primary models for effective benign traffic classification in ResSDN, ensuring optimized QoS while maintaining accurate differentiation among application types.

Model	Random Forest	LightGBM	XGB	KNN
Class 0 – Others				
Precision	0.75	0.63	0.61	0.51
Recall	0.79	0.75	0.76	0.52
F1-Score	0.77	0.68	0.68	0.52
Class 1 - Web				
Precision	0.69	0.66	0.65	0.56
Recall	0.64	0.59	0.57	0.44
F1-Score	0.67	0.62	0.61	0.5
Class 2 – Streaming				
Precision	0.76	0.71	0.71	0.55
Recall	0.74	0.71	0.68	0.55
F1-Score	0.75	0.71	0.7	0.55
Class 3 – Cloud				
Precision	0.83	0.78	0.77	0.63
Recall	0.79	0.71	0.68	0.59
F1-Score	0.81	0.74	0.72	0.61
Class 4 - SocialMedia				
Precision	0.89	0.87	0.85	0.73
Recall	0.83	0.76	0.74	0.68
F1-Score	0.86	0.81	0.79	0.7
Class 5 – Email				
Precision	0.72	0.75	0.74	0.6
Recall	0.81	0.81	0.8	0.62
F1-Score	0.76	0.78	0.77	0.61
Class 6 – SoftwareUpdate				
Precision	0.93	0.93	0.91	0.81
Recall	0.93	0.93	0.92	0.87
F1-Score	0.93	0.93	0.92	0.83
Class 7 – VoIP				
Precision	0.77	0.78	0.76	0.59
Recall	0.81	0.82	0.8	0.67
F1-Score	0.79	0.8	0.78	0.63
Class 8 – Collaborative				
Precision	0.8	0.9	0.88	0.76
Recall	0.91	0.93	0.93	0.87
F1-Score	0.85	0.91	0.91	0.81

Table 5 - Precision, Recall, and F1 scores of Each Evaluated Machine Learning Models in Each Class in Stage 2

4.2.2 Overall Performance

To determine the most effective model for Stage 2, the overall accuracy, CV scores, and ROC-AUC scores of the four machine learning models, as presented in *Table 6*, were analyzed. The Random Forest model exhibited the highest accuracy (0.79), followed closely by LightGBM (0.77) and XGBoost (XGB) (0.76). KNN performed the worst with the minimum accuracy being 0.63, depicting relatively poorer performance in classifying benign application traffic. CV scores also provided knowledge about the generalizability of the models. Random Forest performed the best again with the maximum CV score being 0.82, followed by LightGBM (0.78) and XGB (0.76). KNN also obtained the minimum CV score being 0.63 and possessed the worst generalizability i.e., the performance would not generalize across all the different splits in the dataset. ROC-AUC score that tests the models' capacity for distinguishing the different classes in the application traffic also got compared. Random Forest and LightGBM also obtained the highest ROC-AUC scores (0.97), which were very closely followed by the XGB score (0.96). KNN had a markedly lower ROC-AUC score of 0.89, reinforcing its weaker classification performance compared to the other models.

Model	Random Forest	LightGBM	XGB	KNN
Accuracy	0.79	0.77	0.76	0.63
CV Score	0.82	0.78	0.76	0.63
ROC-AUC	0.97	0.97	0.96	0.89

Table 6 - Accuracy, CV Scores, and ROC-AUC Scores of All Machine Learning Models Trained in Stage 2

An interesting observation in Stage 2 is the consistently high classification performance for Class 6 (Software Update) and Class 8 (Collaborative Traffic) across all models, particularly Random Forest, LightGBM, and XGB. These two classes achieved

the highest F1-scores, with Random Forest and LightGBM reaching 0.93 and 0.91 for Software Update and Collaborative Traffic, respectively. One possible reason for this strong performance could be the distinct traffic characteristics of these application types. Software updates typically generate periodic, high-volume traffic with identifiable payload sizes and update server destinations, making it easier for machine learning models to classify accurately. Similarly, collaborative traffic (e.g., video conferencing, real-time document editing) exhibits unique patterns such as sustained, high-bandwidth connections and frequent bidirectional data exchanges, which may contribute to the models' ability to differentiate them from other traffic types. In contrast, Class 0 (Others) and Class 1 (Web Traffic) showed the lowest classification performance across all models, with KNN performing particularly poorly (F1-scores of 0.52 and 0.50, respectively). The lower performance in these categories suggests a higher overlap of feature distributions between different web-based applications, making them harder to classify distinctly. Web traffic often consists of a mix of static and dynamic content, encrypted and unencrypted data, and varying access patterns, which could introduce classification challenges. Another notable trend is that KNN consistently underperformed across all classes, with the lowest overall accuracy (0.63) and ROC-AUC score (0.89).

This suggests that KNN performs poorly in the complexity of multi-class benign traffic classification, likely because it finds itself vulnerable to high-dimensional input and relying on distance-based measures of similarity that may not accurately capture the fine-grained differences among traffic types. These results highlight the strength of tree-based models, namely Random Forest, LightGBM, and XGB, in performing traffic classification in the ResSDN scenario. Their ability to identify complex patterns and

define decision boundaries in high-dimensional network traffic information makes them better suited than KNN for this specific task.

On the whole, Random Forest achieved the best classification performance in Stage 2 with the best accuracy, CV score, and ROC-AUC score. LightGBM and XGB also achieved good performance and would thus represent good alternatives. KNN had comparatively poor accuracy and generalization ability and would thus be unlikely to be adequate for efficient benign traffic classification in the context of ResSDN.

4.2.3 Recap of Research Question#2 and Hypothesis#2

The major goal of Stage 2 in the proposed two-stage approach is the systematic classification of the diverse types of benign traffic and ranking them based on QoS requirements. By creating clear demarcating borders between application classes, the approach possesses the ability to heighten traffic management within the realm of SDN settings, thus ensuring maximum network performance along with consistent security measures.

This objective aligns with Research Question #2, which investigates the following: *Among Random Forest, LightGBM, XGBoost, and KNN, which model performs best in analyzing and improving the Quality of Service (QoS) in Software Defined Networking (SDN) environments using priority assignment?* The experimental results presented in this section provide an overall evaluation of the performance of these four models. The comparison of classification accuracy, CV metrics, and ROC-AUC scores (*Table 6*) clarifies the relative strengths and weaknesses of each model with regard to the prioritization of different types of application traffic.

