An SDN-Based Traffic Load Prediction using Machine Learning

by

Behdad BIBAK

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT OF A MASTER'S DEGREE
WITH THESIS IN INFORMATION TECHNOLOGY ENGINEERING
M.A.Sc.

MONTREAL, SEPTEMBER 29, 2023

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Michel Kadoch, Thesis supervisor
Department of Electrical Engineering, École de technologie supérieure

Mr. Georges Ghazi, President of the board of examiners
Department of Systems Engineering, École de technologie supérieure

Mr. Zbigniew Dziong, Member of the Jury
Department of Electrical Engineering, École de technologie supérieure

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON SEPTEMBER 19, 2023

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

Une prédiction de charge de trafic basée sur le SDN à l'aide de l'apprentissage automatique

Behdad BIBAK

RÉSUMÉ

Cette thèse est une tentative expérimentale de prédiction de la charge du réseau à l'aide de techniques d'apprentissage automatique dans un environnement SDN (Software Defined Networking).

Comparaison du contrôle de la congestion : SDN et réseaux traditionnels

Une disparité fondamentale entre le Software-Defined Networking (SDN) et les réseaux traditionnels réside dans leurs approches du contrôle de la congestion.

Les réseaux traditionnels adoptent souvent des stratégies de routage fixes, ce qui peut entraîner une utilisation sous-optimale des ressources pendant la congestion. En revanche, la séparation des plans de contrôle et de données du SDN permet une prédiction de congestion en temps réel et un reroutage intelligent. Cette adaptation dynamique optimise l'allocation des ressources, garantissant un service ininterrompu lors des pics de trafic.

La divergence marquée dans l'adaptabilité devient évidente lors de l'évaluation de la façon dont chacun réagit aux événements de congestion. Les réseaux traditionnels sont aux prises avec des réponses lentes en raison de leur dépendance à des routes prédéterminées. En revanche, la gestion de la congestion basée sur l'apprentissage automatique du SDN, telle que présentée dans cette recherche, atténue rapidement la congestion en redirigeant dynamiquement le trafic, améliorant ainsi les performances du réseau sous la contrainte.

Le réseau défini par logiciel (SDN) a révolutionné les environnements de réseau informatique en séparant les plans de contrôle et de données, permettant un contrôle centralisé via des contrôleurs définis par logiciel. Le SDN offre de nombreux avantages, notamment une meilleure configurabilité du réseau, une gestion simplifiée et une flexibilité dans l'allocation des ressources en fonction des demandes en temps réel. Cependant, des défis spécifiques, tels que la sécurité, la fiabilité et la congestion, affectent l'efficience et l'efficacité des systèmes SDN.

La congestion, en particulier, a un impact significatif sur la qualité des services SDN. Bien qu'il existe des programmes d'évitement de la congestion, des améliorations sont possibles pour fournir un service de haute qualité. Cette recherche propose un modèle d'évitement de congestion qui combine les réseaux de neurones profonds et l'algorithme Mijumbi (Mijumbi et al., 2014) pour résoudre le problème de congestion dans les SDN.

Le modèle proposé utilise un algorithme de réseau neuronal profond pour prédire la congestion sur la base d'une analyse approfondie de la communication de données au sein du réseau. Le modèle atténue efficacement la congestion et assure une transmission fluide en redirigeant

intelligemment les données via des itinéraires alternatifs. L'étape suivante du modèle utilise l'algorithme Mijumbi pour détourner les flux de congestion prévus des routes congestionnées, permettant ainsi une communication sans congestion.

L'intégration du modèle proposé dans les SDN se traduit par une prédiction précise de la congestion et une gestion efficace de la congestion. Le modèle détecte et redirige intelligemment la communication de données via des itinéraires alternatifs, en utilisant les ressources disponibles et en améliorant la qualité de service du réseau. Un ensemble de données en ligne, en particulier l'ensemble de données DDOS, évalue les performances du modèle proposé, en tenant compte de paramètres tels que les nœuds expéditeurs, les nœuds récepteurs, la taille des paquets, le temps de transmission et le protocole.

Le modèle proposé est simulé dans Python Jupyter Notebook, et les résultats sont comparés à un modèle de référence (QRTP) pour évaluer les performances. Le modèle proposé améliore considérablement la qualité de service et certaines métriques ; y compris les taux de perte de paquets, le débit, les effets de reconfiguration et l'utilisation des liaisons. Il fait également preuve de robustesse dans les situations de réseau dynamique, en maintenant son efficacité malgré la congestion, les changements de topologie du réseau et les altérations des routes de communication de données.

En conclusion, cette recherche présente un nouveau modèle d'évitement de congestion pour les SDN qui intègre des techniques d'apprentissage automatique et d'apprentissage en profondeur. Le modèle proposé résout de manière proactive les problèmes de congestion, optimise les solutions et améliore l'expérience utilisateur et les performances du réseau. Les résultats contribuent au développement d'un réseau de communication plus fiable et évolutif, offrant une solution complète aux problèmes de congestion du réseau dans les SDN.

Mots-clés: SDN, apprentissage automatique, prédiction de la charge de trafic, réseaux de neurones profonds, réseaux de neurones artificiels

An SDN-Based Traffic Load Prediction using Machine Learning

Behdad BIBAK

ABSTRACT

This thesis is an experimental attempt at predicting network load using machine learning techniques in an SDN (Software Defined Networking) environment.

Software-defined networking (SDN) has revolutionized computer network environments by separating the control and data planes, enabling centralized control through software-defined controllers. SDN offers numerous benefits, including enhanced network configurability, simplified management, and flexibility in resource allocation based on real-time demands. However, specific challenges, such as security, reliability, and congestion, affect the efficiency and effectiveness of SDN systems.

Comparison of Congestion Control: SDN vs. Traditional Networks

A fundamental disparity between Software-Defined Networking (SDN) and traditional networks lies in their approaches to congestion control.

Traditional networks often adopt fixed routing strategies, which can result in sub-optimal resource utilization during congestion. In contrast, SDN's separation of control and data planes enables real-time congestion prediction and intelligent rerouting. This dynamic adaptation optimizes resource allocation, ensuring uninterrupted service during traffic peaks.

The stark divergence in adaptability becomes evident when assessing how each responds to congestion events. Traditional networks grapple with sluggish responses due to their reliance on predetermined routes. In contrast, SDN's machine learning-driven congestion management, as showcased in this research, promptly mitigates congestion by dynamically rerouting traffic, enhancing network performance under duress.

Congestion, in particular, significantly impacts the quality of SDN services. Although existing congestion avoidance schemes exist, there is room for improvement to provide high-quality service. This research proposes a congestion avoidance model that combines deep neural networks and the Mijumbi (Mijumbi et al., 2014) algorithm to address the congestion issue in SDNs.

The proposed model utilizes a deep neural network algorithm to predict congestion based on an in-depth analysis of data communication within the network. The model effectively mitigates congestion and ensures smooth transmission by intelligently rerouting data through alternative routes. The subsequent stage of the model employs the Mijumbi algorithm to divert predicted congestion flows away from congested routes, thereby enabling congestion-free communication.

Integrating the proposed model in SDNs results in accurate congestion prediction and efficient congestion management. The model intelligently detects and redirects data communication through alternative routes, utilizing available resources and improving network quality of service. An online dataset, specifically the DDOS dataset, evaluates the proposed model's performance, considering parameters such as sender nodes, receiver nodes, packet size, transmission time, and protocol.

The proposed model is simulated in Python Jupyter Notebook, and the results are compared with a baseline model (QRTP) to assess performance. The proposed model significantly improves quality-of-service and some metrics; including packet loss rates, throughput, reconfiguration effects, and link utilization. It also demonstrates robustness in dynamic network situations, maintaining its effectiveness despite congestion, network topology changes, and data communication route alterations.

In conclusion, this research presents a novel congestion avoidance model for SDNs that integrates machine learning and deep learning techniques. The proposed model proactively addresses congestion problems, optimizes solutions, and enhances user experience and network performance. The findings contribute to developing a more reliable and scalable communication network, providing a comprehensive solution for network congestion challenges in SDNs.

Keywords: SDN, machine learning, traffic load prediction, deep neural networks, artificial neural networks, congestion, quality of service.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

AI	Artificial intelligence
ANN	Artificial Neural Networks
API	Application Programming Interface
CNN	Convolutional Neural Network
DDoS	Distributed Denial-of-Service
DL	Deep Learning
DNN	Deep Neural Network
ETS	École de Technologie Supérieure
IoT	Internet of Things
Mb	Megabits
MB	Megabytes
ML	Machine Learning
MLP	Multi-layer Perceptron
NBI	Northbound Interfaces
NeuTM	Neural Traffic Matrix
ONF	Open Networking Foundation
Opex	Operational Expenditure
QoS	Quality of Service
QRTP	Quality of service aware Resource allocation Traffic Prediction

RNN	Recurrent Neural Network
SDN	Software Defined Networking
TCP/IP	Transmission Control Protocol/Internet Protocol
WAN	Wide-area Network

INTRODUCTION

This section's objectives are to outline the driving forces behind the study on SDN-based traffic and to give a basic overview of the subject's significance for advancing SDN technology in the future.

The construction of an algorithm that predicts future network traffic using real-time monitoring and analysis of the current traffic is the main goal of this thesis. For use in a Software-Defined Networking environment, the algorithm is created as a stand-alone component. The prediction of network load was implemented utilizing machine learning and deep neural networks. The application was integrated and made to run on top of an SDN controller at the same time, and it is able to connect to the network through the controller's Northbound Interface by using an autonomous network management framework.

The rapid advancement of the Internet, the continuous expansion of computer networks, and the emergence of security concerns have raised questions about the ability of the conventional TCP/IP-based network to address various issues (Sezer et al., 2013). These issues include complexity, scalability, reliability, controllability, security, and quality of service (QoS). Additionally, traffic prediction in conventional networks is a critical area of research, as it plays a vital role in network planning and performance analysis (Kuang, Xu, Huang, & Liu, 2010). However, applying traffic prediction algorithms to the industry's traditional scattered networks poses challenges, as these networks are not easily managed by centralized solutions.

To meet the ever-growing demand for flexible network requirements, reduce network complexity, and accelerate network innovation, there is a need for a more effective and adaptable network deployment strategy (Nunes et al., 2014; Kreutz et al., 2015). As a result, the concept of Software Defined Networking (SDN) has emerged as a novel network architecture and is gaining widespread acceptance (Canini et al., 2015).

Thank you for providing the references. Here's the revised text with proper citations:

A widespread network paradigm, Software Defined Networking (SDN), offers efficient traffic management and ensures the desired Quality of Service (QoS) (A Recent Trends in Software Defined Networking (SDN) Security, 2016). Unlike traditional TCP/IP network designs, where a switch or router tightly couples the traffic control layer and data layer, SDN's foundational technology, OpenFlow, allows for the separation of the control plane from network devices, creating a separate centralized controller that reduces network complexity.

The controller, being a centralized entity, is responsible for managing the entire network. By collecting real-time data from network nodes and packets, the controller gains a comprehensive perspective of the network, enabling it to make routing decisions and dynamically program the network (A Recent Trends in Software Defined Networking (SDN) Security, 2016). The centralization, programmability, and ability to gather real-time data in SDN architecture make it suitable for applying machine learning techniques for efficient traffic prediction (Liu et al., 2019).

Machine learning (ML) is a subfield of artificial intelligence (AI) that focuses on algorithms and statistical models enabling computer systems to perform specific tasks without relying on explicit instructions. Instead, ML utilizes patterns and inference for decision-making ("Classification and Enrichment of Unlabeled Feedback Data Using Machine Learning," 2019). By integrating ML with SDN, particularly for traffic prediction, the network's performance can be optimized, leading to enhanced overall efficiency (Liu et al., 2019).

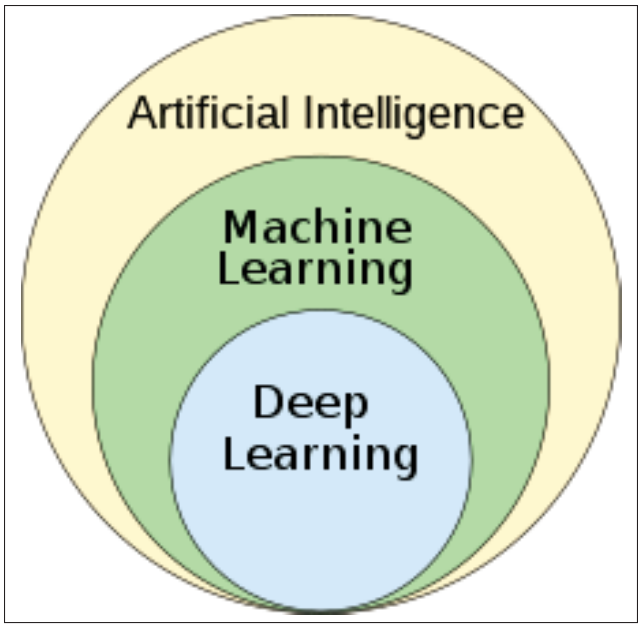


Figure 0.1 Machine learning as subfield of AI, taken from Wikipedia contributors (2023)

Traffic prediction plays a critical role in assessing and improving network performance, as well as in network design. By predicting traffic patterns in advance, it becomes possible to efficiently manage network resources, achieve load balancing, and avoid congestion, leading to enhanced overall network performance. Moreover, traffic prediction informs network planning and design, simplifying the process of network resource management.

However, modeling and analyzing data traffic within a Software-Defined Network (SDN) presents its own set of complexities and challenges. The nature of SDN network traffic is characterized by its fractal and paroxysmal patterns, which can be difficult to accurately capture and predict. As a result, addressing these intricacies and developing effective traffic prediction models for SDN networks is a demanding and crucial task in the field of network research and management.

Contribution

This study makes the following contributions:

- We go into great detail on the SDN traffic measuring procedure and network traffic prediction principles.
- We introduce an algorithm and employ it to create an optimum network load. Then, to acquire a traffic prediction in SDN, we improve this algorithm.
- Our approach achieves the ideal balance between accuracy and variety as two crucial goals, intending to reduce prediction error. This work's main emphasis is on that.
- We contrast our model's output with that of the employed algorithm.

Thesis Organization

The problem statement is presented, the general context is explained, and some of the thesis's fundamental concepts are defined in this introduction section.

The rest of this thesis is divided into the following chapters.

- Background information on SDN and reviews of its concepts are provided in chapter 1.
- The overall approach to addressing the many research questions of SDN, ML and reviews of the earlier work relevant to the issues raised by the research are discussed in chapter 2.
- The benefits and drawbacks of these works in this chapter are highlighted. Our method for predicting network traffic is described in Chapter 3. We provide a thorough explanation of our suggested solution.
- Our contribution, which is an SDN-based network load prediction performance, is presented in Chapter 4. The strategy, the execution, and the outcomes are described.
- We then go over the findings of the experiment and conclude. It also outlines future work that may be done to enhance our job.

