A PROPOSED APPROACH NETWORK INTRUSION DETECTION SYSTEM (NIDS) USING DEEP LEARNING FOR SOFTWARE DEFINED NETWORK (SDN): A FUTURISTIC APPROACH

2022
MASTER THESIS
COMPUTER ENGINEERING

Mhmood Radhi HADI HADI

Thesis Advisor
Assist. Prof. Dr. Adnan Saher MOHAMMED

A PROPOSED APPROACH NETWORK INTRUSION DETECTION SYSTEM (NIDS) USING DEEP LEARNING FOR SOFTWARE DEFINED NETWORK (SDN): A FUTURISTIC APPROACH

Mhmood Radhi HADI HADI

T.C.
Karabük University
Institute of Graduate Programs
Department of Computer Engineering
Prepared as
Master Thesis

Thesis Advisor
Assist. Prof. Dr. Adnan Saher MOHAMMED

KARABUK

June 2022

I certify that in my opinion the thesis submitted by Mhmood Radhi Hadi Hadi titled "A PROPOSED APPROACH NETWORK INTRUSION DETECTION SYSTEM (NIDS) USING DEEP LEARNING FOR SOFTWARE DEFINED NETWORK (SDN): A FUTURISTIC APPROACH " is fully adequate in scope and in quality as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Adnan Saher MOHAMMED .....
Thesis Advisor, Department of Computer Engineering

This thesis is accepted by the examining committee with a unanimous vote in the Department of Computer Engineering as a Master of Science Thesis. June 21, 2021

Examining Committee Members (Institutions)	Signature
Chairman : Prof. Dr Fatih Vehbi ÇELEBİ (YBU)	.....
Member : Assist. Prof. Dr. Adnan Saher MOHAMMED (KBU)	.....
Member : Assist. Prof. Dr. Ferhat ATASOY (KBU)	.....

The degree of Master of Science by the thesis submitted is approved by the Administrative Board of the Institute of Graduate Programs, Karabük University.

Prof. Dr. Hasan SOLMAZ .....
Director of the Institute of Graduate Programs

"I declare that all the information within this thesis has been gathered and presented in accordance with academic regulations and ethical principles and I have according to the requirements of these regulations and principles cited all those which do not originate in this work as well."

Mhmood Radhi HADI HADI

ABSTRACT

M. Sc. Thesis

A PROPOSED APPROACH NETWORK INTRUSION DETECTION SYSTEM (NIDS) USING DEEP LEARNING FOR SOFTWARE DEFINED NETWORK (SDN): A FUTURISTIC APPROACH

Mhmood Radhi HADI HADI

Karabük University

Institute of Graduate Programs

The Department of Computer Engineering

Thesis Advisor:

Assist. Prof. Dr. Adnan Saher MOHAMMED

June 2022, 80 pages

Software defined networking (SDN) is considered one of the most promising solutions for evolving and modifying the architecture of traditional networks. The most notable features of SDN are many improvements and changes to the network and internet architecture. SDN primary solutions are flexibility and centralized control. Therefore, SDN is completely centralized. The centralization of SDN has created many vulnerabilities in this architecture that have made it a target for attackers. The controller is the brain of the SDN, and once it is attacked, the entire network will fall. One of the most significant topics to consider while considering SDN is attack protection. It is vital to put in place a powerful defense system capable of rapidly

detecting attacks. Intrusion detection systems are one of the most significant network security systems. We propose to use these approaches to build an SDN-specific intrusion detection system. The deep learning system has attracted researchers in recent years because of its efficiency. Our approach suggests using an intelligent network intrusion detection system (NIDS) which use deep learning to identify attacks that are trained with deep learning algorithms to achieve reliable and secure systems. We propose to use algorithms (DNN, CNN, RNN, GRU, LSTM) in our proposed approach, which is trained on 12 features extracted from 41 features in NSL-KDD dataset. The results show that the CNN algorithm achieves the highest accuracy, as well as other excellent algorithms with high accuracy. Our approach has been successful so it is expected that deep learning can be effectively used for SDN security in the future.

Keywords: Deep learning, Software Defined Network, SDN, Network Intrusion Detection System, NIDS.
Science Code : 92431

ÖZET

Yüksek Lisans Tezi

DERİN SİNİR AĞLARI İLE ZAMAN SERİLERİNİN SINIFLANDIRILMASI

Mhmood Radhi HADI HADI

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi. Adnan Saher MOHAMMED
Haziran 2022, 80 sayfa

Yazılım tanımlı ağ iletişimi (SDN), geleneksel ağların mimarisini geliştirmek ve değiştirmek için en umut verici çözümlerden biri olarak kabul edilir. SDN en dikkat çekici özellikleri, ağ ve internet mimarisinde yapılan birçok iyileştirme ve değişikliktir. SDN birincil çözümleri esneklik ve merkezi kontroldür. Bu nedenle, SDN tamamen merkezileştirilmiştir. SDN merkezileştirilmesi, bu mimaride onu saldırganlar için bir hedef haline getiren birçok güvenlik açığı yarattı. Kontrolör, SDN beynidir ve bir kez saldırıya uğradığında tüm ağ çökecektir. SDN düşünürken dikkate alınması gereken en önemli konulardan biri saldırı korumasıdır. Saldırıları hızla tespit edebilen güçlü bir savunma sistemini devreye sokmak hayati önem taşımaktadır. Saldırı tespit sistemleri, en önemli ağ güvenlik sistemlerinden biridir. SDN özel bir saldırı tespit sistemi oluşturmak için bu yaklaşımları kullanmayı öneriyoruz. Derin öğrenme sistemi, verimliliği nedeniyle son yıllarda araştırmacıları cezbetmiştir.

Yaklaşımımız, güvenilir ve güvenli sistemler elde etmek için derin öğrenme algoritmalarıyla eğitilen saldırıları belirlemek için derin öğrenmeyi kullanan akıllı bir ağ saldırı tespit sistemi (NIDS) kullanılmasını önerir. NSL-KDD veri setindeki 41 öznitelikten çıkarılan 12 öznitelik üzerinde eğitilen önerilen yaklaşımımızda algoritmaların (DNN, CNN, RNN, GRU, LSTM) kullanılmasını öneriyoruz. Sonuçlar, CNN algoritmasının en yüksek doğruluğa ulaştığını ve kalan algoritmaların çok yüksek performans ve iyi, yakın doğrulukla elde ettiğini göstermektedir. Yaklaşımımız o kadar başarılı oldu ki, gelecekte derin öğrenmenin SDN güvenliği için etkin bir şekilde kullanılabileceği tahmin ediliyor.

Anahtar Kelimeler: Derin öğrenme, Yazılım Tanımlı Ağ, SDN, Ağ Saldırı Tespit Sistemi, NIDS.
Bilim Kodu: 92431

ACKNOWLEDGMENT

Firstly, I thank Allah deeply in my heart and Imam Sahib zaman. I also thank my advisor, Assist. Prof. Dr. ADNAN SAHER MOHAMMED, for his limitless contribution to the preparation of this research. I would like to show my gratitude to my family, who stood by my side throughout my journey, especially my mother for their limitless contribution

CONTENTS

Page

APPROVAL..... ii
ABSTRACT.....iv
ÖZET.....vi
ACKNOWLEDGMENT.....viii
CONTENTS.....ix
LIST OF FIGURES xii
LIST OF TABLES.....xiii
ABBREVIATIONS xiv

PART 1 1
INTRODUCTION 1
1.1. MOTIVATION 3
1.2. PROBLEM STATEMENT 5
1.3. AIMS OF THE STUDY..... 6
1.4. THESIS STRUCTURE 6

PART 2 8
BACKGROUND 8
2.1. SOFTWARE DEFINED NETWORK 8
2.1.1. HISTORY OF SDN..... 9
2.1.2. ARCHITECTURE OF SDN 12
2.1.3. OPENFLOW PROTOCOL 14
2.1.4. SDN CONTROLLERS 17
2.1.4.1. NOX CONTROLLER 18
2.1.4.2. POX CONTROLLER..... 18
2.2. INTRUSION DETECTION SYSTEM 19

Page

2.2.1. INTRUSION DETECTION SYSTEM BASED SIGNATURE DETECTION 20
2.2.2. INTRUSION DETECTION SYSTEM BASED ANOMALY DETECTION 21
2.2.3. INTRUSION DETECTION SYSTEM EVALUATION METRICS 21
2.3. DEEP LEARNING 23
2.3.1. DEEP NEURAL NETWORK 23
2.3.2. CONVOLUTIONAL NEURAL NETWORK 24
2.3.3. RECURRENT NEURAL NETWORK 24
2.3.4. LONG SHORT-TERM MEMORY 26
2.3.5. GATED RECURRENT NEURAL NETWORK 27

PART 3 29
RELATED WORKS 29
2.1. METHODS NON-BASED MACHINE LEARNING 29
2.2. METHODS BASED MACHINE LEARNING. 30
2.2.1. MACHINE LEARNING ALGORITHM BASED INTRUSION DETECTION SYSTEM 31
2.2.1. Intrusion Detection System in Deep Learning 32
2.2.2. MACHINE LEARNING BASED IDS FOR SDN 34
2.2.3. DEEP LEARNING BASED IDS FOR SDN 36
2.3. SUMMARY OF CHAPTER 37

PART 4 39
METHODOLOGY 39
3.1. INTRODUCTION 39
3.2. PROPOSED METHODOLOGY 40
3.3. DEEP LEARNING ALGORITHMS 41
3.3.1. Deep Neural Network (DNN) 41
3.3.2. Convolutional Neural Network (CNN) 44
3.3.3. Recurrent Neural Network (RNN) 46
3.3.4. Gated Recurrent Neural Network (GRU) 48

Page

3.3.5. Long-Short Term Memory (LSTM) 48
3.4. DATASET 50
3.4.1. Exploration Data Analysis 50
3.4.2. Preprocessing 53
3.4.3. Feature encoder (One-Hot Encoding) 53
3.4.4. Normalization 54
3.4.5. Split Data 55
3.4.6. Stage of Evaluation Metrics 55
3.4.7. Chapter Summary 56

PART 5 57
RESULTS AND DISCUSSION 57
4.1. INTRODUCTION 57
4.2. DATASET 57
4.3. FEATURE SELECTION 59
4.4. NUMERICALIZATION 60
4.5. NORMALIZATION 61
4.6. ANORMALY DETECTION 62
4.7. IMPLEMENTATION MODEL CLASSIFICATION 62
4.7.1. DNN CLASSIFIER 62
4.7.2. CNN CLASSIFIER 64
4.7.3. RNN CLASSIFIER 65
4.7.4. LSTM CLASSIFER 67
4.7.5. GRU CLASSIFER 68
4.8. RESULTS OF EXPERIMENT 70

PART 6 73
CONCLUSION AND FUTURE WORKS 73
5.1. CONCLUSION 73
5.2. FUTURE WORKS 74
REFERENCES 75
RESUME 75

LIST OF FIGURES

Page

Figure 1.1 Network intrusion detection system (NIDS) [6]..... 2
Figure 2.1. Active Node Architecture…….10
Figure 2.2. Control and routing protocol (RPC). 11
Figure 2.3. Traditional network. 13
Figure 3.1. Proposed methodology system for NIDS with DL for SDN……41
Figure 3.2. Summary Architecture of DNN model in NSL-KDD dataset..... 43
Figure 3.3. Summary Architecture of CNN model in NSL-KDD dataset 45
Figure 3.4. Summary Architecture of RNN model in NSL-KDD dataset 47
Figure 3.5. Summary Architecture of GRU model in NSL-KDD dataset 48
Figure 3.6. Summary Architecture of LSTM model in NSL-KDD dataset..... 49
Figure 3.7. Splitting NSL-KDD dataset..... 55
Figure 4.1. Confusion Matrix for applied DNN on NSL-KDD dataset…….63
Figure 4.2. ROC Curve for DNN classifer..... 63
Figure 4.3. Confusion Matrix for applied CNN on NSL-KDD Dataset. 64
Figure 4.4. ROC Curve for CNN classifer..... 65
Figure 4.5. Confusion Matrix for applied RNN on NSL-KDD Dataset. 66
Figure 4.6. ROC Curve for RNN classifer..... 67
Figure 4.7. Confusion Matrix for applied LSTM on NSL-KDD Dataset. 67
Figure 4.8. ROC Curve for LSTM classifer..... 68
Figure 4.9. Confusion Matrix for applied GRU on NSL-KDD Dataset. 69
Figure 4.10. ROC Curve for GRU classifer..... 70
Figure 4.11. Result of deep learning algorithms with metrics classification..... 72

LIST OF TABLES

Page

Table 2.1. Summary IDS with Machine Learning. 32
Table 2.2. Summary IDS with Deep Learning. 33
Table 2.3. Summary IDS with ML for SDN. 35
Table 2.4. Summary IDS with DL for SDN. 37
Table 3.1. Features with description of NSL-KDD Dataset.....50
Table 3.2. Classification Attack Categories of NSL-KDD Dataset. 53
Table 3.3. One-Hot Encoding Mechanism. 54
Table 3.4. Confusion Matrix (CM). 56
Table 4.1. NSL-KDD dataset with six rows.....58
Table 4.2. Features extracted from NSL-KDD Dataset. 59
Table 4.3. Categorical feature in NSL-KDD dataset. 60
Table 4.4. Categroical features with One-Hot Encoding. 61
Table 4.5. Normalization of NSL-KDD dataset with 4 rows..... 61
Table 4.6. Result of Metrics applied DNN on NSL-KDD Dataset. 62
Table 4.7. Result & Metrics applied CNN on NSL-KDD Dataset. 64
Table 4.8. Result & Metrics applied RNN on NSL-KDD Dataset. 66
Table 4.9. Result & Metrics applied LSTM on NSL-KDD Dataset. 68
Table 4.10. Result & Metrics applied GRU on NSL-KDD Dataset. 69
Table 4.11. Result of deep learning algorithms with Evaluation metrics 71
Table 4.12. Result of Confusion Matrix in metrics of deep learning algorithms..... 72

ABBREVIATIONS

SDN	: Software Defined Network
DL	: Deep Learning
ML	: Machine Learning
OF	: Open Flow
DOS	: Denial Of Service
IDS	: Intrusion Detection System
NIDS	: Network Intrusion Detection System
DNN	: Deep Neural Network
RNN	: Recurrent Neural Network
CNN	: Convolutional Neural Network
GRU	: Gated Recurrent Unit
LSTM	: Long Short-Term Memory
ACC	: Accuracy
SVM	: Support Vector Machine
AI	: Artificial Intelligence
TP	: True Positive
FP	: False Positive

PART 1

INTRODUCTION

Software-defined networks are one of the emerging services that provide promising solutions to the infrastructure of the internet, as traditional networks no longer meet the continuous increase and flexibility required in the size of the network with the speed and development of many technologies based on the infrastructure of the Internet. Although the structure of SDN networks is considered promising for the world of networks. , that it suffers from security problems and major flaws that make it the focus of many attacks, network targets, and hackers [1, 2].

Therefore, one of the biggest flaws in the SDN network is the presence of one control over the entire network, and its exposure to attack means the destruction of the network and the fall of the entire service from the network. Therefore, it is very necessary to think of an intrusion detection system whose location is in the controller, where it controls the entire network, manages, monitors and sends alerts to the network administrator when there is a defect or the possibility of being attacked, or an intrusion is discovered. We can also talk about intrusion detection systems and their history and development history. Researchers have used anti-attack protection systems for decades to protect their systems from intrusion as well as protect the privacy of their data. One form of protection is firewalled from attacks as well as the development of strong encryption systems [3].

One form of protection method is intrusion detection system against attacks, where complete protection of the system against attacks is provided through IDS [4]. The foundation stone for the first intrusion detection system was discovered and laid in the period between 1984 and 1986 by designing the so-called (IDES) known as Expert Intrusion Detection Systems [5].

Where this approach assumes that the attacker's style and movement in the network differ from the free and normal person's way of using network resources or the nature of the traffic through statistics. This approach shows that the principles of mathematics and statistics were used with computer applications, software, and operating systems, and at the end of the 1980s this thing led to the merging of these technologies with each other to reach a single system that works together.

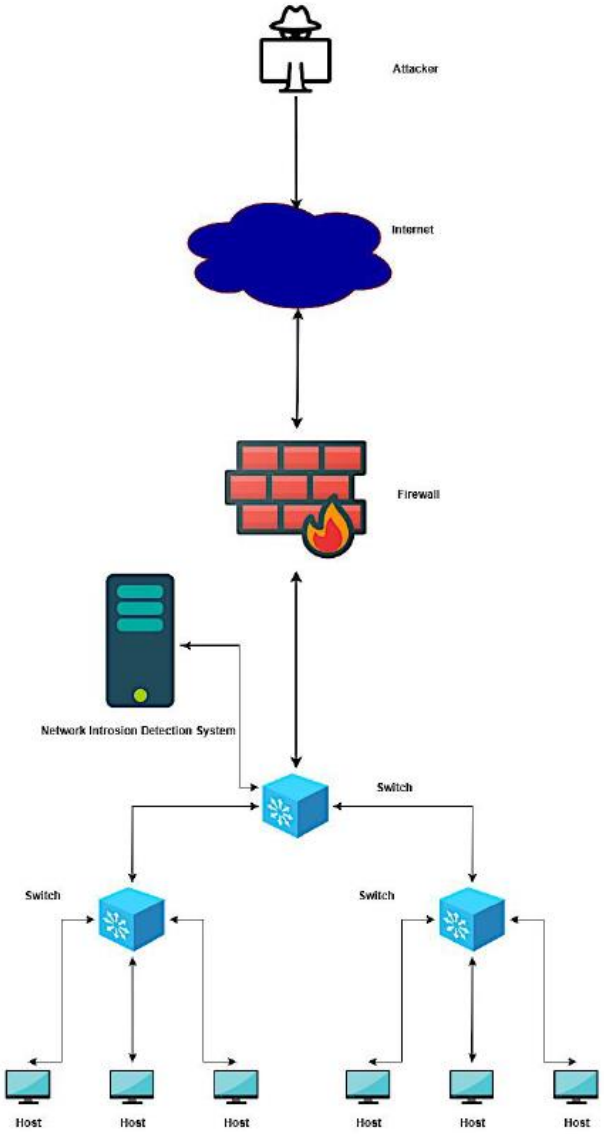

Figure 1.1 Network intrusion detection system (NIDS) [6].

Intrusion detection systems are referred to as systems that can identify attacks through the use of algorithms dedicated to this performance, through which unusual free traffic is detected or the incorrect pattern is recognized through intrusion detection. Report suspicious activity in the network where this system is called NIDS [7].

The development of methods of penetration and the development of tools for hackers, has led to the urgent need to develop intrusion detection and protection systems in general to provide protection and privacy for the data of institutions and government and private sectors [8]. Where it is possible to protect the organization from penetration or exposure to attacks or reduce the impact of attacks through continuous development of Intrusion Detection Systems [6].

Therefore, we focused in this thesis on the design of an intrusion and anomaly detection system whose location is in the controller part of the network for SDN, this system is intelligent and called (NIDS) is designed and evaluated based on deep learning algorithms and compared the results of these algorithms with each other, given the power of deep learning in developing Strong systems capable of dealing with various types of attacks and data.

1.1. MOTIVATION

The (NIDS) development sector notices a significant and remarkable development due to the tendency of many different institutions and governments to protect their data and protect their infrastructure. This increase in development is offset by an increase in the number of electronic attacks and the development of penetration tools. The most prominent and most prevalent attacks are Denial of Service or Worms and malware.

NIDS is designed to detect these and many other attacks such as worm attacks and malware attacks. Designing an intrusion detection system that has reliable, accurate performance and detection speed is one of the necessary characteristics and advantages when considering an effective intrusion detection system against attacks.

Many different approaches that applied machine learning techniques in designing and building intelligent intrusion detection systems assumed that by using deep learning methods, more reliable, robust, capable, and capable systems would be built to detect numerous attacks and assumed that this technology could be applied to protect newly emerging technologies such as IoT and SDN technologies [9]. Therefore, in these approaches, we assume the construction of detection systems with deep learning methods for SDN, so that in the future these approaches are used within the SDN architecture and applied to it.