The best hyperparameter settings for each model, obtained through extensive tuning with RandomizedSearchCV, are as follows:

Random Forest: n_estimators: 300, min_samples_split: 5, min_samples_leaf: 1, max_features: 0.3, max_depth: 30, bootstrap: False

LightGBM: subsample: 0.8, reg_lambda: 0, reg_alpha: 0, num_leaves: 31, n_estimators: 500, min_child_samples: 20, max_depth: 10, learning_rate: 0.1, colsample_bytree: 0.7

XGBoost: subsample: 0.8, reg_lambda: 0, reg_alpha: 0.1, n_estimators: 200, min_child_weight: 1, max_depth: 6, learning_rate: 0.1, gamma: 0.3, colsample_bytree: 1.0

KNN weighting method: 'distance', neighbors: 7, distance function: 'manhattan'. The results achieved using CV are further used to support the generalization capability of the models, and Random Forest achieved the highest result of 0.79, closely followed by XGBoost with 0.76 and LightGBM with 0.77. Further, XGBoost had an outstanding ROC-AUC score result of 0.96, which indicates its excellence in identifying subtle differences in application traffic patterns.

Despite the initial hypothesis#2 that XGBoost would perform better than other models due to its ability to handle structured data, the empirical results indicated that Random Forest showed improved efficacy. Random Forest achieved the best overall accuracy of 0.79 compared to 0.76 of XGBoost, along with better CV scores and ROC-AUC measures. This deviation from the first hypothesis may be attributed to a range of factors:

Generalization Ability and Stability: Random Forest demonstrated superior generalization ability, particularly in handling high-dimensional traffic information.

Unlike XGBoost, which relies on boosting-based tree refinement, Random Forest employs fully grown decision trees with bootstrap aggregation, making it more resilient to overfitting, especially in a dataset with imbalanced traffic classes. The inherent randomness in Random Forest helps improve generalization by reducing variance.

Computational Complexity and Training Efficiency: XGBoost's iterative boosting process, while powerful, is computationally intensive and highly sensitive to hyperparameter tuning. The need for precise tuning increased the risk of suboptimal model configurations, impacting overall performance. In contrast, Random Forest, with fewer hyperparameter dependencies, produced more stable results across multiple CV folds.

Handling of Imbalanced Classes: The dataset used in Stage 2 exhibited a highly imbalanced distribution of traffic classes, with Web traffic dominating the dataset while Collaborative applications had significantly fewer instances. XGBoost struggled to balance classification performance across all classes, exhibiting lower recall in minority classes compared to Random Forest. Random Forest’s ability to assign weights dynamically and handle class imbalances contributed to its superior F1-scores across multiple traffic categories.

Feature Interactions and Interpretability: Random Forest 's ensemble nature enables its ability to readily detect complex feature interactions. While XGBoost has a good capability to distinguish between features of importance, its tendency for the stringent refinement of trees can lead to an over-emphasis on specific patterns, possibly at the cost of detecting more general trends in network traffic behavior. This, in turn, can cause

reduced robustness in the classification of application types with high intra-class variability.

Taking the above results into consideration, Random Forest was eventually chosen for QoS prioritization based on higher classification accuracy, less sensitivity towards hyperparameter tuning, and better traffic class imbalance handling. Although there exist viable alternatives like XGBoost offering high precision-recall trade-off, higher computational complexity and sensitivity towards overfitting by this specific dataset made Random Forest the better choice for real-world SDN scenarios.

4.3 Tool Assessment

To enable comprehensive testing of the proposed Two-Stage Machine Learning Framework for enhanced cybersecurity and QoS in SDN environments, GNS3-based testing was created. GNS3 configuration was selected in order to create and collect a varied set of network traffic types that include legitimate application flows as well as malicious attempts at infringement. Evaluation purpose is the identification of the effectiveness of the framework in the timely detection of intrusions (Stage 1) and the classification of innocuous traffic according to QoS priority ordering (Stage 2). Through the use of legitimate tools for attack simulation coupled with common applications, the test verifies the operational effectiveness in real-world SDN applications.

Overall, 191,658 traffic events were captured using Wireshark, covering a variety of traffic types to develop a large dataset for model training and validation. The assessment involves both attack-based and application-based traffic classification, reflecting real-world network conditions and security challenges.

4.3.1 Simulation Setup

As illustrated in *Figure 4*, the simulation environment was carefully designed to emulate a realistic SDN network. The GNS3-based setup includes a programmable SDN controller, multiple virtualized hosts, and network switches to manage and route traffic dynamically. Traffic was generated using dedicated tools and applications, ensuring a diverse dataset for machine learning assessment. The following categories of network traffic were simulated:

Attack Traffic (To evaluate the intrusion detection capability in Stage 1, six different types of cyber-attacks were executed):

- Brute Force Attack (Patator FTP): Simulates unauthorized login attempts to compromise network credentials.
- DoS Attack (Slowloris): Creates a Denial-of-Service condition by keeping HTTP connections open indefinitely.
- Web Attack (XSS): Injects malicious scripts into web applications to exploit client-side vulnerabilities.
- Infiltration Attack (Nmap): Uses network scanning techniques to probe and exploit security weaknesses.
- Botnet Attack (Ares): Simulates botnet communication used for large-scale cyber-attacks.
- DDoS Attack (hping3): Floods the target with network packets to overwhelm resources and disrupt services.

Benign Application Traffic (To assess the QoS-based traffic classification in Stage 2, nine categories of benign application traffic were captured):

- Web Browsing Traffic: General HTTP/HTTPS traffic from common web usage.
- Video Streaming Traffic: Traffic generated from YouTube and Netflix to evaluate real-time streaming QoS.
- Cloud File Sharing Traffic: Data transfers from Google Drive and Microsoft OneDrive representing cloud storage usage.
- Social Media Traffic: Interactions with Facebook and Instagram, reflecting social networking behavior.
- Email Traffic: Communications through Outlook and Gmail, representing business and personal email exchanges.
- Software Update Traffic: Traffic from Windows Update, ensuring network adaptability to patching activities.
- Collaborative Traffic: Data flows from Microsoft Teams, representing real-time collaboration tools.
- Miscellaneous traffic: Other traffic like Uber Eats that mimic real-world background network traffic.