CHAPTER 1

TECHNICAL BACKGROUND

In this chapter, we provide a general overview of SDN (Software-Defined Networking), Machine Learning, and the DNN (Deep Neural Network) algorithm that we employed in our study. These concepts are essential for understanding the research conducted in this thesis.

Following the introduction of the concepts, we clearly define the research objectives. The primary focus of this thesis is traffic load prediction in SDN networks. To achieve this goal, we outline the general methodology that we will follow throughout the study. This methodology will guide us in developing a traffic-aware load prediction model tailored to the unique characteristics of network traffic behavior in SDN.

By establishing the research objectives and presenting the general methodology, we aim to lay a strong foundation for our study and demonstrate the relevance of our work to the field of traffic load prediction in SDN.

1.1 Background and motivation

Due to the exponential growth in Internet users, the expansion of IP-based access networks and devices, and the advent of new applications like the Internet of Things (IoT) and cloud computing, Internet traffic has increased dramatically during the past ten years, according to the latest report by Cisco, Globally, there will be 26.3 billion networked devices in 2020, up from 16.3 billion in 2015 (Global - 2020 Forecast Highlights - Cisco, n.d.).

Recent trends in the technology sector, such as the widespread virtualization of servers, the advancement of enterprise data centers and cloud infrastructures, the proliferation of network access devices and applications, and the increasing need for parallel big data processing, have highlighted the necessity to reassess network design and operation. Traditional hierarchical physical switches with fixed configurations in network infrastructure have become inadequate to support the dynamic traffic demands and communication paradigms of today. In response to these

challenges, the concept of "software-defined networking" (SDN) has emerged, aiming to provide a network architecture that is highly adaptable, dynamically programmable, customizable, and cost-effective (ProQuest, n.d.).

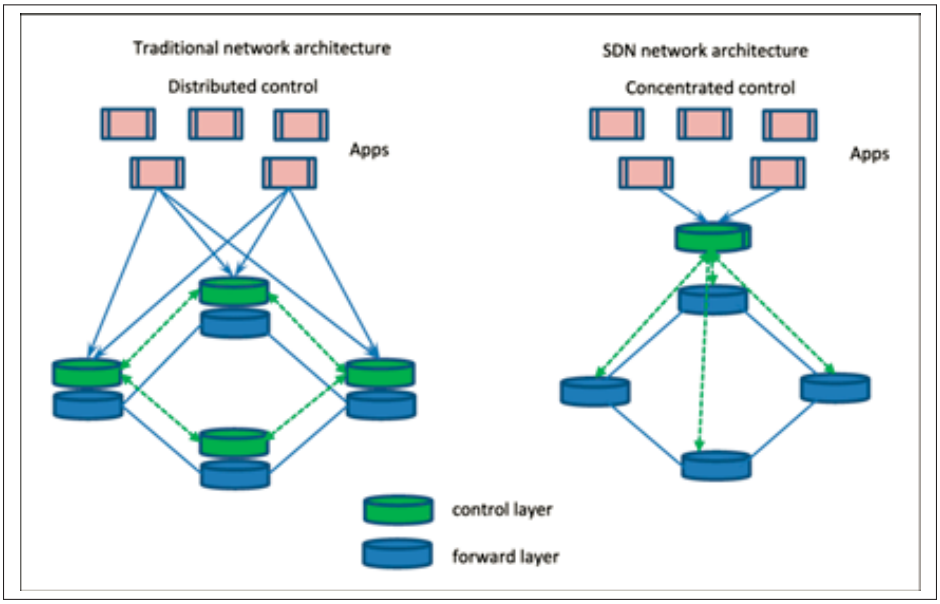


Figure 1.1 Networks with & without SDN taken from Liang & Li(2018)

Along with the significant development of software-based networks and the vast expansion and diversity of their applications, a standardization institute called ONF was established to manage and integrate these networks in collaboration with large companies such as Google and Cisco (“Software Defined Networking,” 2015). Software-based networks have evolved from an academic debate into a thriving industry experience, with Google’s data centers migrating to software-based networks. The results show that, for the company Google, this development has been very beneficial(Google’s internal network is built using SDN technology).

1.2 SDN architecture

The emergence of the software-defined network architecture is a response to the limitations of traditional network architectures in meeting the demands of modern complex networks.

The fundamental idea behind Software-Defined Networking (SDN) is to separate the Control Plane and the Forwarding (Data) Plane on a network switch, and then move the forwarding decisions to a central controller (S. Liu & Li, 2017). In a conventional network, a switch uses its Control Plane logic to determine how packets should be routed across its Data Plane to different destinations based on its local configuration. However, in an SDN, the switches still utilize their Data Plane to forward actual packets, while the methods and decisions governing packet switching are managed externally by a centralized software-based element known as the SDN Controller.

The use of a vendor-neutral, software-based SDN Controller empowers network managers to create and implement their algorithms for regulating data flows and packet forwarding within the network. This provides a high degree of flexibility and dynamic optimization to the network's intelligence. By abstracting the low-level packet switching from network applications and services, SDN enables the network to quickly adapt to changes and unique conditions, catering to its specific requirements efficiently. The centralized control and programmability of SDN allow network administrators to respond rapidly to evolving network demands and provide efficient network management and optimization.

In addition to the SDN Controller, the SDN Applications and SDN Data paths are other fundamental architectural elements of a Software-Defined Network. Several researchers have proposed various SDN architectures based on SDN principles, incorporating the path calculation element as a critical component (Mayoral, Vilalta, Muñoz, Casellas, & Martínez, 2017).

SDN Applications are software entities responsible for executing various network functions and operations. They interact with the SDN Controller through Northbound Interfaces (NBI), which typically provide open and vendor-independent abstract views of the network. The SDN Controller then communicates the logical packet switching mechanisms, residing in physical network devices, to the SDN Data paths. These Data paths consist of forwarding engines and traffic processing capabilities that execute the network actions required by the SDN Applications. Figure 1-2 provides a broad illustration of the SDN architecture, encompassing these components.

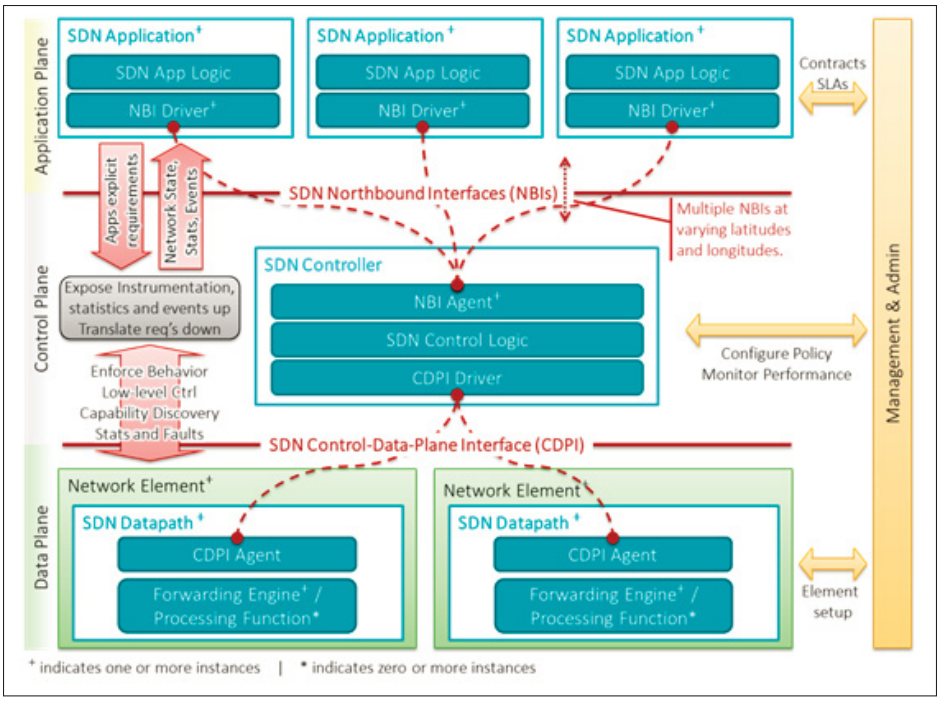


Figure 1.2 The SDN architecture in brief taken from ONF(2013)

1.3 SDN benefits and challenges

Today, many businesses are making use of SDN technology's advantages. Instead of manually controlling networking through hardware like switches and routers, software-defined networking (SDN) enables the management of networking solutions at the software level. Some of SDN's main benefits will be examined below.

1.3.1 Virtualization technology

In virtualization, an environment is provided for using and achieving the reality designed for simplifying in a virtualized way. Virtualization is a technology that does not seem really due to its specific function and logic in its design, but it has a function.

1.3.2 Centralization

The capacity to administer a network from a centralized perspective is one of the key benefits provided by SDN. Said SDN virtualizes the network and data control planes, enabling users to provide physical and virtual components from a single location. This is helpful because managing traditional infrastructure can be difficult, mainly if many different systems must be controlled separately. An administrator can drill down and up at will, thanks to SDN, which removes this barrier (Keary, 2020).

1.3.3 Dynamic resource sharing

The ability to change the size and volume of the network is one of the advantages that SDN offers us. Depending on the requirements and processing load and the traffic generated, some nodes can be added to the network or reduced from the network. One of the advantages of this network model is the possibility of sharing resources without being limited to physical location due to the dynamics of such networks. Resources in this network can be efficiently allocated from one part to another.

1.3.4 Optimization

Optimization is another advantage of SDN networks. Due to the virtualization and dynamism of such networks, all nodes can be used optimally with maximum efficiency. In this case, if there is a load, there will be no more imbalance in the network, and each node will work efficiently.

1.3.5 Power consumption management

In SDN, due to the integration of control systems and monitoring of the entire network and its integration, it is possible to remove or insert physical nodes from the circuit based on the requirements of the network. This dynamism creates the ability to remove unnecessary elements from the circuit when the load is reduced, and some nodes are not needed, thus

not only preventing the nodes from being depreciated when unnecessary and reducing power. Furthermore, consumption is also raised.

In large networks, power consumption is a factor in increasing the price of services, which plays a significant role in competitive debates. If we have a local network, this decrease and increase in dynamic consumption may be insignificant. However, this consumption costs money in an extensive network comprising thousands of servers, switches, and routers and composed of several data centers and infrastructure. In addition to rising power consumption, there are other issues, such as rising equipment temperatures due to power consumption and an increase in the number of nodes that require cooling systems that require power consumption, depreciation, and cost. It is, therefore, essential to reduce power consumption by removing extra and unnecessary nodes.

1.3.6 Security

Although the trend towards virtualization has made it more challenging for network administrators to protect their networks from outside attacks, it has also brought a significant benefit. An SDN controller gives the administrator a centralized location to manage the network's complete security. Users are given a clear view of their infrastructure through which they can manage the security of their whole network, even though doing so comes at the cost of making the SDN controller a target (Keary, 2020).

1.3.7 Simplification

By reducing complexity by implementing the network in two layers of control and data planes, network complexity is easily eliminated because the operator no longer needs to be aware of the content and complexity. The controller layer will do this. From the user's point of view, a layer of application software is given the necessary instructions to configure it. The control layer takes this command or command and communicates with the physical and hardware layer through communication protocols such as Open flow and compiles and executes the command

based on the type of element. The simplification in this method eliminates the need for the user to know the complexities of each element and each brand.

1.3.8 Automation

Mechanization and network automation are other benefits of SDN networks. By layering the network in the controller layer, we control the whole network and its physics. Using the automation software available for the network makes it possible to move from the separation of the elements to the mechanized state and execute commands through the software.

The following benefits can be achieved in automation with SDN:

- Troubleshooting
- Reducing the service time
- Policy implementation

1.3.9 Adding new load

In SDN, regardless of the complexity and the existing network load, if there is a new load, this load can be entered into the network. The control layer is responsible for where this load should be applied, which part of the network has more load, and which part has less.

Another advantage of SDN is the increase in productivity, with the maximum load being achieved by balancing the load between the elements and not idling the nodes. This section can be divided into the following categories.

1.3.9.1 Traffic load

Bandwidth management can be quickly done by integrating the whole network and virtualizing it. For example, if the link between two routers is saturated, the control layer can easily detect this and direct traffic from other routes.

1.3.9.2 Load balancing

Load distribution among all elements and links in the network is one of the concerns that network administrators have always been concerned with. How to find out if part of the network is idle and part of the network is saturated with load is a category that needs to be continuously evaluated and monitored. Now, if each element acts separately, this monitoring becomes more complicated. In addition, the separate function makes it impossible to divide the load between the networks - each element behaves separately - but in the virtual network model, the other elements all serve a single network. This network can be divided into different parts based on the traffic load. The network transferred the new load to the free part.

1.3.9.3 Monitoring

Typically, all network tools are constantly monitored. There is nothing new so far. However, the new problem that arises in SDN is that, in addition to monitoring each element, there is a need to monitor the relationships created by integration and virtualization because, in these relationships, there are relative facilities and limitations that must be met in advance. Design and apply the following configurations. Monitoring in such dimensions requires special equipment and software incomparable to the time when each operates individually.

1.3.10 Reduce costs

It is a certified fact that SDN reduces costs. This reduction takes place in different dimensions. The first and most crucial reduction will be in the Opex layer. With the development of the network, whether in the dimensions of the Internet, intranets or even local area networks, its complexity and volume have increased exponentially. This complexity arose at the software and hardware levels, and each caused many problems for network managers and administrators. As these complexities increased, more and more professional operators were needed, which in turn increased Opex. Nevertheless, many of these tasks will be delegated to the control layer with virtualization and SDN networking.

SDNs reduce costs through the efficient use of resources and the lack of need to add resources in the event of a load or new changes. Each element is considered a part of the whole set in this model. If needed in one part of the network, it efficiently uses other elements in other parts of the network. In contrast, when the network is not implemented as SDN, we cannot share the elements when needed, and we have to add new tools and resources to the network, increasing costs.

1.3.11 Challenges

SDN also faces specific difficulties. The single point of failure is a significant obstacle. Because SDN employs the centralization idea, the entire network crashes when the controller—the network's brain—goes down. A single point of failure is another significant security vulnerability. Attackers can concentrate primarily on bringing down the controller to bring down the entire network. The issue of scalability is another one. One controller cannot adequately manage the network tasks as the network gets big. The network grows as more controllers are added. The problem of where to put the controllers gets increasingly difficult as more controllers are added. The controller placement problem is the common name for this. The controller placement issue has been addressed using a variety of strategies (Lange et al., 2015; Hu et al., 2014). Additionally, much study has been done recently on security in SDN networks.

1.4 Machine learning

The study of algorithms and statistical models that computer systems employ to carry out a particular task without utilizing explicit instructions, depending instead on patterns and inference, is known as machine learning (ML) (Redmond, 2021). Recent research has begun to include machine learning techniques into SDN to increase the effectiveness of network management and conformance or to solve problems that are difficult to handle using conventional techniques due to the rapid development of machine learning technology. There are generally four categories for machine learning, Supervised Machine Learning, Unsupervised Machine Learning, Semi-Supervised Machine Learning, and Reinforcement learning.