Since this paper is concerned with introducing (NIDS) approach using deep learning techniques for SDN, it can be said that one of the most important incentives to protect (SDN) is the controller who is in control of the entire network performance, which in turn is considered this great for hackers to bring down the entire network and apply different attack techniques on this new networking infrastructure. So, controller security is very important and something that must be provided [10, 11].

Many traditional network-based NIDS have been developed, which perform anomaly detection based on incident detection and each task has drawbacks. Zero-day attacks cannot be detected by a signature-based approach [10]. Anomaly-based IDS can also contain errors that may be tolerated in traditional networks, but in SDN, for example, if an attack is assumed that the system has been given a (false negative) warning and assumes that it is a natural attack, it leads to the destruction of the entire network, and they fall under an undetected attack. The architecture of traditional networks may not be affected as much as the SDN architecture, especially attacks that target the controller are considered to be very harmful.

Therefore, continuous improvements to the NIDS design are robust and effective in attack detection, which is critical to enhancing the security of this architecture.

Many methods have tried to use and apply applications for scanning and protection from within the mobile, but they still suffer from several problems and restrictions, as they must protect the network, but the attack or threat occurs, and this request leads to complete information of the network from the console. As he must have statistics and

raw data to use with his (IDS) and this means that the attack took place in the first place, there is also a challenge that the controller's location makes it vulnerable to various attacks that require modern technologies, including a rapid communication mechanism between the controller and NIDS. Able to use the sophisticated and accurate detection techniques used within (NIDS) systems, which are lacking in simple protection applications.

1.2. PROBLEM STATEMENT

The development of (IDS) passed through several stages, the most important of which was the discovery of attacks and determining the type of attack that took place and this development continued for long periods. With the development of data volume and the increase of attacks of various types, the traditional methods became unhelpful and unhelpful, as they were unable to address the huge development in the size of the data, and the modern technologies used, including machine learning techniques, no longer achieve the required accuracy, high performance, or rapid detection to obtain a reliable (NIDS) system capable of dealing with large amounts of data. Machine learning techniques failed to classify five types of attacks out of ten types. From attacks, and this in itself is a defect and a big problem that must be dealt with if it is required to provide a modern network protection system such as the SDN [12].

Modern and powerful algorithms such as (DNN, CNN, GRU, RNN, and LSTM), which are considered the most powerful, performance, efficient, and capable of dealing with data in terms of self-processing and effective performance, as well as flexible dealing with large and large amounts of data. As the most prominent methods currently used are machine learning methods It has become useless, so it is necessary to quickly shift towards deep learning methods in the application and creation of intrusion detection systems capable of detecting various attacks and built using big data to be strong in performance and high reliability. We hypothesized the use and evaluation of the performance of five types of famous deep learning algorithms Which leads to higher quality results than using a single type of algorithm.

1.3. AIMS OF THE STUDY

Represents the main objectives in our approach to the application of anomaly detection systems using deep learning techniques in:

- Making a study and clarification of the intrusion detection systems in the traditional structure and an attempt to study the approach and its application of SDN.
- The approach consists of several powerful deep learning algorithms such as (DNN, CNN, RNN, GRU, LSTM) to be used in building the detection system and its application to detect anomalies in the dataset.
- Evaluating the possibility of applying the (NIDS) approach in the SDN network and the most significant things that help to apply it in the future.
- Making an evaluation of the results, and this is considered the main objective in this study, which is an evaluation study that measures the performance of these algorithms and the efficiency of each type and its difference from the other type, where performance measures are used such as (Accuracy, Precision, Recall, F1-Score).

1.4. THESIS STRUCTURE

Part 1. Introduction: Contains the background, motivation, and problem statement. It also describes the aims of this study with a thesis structure.

Part 2. Background: Surveys A description of the program-defined networking was discussed, a historical review of SDN was conducted, and the SDN controller and SDN protocol, as well as NIDS and Deep learning classifiers, are used in our approach are related developments were covered.

Part 3. Related Work: This section described a Survey of previous security SDN methods using machine learning with NIDS using deep learning. In addition, a comprehensive review of these studies was conducted to discuss the author's findings.

Part 4. Methodology: This section provides a detailed description of the approach method used in this study, the deep learning classifiers used, the method used to process the dataset, and the contributions to this study.

Part 5. Results and discussion: We will explain the results obtained by our approach in terms of the results of the dataset and the results of various deep learning classifiers.

Part 6. Conclusion and future work: This section presents the results in a summary format and shows what research and future approaches to consider.

PART 2

BACKGROUND

2.1. SOFTWARE DEFINED NETWORK

The Internet's development in our day has resulted in a spectacular rise of network technologies and adherence, as it has benefited the lives of many people in various ways, including social, service, and economic aspects, as well as technologically [7]. There have been many changes and developments in the private infrastructure in networks where it is no longer what it used to be and the same services that were offered in the past have not become the same, and besides, the number of private users in the network has recently increased. Also, services have become network-provided why not like before because traditional networks depend on specialized algorithms in those networks specialized to work inside devices Because there is knowledge of dedicated architecture and network data path routing algorithms that work exclusively for the network [8]. It is often difficult to control and manage an entire network because an algorithm is a specific piece of hardware or a network, so a computer network is made up of complex algorithms and the network itself is complex [9]. Network systems are often less flexible and require the network administrator's effort and preparation when preparing equipment. Although some vendors of these devices provide providers with centrally managed services, their work is limited to configuring and managing their own mechanisms and protocols [9]. These problems have hindered the growth and development of the network for decades, and with that, they do not contribute to reducing the complexity of the network because there is a backlog of solving many of the problems. problems that the network must deal with, such as delays in processing intrusion detection or analysis of communication patterns. In addition to the development of many network tools, it is necessary to radically improve the structure of the network and reduce these defects as much as possible [10]. Currently, networks are characterized as mature in such a way that it is difficult to develop and change their infrastructure because it is often difficult to change IP

networks. This structure provides more evolving needs for easier network management [9]. The SDN network is a network that has made a breakthrough in the world of traditional networks as it provided a separation between the control layer from the data layer, where the control layer monitors the general performance and full control of the network, which means that the SDN network is a dynamic network and its structure is subject to change based on the requirements of the required design, so it can be considered It paved the way to solve many problems that were considered complex in networks by providing the central infrastructure and monitoring the general behavior of the network [9]. The operating principle of an SDN network is based on a centralized approach to control that facilitates complete control of the network through the control level.

The control layer is defined by its functionality by handling the full range of switches, routers, and middleware boxes, and each data control level has an application programming interface (API) for receiving network administrator's rent and directing them to the controller [11]. Since SDN works on the principle of separating the control plane from the data plane, it allows us to innovate back into this network's infrastructure through a software application interface. Where it is possible to program the console and send its instructions, as this allows the operator to upgrade the infrastructure and facilitate its management.

2.1.1. HISTORY OF SDN

The concept and terminology of SDN aim to make the network environment easy to control and manage in its components. However, the evolution towards SDN started with programmable and controllable networks where it could go until the mid-90s when the first programmable networks started to control the functions of the network. them programmatically in the network [10]. During the 1990s, Internet service was gradually increasing in speed, as it attracted the attention of researchers, interested people, and developers, and paved the way for the launch of many efforts. serious development, as a programming-based approach. emerged as the Active Network. Where this study acts as an alternative to traditional network systems, as it opens up the possibility of story-driven storytelling of programmatically defined interfaces.

Active networks have been studied to identify the constituents and their guests regardless of the comprehensive services they provide. The goal of this technology is to pave the way for comprehensive services through the proper preparation of nodes. [13]. As we can see in Figure 2.1. show a working node architecture referring to the components and nodes that make up this network, where each node can verify and deploy separate environments or multiple environments (EEs) where users can against these active applications (AA) enable end-to-end service expression or more through (EE).

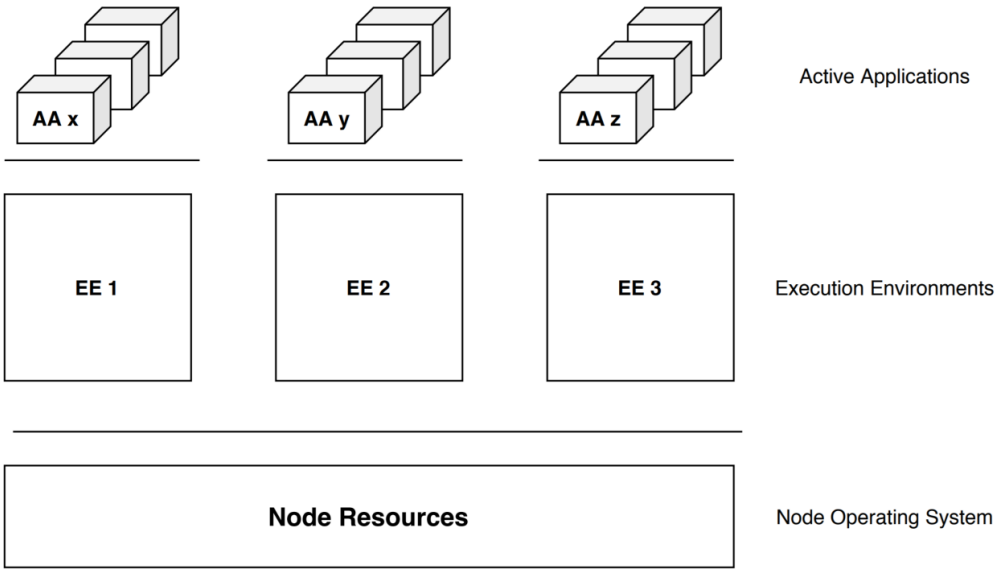

Figure 2.1. Active Node Architecture.

Increasing the number of network users has led to network growth and increased traffic, with more emphasis on network reliability. This was the beginning of the 21st century, and we needed to focus on controlling some of the assigned network functions. B. Concentrated data traffic. This is difficult because the structure of these networks is so closely related. One of the reasons that led to the separation of the control layer and the data layer raised the issue of controlling routing and data movement behavior [10]. Two approaches were devised as RPC (Routing Control Platform) [15] dedicated for making centralized control and for CES (Forwarding and Control Element Separation) for the data plane [14]. Figure 2.2 shows Control and routing protocol (RPC). The concept and term RCP appeared specifically to solve the

problem of routing control between domains for addressing networks where they were identified through the exploration of the potential layouts for the centralized control system. A central server is also provided with the BGP protocol. We get high flexibility and better control for network operators to control the routing between the tracks in addition to flexibility and higher control with network operators [16]. In summary, numerous services based on the RCP protocol were available, including picking the best route, determining the network level and policy level, and re-defining the pathways between the AS (inter-routing). This strategy concentrates on the strong side and the ability of this platform to grow transiently to face obstacles since it has the qualities and advantages of a single point control [16].

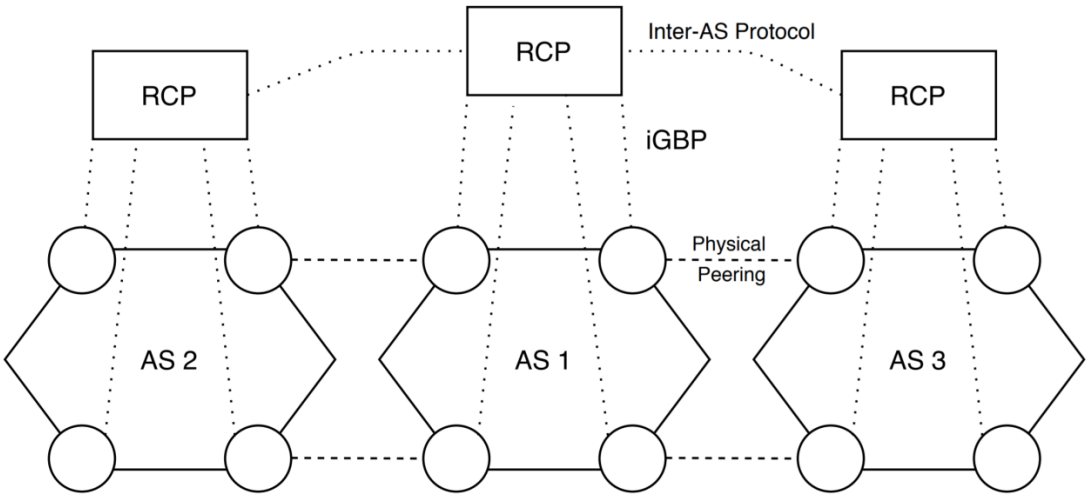

Figure 2.2. Control and routing protocol (RPC).

Like any new development process or innovative technology, technologies that attempt to change network infrastructure have been criticized. In particular, many researchers believe that the idea of separating the control layer and the data layer is undesirable, as many researchers believe that this method is not a building method at the required level or that it is a strong building method that can be relied on in the future. Extensive research into the achievable insights and the results obtained after the process of separating the control layer from the data has uncovered a robust and scalable structure based on the control layer. For example, you can mention a (4D) 4D project. The project consists of four layers: publishing, decision making, data control,

and discovery, each layer with specific features and levels. Bitumen contains logic that controls the width of the level's structure. For networks, the discovery layer's function is to act both as a data provider and as a leader at the control level of the data layer. Also, the entire data layer is at the network level, as 4D projects have many advantages. For example, there are many benefits such as higher capacity, scalability, innovation and infrastructure development, better and stronger security. [17]. One of the successful projects [18]. The Ethan project is considered a breakthrough in the world of programmatically defined networking, through which the OpenFlow protocol, one of the most famous in the world (SDN), has been released and operational. In addition, the traditional OpenFlow protocol has provided an open-source platform and platform for developers through which to create flow tables for controllers and switches [19].

2.1.2. ARCHITECTURE OF SDN

The traditional network consists of three branches or levels and each layer has a dedicated function, that is the management, control, and navigation layer, where the control level provides control over the data to form tables according to the requirements. purpose redirection, used by redirection to direct packets to ingress and egress ports as well as the administrative layer or layer Its function is to provide network configuration and performance monitoring services. Similarly, it can also be observed that in traditional networks, the routing and control layers are combined in a single device such as (routers and switches), but with an increase in the size of users. network usage and the need for a faster connection. and more and more services extend, this becomes an inefficient model in terms of operations, so infrastructure needs to be developed.

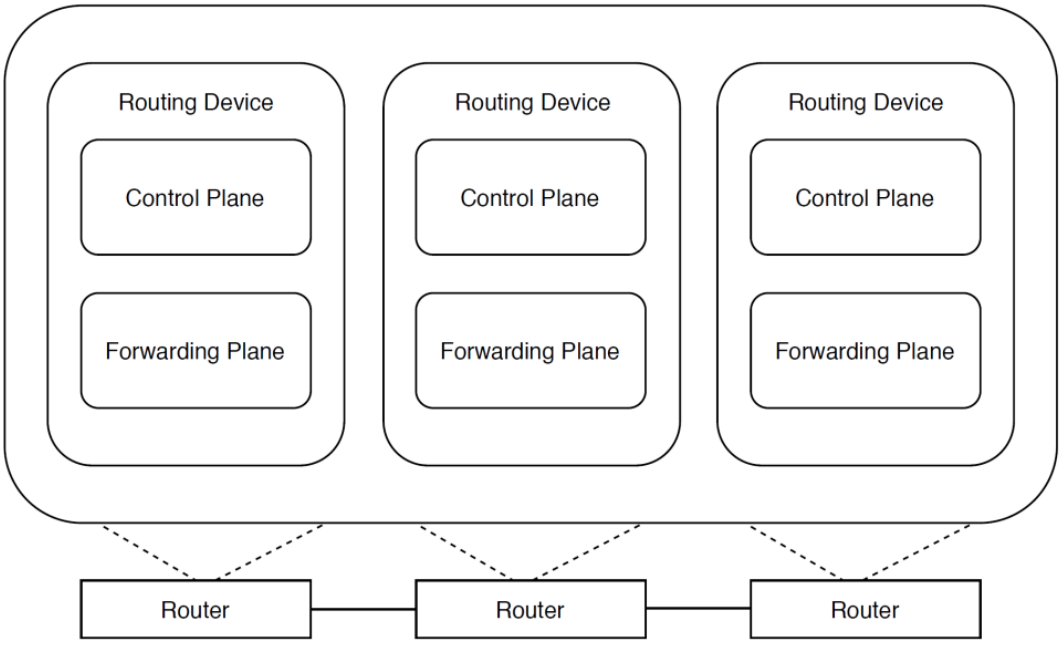

Figure 2.3. Traditional network.

Figure 2.3 presents a unique topology for a typical network, in which the control and routing layers are combined. Because traditional infrastructure is analogous to the application of a single integrated circuit and a dedicated wire is subject to change, this structure is fully implemented using dedicated devices consisting of more than one switch and one router, as the traditional infrastructure is analogous to the application of a single integrated circuit and a dedicated wire is subject to change, resulting in the formation of a network limited by many functions. In Figure 2.4, we can see the infrastructure dedicated to the SDN network with which we can expertly view the content of the SDN from the three layers. The control layers and the application layer and the infrastructure layer where all the layers are interconnected (API) and the control layers connect to the southern API to the infrastructure layer and connect the control layers' control with app d through a third API.

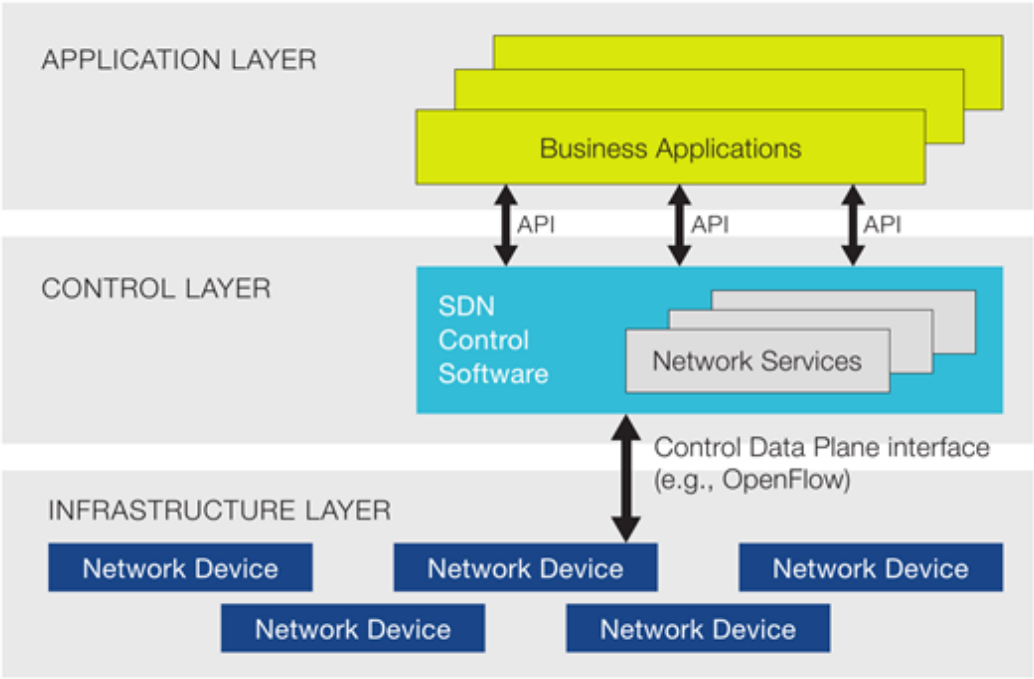

Figure 2.4. Software defined network architecture.