All the network traffic was sniffed using Wireshark, allowing the analysis of packets and extracting the dataset in detail. Preprocessing the captured traffic for extracting useful features for training the models, the task targets accurate classification for both intrusion detection and QoS-based traffic prioritization. This setup ensures that the Two-Stage Machine Learning framework is tested stringently in real-world scenarios, validating the effectiveness in both security and network performance improvement in SDN scenarios.

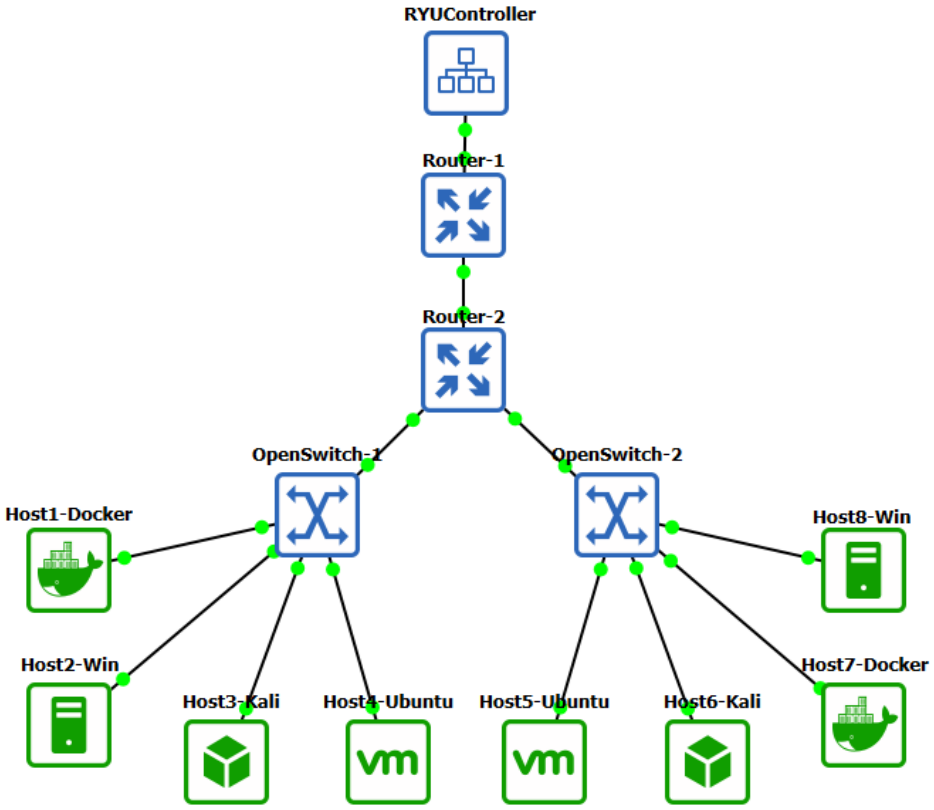

Figure 4 - Simulated SDN Topology in GNS3

4.3.2 Data Preprocessing

Data preprocessing forms an essential component within the field of machine learning because it ensures that the dataset is properly smoothed, structured, and processed for training the models. Subsequent sections outline the specific preprocessing steps performed on the simulated SDN traffic dataset before being included in the constructed two-stage machine learning models for classification and prioritization. First, the dataset is imported, and structural analysis performed to identify the number of entries, and the number of columns included. To enhance consistency, the column names are standardized by the removal of extraneous whitespace. It is also critical in

maintaining consistency within the dataset and preventing the likelihood of mistakes within the subsequent process steps.

These types of network traffic within the dataset are reclassified into two general types: Benign and Malicious. This reclassification of the traffic into these types is achieved using the mapping technique in order to maximize the identification of the security threats in the network traffic. Numerical encoding is performed after the classification process in order to convert the categorical labels into numeric representations needed by the machine learning models. Priority levels are labeled for each traffic type based on the importance and the effect within the network. Proper traffic management based on these labeled priority levels ensures that the essential services receive higher priority scores. Transcoding these priorities into numeric representations makes the dataset excellently preprocessed for analysis and classification.

In order to strengthen the dataset, the irrelevant categorical variables are dropped and only the features that will prove useful for subsequent analysis are kept. The dataset is segregated into feature variables and target labels to facilitate two distinct stages of classification. Non-numerical features are converted into numerical representations by the use of label encoding, making them amenable to incorporate into machine learning models. Handling missing values and outliers is done by replacing infinite values with a neutral placeholder, as well as filling missing values with their respective means. This approach reduces inconsistencies and ensures that the dataset remains robust for training.

Imbalanced datasets are common in real-world situations, and the Synthetic Minority Over-sampling Technique (SMOTE) must therefore be applied in order to balance class distribution. SMOTE creates synthetic examples of the minority class so

that the generalization ability of the model across traffic types improves. SMOTE is complemented by the addition of a safeguard mechanism that verifies the presence of one class so that SMOTE does not accidentally target normal operation. Where necessary, missing values are replaced by a placeholder for consistency and SMOTE applied to correct dataset imbalance. The process makes the development of the priority model based on balanced priority levels. It effectively prevents biasing towards the most common traffic types.

The dataset, now preprocessed, is divided into training and test subsets using an 80-20 split ratio. This technique ensures that the model is trained on a diversified set of data while, at the same time, maintaining a portion for purposes of evaluation. Stratified sampling is used to maintain the class distribution in the training and testing sets, hence allowing for an unbiased assessment of the model's performance. Importance of features is assessed using the assistance of an XGBoost classifier, whose purpose is to identify the most relevant features as related to classification, in correlation with the practice used in the training of the best model for each stage. Upon feature importance score examination, the top 50 features are selected for the next training. Feature selection technique improves model performance by reducing dimensionality while focusing on the most relevant predictors. The datasets are then trimmed down to contain solely the chosen features, with a view to the model being trained using the most informative features available. This practice improves operational effectiveness and fosters better generalization in classifying new traffic patterns.