1.5 Artificial Neural Networks

Artificial Neural Networks (ANN), an algorithm, belongs to the supervised machine learning category. Artificial neural networks are information processing systems with the capacity to model linear functions in parallel, learn from data and information received, and have the ability to tolerate ambiguity (fault tolerance). Artificial neural networks have various applications, including forecasting, data processing, and pattern recognition (Chen-Xiao & Yabin, 2016). Since the primary function of load balancing in SDN networks is to determine and change weights, which means the ability to process data - input data without the need for an optimization target, artificial neural networks (ANNs) have been chosen as the method for load balancing. ANNs can classify and select input data into defined categories.

1.6 Objectives and general methodology

This thesis is centered around four specific objectives: predicting traffic load, avoiding network reconfigurations during repeated network optimization, traffic modeling, and congestion reduction.

The primary objective is to develop a traffic-aware load prediction model, which requires a comprehensive understanding of traffic behavior and characteristics. The research focuses on traffic characterization to create a custom forecast algorithm tailored to network traffic behavior. This entails a thorough investigation of traffic characteristics to enhance the precision of the prediction algorithm. As traffic time series exhibit diverse patterns at different times, links, and time scales, their behavior is dynamic and constantly evolving. Therefore, understanding and comprehending traffic characteristics are essential for developing an accurate traffic model.

To achieve this, a traffic model based on a machine-learning algorithm that can effectively handle traffic characteristics is proposed. The research leverages the Mijumbi algorithm (Mijumbi et al., 2014) as the foundation for the network traffic predictor, considering traffic characteristics to improve the accuracy of traffic load predictions.

1.7 Summary

Traffic prediction is incredibly significant in providing high-quality compunction in a network. Utilizing intelligent algorithms is one of the essential aspects that help make the best solution for the more complex issues in the network. Moreover, forecasting possible issues can maintain and improve the QoS. The DNN algorithm can be applied in the software-defined network prediction using historical network data for better decision-making and accuracy.

CHAPTER 2

RELATED WORKS

This study employs SDN and machine learning to classify and predict network traffic. In this chapter, we will provide the reason for selecting SDN and the importance of Machine Learning in SDN and traffic prediction. In addition, we present a review of previous research on traffic prediction in software-defined networks.

2.1 Importance of traffic prediction

Traffic prediction is essential for ensuring high-quality communication in networking. Different issues may exist in the network, so it is impotent to make the network more efficient. These issues need to be resolved. For example, the congestion issue arrives in the data communication; in that case, the reduction of the helpful traffic to avoid congestion and attain better results. Traffic prediction is essential by using historical data to predict the network help to avoid problems in the network. The neural network can perform prediction analysis since its historical data proves helpful in making better decisions.

As per the supposed problem, the traffic prediction will enable us to determine the possible congestion on the link before they decrease the quality of services. The primary aim is to route the traffic on the less congested routes, which will help avoid congestion in the network. The Neural Network model will make the process intelligent. Hence it is a significant problem of the network for route optimization.

2.2 Related studies

(Bouzidi et al., 2021) proposed an SDN-based rules placement model to dynamically predict traffic congestion through neural networks and optimal routes to reroute traffic performance. This model helps improve network utilization by developing a deep network agent. Initially, the quality-of-service aware routing problems are linear to reduce the end-to-end delay and

link utilization. After that, a simple heuristic algorithm was used to solve this problem. The significant contribution of this study is the rule placement in SDN based on real-time statistics measurements and traffic prediction.

(Shu et al., 2016) conducted research on traffic engineering technology based on Software-Defined Networks (SDN). They introduced an initial framework for traffic engineering in SDN, comprising two main components: traffic measurements and traffic management. The traffic measurement aspect is responsible for real-time monitoring and analysis of network traffic, taking into account traffic requirements for effective traffic management. This comprehensive traffic framework includes various parameters, such as the general framework, analysis prediction, load balancing, Quality of Service (QoS) considerations, guarantee scheduling, energy-saving techniques, and traffic management for hybrid IP/SDN environments. A significant contribution of their research lies in proposing a reference framework for traffic engineering, which encompasses both traffic measurements and traffic management components.

(Tajiki et al., 2017) defined development relates to the judgment of the flow matrix that enables the scheme to minimize the total packet loss and enhance the network's throughput. The study uses a mathematical formulation of the QoS-aware resources reallocation in SDN based on traffic prediction. Two schemes were proposed to solve the issue. One is the exact solution, and the other is fast optimal. The proposed schemes are compared with the accuracy perspective.

Furthermore, the impact of prediction on resource reallocation is discussed. In this regard, it is shown that the packet loss rate decreased, and network throughput increased significantly. The significant contribution of the study is a mathematical formula for the quality of service-aware resource allocation in SDN based on traffic prediction.

(Mijumbi et al., 2014) took advantage of software-defined network control plane capability to perform resource management more efficiently and effectively in the virtual network by dynamically adjusting the virtual networks by subtracting the network mapping through network status. In the study, the SDN controller is extended to monitor the resource utilization of the virtual networks, increase the average loading of the nodes and switching, and use the information

to proactively add or remove the rules for flow from the switches. The main contribution is that they proposed a dynamic resource management technique that is aware of the link and switches resources in SDN based virtual network model.

(Azzouni & Pujolle, 2017) They presented NeuTM mode based on the network traffic matrix and prediction agenda based on the long and short-term memory decrement neural network. This prediction model is defined as the problem of estimating future network traffic matrix from the previous and achieved network traffic data. It is extensively used in planning the network and resources management and security. Short-term memory is a specific recurrent neural network architecture helpful in learning data and classifying and predicting time series with time lags of unknown sizes. This model shows some long-range dependencies more accurately than the conventional RNN model.

(Parsaei et al., 2017) They defined an intelligent algorithm-based scheme that uses the genetic and ant colony algorithms to optimize the flow rules placement. It is a heuristic traffic engineering approach. This study explained that these approaches are limited according to the situation. This model must cover the more complicated routing issues, which provides the quality of services in the SDN.

(Yoonseon et al., 2015) proposed a scheme to minimize the network's energy spending and congestion. For that purpose, this model computes the optimal topology that can accommodate the expected traffic demands and then load balancing by distributing the flows across the k-shortest routes of the optimal topology by resolving the integer linear program issue. The significant contribution of this study is the energy efficiency through optimal computing topology used for load balancing.

These techniques deal with route optimization through the routing algorithm to discover efficient routing policies in the dynamically changing network without having the information in advance of the network topology and traffic patterns. The algorithm stores the values in packets that pass through the network. The utilized algorithm helps obtain low latency and high throughput and

adaptive routing. Also, more is needed regarding the storage table space and time to reach the best policy (Khodayari & Yazdanpanah, 2005).

In (Yan et al., 2015) the proposed model provides a guaranteed solution for the quality of services by identifying the multiple paths between communication devices by queue mechanisms to guarantee the quality of service for multiple types of traffic. The presented approach efficiently finds the fallouts with the minimum delay and high throughput. Its primary approach is to provide high-quality service. It can recover the link failure and reroute the traffic from the failed paths to the other available paths.

(Wang et al., 2019) introduced a lightweight traffic prediction algorithm specifically designed for software-defined network (SDN) applications. This algorithm distinguishes itself from traditional network measurements by adopting a flow-based forwarding concept within SDNs, which leverages the network states to shape the data plane. By using static variables to describe the network flow information forwarded in SDN, more accurate measurements of flow traffic are achieved with minimal overhead.

Additionally, the algorithm takes into account the temporal nature of traffic by incorporating time correlation to model flow traffic. Time series analysis, employing a regressive modeling approach, is used to characterize the network traffic patterns. Subsequently, the traffic forecasting in SDN relies on the flow-based forwarding traffic prediction algorithms developed as part of this research.

Networking is continuously developing with the massive expansion of network traffic. With time software-defined networks approach was proposed that provides the central control mechanism for network traffic measurements control and prediction. Still, the data received by the controller in SDN is excellent in volume; the vast amount of the data is needed to be processed more efficiently; for that purpose, the field of artificial intelligence machine learning is suggested. This research provides a detailed analysis of the machine-learning approaches for traffic prediction. More specifically, deep learning for traffic prediction is also discussed. More importantly, the

issues and challenges in the software-defined networks are highlighted with the future direction for research in the SDNs (Mohammed et al., 2019).

(Tang et al., 2018) introduced a novel deep-learning algorithm aimed at forecasting future traffic load and congestion within the network. Initially, the training process involved employing a basic deep belief architecture and deep Convolutional Neural Network (CNN). Subsequently, a deep-learning-based prediction algorithm was integrated with a DL-based channel assignment algorithm to intelligently route network traffic. This combined approach enables the algorithm to accurately predict prospective traffic load and efficiently avoid congestion by selecting an appropriate channel.

(Baz, 2018) centered their research on utilizing the Bayesian Machine Learning (ML) algorithm to empower switches in Software-Defined Networking (SDN). The objective was to enable these switches to identify the controller responsible for creating flow rules and anticipate unmatched packets in the flow table. They proposed an algorithm that allows SDN switches to efficiently determine the appropriate controller for flow rule creation and anticipate any unmatched packets in the flow table.

(Mestres et al., 2018) conducted a study on using neural networks to model delays in networks. They performed training using diverse neural network architectures under various scenarios, considering critical network components such as topology, network size, traffic intensity, and routing. The aim was to derive guidelines for effectively training such neural networks to accurately model network delays.

2.3 Summary

This chapter delves into the topic of traffic prediction in a software-defined network, providing a comprehensive discussion on the reasons behind selecting this particular subject. The significance of this research area is thoroughly addressed. Furthermore, an extensive review of related works is presented, focusing on traffic prediction methodologies. The proposed methodology is compared and contrasted with existing approaches, highlighting the unique

contributions of each study. To aid in understanding the distinctions, a comparison table is included, summarizing key aspects of the reviewed works.

CHAPTER 3

METHODOLOGY

This chapter presents a detailed discussion of the proposed methodology for the thesis, which encompasses the following steps. Initially, the phase diagram of the proposed methodology will be illustrated in Figure 3-1. Subsequently, an overview of existing studies related to the research area in the thesis will be reviewed. From the literature review, the problem statement will be derived and thoroughly discussed.

The proposed methodology employs a Machine Learning/Deep Learning (ML/DL) approach to address the identified problem. The application of ML/DL techniques in the thesis will be described to tackle the research objectives effectively.

Finally, the algorithm of the proposed methodology will be presented, and a summary of the chapter will be provided.

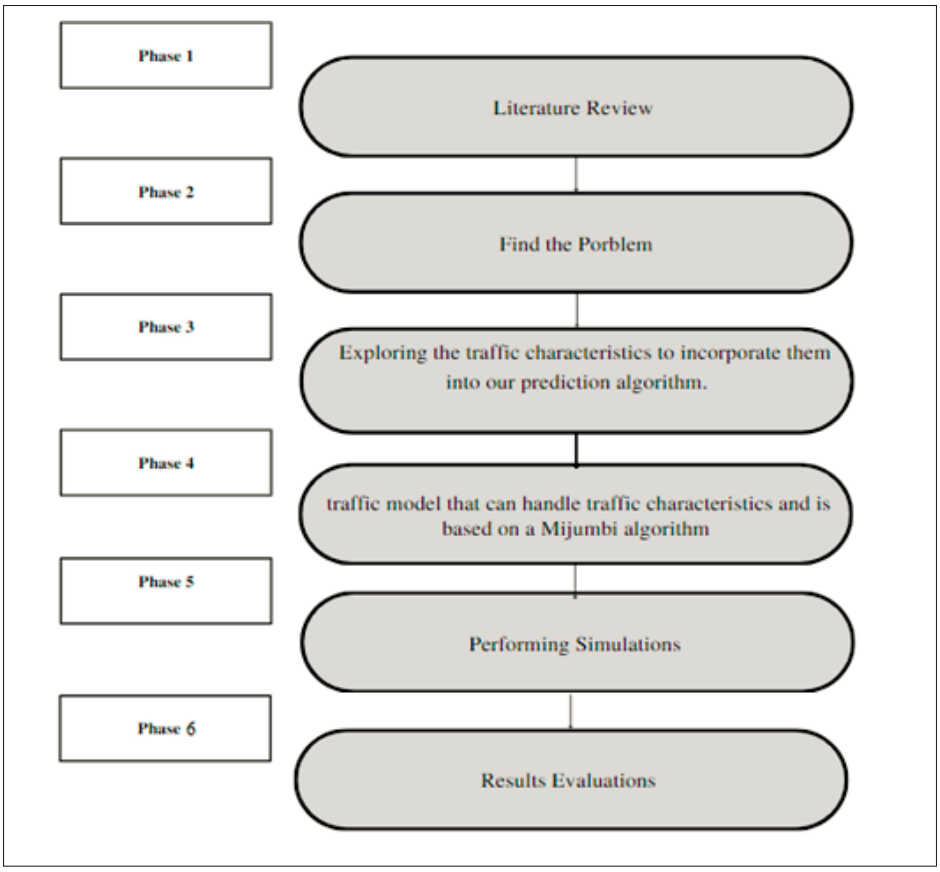


Figure 3.1 Proposed Methodology Phase Diagram

3.1 Contributions extracted from other studies

The software-defined network model is one of the emerging technologies in the field of networking. However, SDN provides a centralized control mechanism for traffic prediction measurements, still a large amount of the data is received at the controller. For data processing, ML approaches are suggested. The literature review reviews some relevant software-defined network prediction modes using the ML/DL algorithms. Recent studies have begun to include ML techniques in SDN to enhance network management effectiveness and conformance or resolve complex problems. We have found several studies in the literature that use the ML/DL algorithm in SDN to solve complex problems and get efficient results. Here we have to discuss a few contributions of the most essential reviewed prediction model based on intelligent algorithmic

SDN. In (Bouzidi et al., 2021), a rule placement in SDN is based on the real-time statistics measurements and traffic prediction performed. In (Shu et al., 2016), a reference framework for traffic engineering is performed in two parts: traffic measurements and management. In (Tajiki et al., 2017), a mathematical formula for the Quality of service-aware resource allocation in SDN based on traffic prediction is designed. In (Mijumbi et al., 2014) uses a dynamic resource management approach that is aware of the link and switches resources in SDN based virtual network model. In (Azzouni & Pujolle, 2017) network traffic matrix is a program based on the long and short-term memory decrement using a neural network. Another study found that intelligent content awareness facilitates traffic engineering in multiple aspects like network cashing, that help removes redundant traffic and content awareness. In (Parsaei et al., 2017) optimize the flow rules placement using the heuristic traffic. In (Yoonseon et al., 2015) provides the granted solution for the Quality of service by providing the different parts for communication devices by queue mechanism. In (Mohammed et al., 2019), a detailed analysis is performed for the machine learning approaches for network prediction and reviewed the different ML/DL approaches used in network prediction, and in the study conducted by Tang et al. (2018), they introduced a deep learning algorithm aimed at predicting potential traffic load and congestion within the network. On the other hand, Baz (2018) focused on employing a Bayesian machine learning algorithm to empower switches in SDN. The goal was to enable these switches to identify the controller responsible for creating flow rules and to anticipate unmatched packets in the flow table. In the research by Mestres et al. (2018), they analyzed the use of neural networks to model network delays. They trained various neural network architectures under different scenarios involving topology, network size, traffic intensity, and routing to provide guidelines for effectively training such neural networks.