2.1.3. OPENFLOW PROTOCOL

In an SDN, there are many types of protocols dedicated to the routing and processing of elements and data in the network. The most commonly used is OpenFlow (OF), so we will discuss it in detail. Figure 2.5 shows this protocol is dedicated to communication because it acts as a data interface through a dedicated application interface (API). There are two interfaces, a south window allows communication from the control unit to the switch. In addition, control class rules are managed by the Open Organization Network (ONF) [29], invented at Stanford University. One of the most important features of this protocol is that it is open source, allowing multiple control devices from multiple vendors to be connected at the same time, and everything is managed by the OF protocol. (OF) is the process of managing the communication between a controller and a switch, as it is done by defining rules and definitions, as well as exchanging packets and data, and controlling the Messages transmitted over secure channels in (OF). Where it handles the switches with the routing table and does the data conversion, sending the routing table to the controller and sending the data back to the switch, through a dedicated application interface (API) includes the

connection because it is completely dependent on the protocol (OF) to complete the communication.

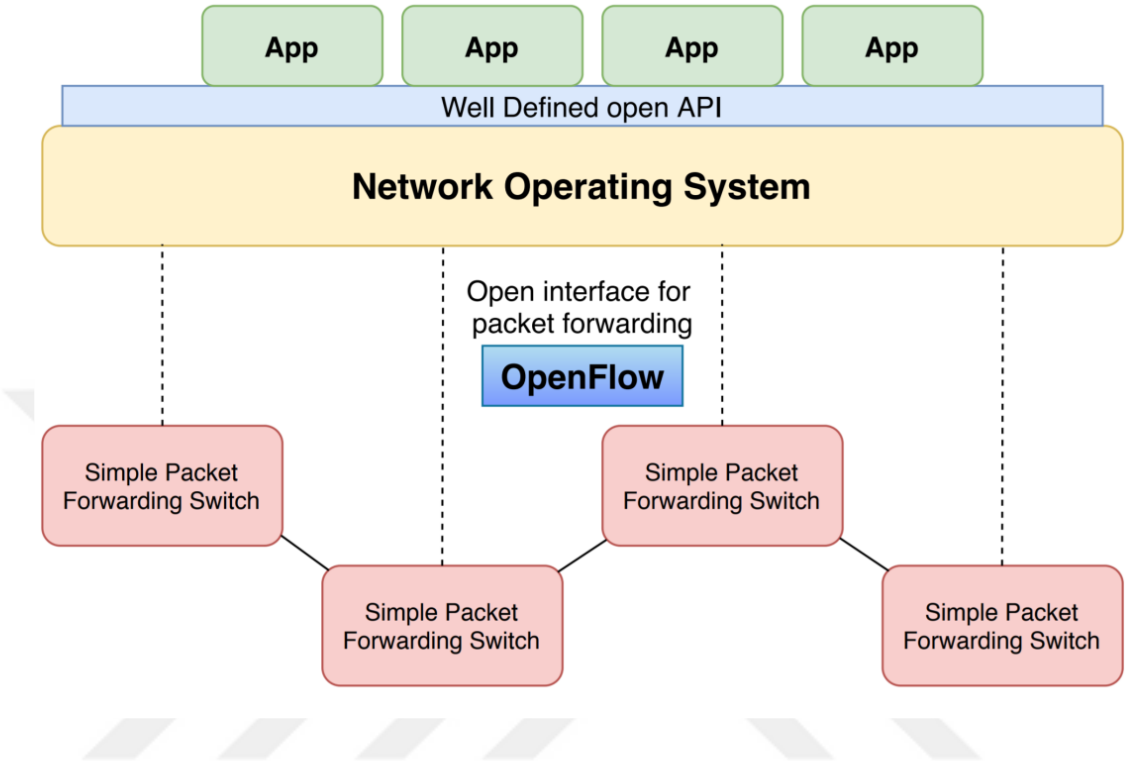

Figure 2.5. OpenFlow Protocol (OF).

As already mentioned, the (OF) protocol is responsible for the communication process between the control layer and the data layer. Figure 2.6 shows the main components of this protocol and some of the main functions that each part performs. The flow table through the assigned port and the protocol (OF) that allows the control layer to communicate with the switch layer is also in the (OF) entry in the routing table and sent to the switch. The table entries are reviewed for consistency. Match fields, summary fields, and instruction sets that are executed in the package are among the entries. The custom action is then performed after a comparison with the new package. The packet is sent and sent to the control layer through a secure connection after the switch receives it through the table of entries. In addition, by following the basic methods for incoming packets, packets can be modified and delivered. The packet is encapsulated and transferred to the control layer via the connection after the packet

forwarding process. Security can be adjusted before being sent, or packets can be dropped and routed to a shared translation path line.

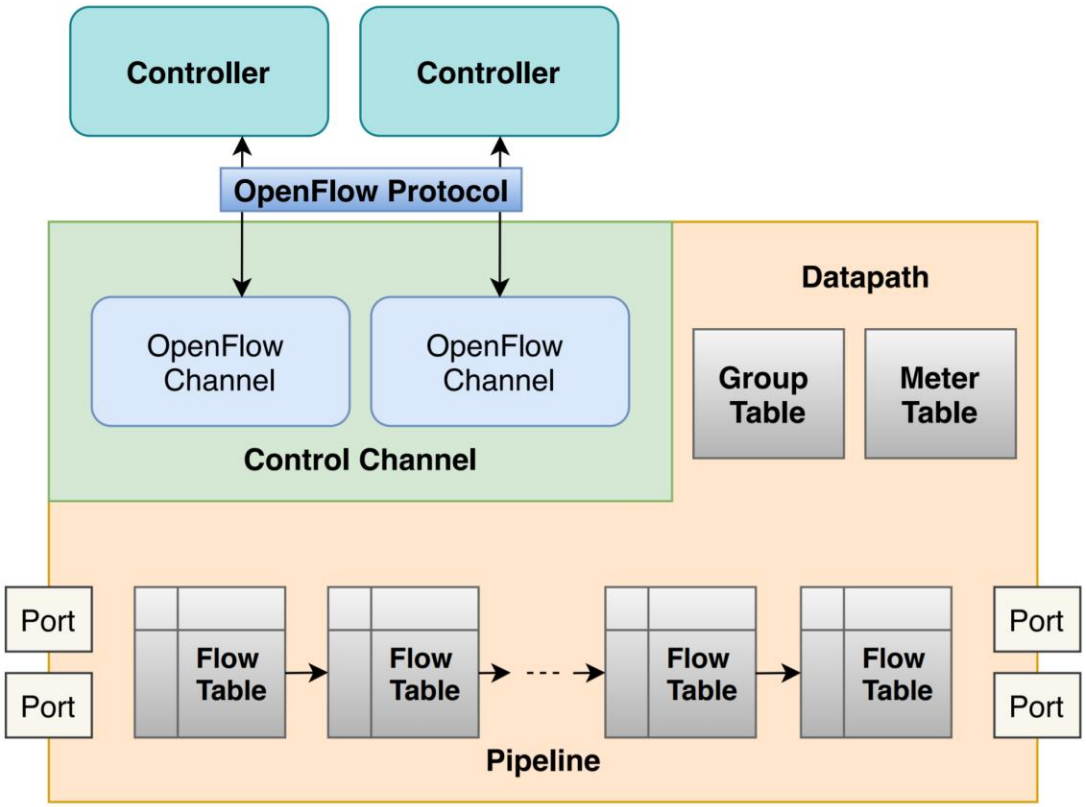

Figure 2.6. Characteristic switch of OpenFlow.

In addition to the data layer, the structure of a network (SDN) can be represented by three main layers: the control layer that controls the entire network and satisfies the presence of the controller, which is the control spirit of the network (SDN). Data layer, and the secure bearer channel that connects this network. Layers (controllers and switches). A secure and reliable method of communication using SSL and TSL cryptographic protocols is used for communication between the controller and the switch. The communication process is carried out by implementing security measures to protect the connection from various attacks, and as is known, the communication process between the switch and the controller is carried out using the (OF) protocol. Sends data and requires a secure communication path. This protocol was released in 2007, and its first official version, OpenFlow 1.0, was released in 2019. Later, an additional update was issued that added the routing table to the new version titled

OpenFlow 1.1, and a new version was also created. New popular title, OpenFlow 13. Until the latest version was released, it was OpenFlow 15.1 from (ONF) company [31].

2.1.4. SDN CONTROLLERS

As previously noted, the control unit is the brain of the network, controlling all of the various functions through a variety of functions and internal signals that are exchanged with the network's lower levels. Also, in the process of transmitting instructions, the (OF) protocol is the most prevalent and used effect. Data transformations and flow. In addition to managing a number of essential responsibilities. Figure 2.7 illustrates the controller's internal structure and core operations, which include multiple process and package controllers, the data processing unit, and protocol implementation units (OF), as well as many extra components and management units for system components.

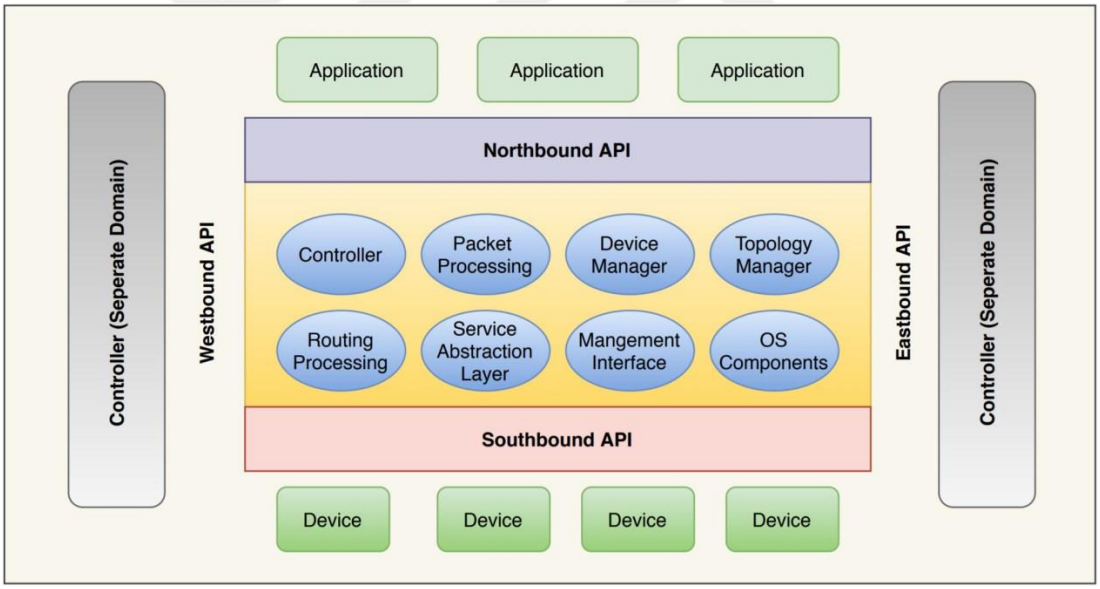

Figure 2.7. The core of controller in SDN.

There are many microcontrollers used in (SDN) networks, some of which are dedicated to the enterprise or stakeholder, some of which are dedicated to the field of study, and they are distinguished by the programming language used. Perhaps the most well-known of these used in research are the NOX compiler and (POX) controllers, in addition to the Ryu controller. These controllers differ from each other depending on

the library used, the programming method, and ease of use, which is one of the main features of use.

2.1.4.1. NOX CONTROLLER

NOX protocol stands for (Network Operating System). It is also considered the first console designed and developed by Nesria [33]. It is also considered that this protocol is developed and translated on the basis of the problem-based programming model, in addition, it is also the basic controller of the protocol (OF), so this protocol is considered to be the basis of the innovation network (SDN).

2.1.4.2. POX CONTROLLER

This type of control is open source because it is written in the Python [18] language. This is the kind of version with improved performance and developed features. This controller uses the (OpenFlow) protocol in the process of message forwarding, data forwarding, and communication between the controller and the switch. The main feature of this microcontroller is that it does not require a large amount of memory, but it is not efficient. It also supports a graphics interface and an OFV1 switch [19]. Due to the fact that the (POX) console supports (Python), a (Mininet) emulator is used to implement and ease of use with the environment Occasion. Figure 2.8 that showing architecture of POX controller.

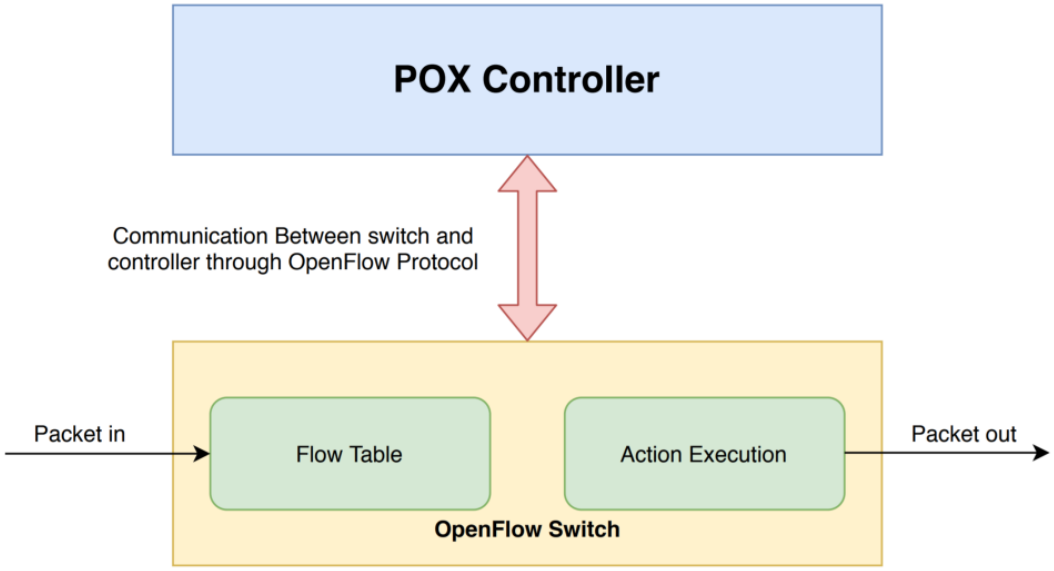

Figure 2.8. POX Controller.

2.2. INTRUSION DETECTION SYSTEM

Active attacks are characterized by unwanted data access or network infiltration 47. The majority of attacks and intrusions aim to destroy network infrastructure or steal data. Intrusion attacks have increased in recent years as the number of Internet users has grown, increasing the number of attacks and a wider range of attack tactics. Intrusion detection is based on looking for any anomalies in network traffic by monitoring network behavior and performance. In network systems, abnormal behavior is identified using specific probes [12]. One such system dedicated to detecting anomalous behavior is called (IDS) [13]. There are different types of IDS systems, which can be anomalous or signature-based systems. Figure 2.9 that shown IDS.

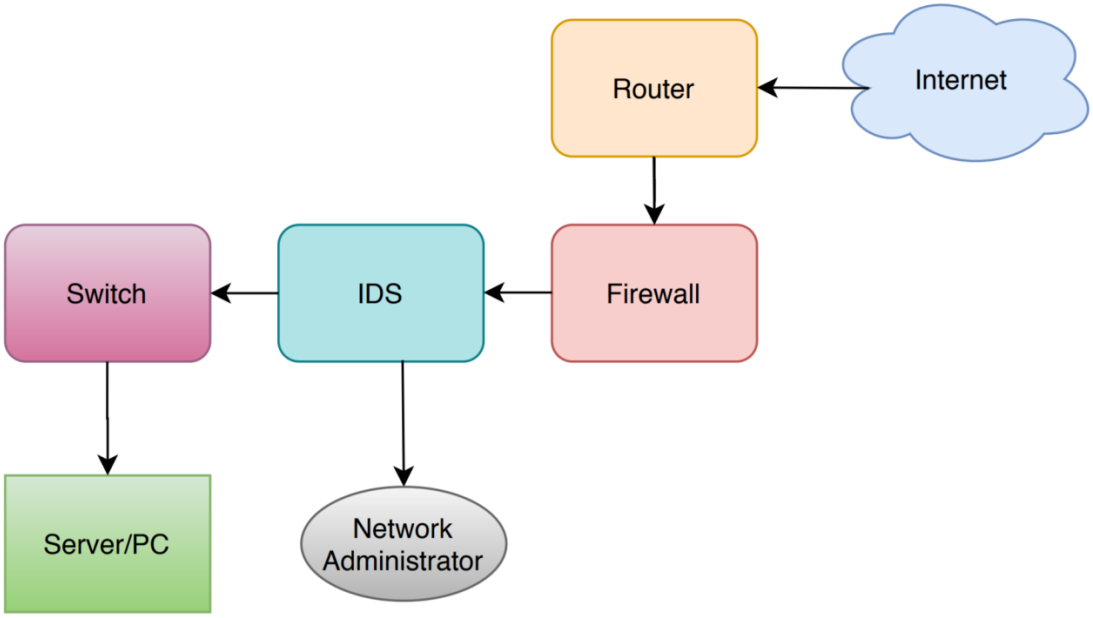

Figure 2.9. Workflow intrusion detection system.

2.2.1. INTRUSION DETECTION SYSTEM BASED SIGNATURE DETECTION

The signature-based approach relies on comparing previously stored attack patterns with existing network patterns. This type of attack detection depends on network traffic and works in a match pattern. The simplest example of this type of approach is short noses only rescue team (SNORT) [22], which is a signature-based in-process detection system. This method makes advantage of large databases that contain several forms of attacks. The signatures that are saved and updated over time assist the network administrator in determining what type of attack is being targeted. The signature-based intrusion detection system has several flaws, the most notable of which being the requirement to update the database regularly, which is important because threats change often. Furthermore, storing adds data to the database, slowing down the system.

2.2.2. INTRUSION DETECTION SYSTEM BASED ANOMALY DETECTION

This type of attack detection relies on understanding network traffic and recognizing traffic anomalies. Therefore, an alert is issued when an anomaly is detected in the monitored traffic. Anomaly detection methods can be divided into two methods: using statistical techniques for detection and identification, and using machine learning techniques to create models that can be used to identify attacks. Machine learning methods can be classified into two types, one that uses supervised learning and the other that uses unsupervised learning. Where algorithms are trained through different datasets dedicated to building such systems. These systems are then limited to detecting anomalies in the data on which they are trained. The IDS system can train them to detect different types of attacks, such as denial of service attacks, worm attacks, etc.

2.2.3. INTRUSION DETECTION SYSTEM EVALUATION METRICS

To evaluate the performance of intrusion detectors, researchers use some statistical and computational methods and methods to evaluate their systems. Several of these scales are used to evaluate the model's performance in terms of accuracy, which is the most important metric on which to base. Other metrics used like (Accuracy, Recall, F-score, and Confusion Matrix), will be discussed with their respective equations. It all depends on the input and metrics like (TP, TN, FP, and FN).

True Positive (TP): A positive result is true when the model predicts it, and it is a positive result.

True Negative (TN): It is when the model correctly predicts it as a negative result and it is a negative result.

False Positive (FP): Represents the missing state of the model when a result appears as positive until it is negative.

False Negative (FN): It is also considered a miss in the model where the result is classified as negative to the fact that it is positive.

Accuracy (ACC): The accuracy metrics are used to measure the correct ratings and to measure their total number.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

Precision (PR): It is done by measuring several incorrect ratings with the real ones, but sequentially.

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall: It is considered one of the accuracy measures of the model by which the missed entries are measured, and the real classifications come before them.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

F1-score: It is also a measure of model accuracy when used with a data set. It is used to evaluate binary classifiers as either positive or negative.

$$F - score = 2.\frac{Precision.Recall}{Precision + Recall} \tag{2.4}$$

2.3. DEEP LEARNING

2.3.1. DEEP NEURAL NETWORK

A deep neural network is considered to be a more complex network than a regular neural network, which is a simple type whose architecture we can see in figure 2.10. A simple neural network is characterized by a network consisting of limited inputs and at most one or two hidden layers and an output layer where its operation simulates the behavior of a human neural network. This algorithm or style suffers from some of the simplest problems and flaws, the most important of which are oversaturation and inability to handle large data with high accuracy and its performance is poor and good. Therefore, the most important feature of the deep neural network is its ability to process big data with high efficiency because it contains many different hidden layers. It also contains multiple entries.