4.3.3 ResSDN Model Training

Following preprocessing of the dataset for integrity, balance, and accurate selection of the features, the next phase comprises model training. Model training comprises the use of pre-trained models that are fine-tuned using the processed dataset to enable accurate classification of network traffic and packet prioritization. Model training process is also complemented by the use of optimized and stored machine learning models that are fine-tuned by using the application of hyperparameter optimization for optimal performance. Two-stage classification comprises:

Stage 1 Model: It distinguishes the network traffic into malicious or benign traffic. Adding the Stage 1 model into the preprocessed training dataset starts the training process. Distinguishing between malicious and benign traffic is trained by the model by learning the pattern from the feature set. After training, the model makes the prediction on test set examples that are unseen. In addition, the adaptive security threshold in intrusion detection is also utilized by ResSDN. Real-time optimization of the classification accuracy by the system is enabled by dynamic adaptation based on the prediction threshold and the confidence scores obtained from the model. Predictions based on confidence scores >= 0.7 are marked as malicious reducing the rates of the false positives and boosting the detection reliability. Isolation Forest-based anomaly detection also finds use in the detection and processing of the data that has the patterned anomaly in real-time. Incorporating the feature of anomaly detection optimizes the security by invoking re-modeling in the event of any anomaly in the dataset existing, keeping the model for intrusion detection adaptive and up-to-date. Predicted Output includes Class Labels (A two-class classification between benign or malicious traffic) and Probability

Scores (A confidence score that includes the prediction probability for any prediction assisting in the further decision-making process). Utilization of the pre-trained model optimizes the training for computational cost-effectiveness ensuring maximum accuracy in classification.

Stage 2 Model: This stage continues to classify network traffic based on its respective priority levels, thus enabling the effective distribution of network resources. Following the classification of malicious traffic in Stage 1, this second stage augments the classification by prioritizing legitimate traffic. The Stage 2 model is trained on a separate dataset that includes traffic labeled with priority levels, so that services labeled as high priority are given resource allocation accordingly. Complementary to the above stage, predictions are produced from the test dataset that include **Priority Labels** (numerical values that indicate the priority of different types of network traffic) and **Probability Scores** (an indicator of confidence that aids in evaluating the reliability of the priority assignments). Utilization of the pre-trained models simplifies the training process and improves the overall effectiveness of the models by combining new input data that transmits new information. Segmented classification ensures systematic treatment where optimization of the security and QoS is handled prudently. Model training process depends on the preprocessing dataset that enables effective traffic classification and simultaneous priorities assignment. Utilization of the pre-trained models promotes process optimization for enhanced effectiveness while maintaining strict cybersecurity measures and ensuring capable traffic handling capabilities. Trained models are now set for testing and confirmation in order to determine the accuracy, precision, and overall effectiveness in real-world applications.

4.3.4 ResSDN Model Evaluation and Validation

In order to test the performance and reliability of the constructed models, there exists a complete test process. Performance measures are highlighted by the analysis to test the robustness of the model. Performance measures are utilized in both the Stage 1 and the Stage 2 models: **Accuracy** measures the overall accuracy of the predictions by the model. **Classification report** includes precision, recall, and the F1-score utilized for the calculation of the balance between the specificity and the sensitivity. **ROC-AUC Score** is the measure that computes the performance of the model in distinguishing between the classes well. To improve the generalizability of the model, the method utilizes the 10-fold stratified CV. It splits the dataset into folds that enables the iterative train and validate process in order to minimize the occurrence of overfitting and analyze the robustness of the model. Mean and the standard deviation of the scores of the validations provide informative information about the consistency of the model. The ROC-AUC scores are calculated for each classification stage that shows the balance between the trade-off in the rates between the true positive and the false positive. ROC-AUC depicts the ability of the model in distinguishing between the classes. Higher values of the ROC-AUC represent better classification performance.

Analysis and comparison on the Stage 1 model illustrates the effectiveness of the model in classifying malicious and benign traffic with great accuracy. Evaluator measures that are common in classification such as precision, recall, F1-score, accuracy, CV score, and the ROC-AUC score are utilized to measure the performance of the two traffic classes. Under the **Class 0** that signifies benign traffic, the model has a precision of 0.98, which illustrates the low percentage of the occurrence of false positives, and

recall of 0.94, which illustrates the capability of the model in classifying most benign traffic. Additionally, the F1-score at 0.97 illustrates good recall and precision balance. Under the class that denotes Malicious Traffic as **Class 1**, accuracy in the case of malicious traffic by ResSDN stands at 0.94, which illustrates the high proportion of the malicious traffic that gets classed effectively. A recall of 0.98 illustrates the high capacity by the model in detecting malicious traffic. Additionally, the F1-score at 0.97 illustrates consistency in the accuracy of the classification. The model has an **Overall Accuracy** level of 0.9686 that illustrates the accuracy by the model in classifying the traffic in most scenarios. The 0.9689 CV accuracy strengthens the validity and universal applicability of the model to different sets of data, thus reducing the possibility of overfitting. The model proves its effectiveness through a **ROC-AUC** measure of 0.9711, which indicates a strong ability to distinguish between malicious and harmless traffic, thus highlighting its effectiveness.

From *Tables 7* and *Table 8* results, the Stage 1 model clearly classifies the network traffic with significant precision, recall, and overall accuracy. In addition, the CV results and the ROC-AUC values obtained further support for the consistency and usability of the model. These results support the correctness in classifying malicious traffic by the model and provide a solid foundation for the following Stage 2 classification based on the prioritization of network traffic.

ResSDN Stage 1	
Class 0 - Benign	
Precision	0.98
Recall	0.94
F1-Score	0.97
Class 1 - Malicious	
Precision	0.94

Recall	0.98
F1-Score	0.97

Table 7 - Evaluation Scores of Each Class in Stage 1

ResSDN Stage 1 Summary	
Accuracy for Stage 1	0.9686
Precision for Stage 1	0.9705
Recall for Stage 1	0.9686
F1-Score for Stage 1	0.9686
CV Score for Stage 1	0.9685
AUC Score for Stage 1	0.9711

Table 8 - Performance Summary of Stage 1

Stage 2 model testing and verification in the ResSDN shown in *Table 9* determines the accuracy by which the model classifies network traffic into priority levels for resource allocation optimization. Priority-Level Classification accuracy of the model determines precision, recall, F1-score, accuracy, CV score, and ROC-AUC score. Priority levels have been effectively classified by the model for optimal QoS management:

Priority Level 4: Possesses superior classification ability with a precision level of 0.97, recall level of 0.98, and F1-score level of 0.98, which indicates strong dependability in identifying high-priority traffic.

Priority Level 3: Achieves 0.82 precision, 0.94 recall, and an F1-score of 0.88, demonstrating good sensitivity in classifying this priority category, though precision is slightly lower.

Priority Level 2: Maintains a balanced performance with 0.93 precision, 0.95 recall, and an F1-score of 0.94, ensuring accurate classification for mid-priority traffic.