3.2 The Problem - Traffic load issue

The increasing utilization of internet users and internet protocol-based connected devices on the internet; encourages the development of cloud computing and internet of things types technologies which is also the reason behind this more significant spreading out in internet-

connected devices. It needs time for emerging technological advancements such as cloud infrastructures, virtualization of servers, variability of network devices and applications, big data processing, etc. The old switches and hardcoded configuration must effectively meet the latest trends and technologies, a big problem that must be resolved. The software-defined network architecture concept brings adaptability, affordability, customizations, and dynamic programmability, which traditional network models miss. The SDN model justifies its efficiency even in collaborating with Google and Cisco. SDN models are utilized practically in industries. Even now, Google data centers comply migrating to software-based networks with and gaining positive results ("Google's Largest Internal Network Interconnects Its Data Centers Using Software Defined Network (SDN) in the WAN – Technology Blog," 2012).

SDN architecture is designed to resolve the complications of traditional networks, having the idea of forwarding the decision of the central controller toward the control and data plane by separating them on the network switch. A conventional network delivers a data plane toward several receivers with its logic. At the same time, an SDN switch uses a data plane to forward the actual packet, but a program governs its packet switch. That is why programmers can handle the data flow and packet forwarding using their algorithms. Network intelligence makes the network more flexible and dynamically optimized. To minimize the complexity, the SDN Open flow protocol enables the control and data flow separately for the centralized network control. After the data collection, network control completes its viewpoint about the whole network.

However, the SDN faces some issues that we have considered resolving, which is the research question for this thesis. Long-term consistency in traffic needs is not guaranteed. Secondly, we must develop an accurate mathematical model that helps capture the network characteristic very well. This is because cloud service users aim to utilize the resources with time and cost efficiency that can achieve through better network prediction. The problem considered for the research is Network prediction, which is one of the central issues in conventional networks and must be used for network assessment and design. The software defines a network as one of the most efficient network models that can resolve all the problems in conventional networks more effectively. SDN network is based on an Open flow protocol that separates the network's control

and data plane to minimize the network's complexity. The central control of the network helps to manage the network more effectively. However, SDN needs to provide the solution through program-based control and still has research gaps to be improvable.

3.3 Proposed Methodology

As reviewed, the ML/DL fields of Artificial intelligence are an advanced tool in providing potential network solutions to make them self-aware, adoptive, secure, and more intelligent. Its approaches enable the network program to learn from its experiences and empower them strongly against all vulnerable activities, such as the security reasons of networks and complexity and congestion in data communication more effectively (Gebremariam et al., 2019). To provide a better solution to the identified problem in the literature, we use the DNN algorithm to obtain efficient results compared to the other simple algorithms. The intelligent algorithm helps to optimize a control network through accurate data analysis and continuously optimizes algorithms to improve network resource utilization (Guo et al., 2018).

In order to solve the problem, proposed intelligent algorithm for the SDN program that enables it to make the decision intelligently for better network perdition to avoid network congestion and ensure the Quality of the services. We use the algorithm to make the network communication flow for a long time. We will make a mathematical model to characterize the network more efficiently for better results. The efficient prediction of the SDN will help the network resource utilization in real time.

The fundamental principles of network traffic in software-defined networks are derived from the basic rules governing network traffic. Network prediction plays a crucial role in anticipating network trends and proactively balancing the load for better control. By making timely adjustments, network congestion can be avoided, leading to load balancing, which significantly impacts network quality and performance improvement.

In the thesis, we are going to solve the SDN problems through the utilization of the artificial intelligence algorithm DNN to predict traffic that avoids the problems more efficiently using the

historical data, remove the actual problems, and make the network flow better than compared approaches (Azzouni & Pujolle, 2017). The adaptive mechanism of intelligent algorithms facilitates learning from their own experiences. The proposed algorithm is the multilayer and multi-node learning algorithm from the historical data and predictions.

AI algorithms are efficient and utilizable tools to make the network self-aware, secured, managed, and governed more effectively. These approaches make the network learn from the experiences that will strengthen the network against the vulnerable activities in the networks, making the network more beneficial to work self-governing and solving the issues without human support. Most artificial intelligence-based networking applications are resource utilization, traffic load balancing, network planning, fault failure management, fault detection, management and operation of the networks, security, breaches detection, etc. (Anteneh A. Gebremariam et al., 2019).

Software-defined networks and Open flow architecture are incredibly impactful in providing an efficient solution for controlling and managing the network by separating the control and data layers (Allan Schmidt., 2019). That is why AI approaches are applicable in the SDN models for these several reasons, such as the advancements in computer technologies, data-driven technologies, the utilization of real-time and historical data to make the network more intelligent, real-time working, and the programming capability that provides the optimal solution for the network (Junfeng Xi et al., 2018).

The AI algorithm is more beneficial for performing the network prediction using the SDN controller's collected real-time and historical data. This will dynamically adjust the network traffic link states, such as bandwidth, cost, and reliability. This will be realized the network traffic is intelligently controllable and optimize the network, helping to accurately and timely schedule the path for the traffic; such things enable the network to save itself from the vulnerabilities such as congestion, security and complexity, Quality of services (Aipeng Guo et al., 2021).

3.3.1 Proposed Methodology Flow Chart

The main objective of network prediction is to anticipate potential congestion in the network by leveraging historical and real-time data. By accurately predicting network conditions, congestion can be avoided, leading to enhanced network performance and the provision of Quality of Service (QoS) for communication. The predictability of network congestion is crucial for ensuring efficient network management. Based on the prediction results and examination of traffic data, the SDN controller can intelligently direct flows to less congested links, optimizing the overall network performance and enhancing the user experience.

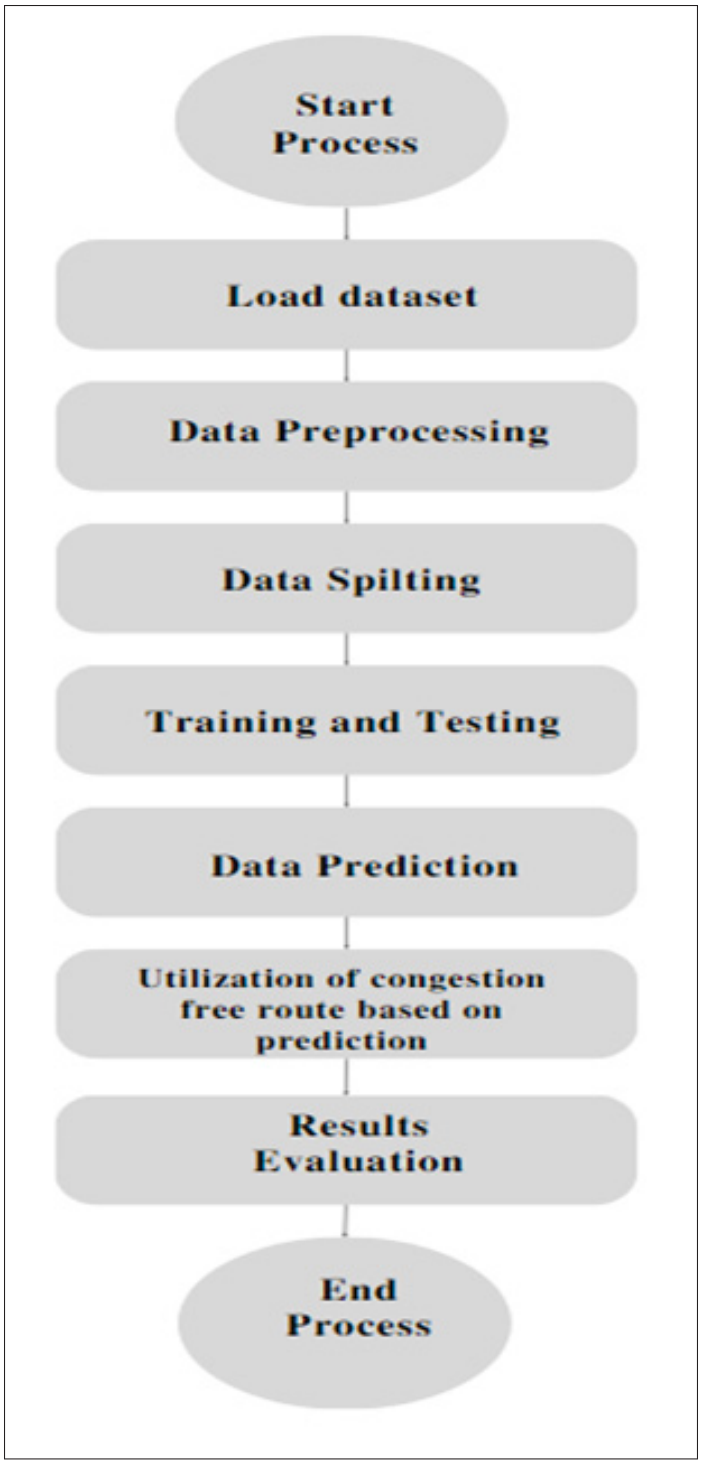


Figure 3.2 Proposed model's flow chart

The proposed model's flow chart illustrates the thesis's mechanism in steps initiated for the problem finding in the literature review and the proposed methodology of solving that problem through the artificial intelligent approach. The complete procedure of the proposed methodology will be explained in detail. Here we initiate the data collection for the intelligent algorithm dataset, which is the primary and essential part.

3.4 Dataset

In the context of evaluating congestion avoidance mechanisms, a novel model was examined, incorporating an intelligent algorithm inspired by Mijumbi and a deep neural network algorithm. This investigation utilized the 'DDOS attack SDN Dataset' sourced from the internet (Ahuja et al., 2020). Before implementing the model, relevant features pertaining to the proposed approach were removed during the initial data preprocessing stage. However, the remaining dataset was effectively employed to train and test the deep neural network's ability to predict congestion. Here is a figure of the utilized dataset for the proposed model shown below.

Switch	rc	dt	pktcount	bytecount	dur	tot_dur	flow	packetin	pktperflow	byteperflo	pktrate	Pairflow	Protocol	port_no	tx_byte	rx_byte	congestion
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	3	1.44E+08	3917	0
1	1	8	126395	1.35E+08	280	2.81E+11	2	1943	13531	14424046	451	0	2	4	3842	3520	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	1	3795	1242	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	2	3688	1492	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	3	3413	3665	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	1	3795	1402	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	4	3665	3413	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	1	3775	1492	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	2	3845	1402	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	4	3.55E+08	4295	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	1	3775	1242	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	2	3413	3665	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	1	4047	1.44E+08	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	4	3665	3413	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	2	5.81E+08	2586	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	4	3665	3413	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	2	3795	1402	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	2	3795	1492	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	2	4047	1.93E+08	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	1	3879	48410560	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	2	4055	96412666	0
1	1	8	45304	48294064	100	1.01E+11	3	1943	13535	14428310	451	0	2	3	3413	3665	0
1	2	8	90333	96294978	200	2.01E+11	3	1943	13534	14427244	451	0	2	1	3570	1492	0
1	2	8	103866	1.11E+08	230	2.31E+11	3	1943	13533	14426178	451	0	2	1	3775	1242	0
1	1	8	126395	1.35E+08	280	2.81E+11	2	1943	13531	14424046	451	0	2	3	4766	3.53E+08	0
1	1	8	85676	91330616	190	1.91E+11	3	1943	13306	14184196	443	0	2	2	3795	1492	0
1	1	8	85676	91330616	190	1.91E+11	3	1943	13306	14184196	443	0	2	4	4.99E+08	4715	0
1	1	8	85676	91330616	190	1.91E+11	3	1943	13306	14184196	443	0	2	3	2.31E+08	4169	0

Figure 3.3 Data set utilized to test the performance of the proposed model,extracted from Ahuja et al., (2020)

3.5 Deep Neural Network

The Deep Neural Network (DNN) represents a highly effective algorithm for capturing and understanding complex relationships between inputs and outputs, especially in non-linear systems. By rapidly minimizing the error between predicted and actual outputs, the DNN can efficiently learn from training data, enabling accurate forecasts. Given its adaptability, DNN stands as a versatile algorithm in prediction scenarios. In the context of SDN, this study leverages the DNN algorithm to predict network behavior, achieve network balance, prevent congestion, and facilitate network reconfigurations. The proposed algorithm employs the SDN dataset, specifically utilizing eight keys (indicated in table 3.1) and relevant features to forecast congestion accurately.

3.6 Topology of the proposed DNN

The deep neural network (DNN) architecture comprises multiple interconnected layers of neurons, including the input, hidden, and output layers. Each neuron in the network is connected to others through connection links, with each link associated with a weight representing the importance of the input. The neuron's internal state is referred to as the activation signal. The output signal results from combining inputs and activation rules and is sent to other units in the network. Among various network topologies, the feed-forward topology is essential, involving input, output, and hidden layers.

Artificial intelligence's fundamental concept lies in learning, where deep neural networks adapt to changes in their environment. DNN, as a complex adaptive system, can alter its internal structure based on the information it processes. As environmental conditions change, the DNN can adjust its input/output behavior. This adjustment typically involves modifying the weights in the neural network. Specific rules govern the process of weight modification to accommodate changes in input/output behavior.