Thus, deep neural networks can be considered as modernization and a major development in the field of deep learning, and a major step forward in the field of big data, provided that the available resources are available. high computational resources for processing. Continuous processing occurs between hidden layers and continuous improvement in accuracy and performance. Therefore, the depth of a neural network is the number of its component layers [23].

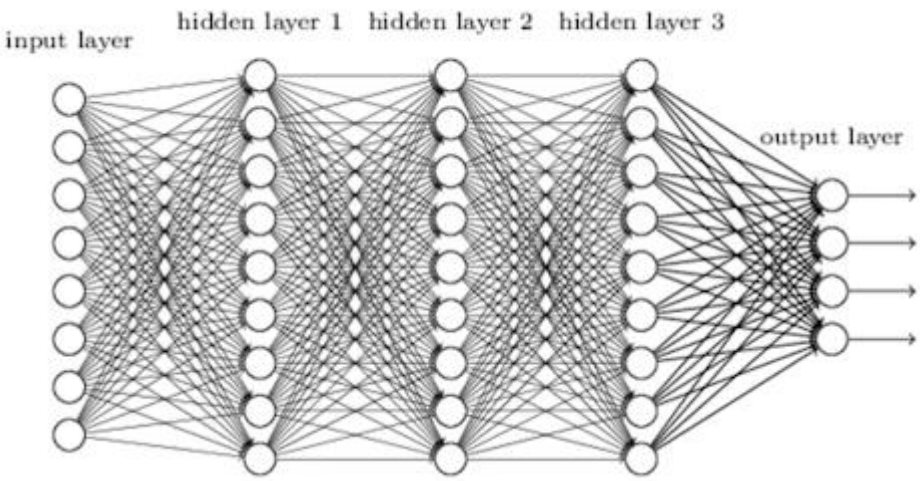

Figure 2.10. Deep neural network architecture.

2.3.2. CONVOLUTIONAL NEURAL NETWORK

A convolutional neural network is a type of network that depends on image inputs where it uses more than one type of hidden layer that perform several functions including convolution and aggregation. The convolution function also helps to solve problems and create and clarify maps, as we can see illustrated in figure 2.11.

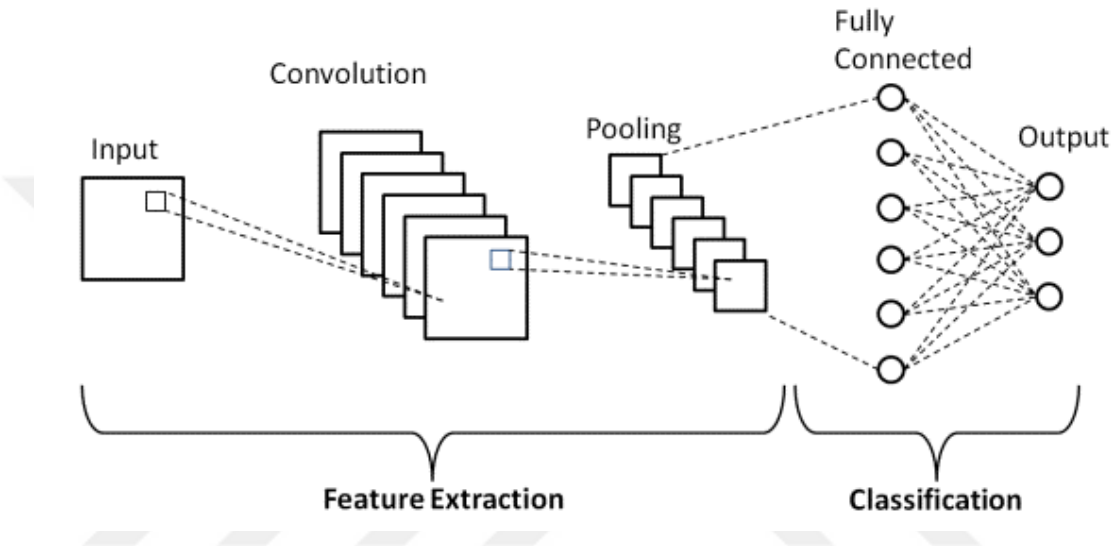

Figure 2.11. Convolutional neural network architecture

It can be said that CNN requires a small amount of processing compared to other methods, and this is one of the advantages of ConvNets. It is considered that the previous methods are primitive and need more training to achieve a good filter design. The ConvNet filter has been designed to simulate the transmission mechanism in human brain cells, so that these cells can be influenced by the visual field to receive limited stimuli in the limited visual field, also in this filter it is shown specified on the relevant area just to get the filtered data.

2.3.3. RECURRENT NEURAL NETWORK

A Recurrent neural network is another type of neural network, which is characterized by its power in time series domains with sequential data, this type of algorithm is used in many fields, of which the most important is natural language processing, and also

depends on this type on a feedback mechanism where the inputs of the current network are the outputs of the previous network, where this algorithm uses the handle change of data type based on the previous type [24, 25]. Figure 2.12 shows an illustration of a Recurrent Neural Network (RNN).

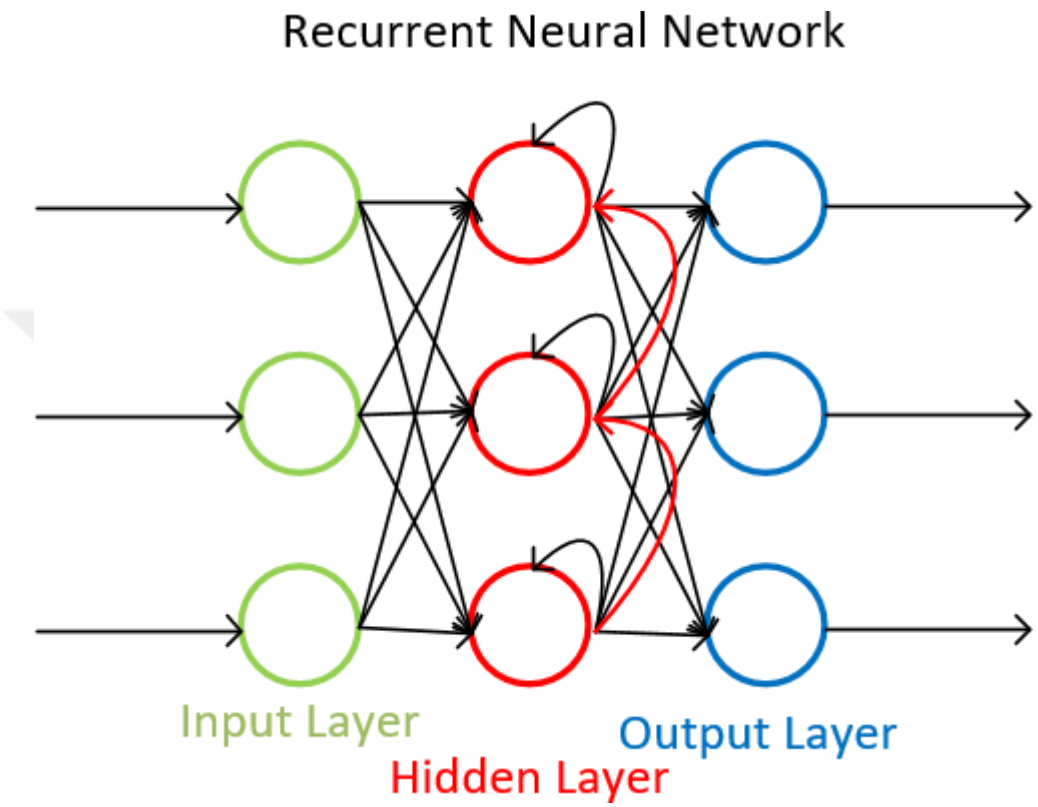

Figure 2.12. Recurrent Neural Network (RNN) illustration

It can also be compared with feedforward neural networks. Use the type that determines the neural network, and if it is an internal memory type, process it if the input is time series. These inputs are different from each other because they are used in the development of neural networks. Finally, the input is connected to the recurrent neural network, as shown in figure 2.13, which shows the inside of the RNN.

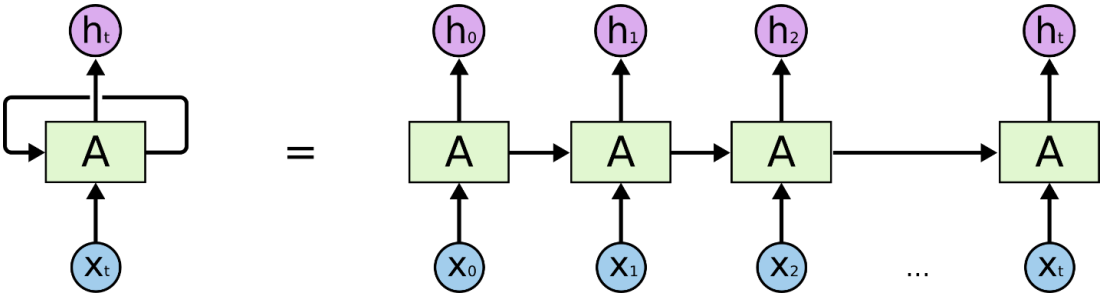

Figure 2.13. The inside illustration RNN.

Weights are represented by the symbol W, secret and singular variables are represented by the symbol h, W_{hh} is the weight of the currently represented input state, and tanh is the activation between the two values. The activation function used when is [1.1].

2.3.4. LONG SHORT-TERM MEMORY

Long short-term memory (LSTM) is one of the types of cyclic neural networks, where this type depends on feeding input with the help of outputs from the previous cell. This type of neural network has solved the problem faced by type networks (RNNs) in the past when they cannot store data for long periods of time. LSTM are capable of handling stored data. long-term storage because they use a memory called shadow term, which means they are capable of storing data for long periods. This type of neural network was discovered by the authors (Hochreiter & Schmidhuber). LSTM has the ability and ability to store data for a long time, unlike RNN which performs well in this respect. The most important applications of LSTM are time series dependent applications where it provides good performance. In figure 2.14, we can see that the LSTM structure includes the forget gate, using a sigmoid function, set to (1,0). In this portal, unwanted or useless information is deleted, after processing and passing through the activation function. For input gates where the input data is organized, if the input data is only used data for the neural network, the data is organized by the sigmoid function and then the (tanh) function is used to get useful values. The (tanh) through the property. Transforms the data and returns the result between (1, +1). As for the output gate, it is responsible for extracting useful or good data from the cell, as

is done using the (tanh) function. In this function, the data is organized using the sigmoid function and multiplied to output the next cell.

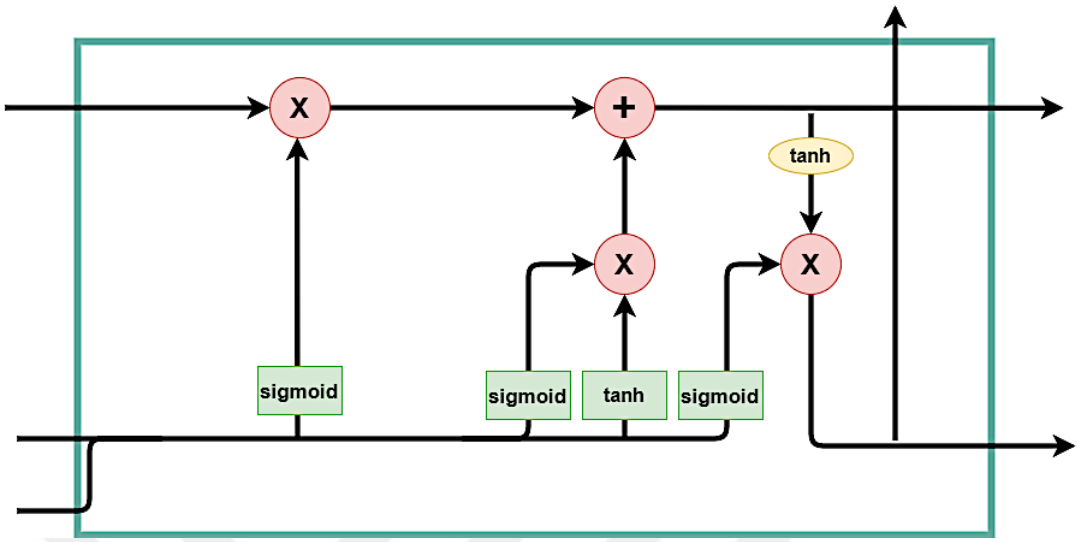

Figure 2.14. Long Short-Term Memory (LSTM) Architecture

2.3.5. GATED RECURRENT NEURAL NETWORK

The Recurrent Neural Network suffers from several problems including the following gradient as well as the exploding problem, and hence several developments have been made on the Regenerating Neural Network, perhaps the most important of which is long short-term memory, which is a well-known part of this distinction, but isn't as efficient and efficient in closed neural networks GRU, which is less well known. In figure 2.15 is the internal structure of GRU consists of three internal gates, and the state of the inner cell is not maintained, because the gate fusion method is used in the case of a hidden inner cell LSTM. The GRU gate consists of three different entities:

Update Gate: The work is exactly the same as the work of the LSTM output gate. The purpose is to determine the amount of work and prior knowledge required. It was passed on to the future.

Current Memory Gate: When referring to the recursive unit array with gates, it has not been discussed, mentioned, or explained in detail yet, it is associated with re-entrance, modification, or reset gate, in which the most important achievement is to correct the

errors in the input and can make it null. Often used to reduce the influence of information from the front gate on the influence of the latter, so it is combined with a reset port. Most of the basic explanations in the regular unit network are the same about work, which leads to an increase in job similarity where the difference lies only in the internal structure, where the structure the inner only includes the internal unit gates and the previous structure of hidden loop. Most of the basic explanations in recurrent unit networks are similar in work, which leads to a rise in task similarity where the only variation is within the internal structure, which is made up only of internal unit gates and the preceding concealed loop.

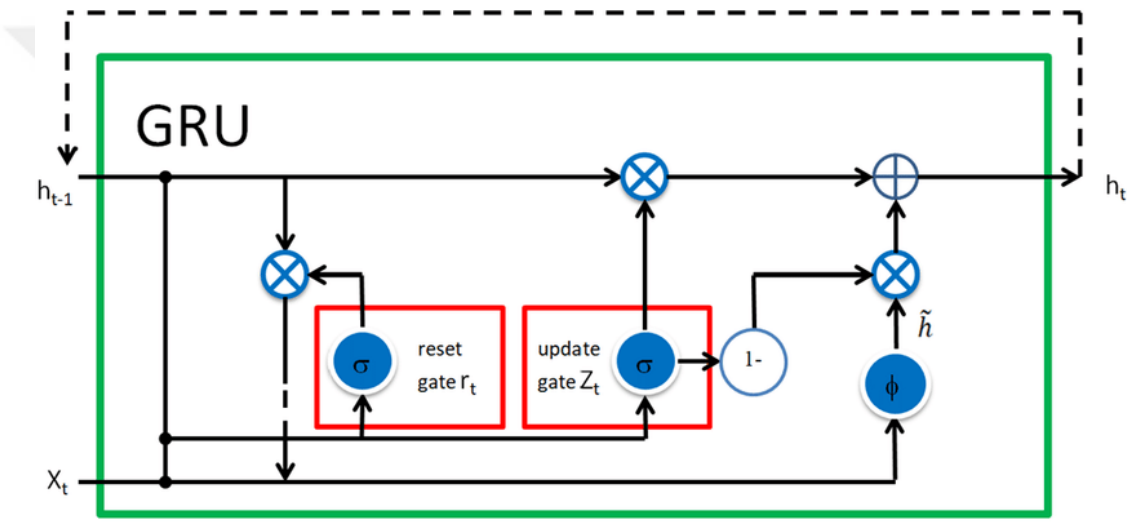

Figure 2.15. Gated recurrent unit (GRU)[14].

Update Gate formula:

$$Z = \sigma(W_z \,.\, x_t \;+\; U_Z \,.\, h_{(t-1)} \;+\; b_z \tag{2.14}$$

Reset Gate formula:

$$r = \sigma(W_r \,.\, x_t \;+\; U_r \,.\, h_{(t-1)} \;+\; b_r) \tag{2.15}$$

PART 3

RELATED WORKS

This study demonstrates the use of deep learning and data preprocessing techniques applied in several approaches. This method is used for applications in intrusion detection systems in software defined network environments to detect attacks. Contributions of this technique are associated with the detection of attacks. Deep learning with the data preprocessing to build (NIDS-DL) system for SDN.

2.1. METHODS NON-BASED MACHINE LEARNING

In [27] A method, approach, and method that is not related to machine learning has been proposed, meaning that the statistical analysis approach or the simple statistical approach is adopted, as this approach is based on entropy, which is an approach where a comparison of the entropy packages is neglected to discover the randomness between the different packages. As the arithmetic comparison process was done, where a threshold value was set, and if this increased, it means that an attack will occur and looked at it through a window. It was suffering from some problems, which is that it can detect the attack for one user or one guest only. The question that remains on the table is whether it can solve and detect the attacks on the whole system.

Authors in [28] employed method based on excluding FP and FN from the network where a threshold value consisting of three different levels is relied upon. The disadvantage of this method is that it takes up many resources from the network. Authors in [29] This method was faster than the previous one although it did not mention the resources used. It also used different types of data to measure the values of entropy and threshold. The FNs are higher while the values of FBs are lower. The authors in [30] was based on the work [27] Where the authors explained the use of a packet size of 50 about entropy, the statistical analysis was also of a short-term type.

The focus of the researchers was on whether or not the attack would happen, as a varied and different size of windows was used as it was proved that this method was not effective in a way that was to some extent increasing the attack percentage to 75% from the rate used for the regular attack, as the source IP of the incoming packets was adopted to identify the size and measurement of the different entropy. Study in [27] His method, along with other methods, was specifically dedicated to the SDN environment.

Also, there is research into [31] producing better detection accuracy for analysis in addition to faster computation of entropy for each flow and detection speed and analysis. The improvement was centered on the signature-based anomaly-based approach, as the anomaly detection method is by observing the entire traffic movement and comparing the normal movement with the normal movement to trigger an alarm, and this is what is known as FP when the use of the signature-based is available. In that it has a data set known as a threat as it can pass a set of threats without being detected until they are placed for them in a dedicated database. Specifically, about denial-of-service attacks in a software-defined network environment without using machine learning methods. Some attempts have focused on this area, such as in [32, 33] this research which focused on using a specialized framework to detect potential flow attacks.

Study in [32] Whereas, he focused on saturation attacks in the control and data layers. This framework suffers from some problems, including excessive delay, although it has high accuracy, as it works on the principle of proactive analysis of the evidence before entering it.

In this section, the methods used in intrusion detection systems were discussed without mentioning the methods of machine learning and deep learning.

2.2. METHODS BASED MACHINE LEARNING.

In addition, with the development of the new generation of software-defined networks, it is necessary to design a protection and detection system based on machine learning

and deep learning systems developed by several researchers. Study from the design of intelligent intrusion detection systems. In the next subsection, approaches or studies focusing on the use of machine learning and deep learning methods in intrusion detection systems are discussed and the methods attempted to implement the system. This for SDN will be discussed in the following subsections.

2.2.1. MACHINE LEARNING ALGORITHM BASED INTRUSION DETECTION SYSTEM

Several authors have used machine learning to design intrusion detection system models, and the performance of these models in detecting attacks has been evaluated using the dataset that created in specialized laboratories.

The bee colony algorithm was used in [34], and the approach or method combined the SVM algorithm with a cluster based on the bee colony method, which was named CSVBC and applied CSVAC. This is also a cluster based on SVM and methods. Ant colony that author used with KDD99 dataset. Author try applied classification rate and implementation time efficiency, the results show that the combination of the two approaches yields better results than either method alone. When comparing (CSVBS) with (CSVAC), the proposed system (CSVBC) performed better, with an accuracy of 88.46 percent.

Study in [35] that using approach proposes the use of machine learning techniques in intrusion detection based on size reduction by the Principal component analysis (PCA) method, using two types of supervised learning algorithms K-Nearest Neighbors (KNN) with Suport Vector Machine (SVM), then the accuracy of these two algorithms is the same. The KDDCup99 dataset was used. The KNN-based method achieves a high detection accuracy of 90%.