Priority Level 1: Records 0.98 precision, highlighting minimal false positives. However, the 0.73 recall suggests some misclassification of lower-priority traffic, leading to an F1-score of 0.85.

Overall Accuracy: The model achieves an overall accuracy of 0.9249, confirming its effectiveness in traffic prioritization.

CV Score: 0.9549 indicates that the model performs consistently across different subsets of the dataset.

ROC-AUC Score: The model achieves an outstanding ROC-AUC score of 0.99, reflecting its strong discriminatory power in priority classification.

The evaluation results as shown in *Table 9* affirm that the Stage 2 model efficiently classifies benign network traffic into appropriate priority levels, ensuring effective QoS management. While high-priority classes exhibit excellent performance, improvements in recall for lower-priority levels could further enhance classification balance. Overall, the approach is sound and consistent and provides the critical ingredient towards traffic smoothing and asset allocation on the network for ResSDN as evidenced by *Table 10.*

ResSDN Stage 2	
Priority Level 4	
Precision	0.97
Recall	0.98
F1-Score	0.98
Priority Level 3	
Precision	0.82
Recall	0.94
F1-Score	0.88
Priority Level 2	
Precision	0.93
Recall	0.95
F1-Score	0.94
Priority Level 1	
Precision	0.98
Recall	0.73
F1-Score	0.85

Table 9 – Evaluation Scores of Each Class in Stage 2

ResSDN Stage 2 Summary	
Accuracy for Stage 2	0.9249
Precision for Stage 2	0.9314
Recall for Stage 2	0.9249
F1-Score for Stage 2	0.9230
CV Score for Stage 2	0.9549
AUC Score for Stage 2	0.9900

Table 10 - Performance Summary of Stage 2

The test framework includes the examination of latency in both the classification stages. Latency (millisecond), which measures the duration taken by each prediction, and False Positives, another essential parameter for evaluating the implications of misclassifying benign traffic, are now included in the contemporary performance metrics. These measures ensure that the real-world network situations where latency and false positive rates have significant impacts on QoS are well-supported by ResSDN. Examination of the Security-QoS trade-off by ResSDN, as presented in *Table 11*, offers the complete analysis of latencies and false positives at each stage. Intrusion detection systems directly affect QoS based on several essential trade-offs:

Intrusion Detection Latency (1055.69ms): This indicates a moderate delay in processing in Stage 1, which is expected due to extensive inspection functions like anomaly detection, feature selection, and adaptive threshold evaluation. While this amount of latency is still tolerable, further optimization may be able to reduce delays, especially for time-sensitive traffic classes like VoIP and video conferencing.

QoS Classification Latency (2053.81ms): The significantly higher latency in Stage 2 shows that the complexity of the prioritization model or the workload of data processing is high. This latency can be enhanced by reducing the feature set, reducing redundant calculations, or enhancing model efficiency.

Also, false positives are also the key metric that should be analyzed when reviewing the effectiveness of the tool because such incidents might impair network performance. Therefore, these need to be minimized by careful model selection and tuning. Random Forest was chosen for having the lowest false positive rate along with high interpretability. Secondly, threshold tuning was also performed in order to vary the decision boundaries. CV methods ensured that the model performs well in respect to several attack scenarios. Lastly, the two-stage architecture of ResSDN also aids in the process—by first filtering the malicious traffic, the following stage makes sure that legitimate traffic is classified properly for QoS management.

False Positives Rate at Stage 1 (0.0628): A relatively high number of false positives at this stage suggests a conservative intrusion detection mechanism. While this ensures security by minimizing false negatives (undetected malicious traffic), it may inadvertently affect benign traffic. Refining the model's threshold, improving feature selection, or integrating additional context-aware filtering may help reduce these false positives.

False Positives Rate at Stage 2 (0.0188): This result is optimal, indicating that the QoS prioritization model accurately assigns priority levels without misclassifying benign traffic. This precision contributes positively to network performance and resource allocation.

ResSDN Security-QoS Trade-Off Analysis	
Latency from Intrusion Detection	1055.69ms
Latency from QoS Classification	2053.81ms
False Negatives at Stage 1 (Packet Drops)	0.0062
False Positives at Stage 1 (Packet Drops)	0.0628
True Negatives at Stage 1 (Packet Drops)	0.9372

True Positives at Stage 1 (Packet Drops)	0.9938
False Negatives at Stage 2 (Misclassified Traffic)	0.0751
False Positives at Stage 2 (Misclassified Traffic)	0.0188
True Negatives at Stage 2 (Misclassified Traffic)	0.9812
True Positives at Stage 2 (Misclassified Traffic)	0.9249

Table 11 - ResSDN Security-QoS Trade-Off Analysis

4.3.5 Recap of Research Question#3 and Hypothesis#3

The third research question sought to determine whether the proposed joint QoS-Security Optimization tool, ResSDN, effectively enhances cyber maturity and QoS within SDN environments. The corresponding hypothesis posited that ResSDN can enhance cyber maturity and QoS by mitigating security risks while optimizing resource allocation. The evaluation of Stage 1 (Intrusion Detection) and Stage 2 (Traffic Prioritization) of ResSDN demonstrates its capability to fulfill this objective.

Stage 1 model, which was responsible for harmful traffic categorization, achieved the accuracy score of 0.9686, the CV score of 0.9685, and the ROC-AUC score of 0.9711 that signifies high performance in distinguishing harmful and benign traffic. Additionally, high recall measures for Malicious traffic, 0.98, and Benign traffic, 0.94, further vindicate the ability of the model in threat detection related to the improvement in cyber maturity in the scenarios involving SDN.

The Stage 2 model, tasked with the prioritization of benign network traffic based on QoS demands, achieved an accuracy level of 0.9249, a CV score of 0.9202, and an ROC-AUC score of 0.99. High-priority traffic classes (e.g., Priority Level 4) showed excellent classification effectiveness, with precision, recall, and F1-scores exceeding 0.97, thus ensuring that critical network services are afforded the necessary resource allocation. While Priority Level 1 had a somewhat lower recall rate of 0.73, the

classification performance overall indicates that ResSDN excellently optimizes network traffic flow while maintaining the integrity of security.