Below is the adopted topology of the DNN as illustrated in the figure:

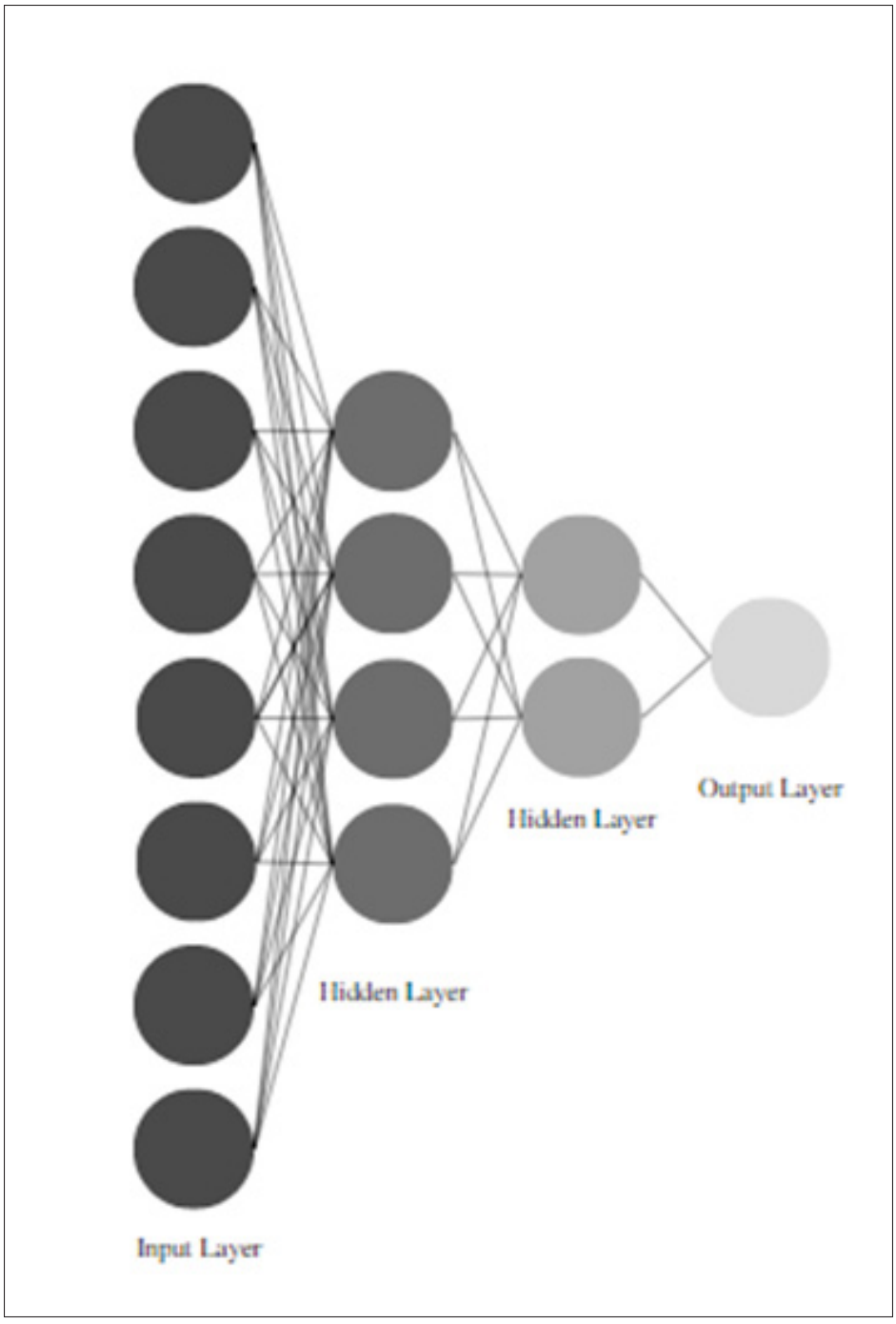


Figure 3.4 The DNN topology in the proposed model

In the thesis, the deep neural network (DNN) is employed to forecast network behavior, facilitating congestion avoidance, traffic load prediction, and network modeling. By leveraging the DNN

algorithm with the SDN dataset, the network can make intelligent decisions, leading to enhanced performance in terms of congestion avoidance and improvements in quality of service.

3.6.1 Addressing Optimality and Need for Deeper Network

Addressing Suboptimality:

The current work, while contributing substantial insights and advancements in employing DNN for network management, is acknowledged as a suboptimal solution. The pursuit of optimality in a DNN model necessitates an extensive and meticulous exploration of network architecture. A model is considered optimal when any additional refinements do not result in improvements or alterations in the output. This typically involves a detailed investigation, including the addition of hidden layers until the results reach a saturation point, indicating the optimal state of the model.

Optimal Solution and Future Directions:

The pathway to optimality in network management using DNNs lies in further refinement and continuous enhancements of the network topology, marked by rigorous assessments and adaptations. The exploration to deepen the network by incorporating more hidden layers and evaluating the subsequent modifications in the performance and results is pivotal. Such iterative refinements and evaluations are the stepping stones towards achieving a truly optimal DNN model in network management tasks.

The present model lays down the groundwork and exemplifies the potentialities of employing DNN in network management domains, and future iterations and enhancements of this model are anticipated to march progressively towards optimality in network management and congestion avoidance strategies.

3.6.2 Components

In the network, several layers are made of nodes; all the computation happens over the nodes' locations. Nodes in the network combine the input data from the dataset in the scenario f the coefficient or weights that either amplify the input or enable the algorithm to learn it as a task, like which one input is helpful to predict the thing more efficiently. The input weight is summed and passed through the activation function to conclude which input helps obtain the most recommended output for the prediction. In the network, the neurons are the switches, the initial input layers receive the data, and the other layers are subsequent layers and continuously receive the data. It is used to predict and explain how it is used in predicting the SDN network to obtain congestion-free traffic.

Table 3.1 The input values

Input Variable	**Details**
I1	Sender node IP
I2	Receiver Node IP
I3	Data Packets
I4	Bytes
I5	Duration
I6	flow
I7	Packets received
I8	Protocol

Table 3.2 The output values

Output Variable	**Details**
P1	Predict congestion-free routes (Predict R1)

3.6.3 DNN Pseudo-code

Input: Features of the dataset (X=X1,X2,,,,,,,Xn) and (Y=Y1, Y2,,,Yn) input

Output: Congestion prediction

Initialization: Load Dataset having Features (nodes, transmission rates time data sizes protocol)

Build DNN: DNN (Input layer, Hidden layers, Output layers) Compilation: Loss function

Split the data into training and testing: (X train, X test, Y train, Y test)

Data processing: by scaling the input features X test, X train and convert into the Y train in the form of binary (0, 1)

Train DNN: Use X train and Y train to train DNN, set iterations numbers, monitor the training and validation process.

Evaluating: Evaluate (X text, Y test)

Congestion Prediction: predict the congestion (0, 1, 2)

End process

3.7 The Mijumbi algorithm

In this model, the combination of the Mijumbi algorithm and the DNN plays a crucial role in rerouting predicted routes to address congestion issues in the software-defined network (SDN). The Mijumbi algorithm addresses congestion in three different ways, taking action based on the predicted routes, which are referred to as A, B, and C possibilities. These actions effectively overcome congestion problems and significantly improve network performance.

For action A, the Mijumbi algorithm adds a new substrate path and ensures the availability of resources, enabling congestion-free data transfer within the SDN. In the B condition, the controller utilizes a better path to minimize congestion and facilitate smooth data transmission. In the C condition, the algorithm removes the flow rule from the network.

The Mijumbi algorithm's approach, when integrated with DNN congestion prediction, ensures efficient handling of congestion issues, leading to congestion-free data communication within the network. This combined strategy enhances the overall network performance and helps to maintain smooth and reliable data transmission.

3.7.1 Mijumbi Algorithm Pseudo-code

Input: Predicted condition (0, 1, 2) obtained from the DNN

Output: Execute one of these (0, 1, 2) and perform its job

Initialization: Initialization of the action for (0= delete substrate path, 1= modify substrate path, 2= add new substrate path)

Start Process: (Check for the predicted values)

If (Pred_val=0)

Delete the substrate path

End if

If (Pred_val==1)

Modify the substrate path

End if

If (Pred_val==2)

Add new substrate path

End if

End Process

In the proposed model for congestion avoidance in the SDN system, the Mijumbi algorithm, as depicted in the provided pseudo code, plays a crucial role. Its purpose is to reroute traffic from congested routes to congestion-free routes, ensuring smooth data transmission within the network while adopting a congestion avoidance approach.

The Deep Neural Network (DNN) algorithm is responsible for predicting congestion, representing congestion levels using numerical values such as 0, 1, and 2. Based on these predictions, the Mijumbi algorithm takes appropriate actions to alleviate congestion.

The Mijumbi algorithm's actions include:

1. Deleting unnecessary transmissions, which could be contributing to congestion.
2. Adding new routes to enable data flow on alternative paths that are not congested.
3. Modifying existing routes to optimize data transmission and alleviate congestion.

3.8 Algorithm implementation structure

1. Start process
2. Initialization of the essential variables and data structure
3. Load Data set which contains the complete information of data communication in the SDN network
4. Split the data into the set for the training and testing data
5. Performing the Deep neural network algorithm that do prediction congestion based on the different scenarios
6. Train the algorithm using the training data of the dataset
7. Input variables to test the training algorithm
8. Output the congestion in the network
9. Simulation results using the Python Jupyter Notebook tool
10. End process

3.9 Summary

This chapter provides a description related to the proposed methodology for SDN. Initially, we have arguments on the requirements of the problem, why this issue arises, and why it is needed to be resolved. After that, we discussed the objectives of the thesis and how we will use the ML/DL schemes in the proposed model. After that, we have a details argument on the approaches of ML/DL and how these are useful for the considered problem in the thesis. Moreover, the details

have been discussed the proposed methodology in the ML/DL approach context. Ultimately, this question of why the ML/DL algorithm is used in the software-defined network has been addressed. In the next chapter, the simulation of the proposed methodology will be described, and the results will be shared in detail.

CHAPTER 4

SIMULATION AND RESULTS

This chapter presents the simulations performed for the performance evaluation of the proposed algorithm for congestion avoidance for software-defined networks. The proposed algorithm is based on the Mijumbi and deep neural network algorithms that predict the congestion and performs some modification in the network to perform congestion-free data communication with high-quality service. To test the performance of the designed algorithm, the proposed algorithm is applied to the software-defined network dataset, which is available on the internet. The performance of the proposed algorithm for congestion avoidance in SDNs is evaluated using the Python language tool Jupyter Notebook, and results are obtained from the considered SDN dataset.

The utilized parameters for the proposed algorithm performance evaluation in this thesis are relevant to the congestion of the network. All of these evaluation parameters results obtained from the simulations are correctly defined, and then these are illustrated in the plots and complete analysis on these plots. These plots will demonstrate the efficiency and improvements of the proposed model and show the improved results clearly described in detail. In the end, the summary of this chapter will give a comprehensive explanation of the simulations of this study. The results efficiently predict the network through the proposed algorithm, showing the improvement in the results obtained in simulations for congestion avoidance and network modelling, as well as the improved quality of service.

4.1 Contributions

The following points outline our main contribution to this study for congestion avoidance in software-defined networks using the proposed algorithm.

- Utilization of Machine learning and Deep learning approaches for the software-defined network that helps improve performance efficiently.

- Development of a novel SDN congestion avoidance model using advanced algorithms to enable the network congestion free and improve its quality of service. The performance of the proposed model is evaluated through the factors which are the throughput, delay, and packet loss in the network.
- The above factors are improving the performance of the software-defined network by dealing with the problem of congestion and by utilizing the proposed algorithm, which is based on the concepts of the Deep neural network and the Mijumbi algorithms. Deep neural networks more effectively predict congestion in software-defined networks, and the Mijumbi algorithm features help to modify the routing, ensure congestion-free routing, and attain high-quality services.
- Comprehensive experiments and evaluation of results for assessment of the designed algorithm's performance by utilizing the software-defined network SDN dataset.
- Preprocessing the utilized Software-defined network dataset to perform the proposed algorithm for network congestion avoidance and improving the quality of service.
- The performance evaluation is done by comparing results with the baseline model results identified as QRTP (quality of service aware resource allocation traffic prediction algorithm) results evaluation is based on these parameters throughput, delay, and packet loss, and providing the deep analysis of the obtained results and their improvements in detail.

This study significantly contributes to making an efficient model that provides congestion-free software-defined networking with high-quality service. This will be a positive impact on the applications concerning the software-defined networks.

4.2 Simulations

The proposed model is designed for congestion avoidance in software-defined networking and increases its quality of services. The proposed model is experimented on the dataset of the software-defined network available online and simulated using Python in the Jupyter Notebook tool. Simulating the proposed model aims to evaluate its performance to justify its improvement. In order to attain the goal, the proposed model utilizes different attributes to predict the network

performance more accurately: throughput, delay, and packet loss. In the simulation, the relevant features from the dataset are used in the algorithm and produce the results with modified routes for the congested flows. Then the results are compared based on the before and after prediction findings. The accuracy of the proposed model is tested, and it has the potential for significantly improved network performance by congestion prediction using ML/DL techniques and network data.

4.2.1 Evaluation parameters

Simulation of the designed model is a vital source to estimate the models' performance. More specifically, the newly proposed models are assessed through the simulation performance. In software-defined networks, estimating the network behaviors through simulation performances in different scenarios is significantly possible. Python is one of the most flexible tools for developing the simulation and evaluating the results for the proposed models also beneficial to simulate the software-defined network for the congestion perdition model.

To estimate the performance of the proposed model in this study, these are the network characteristics of the network traffic, like the packet size, time to deliver the data between devices, and the total data transferred between sender and receiver and transmitted data in bytes and protocol. The network traffic is controlled and optimized using these possible approaches, such as load balancing in the network also, path optimization, etc. Moreover, the utilized algorithm is used to predict the network congestion and help optimize the path for the data transmission in the network with congestion avoidance. The proposed algorithm loads the data and performs Deep neural network operations, such as data preprocessing and splitting. After that, the training and testing phase comes and then predicts the congestion of the network. Then the network uses the Mijumbi algorithm features to reroute the congested flow and transmit the data without congestion. This overall process is programmed and optimized, allowing dynamic and effective routing decisions based on the current network situation.

To conclude, the parameters are explained and discussed above correctly, and performing simulations for the software-defined network congestion prediction to enable the network congestion free.

The table below shows the parametric values utilized in the simulation performance for the proposed model of congestion prediction to avoid congestion in the SDN.

Table 4.1 The Simulation tools

Tools	**Name**
SDN Controller	Mininet
DNN Simulation	Jupyter Notebook
Data-set	DDoS attack SDN Dataset
Programming language	Python
Libraries & Packages	Pandas, Numpy, Matplotlib, Conda.

The table provided above illustrates the tools and dataset utilized in conducting simulations for the proposed model in software-defined networks (SDN). The table consists of two columns, with the first column containing the names of the simulation parameters, and the second column providing details about each parameter.

The simulation tool used for this study is Jupyter Notebook, which is an interactive computing environment enabling users to create and share documents containing code, text, and other elements. Jupyter Notebook primarily supports the Python programming language and is widely used due to its user-friendly nature. Code can be written in individual cells, making it more manageable and comprehensible, thus facilitating development and experimentation.

Jupyter Notebook's strengths lie in its efficiency in generating outputs and visualizations, including graphs, charts, and widgets. It provides a powerful means for data exploration without relying on external plotting tools for result evaluation. This feature-rich environment enables researchers to effectively analyze and visualize simulation results within the notebook itself, contributing to an efficient and productive research process.

Furthermore, the utilized dataset comprises various columns containing data communication information, including time, data size, protocols, and other relevant details. This dataset was downloaded from the Google dataset repository, providing a comprehensive context for the experiments conducted in the proposed model.

In addition to the dataset, the programming language Python was chosen as the primary language for implementing the simulations. Python is widely adopted and highly favored for its ease of understanding, versatility, and extensive library support. Its usage enables efficient development and testing of network control algorithms.

By leveraging Python in the simulation process, the designed software-defined network congestion avoidance model can be thoroughly evaluated and fine-tuned to enhance the overall quality of services. Python's powerful capabilities and user-friendly nature contribute to a seamless and productive simulation experience, aiding researchers in achieving accurate and reliable results in their SDN congestion avoidance studies.