Authors in [36] used a machine learning classifier-based intrusion detection attack approach. The researcher used (MLP, J48, and Bayes classifier). The classifier was applied to a dataset of type NSLKDD. This method achieves good recognition results

with many classifiers. It also achieved the corresponding correct detection rates (93.1%, 91.9%, 90.7%).

Table 2.1. Summary IDS with Machine Learning.

Ref.	Dataset	Type Method	Accuracy	Result
[34]	KDDcup99	SVM, Bee Colony	88.46%	CSVAC get better accuracy from CSVBC
[35]	KDDcup99	KNN, SVM	90%	The approach achieved high detection accuracy after using the PCA method
[36]	NSL-KDD	J48, MLP, Bayes Network	93.1%, 91.9%, 90.7%	High result compared another study related

To summarize this section, a discussion has been made about most of the methods of using machine learning in intrusion detection systems. The types of algorithms used are discussed, along with the datasets summarized in Table 2.1.

2.2.1. Intrusion Detection System in Deep Learning.

Authors in [37] in this approach, the author used self-learning with its application with deep learning to develop an effective intrusion detection system in detecting attacks, the researcher used a dataset of NSL-KDD type. The approach achieved a good performance in intrusion detection when compared with other methods in the previous methodologies.

Study in [38] This approach suggested using the deep neural network approach in designing an intrusion detection system, the approach was based on the detection

method using binary classification and multi-layer classification, the approach used a KDD Cup 99 dataset, and the approach achieved a good detection accuracy with an average of 99.98%.

Study in [39] the authors used an intrusion detection method based on a deep learning algorithm. In his approach, the researcher used the LSTM algorithm with dataset of type NSL-KDD. The approach is evaluated based on binary and multiple classifiers. The classifier achieved a detection accuracy of 99.2% for binary classifiers and 96.9% for multiple classifiers.

Study in [40] The model used in this research suggested the development of a system to protect the network infrastructure and protect it from intrusion using modern deep learning algorithms and tricks, where the approach applied the use of RNN and LSTM algorithms in intrusion detection. A data set of NSL-KDD type was also used. The approach achieved high detection accuracy and good performance when compared with other studies in the same field.

Study in [41] focuses on applied autoencoder for non-symmetric using unsupervised learning feature the model was evaluated using the NSL-KDD and Cup99 datasets, the model showed good and promising results. Table 2.2 shows the summary IDS with Deep Learning.

Table 2.2. Summary IDS with Deep Learning.

Ref	Dataset	Type Method	Accuracy	Result
[37]	NSL-KDD	STL, SMR	88.39%, 79.10%	The performance of the model was good compared to the performance and other methods used previously
[38]	KDDCup99	DNN	99.98%	Amazing accuracy and performance in detection
[39]	KDDCup99	LSTM	99.2%	Used binary classification and

				multiclass classification
[40]	NSL-KDD	RNN, LSTM	90.53%, 96.48%	The model used provided a high and good detection accuracy compared to other models
[41]	NSL-KDD, KDDCup	Autoencoder	98.81%	The approach used has achieved promising and high-accuracy detection results

In a summary, we presented a discussion of the methods and algorithms used in intrusion detection using deep learning methods, and the accuracy of each user model was shown. we can see a summary of the methods used with the data sets and the results obtained.

2.2.2. MACHINE LEARNING BASED IDS FOR SDN

Most of the researchers who specialize in this field have, for many years, relying on traditional machine learning algorithms, and the most famous of these algorithms is the algorithm of neural networks or random forests, K-nearest Neighbor (KNN), or Support Vector Machine (SVM) These algorithms and methods achieved various successes while demonstrating some weakness and restriction from some.

Study in [42] these approaches suggested designing an intrusion detection system in software defined network environment using machine learning methods. Use the NSL-KDD dataset to evaluate the results. Classical methods were used for classification such as (XGBoost, Random Forest, and Decision Trees) algorithm, the approach used focused on case detection. Abnormal protection for SDN system.

Study in [43] it was suggested that a DDoS attack was detected in an SDN environment using two types of SOM algorithms with KNN. This approach was focused on mathematical operations while maintaining specific classification accuracy, The

approach used a CAIDA dataset to generate attacks, and the approach yielded good detection results and adequate accuracy.

Study in [44] The approach suggested developing an intrusion detection system based on anomaly detection using machine learning techniques. The approach focused on using it to detect potential attacks in the SDN environment. The approach achieved a high detection accuracy in identifying attacks with an accuracy of 97%.

Study in [45] The researcher and author used a Floodlight control unit based on the Java environment, in addition to using 6 types of features with Naive Bayes algorithms with SVM and KNN. The maximum accuracy was obtained in the KNN algorithm with an accuracy of 81%. The rest of the algorithms were not accurate enough. Table 2.3 shows the summary IDS with Machine learning for SDN.

Table 2.3. Summary IDS with ML for SDN.

Ref	Dataset	Type Method	Evaluation	Result
[42]	NSL-KDD	XGBoost	ACC= 95.95%	The approach achieved good detection accuracy, which proves the efficiency of the system that was designed
[43]	CAIDA	SOM, KNN	DR=98.24, DR=99.05	The approach has shown that it can maintain results and evaluate based on the decision rate
[44]	NSL-KDD	Neural Network	ACC= 97%	The approach performed well when used with pattern detection to improve accuracy
[45]	Aggregate data	SVM, KNN, Naïve Bayes	ACC= 81%, 97%, 83%	The limitation of this approach is that it is applied exactly to three algorithms, which can be applied to more than one type in the future

2.2.3. DEEP LEARNING BASED IDS FOR SDN

We will briefly summarize the results of the use of intrusion detection technology in conventional networks with our own work in SDN networks. Some research in [46] the authors used and focused on the NSL-KDD data type, with the use of a technique of deep learning or deep neural networks of the type of recurrent neural network with (Gated Recurrent Unit Neural Network) GRU-RNN.

Study in [47] This paper discusses the possibility of applying intrusion detection systems with deep learning in SDN, where the approach presented the application of the GRU algorithm for intrusion detection using a data set of CSE-CIC IDS 2018. The approach achieved good detection accuracy in identifying attacks using deep learning technology.

Study in [48] the approach suggested building an intrusion detection system based on deep learning systems for SDN, where it used a data set of type NSL-KDD where only 6 features were relied on in the detection process, and a DNN algorithm was used in the approach used. The approach achieved strong potential in anomaly detection.

Study in [49], the author used a new approach to SDN. This approach used deep learning to build an intruder detection system. The performance of the deep learning classifier was evaluated on three datasets (KDD99, NSLKDD, USNW). In addition, the author has applied multiple types of optimization techniques to achieve the best results. In [50] This paper suggested building an intrusion detection system to identify attacks on SDNs using deep learning. CNN classifier was used, and it was based on binary classification in the detection process. Features extracted from KDD 99 dataset were used. The approach focused on detecting unknown attacks.

In [51], the author used an attack detection approach using deep learning from SDN. This approach has been applied to ISDN datasets. This dataset has been applied to the CNN classifier. This approach has achieved good results. Table 2.4 shows the summary IDS with DL for SDN.

Table 2.4. Summary IDS with DL for SDN.

Ref	Dataset	Type Method	Evaluation	Result
[46]	NSL-KDD	GRU-RNN	ACC=89%	The approach showed its capabilities in intrusion detection and strong identification of attacks
[47]	CSE-CIC IDS 2018	GRU	-	The approach introduced new ways of detecting attacks using a modern data set
[48]	NSL-KDD	DNN	ACC=75.75%	The approach showed a strong potential in recognizing attacks and detecting anomalies
[49]	KDD99, NSL-KDD, UNSW	DNN	ACC=99.69%, ACC=95.38%, ACC= 81.70%	Result is good but performance of evaluation is worse due depending on low learning rate
[50]	KDD99	CNN	-	Proposed approach to detection attack in SDN
[51]	InSDN	CNN	96.43%	New study that compares DL with another ML approach using

As a summary, we discussed in this section the tests that implemented intrusion detection systems using deep learning with software-defined networks.

2.3. SUMMARY OF CHAPTER

At the beginning of this unit, a brief description of the SDN technology was discussed, then the IDS technology and its types were discussed and explained in detail, the branches that consist of it, and the various attack techniques, as this thesis will deal with anomaly detection technology and will focus on it during the detection process also the deep learning techniques used are discussed and the algorithms that are used in this thesis are clarified. This is a detailed description of reviewing the previous literature and discussing several types of machine learning and deep learning techniques when used with intrusion detection systems, discussing the types of algorithms used and data sets applied, then moving on to these technologies and future directions for their application in SDN technology. All these details are in order to

reach a comprehensive review to measure the performance of the user for each method and evaluate it. As we can see in Figure 2.16 the diagram summary of related work.

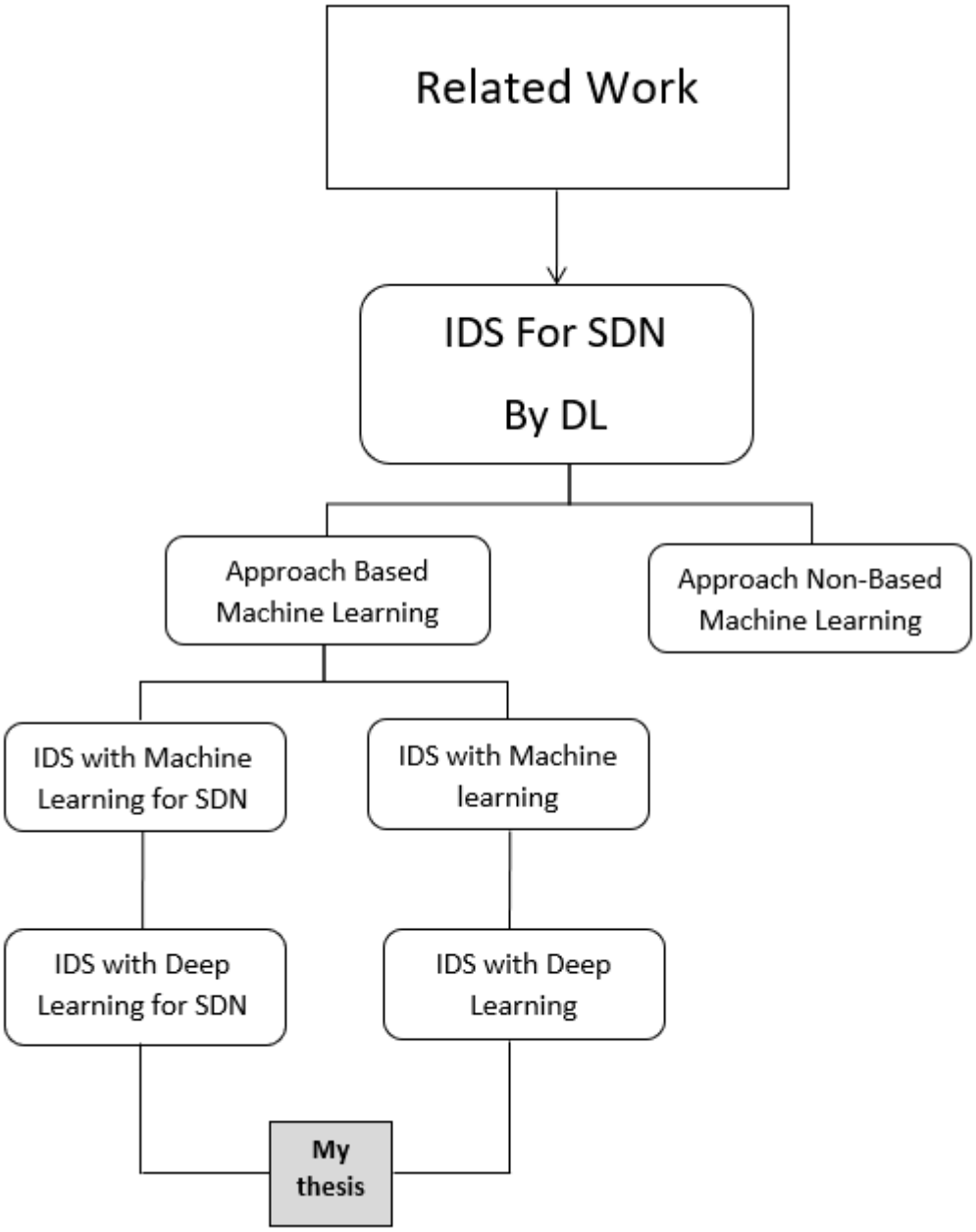

Figure 2.16. Diagram Summary of Related works

PART 4

METHODOLOGY

3.1. INTRODUCTION

Although SDN technology is considered a modern technology, it suffers from many gaps and weaknesses in its infrastructure, especially with regard to the controller part. Where the controller is the main part of the network and its exposure to attack means the fall of the entire network. Therefore, it is important to protect this controller from A very common security system called (IDS). It is located in the network part in order to monitor the performance of data entering and leaving the controller from the external network.

Sometimes there are some programs and tools that the network administrator uses in order to notify him of attacks when they occur or when trying to penetrate the network. According to the mechanism of work, intrusion detection systems can be classified into two main types depending on the type based on the host and the other based on the network. Where the type based in the network (NIDS) on which our work will be devoted and its development in this thesis.

In this chapter, the basic concepts will be introduced. As the main topics that will be clarified and suggested in this chapter are: The first section is an introduction to the topic and the mechanism of work that will be followed. The second section will explain the proposed system in detail with important plans. And the third section will mention and detail the deep learning algorithms used in this approach. The fourth section is a detailed explanation of the data set (NSL-KDD) used in the approach. And the fifth section is the mechanism that will be relied upon in evaluating the research results and the conclusions that will be reached. And the last section is a summary of the chapter in its entirety.

3.2. PROPOSED METHODOLOGY

The proposed system can be observed and viewed through the Figure 3.1. Which carries out the process of detecting and classifying the attacks coming to the controller using a network-based intrusion detection system based on anomaly detection. The data was collected in the laboratory of the university of (new brunsuick). The NSL-KDD data set was a development of another previous dataset KDDCup99 [52] , which was suffering from several problems such as an increased number of duplicate data and inaccuracy and many other problems that were solved in the NSL-KDD [53] dataset which will be detailed in the following sections. Although this data set is new, it does not mean that it is completely free of any problems or does not need constant updating, but it is believed that it will help researchers in building their systems well and efficiently to build intrusion detection systems. The most prominent feature of this set of data is that it has a reasonable number of records, and thus models can be developed and trained without the need to increase the cost of work and evaluate the results. After the data is processed and analyzed, it will be entered into the five classifiers that represent deep learning algorithms, where these classifiers will be trained on data sets to identify attacks, and then the efficiency of each classifier in detection and classification will be measured. The experiments were conducted entirely using the Python language through the Google Collab platform. The data set includes 42 main features that will be discussed in detail in the data sets section.

The application of this approach is a comprehensive study of the application of deep learning methods and its various algorithms in intrusion detection systems, as it is necessary to use deep learning methods and take advantage of their great efficiency. In this approach, five types of famous deep learning algorithms were used (DNN, CNN, RNN, LSTM, GRU). Where datasets are entered into these different classifiers and the classifiers are trained on datasets, then the efficiency of this system is measured in terms of accuracy of detection and the use of other different measures in the evaluation. This study will provide researchers with a comprehensive view of the latest techniques used and solutions will be presented and other ways to develop the future.

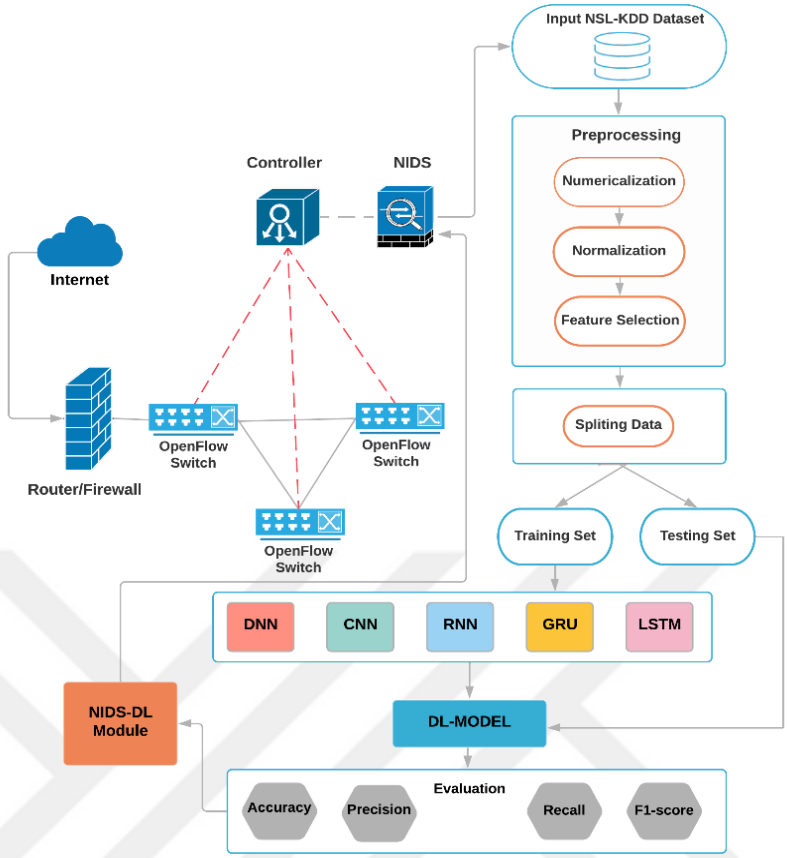

Figure 3.1. Proposed methodology system for NIDS with DL for SDN

3.3. DEEP LEARNING AlGORITHMS

The types of deep learning algorithms used in this study will be mentioned, and the mechanism of these algorithms will be clarified, with the addition of figures specific to each type.

3.3.1. Deep Neural Network (DNN)

In relation to deep neural networks, we use DNN with 3 layers in our thesis. The three-layer structure can be detailed, which is in the foreground. Two parts of the (Flatten) and another part of the Sequential were used, after that the features of the dataset (NSL-KDD) were introduced to the first layer of the neural network, which has a layer (DENSE = 1024) with an activation function of the type (ReLU), then we enter the output from these layers to a layer of (Dropout), which is the equivalent number (0.01),

after that another hidden layer of the (DENSE) is used with the number of entries (768). With an activation function of the type (ReLU) also, then we enter the outputs into an input of the type (Dropout) with a number equivalent to (0.01), after that a hidden layer of the type (Dense = 1) is used. Sigmoid is used in the last layer. as well as an optimizer of type “Adam” with a loss of type "binary_crossentropy" at the end of the model. The general architecture of the DNN used in our approach is shown in Figure 3.2.