A great improvement seen in ResSDN is that it has the ability to self-tune based on the current network condition. It does this by incorporating the following strategies:

Dynamic Adaptive Thresholding at Stage 1: adaptive_threshold() function makes the real-time adaptation of the classification threshold possible using the confidence scores produced by the system. Unlike the reliance on a fixed probability threshold (e.g., 0.5), the system continuously monitors confidence scores and adjusts the threshold in order to minimize false positives without compromising on high detection accuracy. As an example, in situations where traffic changes in the network result in the increase in false positives, the system has the ability to increment the threshold gradually in order to ensure that high-confidence malicious traffic gets classified.

Anomaly Detection and Model Retraining: ResSDN also includes an anomaly detection module that utilizes the Isolation Forest approach for monitoring the traffic pattern in order. When detecting significant discrepancies, the model triggers the process for retraining the model, hence allowing itself to adapt based on the latest network status. This process ensures the effectiveness of ResSDN is not compromised despite the changes in attack mechanisms over time.

Real-time Latency Monitoring and Optimization: The evaluation function monitors latency at both stages to guarantee that predictions fall within acceptable response times in SDN environments. Excessive latency beyond a threshold value prompts the system to dynamically optimize the performance by updating the feature set that is invoked for classification or by invoking lightweight models for real-time traffic.

Priority-Based Processing at Stage 2: In the second stage, the model focuses on the prioritization of traffic based on the assessment of real-time application demands. When congestion is detected, the system dynamically adjusts processing resources to higher priority traffic classes, thus enhancing QoS results. The classifier's probability scores are used to enhance prioritization decisions, thus reducing the chances of misclassification and ensuring fairness in the allocation of resources.

By supporting adaptive thresholding, real-time anomaly detection, and latency-sensitive processing, ResSDN is very responsive. These operations allow the system to manage emerging network threats efficiently without QoS degradation. These kinds of responsiveness are critical within SDN ecosystems where intelligent and adaptive traffic management must be maintained in order to optimize the network. From the Security-QoS Trade-Off in *Table 9*, the ability of ResSDN in ensuring effective security enforcement without significant QoS interference towards the combined optimization goal of the project is demonstrated. Overall, the results prove Hypothesis #3 by demonstrating that ResSDN effectively integrates security and QoS optimization in SDN ecosystems. Through the elimination of security vulnerability by effective intrusion detection and the optimization of resource assignment by intelligent traffic prioritization, ResSDN enhances both cyber maturity and QoS. These results validate the tool’s potential as a strategic platform for effective and secure network management in SDN ecosystems.

Chapter 5:Conclusion

5.1 Results Summary and Analysis

This research introduced and evaluated ResSDN, a two-stage machine learning method for the simultaneous optimization of Software-Defined Networking (SDN) Quality of Service (QoS) and security. Stage 1 involved machine learning-based intrusion detection using machine learning models that were trained on the CIC-IDS-2018 dataset, whereas the benign traffic classification into application classes for QoS prioritization was the focus using the IP Network Traffic Flows Labeled with 75 Apps dataset. The result showed that the Random Forest approach performed better than Decision Tree, Support Vector Machine (SVM), and Deep Neural Network (DNN) in the case study for Stage 1 intrusion detection. Random Forest achieved the highest overall accuracy score (0.89), Cross-Validation (CV) score (0.90), and ROC-AUC score (0.99), thereby ensuring the efficacy in identifying cyber threats. Stage 2 involved the comparative analysis between Random Forest, Light Gradient Boosting Machine or LightGBM, and eXtreme Gradient Boosting or XGBoost and K-Nearest Neighbors or KNN based on QoS classification. Random Forest performed the best when it came to class-based classification among the application types and the most robust model against traffic classification and priority. Model estimation proved that the network security and resource allocation are greatly enhanced by the use of ResSDN.

5.2 Contributions to the Body of Knowledge

This work contributes to existing literature in the following manner. First, the work presents the new two-stage machine learning-based approach called ResSDN that

simultaneously addresses the issue of cybersecurity attacks and QoS in SDN. Unlike previous work that has typically treated these factors in seclusion, this work focuses on the interconnected nature of QoS and security and suggests an integrative approach. Second, the empirical analysis of machine learning models provides insightful results about the performance of the models in terms of intrusion detection and QoS classification. The results identify that the random forest model performs the most efficiently in terms of intrusion detection, while LightGBM and Random Forest prove highly effective in traffic classification. These results provide the foundation for further work and the implementation of machine learning-based network security and QoS improvement. Lastly, the work presents a realistic implementation plan for the deployment process of the ResSDN system in real-world SDN environments. Integration using supervised learning approaches, hyperparameter tuning, and multi-class classification makes the SDN controllers more flexible and effective, offering the scalability needed for enterprise, ISP, and cloud computing environments.

5.3 Recommendations for Future Research

Though positive results have been demonstrated by ResSDN, several fields require further research. Follow-on studies should consider:

Real-Time Adaptability: The use of online learning methods allows for continuous model adjustment as new network threats and traffic patterns evolve.

Clarification and Understandability: Examining explainable artificial intelligence methodologies to enhance transparency in machine learning-based intrusion detection and traffic classification systems.

Integration with SDN Controllers: Developing real-time SDN-compatible APIs to facilitate seamless integration of Machine Learning models with SDN controllers such as OpenDaylight and ONOS.

Improved Feature Engineering: Examining additional features of network traffic to improve the classification accuracy and model generalizability.

Evaluation in Realistic Network Scenarios: Deploying ResSDN in real SDN testbeds or live operational environments for the purpose of testing the scalability, robustness, and effectiveness under typical conditions.

5.4 Conclusion

This work has effectively developed and tested the ResSDN two-stage machine learning method for simultaneous QoS and security optimization in SDN networks. From the results, the reality that Machine Learning -based algorithms like Random Forest and LightGBM are highly effective for QoS classification and intrusion detection were justified. The contribution of this work extends beyond concept because the work offers practical recommendations towards the actualization of Machine Learning-based QoS optimization and security in SDN.

By bridging the gap between QoS management and security, ResSDN makes the next-generation intelligent network optimization possible. They're driven by the outcome towards developing interpretable, scalable, and adaptive Machine Learning methods in such a manner that SDN networks are able to meet growing performance and security requirements in the existing networking environments.