Pandas is one of the employed important libraries for this proposed model simulation, which offers the manipulation tool for the software-defined network simulation to assist in data processing, transformation and analysis for the data. This library also helps structure the data and make it efficient for data organization and manipulation, which is essential for software-defined network preparation scenarios. Another one is the Numpy, which provides t

4.2.2 Proposed algorithm simulations

The proposed algorithm is based on the concepts of two algorithms, the Deep neural network (DNN) and the Mijumbi algorithm, which are essential and valuable algorithms used for the prediction and modification of routes with that is based on the concept of the Mijumbi algorithm from congested routes in the network for communication. Python is a highly recommended and popular programming language for different purposes, such as ML and AI.

The basic concept of the thesis is to train the neural network for congestion prediction in the SDN for congestion avoidance, which helps improve the network's performance. Using the SDN data helps to find patterns and trends in network behavior that help to make the prediction more accurate. When the network faces congestion, certain states can increase the packets directed toward the receiver or have less time to transfer the high volume of data on the same path. In these cases, the algorithm is utilized to predict the occurrence of congestion using these factors.

The utilization of the deep neural network technique of ML/DL for the perdition of the congestion in SDN trends is increasing in the field of computer networking, and it is highly adaptable and helps to attain more accurate results. Moreover, the neural network algorithm can quickly process and deal with a large amount of data, producing more efficient results. That is most recommended for real-time network prediction.

4.3 Results

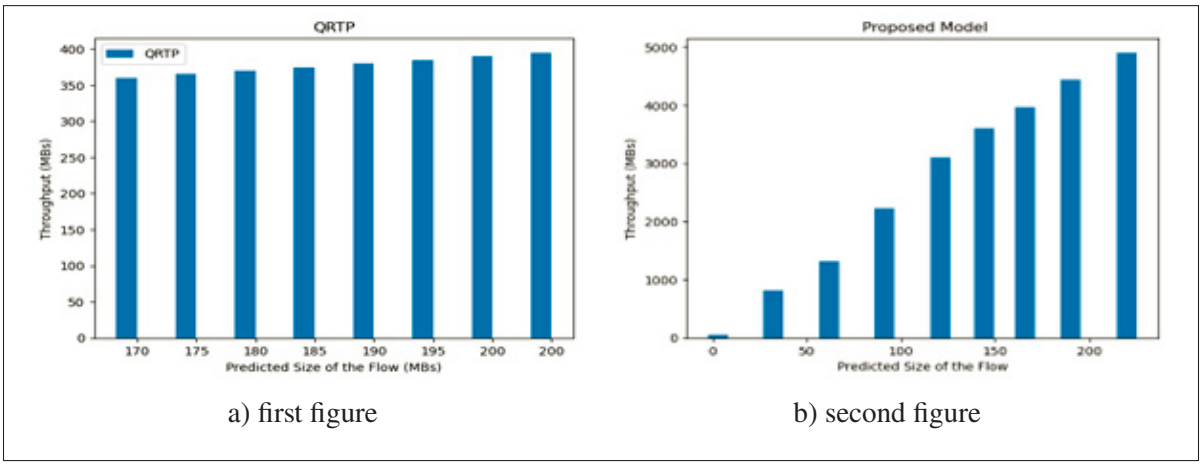
This study is based on prediction-based congestion avoidance and uses an algorithm based on Mijumbi and a deep neural network for congestion prediction. The proposed model is implemented in Python using the Jupyter Notebook tool using the 'DDoS attack SDN dataset'. The proposed algorithm illustrates the results in the plots after and before prediction; the proposed model result is evaluated for these parameters such as throughput, delay, packet loss, and quality of services.

4.3.1 Throughput

Throughput can define as the total data packet transmission in the network for the time intervals. In this model, the proposed algorithm predicts the network congestion. It helps to optimize the path for the congestion-free network using the intelligent algorithm and Mijumbi algorithms as integrated algorithms to ensure congestion-free routing. To estimate the algorithm performance, the throughput used in the data communication is relevant to the data measuring how a small

amount of the data is transmitted in a specific period. Throughput is a crucial factor in finding the reliability of network communication.

The below-stated figures (4.1) show the graphs' results that depict the network system's throughput over time. The movements of the line in the plots show how the network operated for each time interval and how much the throughput was at that time. In this context, the network communication, if the throughput is high, states that the high volume of the data is transmitted in the network. If it remains low, then it says that the low volume of the data is delivered over a specific period; to assess the network performance, the simulation-based results are employed and visually depict the network performance.



a) first figure b) second figure

Figure 4.1 Comparison of the proposed model and The QRTP Congestion prediction models in the context of throughput versus data flows

The above Figures present the throughput in the network. They are evaluated with the predicted size of the flows in MBs versus the network's throughput in the MBs, as clearly figured out in the simulation results. The above graphs show the prediction-based model as a baseline model with the newly proposed model in this study. The results obtained justify its improvements over the existing model. The x-axis represents the presenting predicted size of the flow in MBs, and the y-axis illustrates the results of the throughput in the MBs; however, the baseline model and the proposed model's results are shown separately for the figure titled QRTP, leading the

development of the baseline model, and the plot on the right is showing the results for the proposed model. The results shown in the graphs are one of the best ways for QRTP to show the network performance.

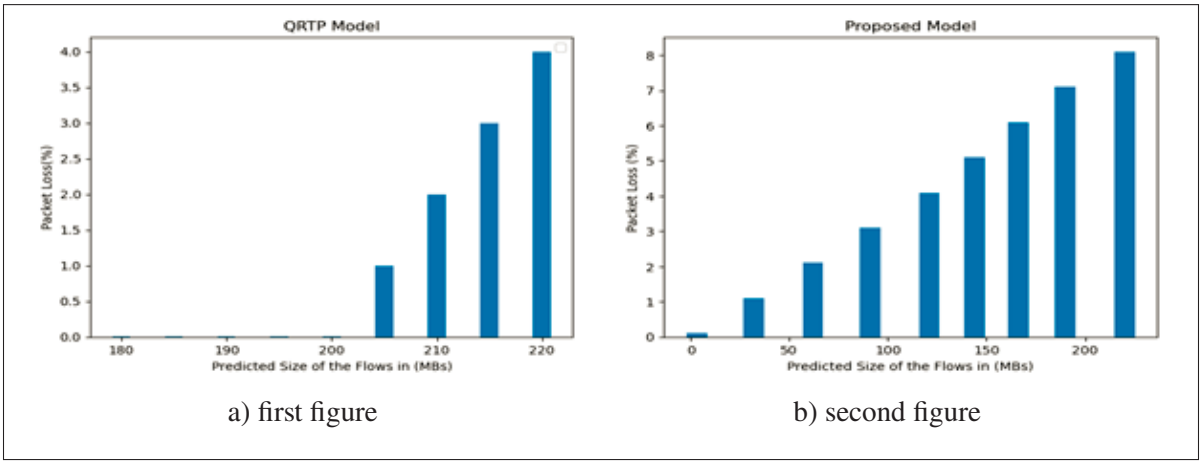
Figure 4.1 shows the proposed model, and the QRTP correctly considered the baseline research model results. Two plots are shown in Figure (4.1). Both of the schemes compare based on the throughput of the network software-defined network model. The throughput of these models in the simulation is evaluated in the megabytes per second, which are asses against the predicted flow sizes in MBs. The baseline model results are around 350 MBs transmitted. Compared with the proposed model, it is a very highly efficient result, and the newly designed model transited the 4000 MBs with the same numbers of the predicted flow sizes in MB. The finding illustrates that the proposed model's results are efficient in the context of the network's throughput. So, these results satisfy the improvements of the proposed model over the baseline model.

4.3.2 Packet loss

Packet loss is a fundamental problem in the software-defined network model, and its performance is necessary to obtain the network performance. The delivery of the data packet from the sender nodes to the receiver nodes should accurately remain the maximum packet delivery ratio evaluated by the packet loss in the time in seconds to estimate the performance. There are several possibilities for packet loss in the software-defined network, such as the fewer energy resources for data communication over long distances among the communication devices, also the overloading of the data transferred toward a receiver node that is not able to receive all packets in the specific period, the transmission of the data on the same route called congestion. The high packet loss ratio is not reasonable for the network performance, and even it can move toward the overall failure of the SDN. The impacts of packet loss could be better, and it encourages the researchers to mitigate its effects.

Figure (4.2) Shows the simulation results in the proper graphs that indicate the x-axis, the iteration numbers, and the y-axis, the packet loss of the software-defined networks. The proposed

model shows the performance of the proposed model and its comparison to clear the difference of the packet loss rates in the graph before and after the prediction through the proposed algorithm in the thesis. This is important in estimating the performance of the software-defined network and its reliability. The results of the proposed model are evaluated with the baseline model to estimate the improvements of the proposed model and then the baseline model.



a) first figure b) second figure

Figure 4.2 The Packet Loss of the proposed algorithm versus the QRTP model for the packet loss rate.

The above-shown figure (4.2) presents the packet loss rate versus the predicted size of the flows in (Mb) for the baseline model QRTP, the proposed model, and the network results obtained from the simulation. The above graphs show the results on the x-axis for the predicted size of the flows in MBs and the y-axis of the plots showing the packet loss rate for both models. In this context, packet loss is an essential factor in determining the network's quality of service. The plots present the prediction finding for the packet loss in the network. The high value of the packet loss shows the network's poor quality and fewer packet loss numbers show the network's better quality. These parameter values significantly impact the results, which are obtained results based on different values.

Figure (4.2) illustrates the results for the QRTP and the proposed model that is compared based on the packet loss rates versus the size of the flows in the MBs. The purpose model shows fewer

packet loss numbers than the QRTP, indicating that the proposed model is more efficient than the baseline model with a low packet loss rate. The predicted size of the flows in MBs was 220 for both models, their packet loss percentage was 4 percent; however, in the baseline QRTP model, the flow size was 200 MBs, and the percentage was 4 percent, showing the improvements over the baseline model and its more efficient model.

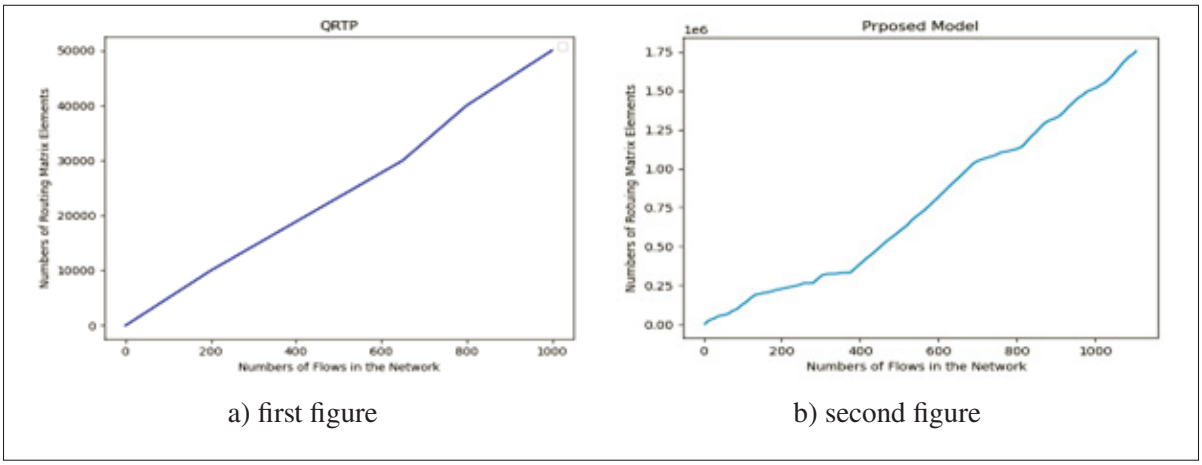
4.3.3 Reconfiguration effects

Routing matrix evaluation is a critical evaluation parameter to assess the performance of the software-defined network. The routing matrix represents the route that transmits the data from a source to the destination. Also, it consists of the grid elements that make the routing decision on each node in the network. The routing matrix impacts the routing algorithm's efficiency and helps get the algorithm's performance. A large amount of the routing matrix shows the better performance of the algorithm in the appropriate network. In this scenario, the overall control remains better, and the optimization of the network traffic flow. The more the routing matrix utilizes, the network can accommodate a considerable number of valuable paths and ensure a flexible route in the sense of congestion and link failure.

In the congestion, the predicted accurate network administrator can measure to avoid it proactively. Analysis of the routing matrix numbers helps estimate the granularity of the routing information available. Moreover, the complete information for the routing matrix enables more efficient traffic management/engineering and load balancing, and congestion avoidance strategies. Furthermore, the routing matrix also impacts the routing algorithms' computational and memory needs. Suppose the amount is higher than the complexity of the routing; In that case, it leads the high overhead, but getting the right balance of the routing matrix is better for obtaining network stability.

To conclude, the routing matrix elements are a proper evaluation parameter in the congestion avoidance prediction models to assess the network's performance. Also, it influences the flexibility, efficiency, and scalability of the routing algorithms. It enables the network administrator

to make informed decisions for optimizing the network performance and mitigating the relevant problems.



a) first figure b) second figure

Figure 4.3 The Comparison of the Proposed model with the QRTP model based on the numbers of the routing matrix elements versus the numbers of the flows in the network.

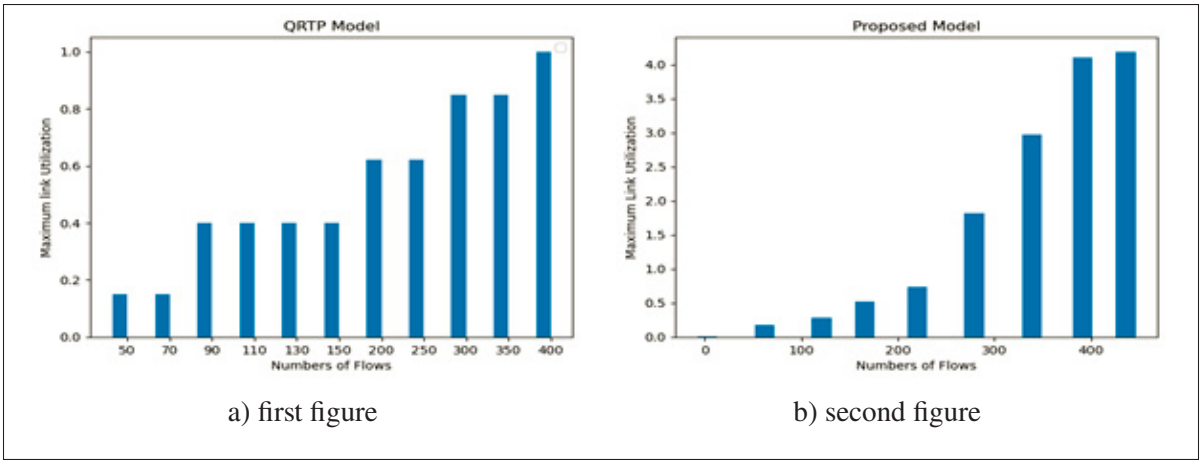
The above figure (4.3) shows the finding of the proposed model and the baseline model in the software-defined network. These plots show the results of both models for the numbers of routing matrix elements versus the flow numbers in the network. The results for the baseline model are up to 5000 in its values. However, the proposed model shows a value up to 1750000, which is a very high value compared to the baseline model, so we can conclude that the proposed model is more efficient and can solve the congestion problem and routes the data more effectively than the baseline model. These results are greatly enhanced than the QRTP model and more efficient. These are the findings that show the insights of the newly designed model that help in the congested routes to optimize the congestion-free routing and transmit the data from the sender nodes to the receivers in a beneficial manner and provide more excellent results. This funding is more helpful for the network administrators when they analyze the software-defined congestion prediction models to utilize the model on their SDNs in real-time. Moreover, accurate prediction through advanced artificial intelligence approaches is greatly acknowledged, and these approaches are helping researcher to solve their issues more efficiently.