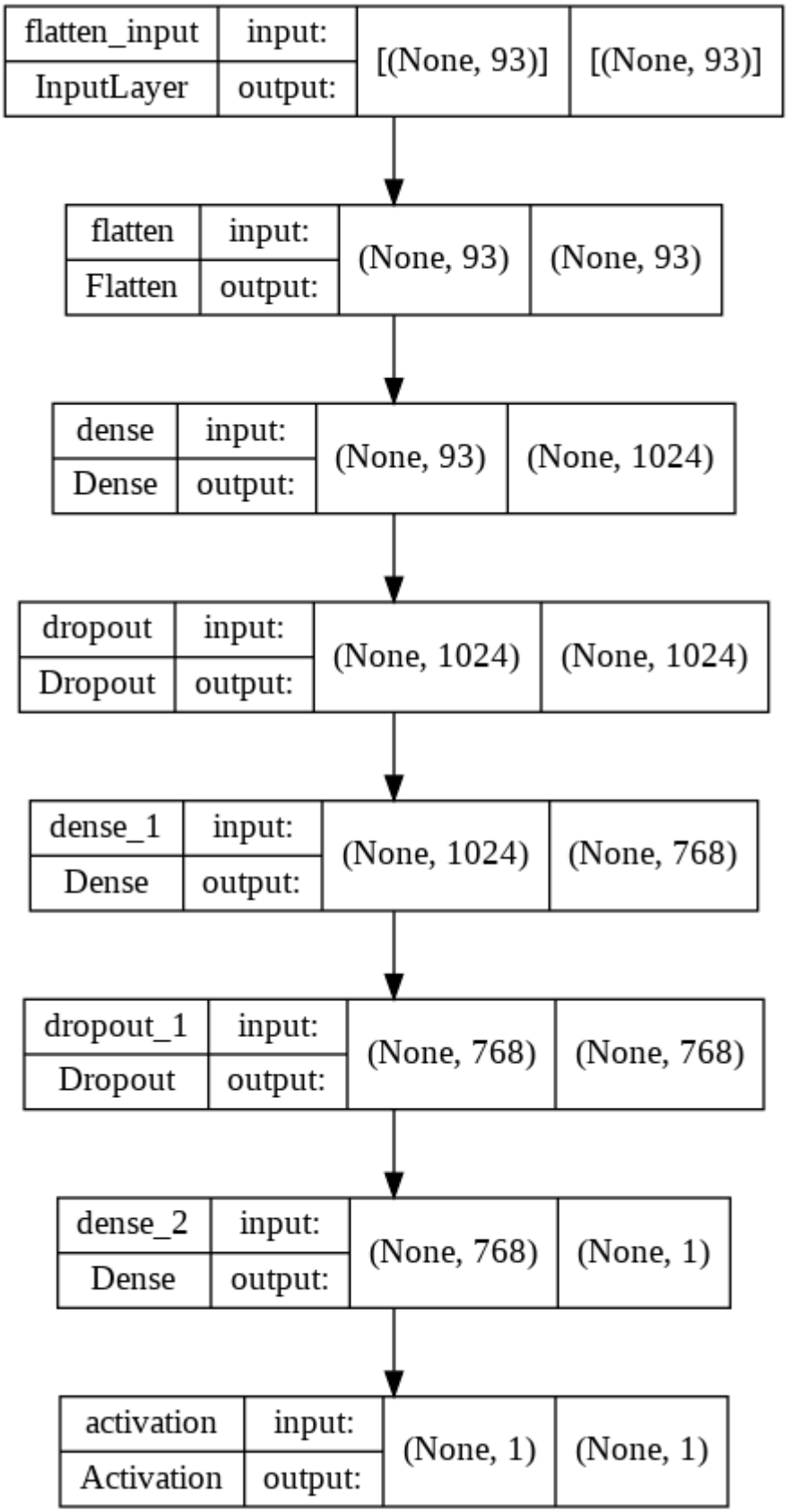

Figure 3.2. Summary Architecture of DNN model in NSL-KDD dataset

3.3.2. Convolutional Neural Network (CNN)

The Figure 3.3 shows the convolutional neural network classifier. Where (NSL-KDD) data is entered into the classifier that consists of several types of layers. The first and included layer is the CNN layer, and then (Conv1d) is used to initialize the data to be entered in the form of a VECTOR For the network where the function (ReLU) is used with the "input_shape" with a number equivalent to 1. The second layer (Conv1D) with the number 64 is also entered and used, with padding equal to “Same”, as well as the use of the (ReLU) function itself. Then also used other layers of (Conv1D) with a number (128) in three layers, then using (MaxPool1D), then using (Dropout) with a value of "0.5", and the last layer was a density of (1) with the use of the (Sigmoid) function. as well as an optimizer of type “Adam” with a loss of type "binary_crossentropy" at the end of the model.

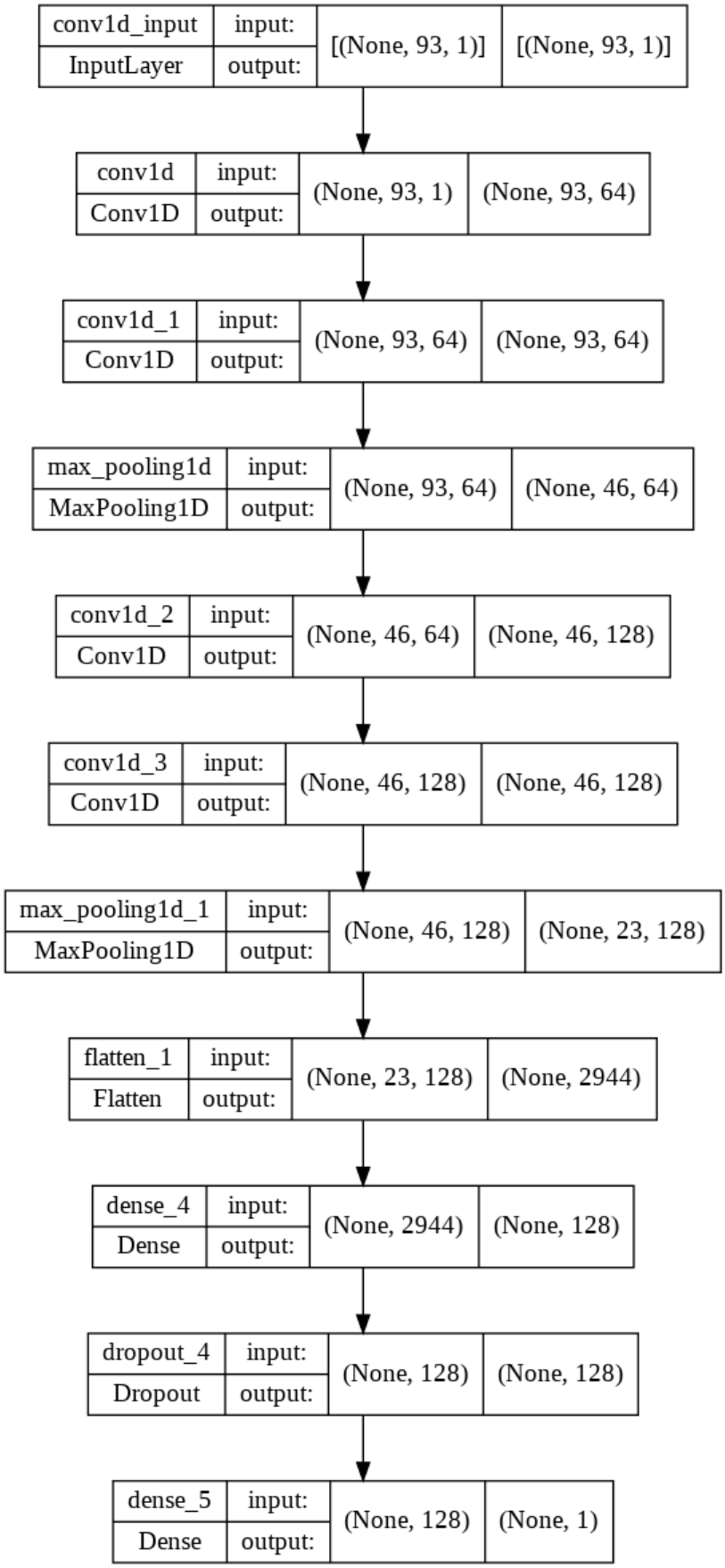

Figure 3.3. Summary Architecture of CNN model in NSL-KDD dataset.

3.3.3. Recurrent Neural Network (RNN)

Figure 3.4 shows the classifier of recurrent neural networks. The NSL-KDD data set was entered via 'Input layer' with a total of 12 features that are become 93 due to the One-Hot encoding method that transfer the number of features, also using input shape that converted shape of the matrix before inputting the classifiers. where the data was processed before being entered into the recurrent neural network, and a simple recurrent neural network, with the number of units of "50", was used. A number of "dropout" layers were used after the RNN layer with a number equivalent to '0.1' and then a layer of the RNN was also used with 8 hidden layers, Then a layer of “Dropout” was used with a number equal to the previous count, then a layer of “Dense” was used with a number of 1, and in the end, a layer of activation function of type “Sigmoid” was used, as well as an optimizer of type “Adam” with a loss of type "binary_crossentropy" at the end of the model.

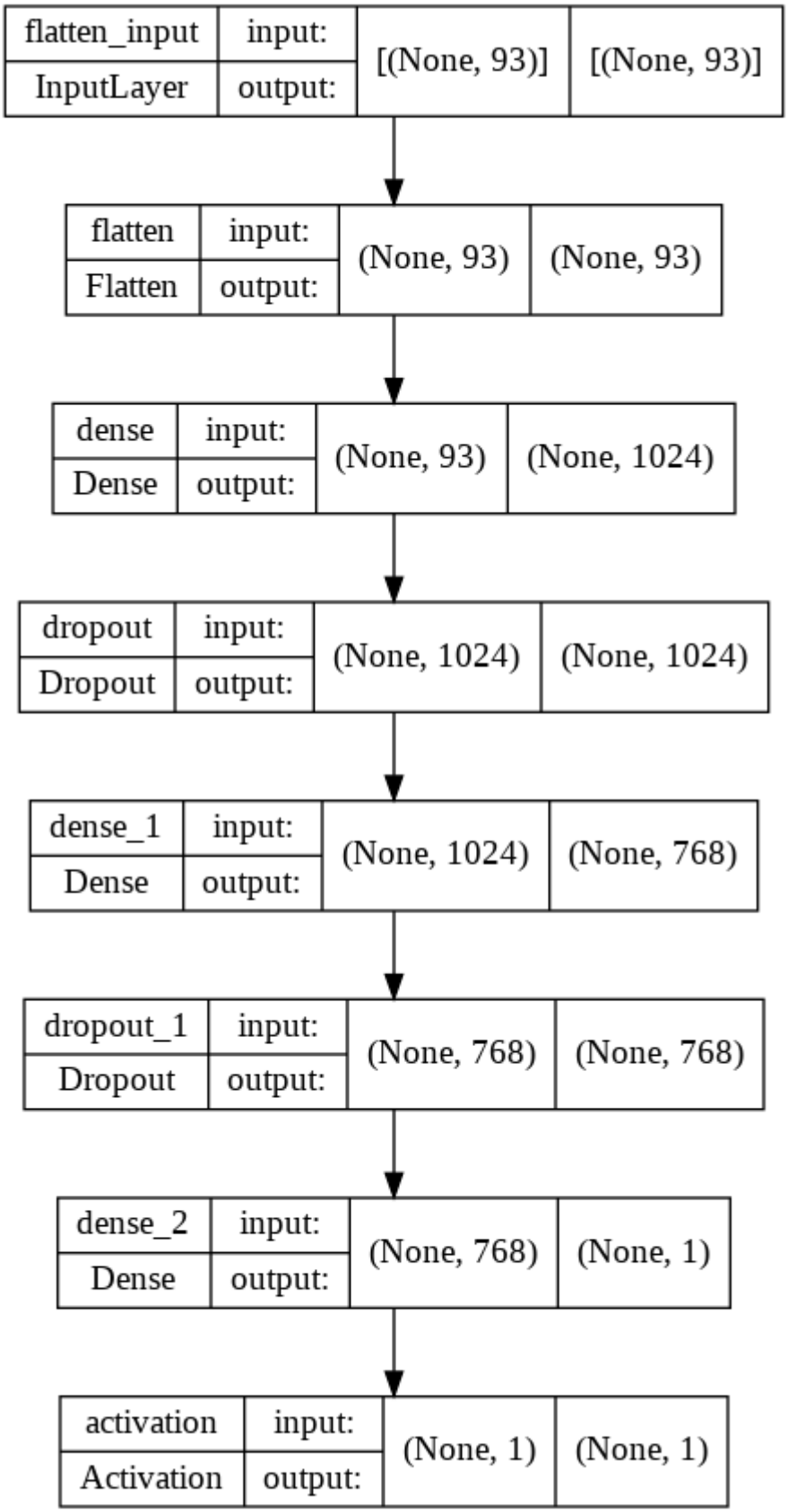

Figure 3.4. Summary Architecture of RNN model in NSL-KDD dataset

3.3.4. Gated Recurrent Neural Network (GRU)

The Figure 3.5 shows the gated recurring unit classifier. It consists of several types of layers and starts with processing and entering data sets "NSL-KDD" and then inserting them into the GRU classifier, where the first layer consists of 128 multiple-gated repeating caches, after that a layer of the "Dropout" to reduce the Overfitting, then a layer of "Dense" is used with One layer until the last layer of the activation function of type "Sigmoid" is used. Also, several methods and improvements were used for the classifier, including the use of an improved "adam" type.

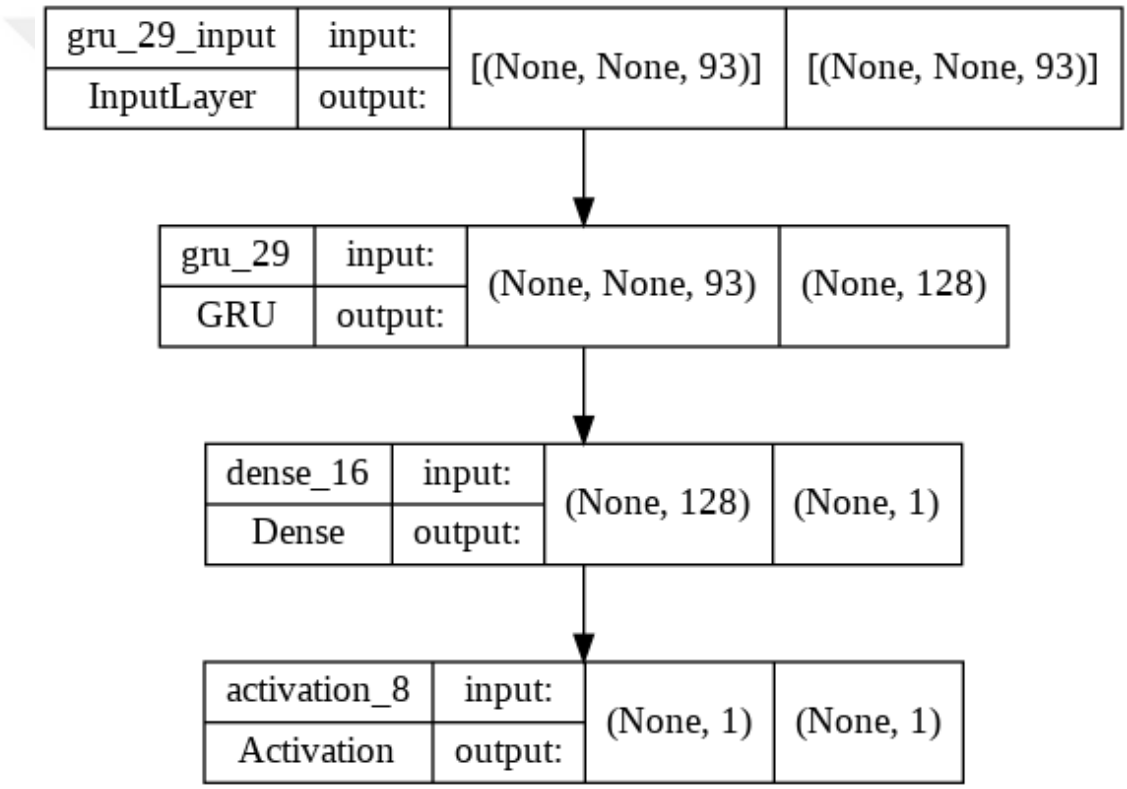

Figure 3.5. Summary Architecture of GRU model in NSL-KDD dataset

3.3.5. Long-Short Term Memory (LSTM)

Figure 3.6 shows the Long-short term memory classifier. Where this workbook consists of an entry for the NSL-KDD data set after it is processed and entered into the workbook, as it consists of three main layers, the first layer contains 128 data input layers, and then a layer of “Dropout” is used with an equivalent number of 0.1, then

this data is entered into another layer of the LSTM containing 128 inputs from the previous layer, after that it enters another layer of "Dropout" with a number equivalent to 0.1, then these layers enter into one other layer of the "Dense" with activation function of the type "sigmoid" is used, and the same type of enhancer of the type "adam" is also used.

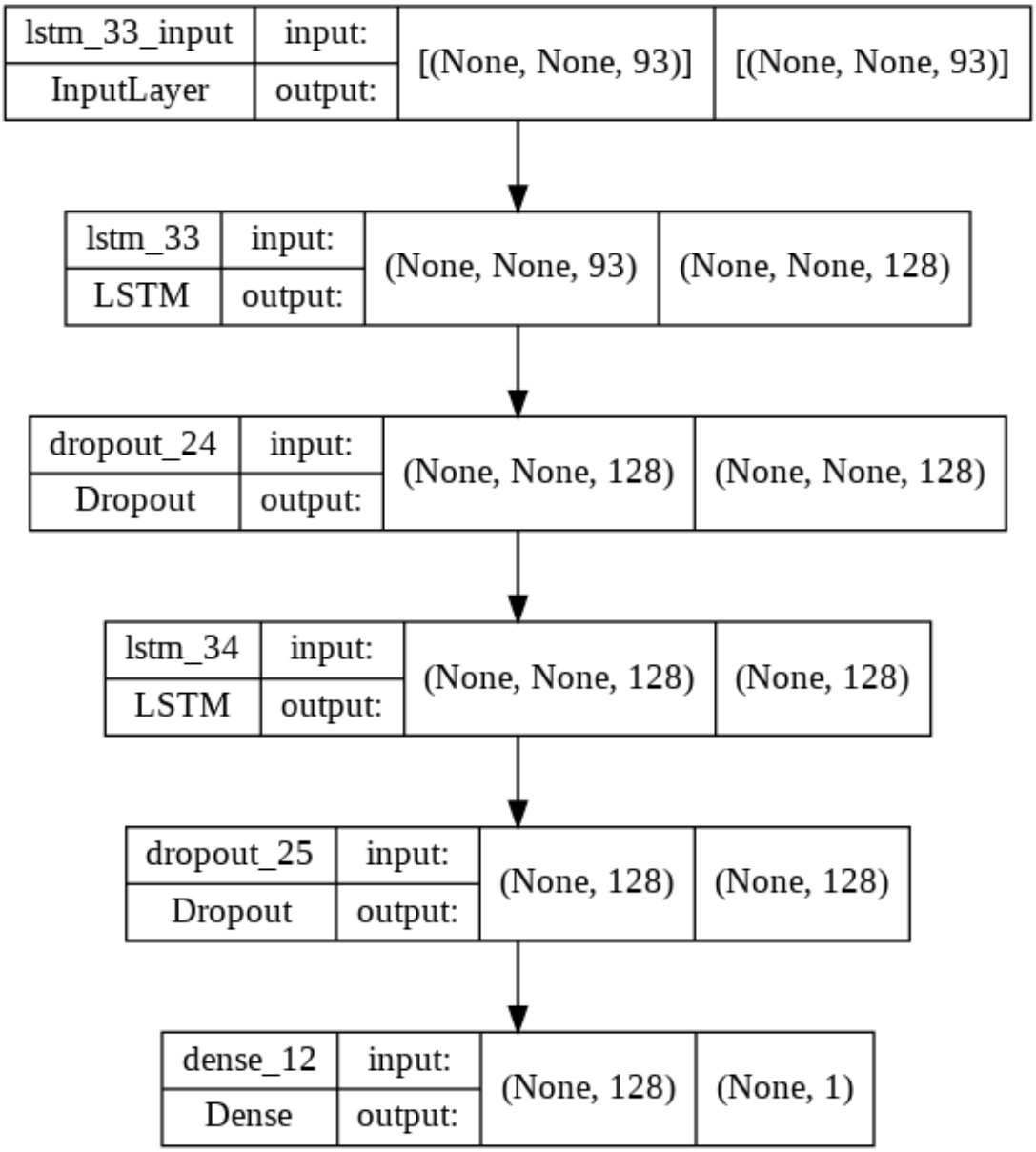

Figure 3.6. Summary Architecture of LSTM model in NSL-KDD dataset

3.4. DATASET

The NSL-KDD dataset is characterized as being developed to solve many of the problems that the KDDCup99 group suffers from, although it does not mean that this version does not suffer from any problems but that at the same time it provides many reliabilities for researchers who applied this data to test the efficiency of their systems and they applied many experiments on this set of data. As it was used as a way to check the reliability of the intrusion detection system in identifying attacks, it is considered flexible in terms of dealing with it by researchers as it provides a reasonable and applicable number of records by researchers and therefore the results are evaluation It is good and can be studied and researched. It was created and developed in the Canadian Cybersecurity Research Laboratory at New BRUNSWICK University.