References

Ahmad, S., Mir, A.H. (2020). Scalability, consistency, reliability and security in SDN controllers: A survey of diverse SDN controllers. *Journal of Network and Systems Management 29*(9). https://doi.org/10.1007/s10922-020-09575-4

Aleem, S. & Ahmed, S. (2023). Unlocking network security and QoS: The fusion of SDN, IoT, and machine learning: A comprehensive analysis. *International Journal of Science and Research in Network Security and Communication, 11*(6). https://ijsrnsc.org/pub_paper/IJSRNSC/3-IJSRNSC-00637.pdf

Bagaa, M., Taleb, T., Bernabe, J. B., & Skarmeta, A. (2022). QoS and resource-aware security orchestration and life cycle management. *IEEE Transactions on Mobile Computing*, *21*(8), 2978–2993. https://doi.org/10.1109/tmc.2020.3046968

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade* (pp. 437-478). Springer.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research, 13*, 281-305. http://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition, 30*(7), 1145-1159. https://doi.org/10.1016/S0031-3203(96)00142-2

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5-32. https://doi.org/10.1023/A:1010933404324

Bozorgchenani, A., Zarakovitis, C. C., Chien, S. F., Ting, T. O., Ni, Q., & Mallouli, W. (2023). Novel modeling and optimization for joint cybersecurity-vs-QoS intrusion detection mechanisms in 5G networks. *Computer Networks, 237*, 110051. https://doi.org/10.1016/j.comnet.2023.110051

Chaudhry, S., & Soni, S. (2022). Machine learning based QoS prediction in network management. *Journal of Network and Systems Management, 30*(2), 243-258. https://doi.org/10.1007/s10922-022-09503-3

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785-794.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273-297. https://doi.org/10.1007/BF00994018

Coman, V.-V., & Dobrota, V. (2022, November 10-11). Open-source software-defined wide area network: An inter-cloud approach [Conference session]. 2022 International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania. https://doi.org/10.1109/isetc56213.2022.10010128

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

Elsayed, M. S., Le-Khac, N.-A., & Jurcut, A. D. (2020). InSDN: A novel SDN intrusion dataset. *IEEE Access, 8*, 165263–165284. https://doi.org/10.1109/access.2020.3022633

Fadlullah, Z. M., Mao, B., & Kato, N. (2022). Balancing QoS and security in the edge: existing practices, challenges, and 6G opportunities with machine learning. *IEEE Communications Surveys & Tutorials, 24*(4), 2419–2448. https://doi.org/10.1109/comst.2022.3191697

Fadlullah, Z. Md., Wei, C., Shi, Z., & Kato, N. (2017). GT-QoSec: A game-theoretic joint optimization of QoS and security for differentiated services in next generation heterogeneous networks. *IEEE Transactions on Wireless Communications, 16*(2), 1037–1050. https://doi.org/10.1109/twc.2016.2636186

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861-874. https://doi.org/10.1016/j.patrec.2005.10.010

Gelenbe, E., Domanska, J., Frohlich, P., Nowak, M. P., & Nowak, S. (2020). Self-aware networks that optimize security, QoS, and energy. *Proceedings of the IEEE, 108*(7), 1150–1167. https://doi.org/10.1109/jproc.2020.2992559

Hammad, M., Hewahi, N., & Elmedany, W. (2023). Enhancing network intrusion recovery in SDN with machine learning: An Innovative approach. *Arab Journal of Basic and Applied Sciences, 30*(1), 561–572. https://doi.org/10.1080/25765299.2023.2261219

Huang, H., Zhang, J., He, W., & Wang, X. (2020). Machine learning-based network traffic classification: A survey. *Computer Networks, 178*, 107345. https://doi.org/10.1016/j.comnet.2020.107345

Hutter, F., Kotthoff, L., & Vanschoren, J. (2011). Automated machine learning: Methods, systems, challenges. *Springer*.

Kaliyamurthy, N. M., Taterh, S., Shanmugasundaram, S., Saxena, A., Cheikhrouhou, O., & Ben Elhadj, H. (2021). Software-defined networking: An evolving network architecture—Programmability and security perspective. *Security and Communication Networks, 2021*, 1–7. https://doi.org/10.1155/2021/9971705

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems, 30*, 3146–3154.

Khamosh, A., Anwer, M. A., Nasrat, N., Hamdard, J., Gawhari, G. S., & Ahmadi, A. R. (2020, October 21-22). Impact of network QoS factors on QoE of IoT services [Conference session]. 2020 - 5th International Conference on Information Technology (InCIT), Chonburi, Thailand. https://doi.org/10.1109/incit50588.2020.9310969

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence* (pp. 1137-1143).

Kreutz, D., Ramos, F. M. V., Verissimo, P. E., 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

Krishnan, Prabhakar & Duttagupta, Subhasri & Achuthan, Krishnashree. (2019). VARMAN: Multi-plane security framework for software defined networks. *Computer Communications*. 148. 10.1016/j.comcom.2019.09.014.

Kuhn, Max & Johnson, Kjell. (2013). Applied Predictive Modeling. 10.1007/978-1-4614-6849-3.

Kumar, J. & Rose P. J., A. L. (2023). Mitigate volumetric DDoS attack using machine learning algorithm in SDN based IoT network environment. *International Journal of Advanced Computer Science and Applications, 14*(1). https://doi.org/10.14569/ijacsa.2023.0140161

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444. https://doi.org/10.1038/nature14539

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R News, 2*(3), 18-22. https://cran.r-project.org/doc/Rnews/

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529-533

Mohammadi, R., Lal, C., & Conti, M. (2022). HTTPScout: A machine learning based countermeasure for HTTP flood attacks in SDN. *International Journal of Information Security, 22*(2), 367–379. https://doi.org/10.1007/s10207-022-00641-3

Netround. (2019). Netrounds' Survey uncovers how IT organizations reduce network brownouts. *PR Newswire: Press Release Distribution, Targeting, Monitoring and Marketing.* https://www.prnewswire.com/news-releases/netrounds-survey-uncovers-how-it-organizations-reduce-network-brownouts-300910508.html

ntop. (n.d.). nDPI - Open and Extensible Deep Packet Inspection Library. ntop. Retrieved February 26, 2025, from https://www.ntop.org/products/deep-packet-inspection/ndpi/

Osman, A., Alsaeedi, M., & Alzahrani, A. (2024). Machine learning-based security solutions for software-defined networking: A survey. *Computers & Security, 132*, 103478. https://doi.org/10.1016/j.cose.2023.103478