4.3.4 Link utilization

Link utilization is a crucial parameter for evaluating the performance of software-defined networks, particularly in congestion avoidance models. It represents the bandwidth or capacity that a specific link utilizes in the network infrastructure. Evaluating link utilization provides valuable insights into the network's efficiency. When the link utilization remains high, it indicates that the link operates close to its maximum capacity, which can lead to congestion. This congestion, in turn, causes delays, packet loss, and a decrease in overall network performance.

Monitoring link utilization helps network administrators gain a comprehensive understanding of the network's performance. By identifying high link utilization, administrators can recognize potential congestion issues and take proactive measures to prevent them. Real-time data provided by network monitoring tools enables administrators to spot sudden spikes and high levels of link utilization that may contribute to congestion. By addressing these issues promptly, administrators can optimize network performance and ensure smooth data transmission throughout the network infrastructure.

To avoid facing the above issues, the proposed algorithm is used to predict congestion, help reroute the data, and avoid congestion by allocating resources more effectively. By proactively managing the links utilization, the network administrator can maintain sub-optimal network performance, reduce downtime, and ensure a stoppable user experience. To conclude, link utilization is a valuable evaluation parameter for predicting and avoiding congestion in network systems.



a) first figure b) second figure

Figure 4.4 The Comparison of the Proposed model with the QRTP model based on the maximum link utilization versus the number of flows.

The above-stated figure shows the results of the QRTP model and the proposed model results in the context of link utilization. Both models show their findings in separate plots for the x-axis, the development of the numbers of the flows, and then on the y-axis, the result of the link utilization obtained for both prediction-based algorithms, and their results are presented. As shown in figure (4.4), the numbers of the increasing flows on the x-axis and the value of the link utilization also increase in the QRTP model in comparison with the proposed model as the numbers of the flows increased the value of the link utilization in enhance but with high numbers as compared to the QRTP model. That shows the proposed model has high numbers for link utilization in comparison.

The increasing link utilization in the proposed model compared to the QRTP model shows that this is more efficient in handling congestion-related tasks. They can utilize the link to ensure the transaction in the network more strongly and obtain the results more efficiently. To conclude the overall analysis, the results demonstrate the superiority of the proposed model over the baseline model in the link utilization with the same number of flows in the simulation performance. The overall finding highlights the proposed model's ability for the real-time network for better performance in the congestion avoidance algorithm. This can maintain the low link factor for reliable and high-quality network services.

The table below shows the parametric values utilized in the simulation performance for the proposed algorithm MLP (Multi-layer Perceptron) classifier.

Table 4.2 Link utilization parameters

Parameter name	**Detail**
Numbers of the epoch	100
Number of neurons in the input layer	8
Number of the hidden layers	2
Number of the neuron in the first hidden layer	4
The number of neurons in the second hidden layers	2
Number of neurons in the output layer	1

4.3.5 Loss

Loss is an important performance evaluation parameter of the deep neural network. The exact value of the loss is the error between the actual and predicted values of the output. For the loss assessment, the error function is specifically used.

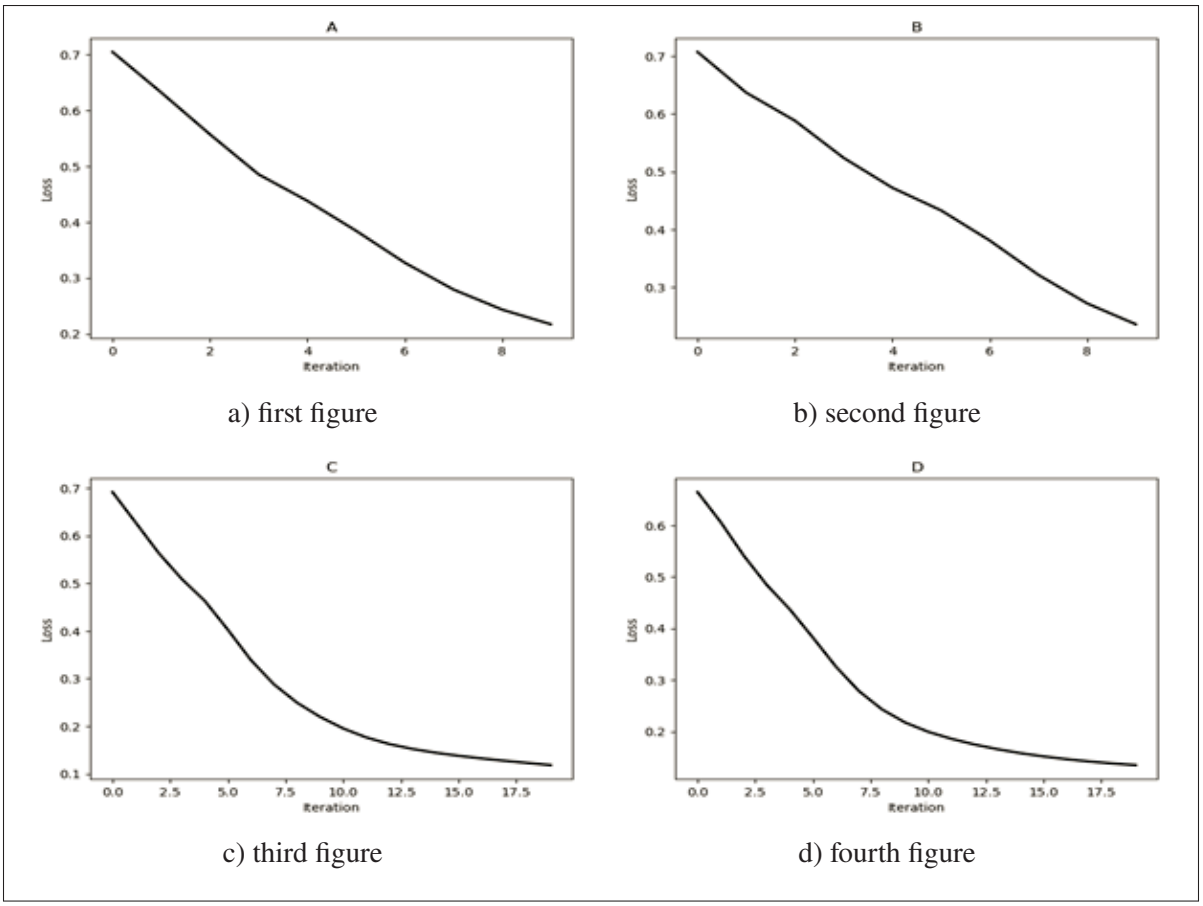


Figure 4.5 The MLP loss results with different parameters

The above-stated graphs show the finding of the proposed algorithm with different values of the testing data percentages and the iteration numbers as measured by MLP loss. The x-axis represents the values of the iteration numbers, and the y-axis for all the showing graphs represents the MLP classifier of the neural network. Plot A shows the results for the 10 percent data as testing and ten iterations for the MLP classifier of the neural network-based results; plot B shows the 20 iterations in the MLP classifier of neural network with the same data amount for testing purposes. Plot C shows the results for the 10 percent data as testing from the dataset with 20 iterations in the MLP classifier of the neural network, and Figure D shows the result of the 20 percent data as testing data from the dataset and the 20 iterations in MLP classifier of neural network-based results. As per these results, the best results are obtained using the parameters of

the 20 percent data as testing from the total data of the dataset and the 20 iterations in the MLP classifier of the neural network in the network.

4.3.6 Accuracy

Accuracy is an assessment parameter mostly adapted to test the neural network's performance. This proposed algorithm uses the deep neural network with the Mijumbi algorithm features to predict and modify the routing based on the prediction values for the SDN congestion issue. To test the performance, the algorithm's accuracy is plotted in the simulation shown in the below figures to understand the accuracy more efficiently with multiples of the iteration numbers. Assess the accuracy of the neural network that performs the data classification, depending on how effectively the DNN (Deep Neural Network) can predict the class labels. The results are shown in the below figure.

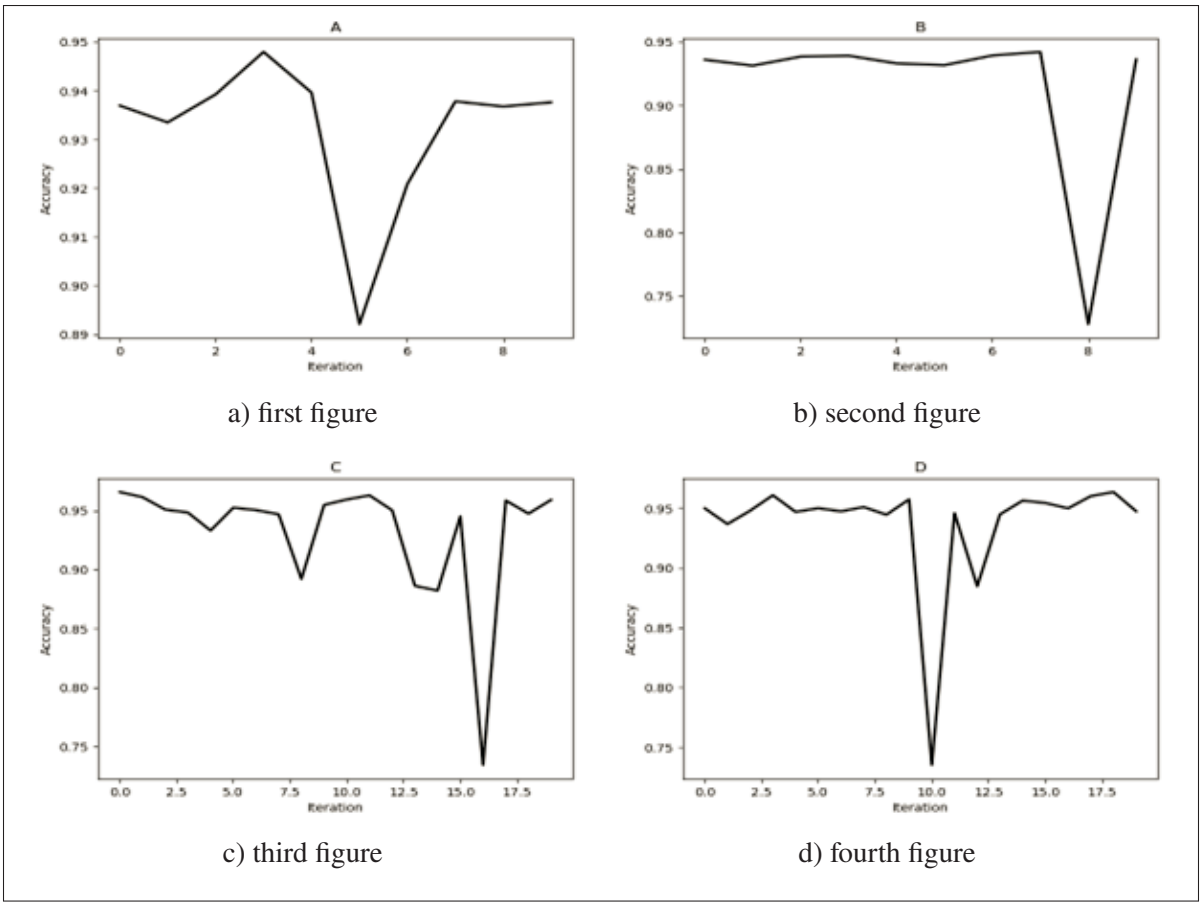


Figure 4.6 The Accuracy of the proposed algorithm with 20% testing data and 20 iterations

The above graphs illustrate the results of the algorithm's accuracy with 10 and 20 percent of the testing data of the total dataset. The data set used in the study is divided into training and testing data. The testing data is 90 and 80 percent, and for the testing, it is considered 10 and 20 percent for the different scenarios, and the result is above Figure (4.6). The values at the x-axis show the numbers of the iterations, and the y-axis shows the accuracy of parentages. The A plot shows the result with the 10 percent of the dataset used in the algorithm as testing data and the accuracy shown in the plot. Plot B, the results are showing satisfying. Still, it is not more than the scenario with the ten precents for the data used as a testing dataset, and this scenario shows accuracy compared with the scenario of the 10 percent. Plot C is based on 10 percent of the data from the dataset for testing purposes with 20 of the iteration in the MLP classifier of

the proposed DNN algorithm. Plot D shows the model's results with the 100 iterations in the MLP classifier with 20 percent of the data for testing offered from the dataset. To compare all of these to get the best of them, the parameter values of 10 percent data as a dataset and the ten iteration-based results show the more efficient results. Finally, these show all the figures are compared to each other, and figure (4.6) shows the more excellent results from all of them. It is helpful to be adopted in the future for the prediction algorithm with these parameter values.

4.4 Summary

Our thesis's simulation and results chapter are presented to predict the congestion in software-defined networks. The simulation environments are explained in reasonable detail, such as how the simulation performs and which tool utilize. The simulation evaluates the improvements in the network after the designed model in the context of the prediction of the congestion to minimize the congestion and enhance the quality of services in the network. The objective of the study areas is congestion avoidance in the network by utilizing the proposed algorithm for load prediction and making congestion-free transmission in the software-defined network.

This chapter is initiated by explaining the simulations. Then the experiments were performed to estimate the SDN performance based on the evaluation parameters such as throughput, delay, packet loss, quality of service, and accuracy. All of the stated evaluation parameters are separately explained, and each is also tested with different values of the performance evaluation in the software-defined network.

The results obtained from the simulations express that the proposed load prediction model is more accurate and has been shown to have an accuracy of more than 90 percent. The prediction is performed on the classification that illustrates the most suitable congestion and its solution to manage the transaction in the software-defined network. In this scenario, 0, 1, and 2 are the prediction classes that respond by working the route for the data delivery by adding a new path, modifying the path, deleting the data packet, and avoiding transmission.