3.4.1. Exploration Data Analysis

The NSL-KDD dataset contains two categories of data types such as (Categorial, and Numeric). Also, the data set contains within it types of attacks such as (Probe, R2L, U2R, and DOS), which will be clarified in Table 3.2 (in this thesis it depends on the Anomaly detection approach, i.e., it does not divide the attacks into categories, but rather a single category). Then the dataset is divided before it is triangulated into training data and test data. The training data contains 42 features and “125973” records, and the test data also contains 42 features and "22544" from the records, where a summary can be made in Table 3.1 showing the number of features and their names and a description of each feature.

Table 3.1. Features with the description of NSL-KDD Dataset.

No.	Name	Description
1	*duration*	duration of connection
2	*protocol_type*	Type of protocol used in connection
3	*service*	Service of the network from side destination
4	*flag*	Case of connection

5	*src_bytes*	Measures the amount of data transfer between source to destination
6	*dst_bytes*	Measures the amount of data transfer between destination to source
7	*land*	If the number of port =1 means yes, or port=0 mean binary refer, used by IP address such as source and destination
8	*wrong_fragment*	Measure the fault and wrong that happens in connections
9	*urgent*	Measure the packets is urgent
10	*hot*	Refer to the event in the system such as creating a program or implement
11	*num_failed_logins*	How many failed logins to system
12	*logged_in*	If logging is successfully referred is to1
13	*num_compromised*	That condition is measuring the number
14	*root_shell*	Make 1 for root shell that gained and 0 not gained
15	*su_attempted*	If need to use the command that using 1, otherwise using 0 if not need to use
16	*num_root*	Measures the num of the process done as root admin or not root
17	*num_file_creations*	Measures how many numbers of file that created during operations
18	*num_shells*	How many shells prompt command
19	*num_access_files*	How many times do access to file controls
20	*num_outbound_cmds*	That's measures how many commands in FTP outbounds session
21	*is_host_login*	Detect if login to the system from admin or root mean is 0,1
22	*is_guest_login*	Detect the away is done from a login is guest or any way
23	*count*	Measures the number of connections to the same destination host is done by the same connection
24	*srv_count*	Measures the number of connections to the same port number or same service using

25	*serror_rate*	Measure the amount of connection in percentages after aggregated connection
26	*srv_serror_rate*	Measures of the percentages of connection are aggregated
27	*rerror_rate*	Number of connections reg flag is aggregated in percentages
28	*srv_rerror_rate*	That measure the percentages of connections flag is aggerated by SRV
29	*same_srv_rate*	The measure of the same service aggregated by in count percentage
30	*diff_srv_rate*	That measures the different connections and services in account and percentage
31	*srv_diff_host_rate*	Percentages aggregated among connections in srv_count
32	*dst_host_count*	Measures the connections with the same IP destination address
33	*dst_host_srv_count*	Port number with same connection measures
34	*dst_host_same_srv_rate*	Used to measure the connection services and calculate percentages
35	*dst_host_diff_srv_rate*	That calculates the percentages of services and calculates a rate
36	*dst_host_same_src_port_rate*	Calculate the same
37	*dst_host_srv_diff_host_rate*	Emphasis on destination port and calculate in percentages
38	*dst_host_serror_rate*	Calculate the number of activated flags in percentages
39	*dst_host_srv_serror_rate*	Percentage of connections that have activated in rate value flag
40	*dst_host_rerror_rate*	Measures the activated reg flag among connections
41	*dst_host_srv_rerror_rate*	Reg flag activated of percentages of connections
42	*label*	Refer with 0 means normal,1 means the attack

Table 3.2. Classification Attack Categories of NSL-KDD Dataset.

Categories of Attack	Name of Attack in Training Set	Name of Attack in Test Set
U2R	Perl, rootkit, load module, Buffer overflow	term ,pst,Buffer overflow ,rootkit ,Perl
R2L	ftp-write, imap, spy ,phf, warezmaster, guess-passwd, multihop	Guess-passed, snmpguess, HTTP tunnel, named, snmpgetattack, ftp-write, send-mail, warez client
Probe	Satan, nmap, ipsweep, portsweep	Saint,ipsweep,portsweep,nmap,mscan,satan
Dos	Neptune, teardrop, back,smurf, land,pod	Worm,back,pod,teardrop,udpstorm,land, mailbomb, apache2, process table,Neptune,smurf

3.4.2. Preprocessing

Before entering the data, it must be processed before it is entered into the classifier and model, as the data must be processed and clear from outliers, null and recurring values. Data is preferred to be consistent. Our dataset contains categorical features need to convert. There are many different ways to convert the category-value features into numerical values. One of the jobs designated to handle that.

3.4.3. Feature encoder (One-Hot Encoding)

In the "NSL-KDD" data set, there are a number of features that are of categorical type, which must be converted to a numerical form in order to be entered into the classifier or dealt with and analyzed. The Table 3.3 shows the use of the "One-Hot Encoding" method.

Table 3.3. One-Hot Encoding Mechanism.

Language	One-Hot Encoding	English	French	Netherlands
English		1	0	0
English		1	0	0
French		0	1	0
Netherlands		0	0	1

3.4.4. Normalization

The term data transformation refers to changing data values and transforming them into other new values, as it is considered a general concept. Data sets are changed and scaled using many different methods, usually symbolized by encoding new values. The data transformation step is one of the most important steps in the analysis process, as it is always referred to as the step It is characterized by which it is possible to speed up the processing of data within the processed classifiers. As its benefits are to increase the accuracy and improve the used model and convert data into different formats, for example from numerical formats to nominal formats or vice versa, and the conversion contributes to changing the size of the data. Normalization or transformation is also considered a tool to remove redundant data from data sets and reduce these redundant values.

The Stander scaler method used to apply on data to made more consistent. The benefit of using this method to change performance of classifiers and avoid missled in algorithms when training. The work of this method can be defined by clarifying its method, where it uses the mean and standard deviation, where the mean is divided by subtracting the variance from the standard deviation. The standard scaler can be affected by outliers, which changes the results of the process as it depends on the standard deviation and means values of each instance.

Equation formula of Stander Scaler:

$$z = \frac{x - \mu}{\sigma} \tag{3.1}$$

$\mu = Mean$

$\sigma = Standard\ Deviation$

3.4.5. Split Data

Before entering the system, the data is divided into training and test data, where the training data is used by the system to train it on the data, or the test data is used to test the quality and performance of the system used. Partition when training data, we used a default method in splitting, where this method makes default divisions in the ratio of 75:25, until we changed the division and made it (0.75) for training and (0.25) for testing using the sklearn library. The Figure 3.7. Shows the splitting mechanism for NSL-KDD dataset.

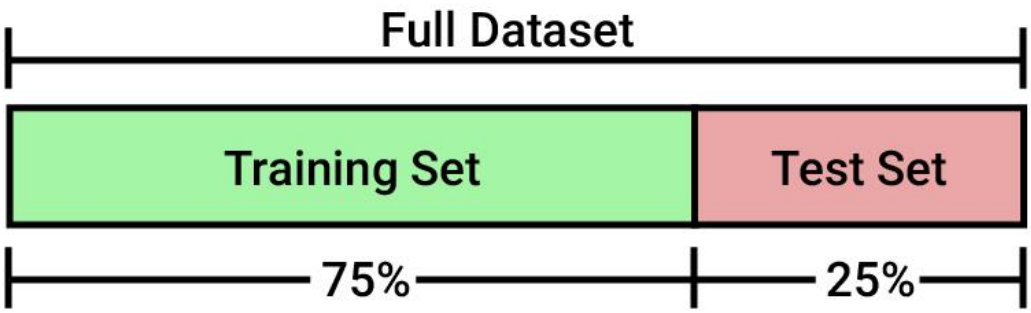

Figure 3.7. Splitting NSL-KDD dataset.

3.4.6. Stage of Evaluation Metrics

It is considered the main part of the results due to the fact that we used evaluation methods for the performance of each model and compared it with the performance of another model, where we used the most common deep learning methods in terms of algorithms, so this study is considered evaluative and depends on a number of metrics that we used, including the accuracy , recall and F-score, precision metrics for each

model in detecting intrusion and identifying attacks We also used the confusion matrix to evaluate the performance of each model used. The Table 3.4 shows the confusion matrix diagram.

Table 3.4. Confusion Matrix (CM).

	Yes	No
Yes	TP	FP
No	FN	TN

3.4.7. Chapter Summary

In this chapter, we will discuss the method that we will use in the project, which is the most important part of the thesis. The structure of the five deep learning algorithms used in our approach is specifically discussed, as the mechanism of their work, the layers that were used from them, and the number of each layer, in addition to that. The data set used in the approach was clarified, analyzed, clarified its details and how to process it before entering it into the model. Then the most important metrics that will be addressed were mentioned, given that the main purpose of this thesis is to build a (NIDS) model for SDN using deep learning methods and have the highest performance. Obtain it and make an evaluation and comparison of the performance of each model.

PART 5

RESULTS AND DISCUSSION

4.1. INTRODUCTION

This chapter presents the results of the proposed model using the NSL-KDD dataset. Where in the beginning, data processing is done through the processing tools that were previously discussed in chapter three, and each step in the processing is clarified. After that, the results are explained step by step and what methods of slicing and segmentation were used, as we will present the results of the evaluation in this chapter. The data will be hashed and configured for the entries of the different algorithm classifiers, where we will get from each classifier a set of results. After extracting the results for each classifier, performance measures are used to measure the efficiency of each algorithm and compare the performance of each type.

4.2. DATASET

The NSL-KDD data set is a set that was used by researchers in designing and building intrusion detection systems capable of simulating real-world attacks, as NSL-KDD is an update to the KDDCup set that was suffering from several defects and problems, so this version of the data set that addresses These errors, but this does not mean that they are free from any errors and do not need to be addressed. Several updates have been made to the NSL-KDD dataset and make it manipulable and trainable by researchers to use in their approach, and the number of records in the NSL-KDD dataset is reasonable, uncomplicated, and understandable. This collection was released in the university of new brunswick laboratories. The NSL-KKD contains logs of attacks that any simple network can see as a type of attack and contains five types of attacks (DOS, Probe, U2R, R2L) and 42 features. Table 4.1 shows the distribution NSL-KDD Dataset. Table 4.1 shows all the features of the NSL-KDD dataset with the addition of 6 columns of records.

Table 4.1. NSL-KDD dataset with six rows.

	Column1	**Column2**	**Column3**	**Column4**	**Column5**	**Column6**
duration	0	0	0	0	0	0
protocol_type	tcp	Udp	Tcp	Tcp	Tcp	tcp
service	ftp_data	Other	Private	http	http	private
flag	SF	SF	S0	SF	SF	REJ
src_bytes	491	146	0	232	199	0
dst_bytes	0	0	0	8153	420	0
land	0	0	0	0	0	0
wrong_fragment	0	0	0	0	0	0
urgent	0	0	0	0	0	0
hot	0	0	0	0	0	0
num_failed_logins	0	0	0	0	0	0
logged_in	0	0	0	1	1	0
num_compromised	0	0	0	0	0	0
root_shell	0	0	0	0	0	0
su_attempted	0	0	0	0	0	0
num_root	0	0	0	0	0	0
num_file_creations	0	0	0	0	0	0
num_shells	0	0	0	0	0	0
num_access_files	0	0	0	0	0	0
num_outbound_cmds	0	0	0	0	0	0
is_host_login	0	0	0	0	0	0
is_guest_login	0	0	0	0	0	0
count	2	13	123	5	30	121
srv_count	2	1	6	5	32	19
serror_rate	0.0	0.0	1.0	0.2	0.0	0.0
srv_serror_rate	0.0	0.0	1.0	0.2	0.0	0.0
rerror_rate	0.0	0.0	0.0	0.0	0.0	1.0
srv_rerror_rate	0.0	0.0	0.0	0.0	0.0	1.0
same_srv_rate	1.00	0.08	0.05	1.00	1.00	0.16

diff_srv_rate	0.00	0.15	0.07	0.00	0.00	0.06
srv_diff_host_rate	0.00	0.00	0.00	0.00	0.00	0.00
dst_host_count	150	255	255	30	255	255
dst_host_srv_count	25	1	26	255	255	19
dst_host_same_srv_rate	0.17	0.00	0.10	1.00	1.00	0.07
dst_host_diff_srv_rate	0.03	0.60	0.05	0.00	0.00	0.07
dst_host_same_src_port_rate	0.17	0.88	0.00	0.03	0.00	0.00
dst_host_srv_diff_host_rate	0.00	0.00	0.00	0.04	0.00	0.00
dst_host_serror_rate	0.00	0.00	1.00	0.03	0.00	0.00
dst_host_srv_serror_rate	0.00	0.00	1.00	0.01	0.00	0.00
dst_host_rerror_rate	0.05	0.00	0.00	0.00	0.00	1.00
dst_host_srv_rerror_rate	0.00	0.00	0.00	0.01	0.00	1.00
label	Normal	Normal	Neptune	Normal	Normal	Neptune

4.3. FEATURE SELECTION

We applied the feature selection method to our approach to select the best features that have a high correlation with the target feature. We extracted the 12 features from 41 features from NSL-KDD dataset. Feature selection has several advantages when apply such as, enhancing the performance of the model and getting the better accuracy, and avoid the miss led when training. Table 4.2 that explain the 12-feature extracted from NSL-KDD dataset.

Table 4.2. Features extracted from NSL-KDD Dataset.

	Column1	**Column 2**	**Column3**	**Column4**	**Column5**	**Column6**
protocol_type	tcp	Udp	Tcp	Tcp	Tcp	tcp
service	ftp_data	Other	Private	http	http	private
flag	SF	SF	S0	SF	SF	REJ
logged_in	0	0	0	1	1	0
count	2	13	123	5	30	121

serror_rate	0.0	0.0	1.0	0.2	0.0	0.0
srv_serror_rate	0.0	0.0	1.0	0.2	0.0	0.0
same_srv_rate	1.00	0.08	0.05	1.00	1.00	0.16
dst_host_srv_count	25	1	26	255	255	19
dst_host_same_srv_rate	0.17	0.00	0.10	1.00	1.00	0.07
dst_host_serror_rate	0.00	0.00	1.00	0.03	0.00	0.00
dst_host_srv_serror_rate	0.00	0.00	1.00	0.01	0.00	0.00
label	Normal	Normal	Neptune	Normal	Normal	Neptune

4.4. NUMERICALIZATION

In the NSL-KDD dataset there are several categorical features (Protocol_type, Service, flag) that must be converted using the One-Hot encoding property as shown in table 4.3. The result of three feature after One-Hot Encoding, we also can see in table 4.4 the result of the Categorical feature after applying One-Hot Encoding techniques.

Table 4.3. Categorical feature in NSL-KDD dataset.

	Protocol_type	**Service**	**flag**
0	Tcp	ftp_data	SF
1	udp	other	SF
2	tcp	private	S0
3	tcp	http	SF
4	tcp	http	SF
5	Tcp	Private	REJ
6	Tcp	Private	S0

Table 4.4. Categroical features with One -Hot Encoding.

	Protocol_ type tcp	Protocol_ty pe udp	Service ftp_data	Service other	Service private	Service http	Flag SF	Flag S0	Flag REJ
0	1	0	1	0	0	0	1	0	0
1	0	1	0	1	0	0	1	0	0
2	1	0	0	0	1	0	0	1	0
3	1	0	0	0	0	1	1	0	0
4	1	0	0	0	0	1	1	0	0
5	1	0	0	0	1	0	0	0	1
6	1	0	0	0	1	0	0	1	0

4.5. NORMALIZATION

The stander scaler method was used to normalize the NSL-KDD dataset. The table 4.5 shows fourth columns of product use after applying the stander scaler method for the 9 features extracted due to the fact that the dataset is unbalanced.

Table 4.5. Normalization of NSL-KDD dataset with 4 rows.

	Row1	Row2	Row3	Row4
logged_in	-0.809262	-0.809262	-0.809262	1.235694
count	-0.717045	-0.620982	0.339648	-0.690846
serror_rate	-0.637209	-0.637209	1.602664	-0.189235
srv_serror_rate	-0.631929	-0.631929	1.605104	-0.184522
same_srv_rate	0.771283	-1.321428	-1.389669	0.771283
dst_host_srv_count	-0.818890	-1.035688	-0.809857	1.258754
dst_host_same_srv_rate	-0.782367	-1.161030	-0.938287	1.066401
dst_host_serror_rate	-0.639532	-0.639532	1.608759	-0.572083
dst_host_srv_serror_rate	-0.624871	-0.624871	1.618955	-0.602433

4.6. ANORMALY DETECTION

Our approach is based on a binary classification to detect the anomaly, that is, classifying the output as either attacks or normal data, meaning that our approach is specialized in detecting anomalies through binary classification. After processing and feature selection to extract the best correlation features with the target, the total dataset contains (126,973 instances) and the size of train data is (107,077 instances) and (18,896 instances) instance the test size data.

4.7. IMPLEMENTATION MODEL CLASSIFICATION

4.7.1. DNN CLASSIFIER

After applying the DNN algorithm and training it on the NSL-KDD dataset, the result of matrices (Accuracy, Precision, Recall, F1-score) is shown in table 4.6. The results of the confusion matrix for binary anomaly detection were as in Figure 4.1.

Table 4.6. Result of Metrics applied DNN on NSL-KDD Dataset.

Accuracy	Precision	Recall	F-score
0.9853	0.983	0.9896	0.9863

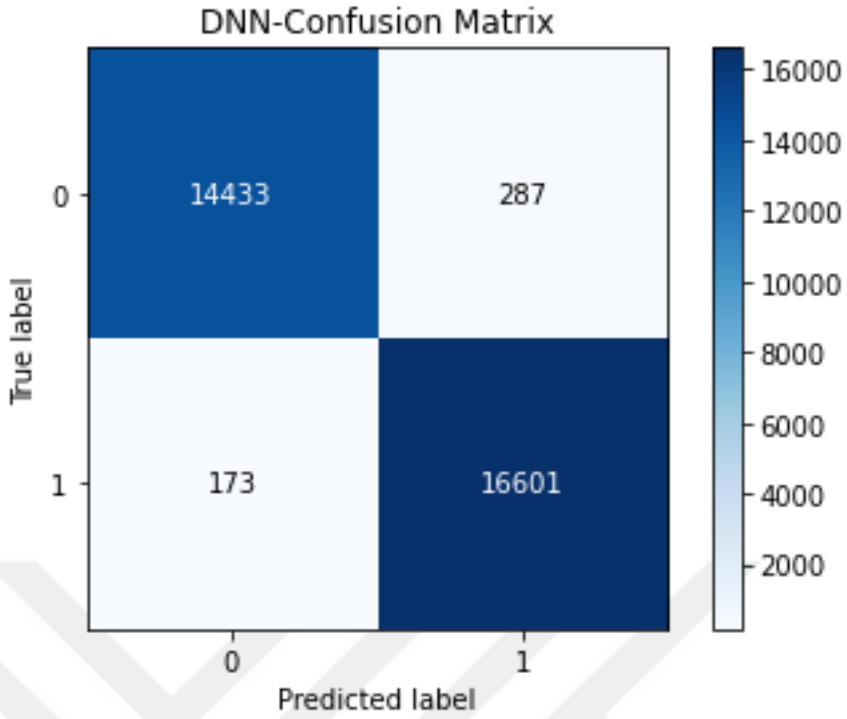

Figure 4.1. Confusion Matrix for applied DNN on NSL-KDD dataset.

The ROC curve is also an important metric that evaluated on our classifiers curve is depending on Sensitivity, Specificity when it gets result (0.998). Figure 4.2 that showing the ROC curve of DNN Classifier.