Osman, M. F., Mohd Isa, M. R., Khairuddin, M. A., Mohd Shukran, M. Afizi, & Mat Razali, N. A. (2024). A novel network optimization framework based on software-defined networking (SDN) and Deep Learning (DL) approach. *JOIV: International Journal on Informatics Visualization, 8*(4), 2082. https://doi.org/10.62527/joiv.8.4.2169

Prasanth, L. L., & Uma, E. (2024). A computationally intelligent framework for traffic engineering and congestion management in software-defined network (SDN). *EURASIP Journal on Wireless Communications and Networking, 2024(1)*. https://doi.org/10.1186/s13638-024-02392-2

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*(1), 81-106. https://doi.org/10.1007/BF00116251

Rajput, Z., Waqas, M., & Rana, M. (2020). A survey on QoS management using machine learning in SDN. *Journal of Communication and Networks, 22*(1), 72-89. https://doi.org/10.1109/JCN.2020.00001

Ring, M., Wunderlich, S., Scheuring, D., Landes, D., & Hotho, A. (2019). A survey of network-based intrusion detection data sets. *Computers & Security, 86*, 147–167. https://doi.org/10.1016/j.cose.2019.06.005

Rui, K., Pan, H., & Shu, S. (2023). Secure routing in the internet of things (IoT) with intrusion detection capability based on software-defined networking (SDN) and machine learning techniques. *Scientific Reports, 13*(1). https://doi.org/10.1038/s41598-023-44764-6

Salau, O., & Beyene, Y. (2024). AI-driven security in SDN: A proactive approach to network threat mitigation. *IEEE Access, 12*, 19562–19578. https://doi.org/10.1109/ACCESS.2024.3295762

Salau, A. O., & Beyene, M. M. (2024). Software defined networking based network traffic classification using Machine Learning Techniques. *Scientific Reports, 14*(1). https://doi.org/10.1038/s41598-024-70983-6

Salman, O., Elhajj, I. H., Chehab, A., & Kayssi, A. (2022). ML-based security and QoS optimization in SDN. *IEEE Transactions on Emerging Topics in Computing, 10*(3), 1564–1575. https://doi.org/10.1109/TETC.2022.3186974

Shafiq, M. Z., Yu, K., & Guo, S. (2021). Anomaly-based intrusion detection in SDN using machine learning. *Journal of Network and Computer Applications, 178*, 102944. https://doi.org/10.1016/j.jnca.2021.102944 Sharma, A. (2024, August 7). Cross validation in machine learning. *GeeksforGeeks.* https://www.geeksforgeeks.org/cross-validation-machine-learning/

Sharma, A., Gupta, B., & Soni, M. (2018). CIC-IDS 2018: A dataset for intrusion detection systems. *Communications and Network, 10*(2), 22-31. https://doi.org/10.4236/cn.2018.102002

Tariq, R., Aslam, M., & Raza, M. (2023). Multi-output classification models for traffic categorization in SDN. *IEEE Transactions on Network Science and Engineering, 11*(1), 72–84. https://doi.org/10.1109/TNSE.2023.3274887

Tu, Z., Zhou, H., Li, K., Li, G., & Shen, Q. (2019, December 9-13). A routing optimization method for software-defined SGIN based on deep reinforcement learning [Conference session]. *2019 IEEE Globecom Workshops (GC Wkshps)*, Waikoloa, HI, USA. https://doi.org/10.1109/gcwkshps45667.2019.9024680

Vaghani, A., Sood, K., & Yu, S. (2022). Security and QoS issues in blockchain enabled next-generation smart logistic networks: A tutorial. *Blockchain: Research and Applications, 3*(3), 100082. https://doi.org/10.1016/j.bcra.2022.100082

Wakili, A., Bakkali, S., & Alaoui A. E. H. (2024). Machine learning for QoS and security enhancement of RPL in IoT-enabled wireless sensors, *Sensors International, 5*, 100289, ISSN 2666-3511. https://doi.org/10.1016/j.sintl.2024.100289.

Wang, S., Guo, Y., & Yang, J. (2021). QoS-aware traffic management in SDN using reinforcement learning. *IEEE Transactions on Network and Service Management, 18*(4), 5143–5156. https://doi.org/10.1109/TNSM.2021.3109782

Wang, Y., Zhao, X., & Li, J. (2021). Traffic classification for QoS management using machine learning. *IEEE Access, 9*, 12945-12958. https://doi.org/10.1109/ACCESS.2021.3055512

Wassie, G., Ding, J., & Wondie, Y. (2023). Traffic prediction in SDN for explainable QoS using deep learning approach. *Scientific Reports, 13*(1). https://doi.org/10.1038/s41598-023-46471-8

Xu, H., Qiu, X., Sheng, Y., Luo, L., & Xiang, Y. (2018). A QoS-driven approach to the cloud service addressing attributes of security. *IEEE Access, 6*, 34477–34487. https://doi.org/10.1109/access.2018.2849594

Zhang, H., Li, Y., & Xu, Y. (2020). Security and Privacy Issues in Software-Defined Networking: A Survey. *IEEE Access, 8*, 170526–170547. https://doi.org/10.1109/ACCESS.2020.3024564

Zhang, J., Luo, J., & Zhang, Y. (2020). Adaptive anomaly detection in SDN using deep learning. *IEEE Transactions on Information Forensics and Security, 15*, 3453–3465. https://doi.org/10.1109/TIFS.2020.2998973

Zhao, J., Pang, L., Kuang, X., Jin, R. (2018). Balancing the QoS and security in Dijkstra algorithm by SDN technology [Conference paper]. In: Zhang, F., Zhai, J., Snir, M., Jin, H., Kasahara, H., Valero, M. (eds) Network and Parallel Computing. NPC 2018. Lecture Notes in Computer Science, Springer, Cham. https://doi.org/10.1007/978-3-030-05677-3_11

Zheng, W., Yang, M., Zhang, C., Zheng, Y., Wu, Y., Zhang, Y., & Li, J. (2022). Application-aware QoS routing in SDNs using machine learning techniques. *Peer-to-Peer Networking and Applications, 15*(1), 529–548. https://doi.org/10.1007/s12083-021-01262-8

Zhou, P., Wang, Z., & Chen, B. (2022). Feature selection techniques for intrusion detection in SDN: A review. *Computers & Security, 118*, 103247. https://doi.org/10.1016/j.cose.2022.103247

Appendix A

The code and dataset used in this praxis can be found in the GitHub Repository:

https://github.com/RubyanneYe/ResSDN