In order to assess the performance of the proposed model, the results of the throughput, packet loss rate, link utilization, and the number of routing matrix elements are evaluated and compared. These results are considered with the model QRTP to test the performance of the algorithm proposed in the thesis. These results are performed in the simulation and obtain the accuracy and the loss rates for the proposed algorithm based on the deep neural network and Mijumbi algorithm, and the predicted congestion modifies the routes for the data transmission to make the congestion-free networking and the results after utilization of the algorithm are evaluated through these parameters such as throughput, link utilization, numbers of the routing matrix elements, packet loss, and QoS.

Overall, the simulation and results chapter provide definitive evidence of the proposed model's efficiency and better performance by utilizing the proposed algorithm in the context of the load prediction algorithm for the software-defined networks.

CHAPTER 5

CONCLUSIONS AND FUTURE WORKS

5.1 Conclusion

In this research, we concentrated on creating sophisticated algorithms that could anticipate traffic and identify normal patterns for each base station. By integrating these algorithms into an SDN controller, the network will be able to design itself and make intelligent recommendations based on anticipated traffic demands.

Software-defined networking is one of the extensively adopted network architectures due to its programmability, flexibility, and network control, providing multiple benefits and improvements in network performance. The integration of advanced technologies and advanced algorithms, such as artificial intelligence, provides more efficient solutions to problems. The combination of more advanced technologies such as software-defined networks SDN and machine learning and deep learning, which are the field of artificial intelligence (AI), provide better approaches to solving problems and achieving high and efficient results in networking.

This thesis is based on the congestion avoidance model in software-defined networks by load prediction and modification procedures utilized in data communication using the efficient algorithm. This flexibility offers the integration of advanced technologies, such as artificial intelligence, which can enhance the network performance and provides a better solution to significant problems. The proposed approach is based on an algorithm that is defined by utilizing the features of the deep neural network and the Mijumbi algorithm to predict the congestion and modification in the software-defined network. The Deep neural network is used in the proposed model to predict congestion. Then the load prediction is used for the data transmission modification for utilized scenario, reroute the data with modified routes to avoid congestion. The utilized scenarios are more valuable and helpful in enhancing the quality of service of the software-defined network.

The proposed model is utilized to perform network activities in the software-defined network, such as the flow. The proposed model's process is initiated from the dataset's loading that is available online. Then the process moves toward the preprocessing of the data, such as removing the irrelevant data attributes that are unimportant for the proposed scenario. After the data preprocessing phase, the algorithm performs training and testing using the dataset for the designed algorithm in this study. The next step is the prediction of the network's congestion, which is performed using the essential and relevant features of the dataset, such as the data packet transmission time to transmit the data, the route of the transmission, etc. The proposed algorithm performs the prediction for the congestion in the network in three scenarios that will move toward the decision of the data transmission into an efficient route without congestion more effectively. This procedure is using the Mijumbi algorithm features used in the proposed algorithm. After prediction the data transmission may decide to modify the data, delete the data packet, or transmit using some extra time. However, some issues may occur in the network in the delay scenarios or others.

In the algorithm, 90 percent of the data from the utilized dataset is used as training data, and ten percent of the data is utilized as testing data to attain the primary goal of congestion avoidance in the SDN among with quality of service. Three parameters are used to evaluate the network's performance and assess the QoS through delay, packet loss rate, and throughput. The simulations of the proposed model are performed using the Python language and the Jupyter Notebook tool to evaluate the network's performance. The basic parameters assess the proposed model's performance with the eight input neurons, 4 for the first hidden layer and 2 for the second hidden layer, and 1 for the output of prediction values. The predicted results of the proposed model compared with the baseline model identified as the QRTP model and after prediction improvement in the network, the results are correctly analyzed and showed in complete detail. These results generated in the network are also developed with the same values parameters to show the improvements in the proposed model, which are helpful to get the more efficient and best results with the scenarios. The obtained network results show significant improvement and justify its performance.

So, its results are improved by significantly comparing the proposed model with the QRTP model with the same values of the predicted flow size that was supposed to be used for evaluation. The throughput in MBs for the proposed model delivered data around 5000 MBs in the 200 predicted flow size. However, the baseline model shows 400 MBs in the same flow size. This thing shows the betterment of the proposed model over the baseline model. The packet loss rate of the proposed model also improved because, in the comparison of the results, the flow size in MBs was 220, and the packet loss rate of the proposed model was 100 percent improved over the baseline model.

Moreover, for the link utilization, the results of the proposed model show significant improvements over the compared model with the same number of flows in the comparison. Further, the numbers of the routing matrix elements in the proposed model show improvements compared to the QRTP model. The QRTP model shows values around 5000, and the proposed model shows 1750000 with similar flows in the network. Hence, it is proposed that the proposed model is better and justifies its simulation performance.

The proposed algorithm was also tested through the accuracy results as the MLP loss, the simulation for them was also evaluated with several values of the parameters to get more efficient results. For the MLP loss, the proposed model shows the results with the10 percent of the data as the testing data of the used dataset, and the 10 iterations of the MLP classifier of the neural network. It shows the best results than other adopted scenarios. However, the overall findings in all the scenarios also justify the performance of the proposed algorithm for congestion avoidance and quality of service in the software-defined networks. Also, the accuracy of the proposed model is evaluated, and the results show satisfaction with high accuracy; however, the parameter of 10 percent as the dataset for the testing data from the dataset and ten iterations in the MLP classifier of the neural network-based scenario show an accuracy of more than 90 percent and its best and more utilizable model.

To finalize, it can be stated that integrated technologies based on artificial intelligence and software-defined networks are more reliable and efficient in finding the solution to several

problems in the SDNs. also, the multiple benefits and improvements in the network performance. The proposed combined algorithm for this research is based on the advanced approaches that help provide more efficient results, specifically, the congestion avoidance problem and the QoS. The proposed algorithm's performance is evaluated in the simulations properly, and the results are obtained in the different evaluation parameters, as discussed earlier. The other results are obtained based on the diversity of the parameters of the simulation for the evaluation of the algorithm because it is essential for the network architecture due to the adaptability and flexibility of the software-defined (SDN).

We think that the contributions made to this work can be leveraged to design network analytics applications that make use of big data SDN programmability to build a decision-making artificial intelligence.

5.2 Future works

This study can be extended using a more efficient and advanced algorithm for more efficient results. We can propose a model that provides the resulting context for more areas. SDN can use artificial intelligence advanced techniques to improve the software-defined network performance and get more efficient for its utilization. Moreover, this study can be extended by dealing with the delay that occurred in the network. It can be resolved through an efficient algorithm in the network.

Below, we offer some ideas for our future projects.

- **Test on Other traffic datasets:**
  For various traffic datasets, performance is anticipated to vary. It would be beneficial to evaluate the accuracy of the same ML algorithms using the selected characteristics on various traffic datasets. Results from training ML algorithms on new datasets versus training them on the present dataset and testing them on the new network can be contrasted.
- **Test of different network topology:**

To verify its effectiveness, the optimized route model can be tested on different network topologies. To determine how the model will function in situations where there are link failures, more research can be done.

- **Test with Additional Algorithms:**
  Better results for the measured accuracy or runtime in the traffic categorization problem might be obtained by testing additional ML algorithms or fine-tuning the tested algorithms' parameters.

BIBLIOGRAPHY

(2012). Google Network. Retrieved from https://techblog.comsoc.org/2012/04/24/googles-largest-internal-network-interconnects-its-data-centers-using-software-defined-network-sdn-in-the-wan.

(2015). Software defined networking. Retrieved from https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/VMDC/SDN/SDN.html.

(2016). Traffic load prediction in SDN/OpenFlow networks - ProQuest. Retrieved from https://www.proquest.com/docview/2715587037.

(2019). Classification and Enrichment of Unlabeled Feedback Data using Machine Learning. *International Journal of Engineering and Advanced Technology*, 9(1), 6647–6650. doi: 10.35940/ijeat.a1912.109119.

(2020). Global - 2020 Forecast Highlights. Retrieved from URL: https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2020_Forecast_Highlights.pdf.

Ahuja, N., Singal, G. & Mukhopadhyay, D. (2020). DDOS attack SDN Dataset (Version V1). doi: 10.17632/jxpfjc64kr.1.

Azzouni, A. & Pujolle, G. (2017). NEUTM: a neural network-based framework for traffic matrix prediction in SDN. *arXiv (Cornell University)*. doi: 10.48550/arxiv.1710.06799.

Baz, A. (2018). Bayesian Machine Learning Algorithm for flow Prediction in SDN switches. doi: 10.1109/cais.2018.8441969.

Bouzidi, E. H., Outtagarts, A., Langar, R. & Boutaba, R. (2021). Deep Q-Network and Traffic Prediction based Routing Optimization in Software Defined Networks. *Journal of Network and Computer Applications*, 192, 103181. doi: 10.1016/j.jnca.2021.103181.

Canini, M., Kuznetsov, P., Levin, D. & Schmid, S. (2015). A distributed and robust SDN control plane for transactional network updates. *Proceedings of the IEEE INFOCOM*. doi: 10.1109/infocom.2015.7218382.

Chen-Xiao, C. & Yabin, X. (2016). Research on Load Balance Method in SDN. *International Journal of Grid and Distributed Computing*, 9(1), 25–36. doi: 10.14257/ijgdc.2016.9.1.03.

Hu, Y., Wang, W., Gong, X., Que, X. & Cheng, S. (2014). On reliability-optimized controller placement for Software-Defined Networks. *China Communications*, 11(2), 38–54. doi: 10.1109/cc.2014.6821736.

Keary, T. (2020). Software-Defined Networking – SDN Guide. Retrieved from https://www.comparitech.com/net-admin/software-defined-networking/.

Khodayari, S. & Yazdanpanah, M. J. (2005). Network routing based on reinforcement learning in dynamically changing networks. *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*. doi: 10.1109/ictai.2005.91.

Kreutz, D., Ramos, F., Veríssimo, P., Rothenberg, C. E., Azodolmolky, S. & Uhlig, S. (2015). Software-Defined Networking: A Comprehensive survey. *Proceedings of the IEEE*, 103(1), 14–76. doi: 10.1109/jproc.2014.2371999.

Kuang, X., Xu, L., Huang, Y. & Liu, F. (2010). Real-time forecasting for short-term traffic flow based on General Regression Neural Network. *Proceedings of the 2010 World Congress on Intelligent Control and Automation (WCICA)*. doi: 10.1109/wcica.2010.5554911.

Lange, S., Gebert, S., Zinner, T., Tran-Gia, P., Hock, D., Jarschel, M. & Hoffmann, M. (2015). Heuristic approaches to the controller placement problem in large scale SDN networks. *IEEE Transactions on Network and Service Management*, 12(1), 4–17. doi: 10.1109/tnsm.2015.2402432.

Liang, G. & Li, W. (2018). A novel industrial control architecture based on Software-Defined Network. *Measurement & Control*, 51(7–8), 360–367. doi: 10.1177/0020294018784310.

Liu, S. & Li, B. (2017). Stemflow: Software-Defined Inter-Datacenter Overlay as a Service. *IEEE Journal on Selected Areas in Communications*, 35(11), 2563–2573. doi: 10.1109/JSAC.2017.2760159.

Liu, Y., Yu, H., Xie, S. & Zhang, Y. (2019). Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks. *IEEE Transactions on Vehicular Technology*, 68(11), 11158–11168. doi: 10.1109/tvt.2019.2935450.

Mestres, A., Alarcon, E., Ji, Y. & Cabellos-Aparicio, A. (2018). Understanding the Modeling of Computer Network Delays using Neural Networks. *Proceedings of the 2018 Symposium on Cloud Computing (SoCC)*. doi: 10.1145/3229607.3229613.

Mijumbi, R., Serrat, J., Rubio-Loyola, J., Bouten, N., De Turck, F. & Latre, S. (2014). Dynamic resource management in SDN-based virtualized networks. *2014 10th International Conference on Network and Service Management (CNSM)*. doi: 10.1109/cnsm.2014.7014204.

Mohammed, A. R., Mohammed, S. A. & Shirmohammadi, S. (2019). Machine learning and deep learning-based traffic classification and prediction in software defined networking. *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*. doi: 10.1109/iwmn.2019.8805044.

Nunes, B. P., Mendonca, M., Nguyen, X. N., Obraczka, K. & Turletti, T. (2014). A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. *IEEE Communications Surveys & Tutorials*, 16(3), 1617–1634. doi: 10.1109/surv.2014.012214.00180.

Open Networking Foundation. (2013). SDN architecture overview 1.0. Retrieved from https://opennetworking.org/wp-content/uploads/2013/02/SDN-architecture-overview-1.0.pdf.

Parsaei, M. R., Sobouti, M. J., Khayami, S. R. & Javidan, R. (2017). Network Traffic Classification using Machine Learning Techniques over Software Defined Networks. *International Journal of Advanced Computer Science and Applications*, 8(7). doi: 10.14569/ijacsa.2017.080729.

Redmond, J. (2021). Artificial Neural Networks - John Redmond - Medium. Retrieved from https://medium.com.

Saxena, M. & Kumar, R. (2016). A recent trends in software defined networking (SDN) security. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 851–855.

Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J. & Rao, N. (2013). Are we ready for SDN? Implementation challenges for software-defined networks. *IEEE Communications Magazine*, 51(7), 36–43. doi: 10.1109/mcom.2013.6553676.

Shu, Z., Wan, J., Lin, J., Wang, S., Li, D. M., Rho, S. & Chen, R. (2016). Traffic engineering in software-defined networking: Measurement and management. *IEEE Access*, 4, 3246–3256. doi: 10.1109/access.2016.2582748.

Tajiki, M. M., Akbari, B., Shojafar, M. & Mokari, N. (2017). Joint QOS and congestion control based on traffic prediction in SDN. *Applied Sciences*, 7(12), 1265. doi: 10.3390/app7121265.

Tang, F., Fadlullah, Z. M., Mao, B. & Kato, N. (2018). An Intelligent Traffic Load Prediction-Based Adaptive Channel Assignment Algorithm in SDN-IoT: a deep learning approach. *IEEE Internet of Things Journal*, 5(6), 5141–5154. doi: 10.1109/jiot.2018.2838574.

Wang, Y., Jiang, D., Huo, L. & Zhao, Y. (2019). A new traffic prediction algorithm to software defined networking. *Mobile Networks and Applications*, 26(2), 716–725. doi: 10.1007/s11036-019-01423-3.

Wikipedia contributors. (2023). Machine learning. Retrieved from https://en.wikipedia.org/wiki/Machine_learning.

Yan, J., Zhang, H., Shuai, Q., Liu, B. & Guo, X. (2015). HiQoS: An SDN-based multipath QoS solution. *China Communications*, 12(5), 123–133. doi: 10.1109/cc.2015.7112035.

Yoonseon, H., Li, J., Chung, J. H., Yoo, J. & Hong, J.-W. (2015). SAVE: Energy-aware Virtual Data Center embedding and Traffic Engineering using SDN. *IEEE NetSoft*. doi: 10.1109/netsoft.2015.7116142.

LIST OF REFERENCES

mAuth1. (2001). mTit1. *mJour1*, 1(1), 42-43.

mAuth2. (2002). mTit2. *mJour2*, 2(2), 42-43.