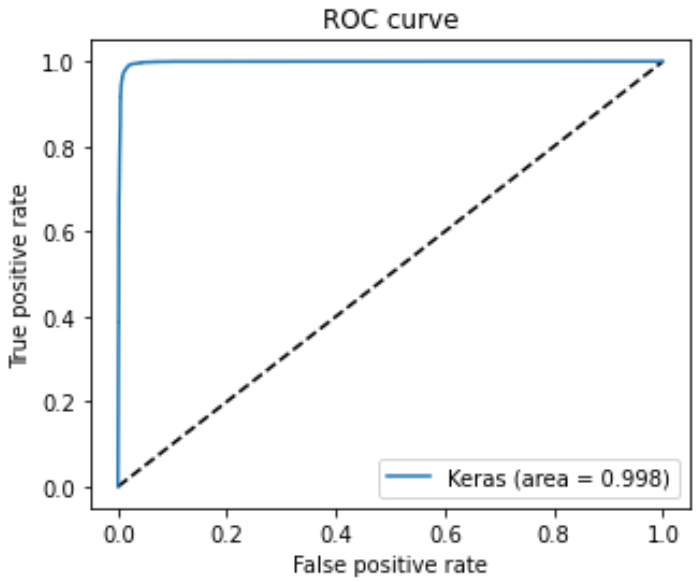

Figure 4.2. ROC Curve for DNN classifier.

4.7.2. CNN CLASSIFIER

After applying the CNN algorithm and training it on the NSL-KDD dataset, the result of matrices (Accuracy, Precision, Recall, F1-Score) is showing in Table 4. The results of the confusion matrix for binary anomaly detection were as in the Figure 4.3.
The result of ROC Curve for CNN classifiers is getting a result (0.998). that is showing in Figure 4.4.

Table 4.7. Result & Metrics applied CNN on NSL-KDD Dataset.

Accuracy	Precision	Recall	F-score
0.9863	0.9845	0.9898	0.9872

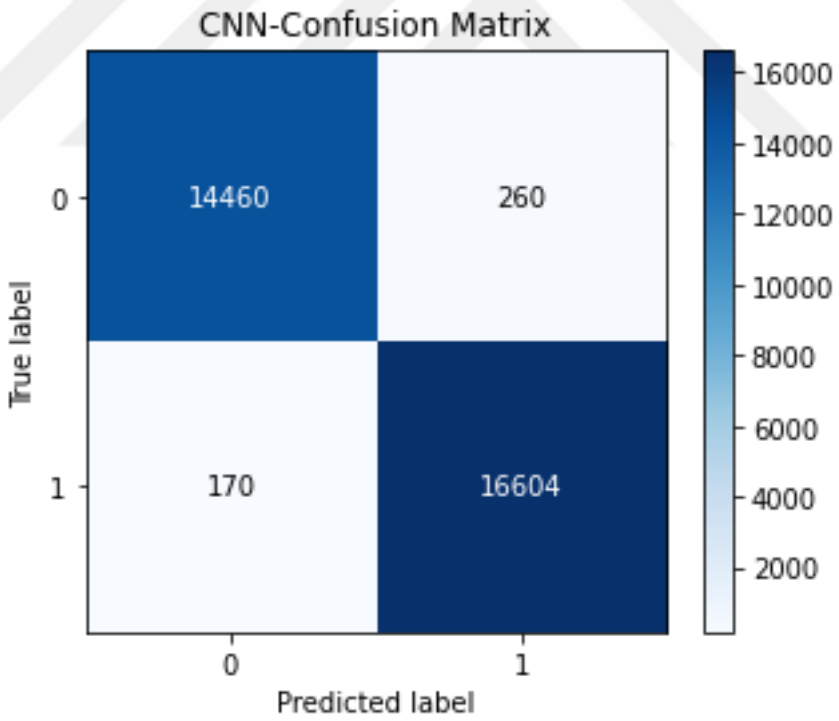

Figure 4.3. Confusion Matrix for applied CNN on NSL-KDD Dataset.

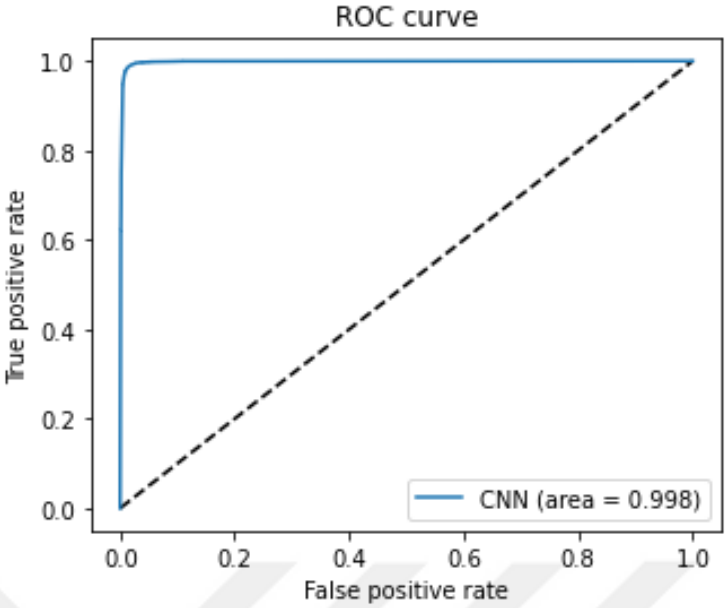

Figure 4.4. ROC Curve for CNN classifer.

4.7.3. RNN CLASSIFIER

After applying the RNN algorithm and training it on the NSL-KDD dataset, the results of the confusion matrix for binary anomaly detection were as in Figure 4.7.

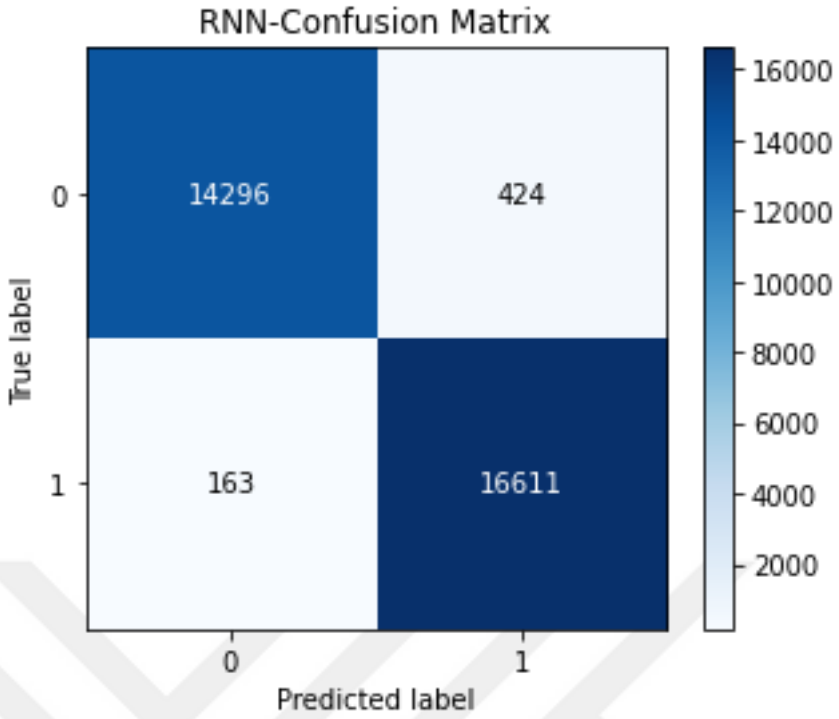

Figure 4.5. Confusion Matrix for applied RNN on NSL-KDD Dataset.

The performance of the RNN classifier was evaluated on the classification of binary anomaly attacks, and the metrics on which the performance of this half was evaluated were used (Accuracy, Recall, Precision F-score). Table 4.7 shows the confusion matrix was also used in the performance evaluation. The result of ROC Curve for RNN classifiers is got result (0.997). that showing in Figure 4.6.

Table 4.8. Result & Metrics applied RNN on NSL-KDD Dataset.

Accuracy	Precision	Recall	F-score
0.9813	0.9751	0.9902	0.9826

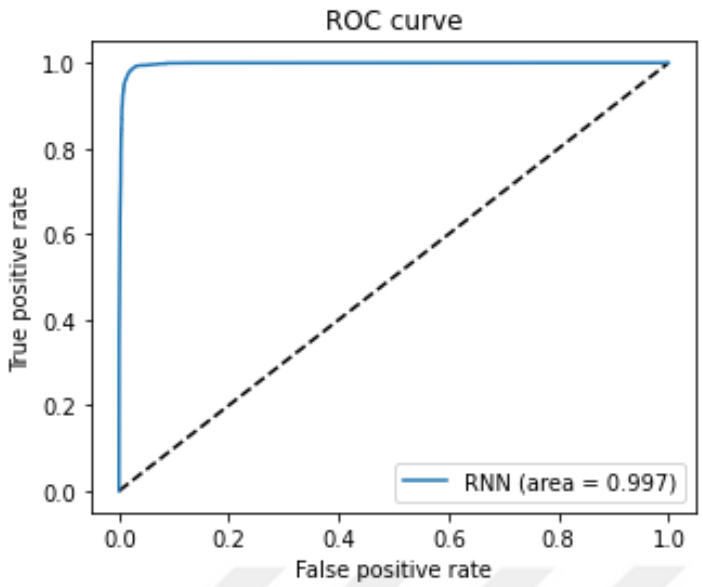

Figure 4.6. ROC Curve for RNN classifer.

4.7.4. LSTM CLASSIFER

After applying the LSTM algorithm and training it on the NSL-KDD dataset, the results of the confusion matrix for binary anomaly detection were as in the Figure 4.7.

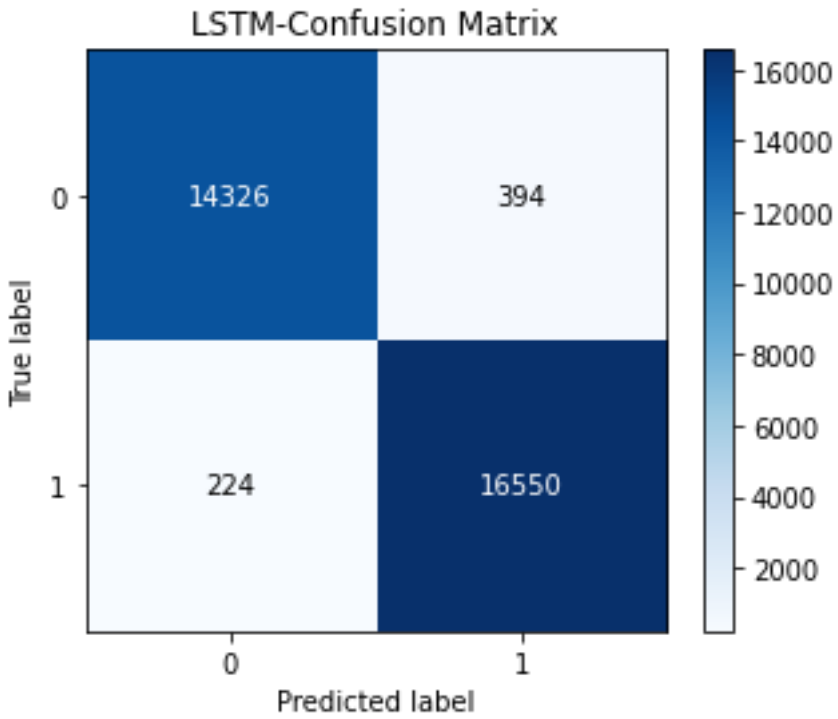

Figure 4.7. Confusion Matrix for applied LSTM on NSL-KDD Dataset.

The performance of the LSTM classifier was evaluated on the classification of binary anomalies attacks, and the metrics on which the performance of this half was evaluated were used (Accuracy, Recall, Precision F-score) as showing the result in Table 4.9. The result of ROC Curve for LSTM classifier is got result (0.997). that is showing in Figure 4.8.

Table 4.9. Result & Metrics applied LSTM on NSL-KDD Dataset.

Accuracy	Precision	Recall	F1-score
0.9804	0.9767	0.9866	0.9816

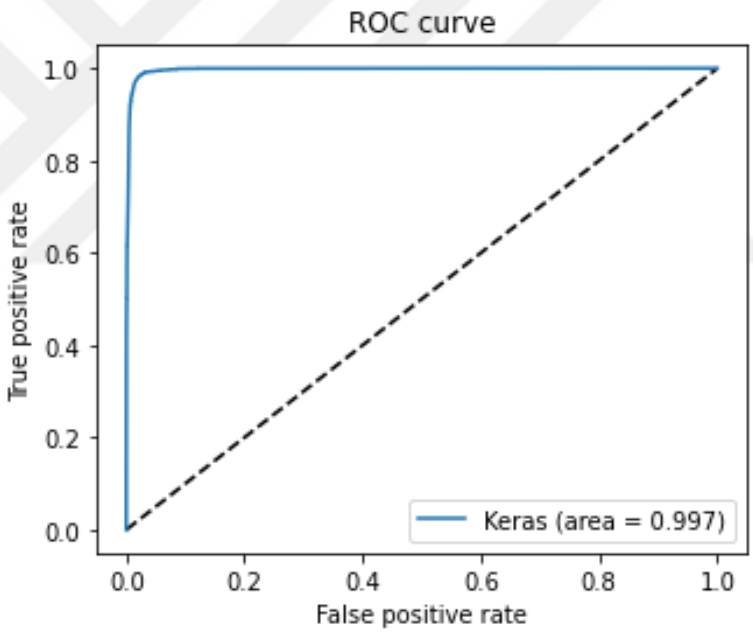

Figure 4.8. ROC Curve for LSTM classifier.

4.7.5. GRU CLASSIFER

After applying the GRU algorithm and training it on the NSL-KDD dataset, the results of the confusion matrix for binary anomaly detection were as in Figure 4.9.

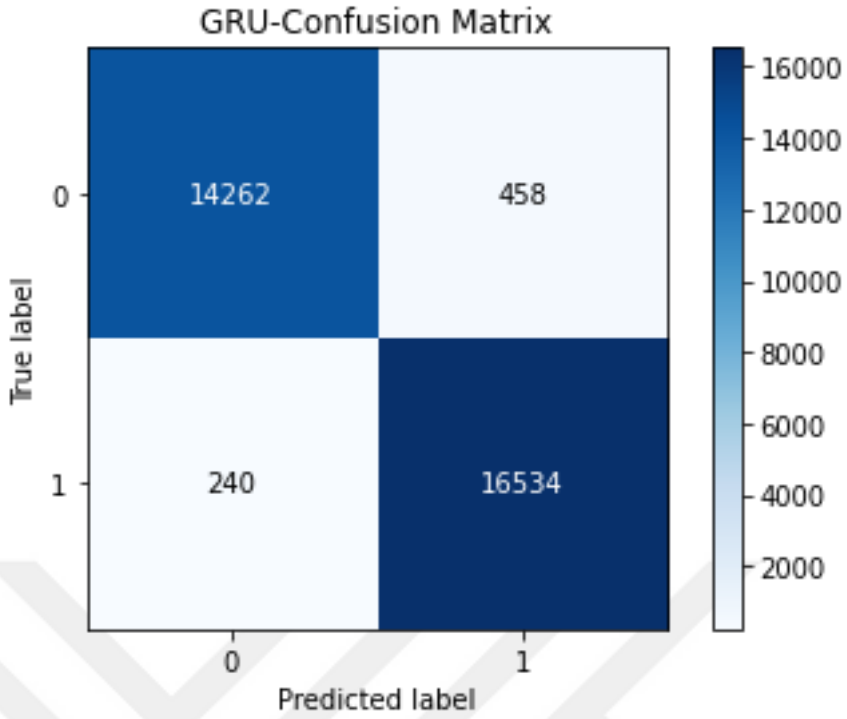

Figure 4.9. Confusion Matrix for applied GRU on NSL-KDD Dataset.

The performance of the GRU classifier was evaluated on the classification of binary anomaly attacks, and the metrics on which the performance of this half was evaluated were used (Accuracy, Recall, Precision F-score) that result is showing in Table 4.10. The result of ROC Curve for RNN classifiers is get result (0.997). that shown in Figure 4.10.

Table 4.10. Result & Metrics applied GRU on NSL-KDD Dataset.

Accuracy	Precision	Recall	F1-score
0.9778	0.973	0.9856	0.9793

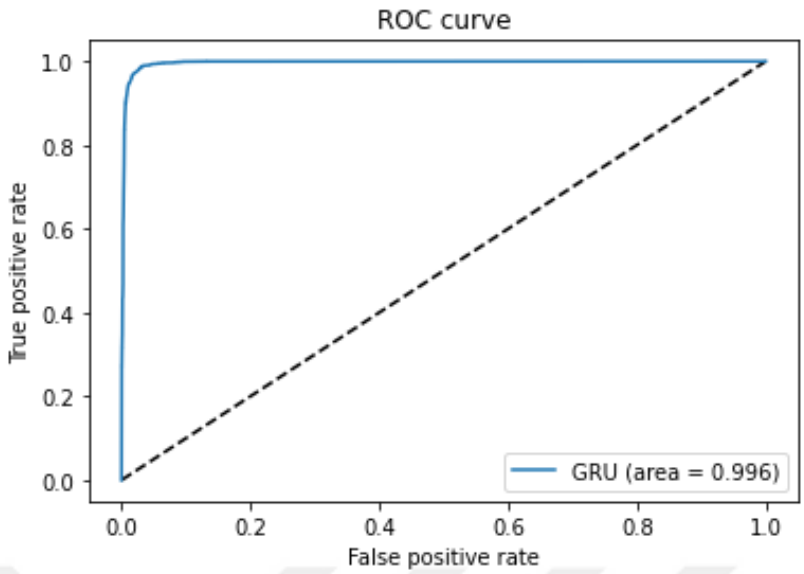

Figure 4.10. ROC Curve for GRU classifier.

4.8. RESULTS OF EXPERIMENT

In this section, a comparison was made for the performance of deep learning algorithms (DNN, CNN, RNN, LSTM, GRU). Where the results were compared with each other, and the results were evaluated on the basis of several different measures, then extracting the most famous and used measures in evaluating the performance of Each classifier (Accuracy, Precision, Recall, F1-score). The CNN classifier achieved the best result in (Accuracy, Precision, and F1-score) compared another classifier, as we see in Table 4.11 and in Figure 4.11. DNN classifier achieved good performance with the measures (Accuracy, Recall, F1-score), then came after the CNN classifier. Showing that in Figure 4.12 and the Table 4.10. RNN Classifier is getting high value in (Recall) metric that best other classifiers when compare, and in (Accuracy, F1-score) matrices is get value best than LSTM and GRU classifiers. LSTM classifier is got in (Accuracy, Recall, F1-score) metrices value best than GRU, also in (Precision) the value is height of RNN classifier. GRU classifier in (Accuracy, Precision, Recall, F1-score) was get results lower than rest classifiers when compared, as showing Table 4.11 and Figure 4.11.

Confusion Matrix is also important metrices that is depending on (TP, TN, FP, FN) parameters. Our aim is trying to get high values in (TP, TN) parameter and lower values in (FP, FN) parameters. CNN classifier is got high score in (TP, TN) parameters and lower score in (FP). DNN classifiers is get score coming after CNN in (TP, TN) and also for (FP) parameters. RNN classifier is get lower score in (FN) parameters compare to another classifiers. All results of rest classifiers are clarified in Table 4.12.

Table 4.11. Result of deep learning algorithms with Evaluation metrics.

	Accuracy	**Precision**	**Recall**	**F1-score**
DNN	0.9853	0.983	0.9896	0.9863
CNN	0.9863	0.9845	0.9898	0.9872
RNN	0.9813	0.9751	0.9902	0.9826
GRU	0.9778	0.973	0.9856	0.9793
LSTM	0.9804	0.9767	0.9866	0.9816

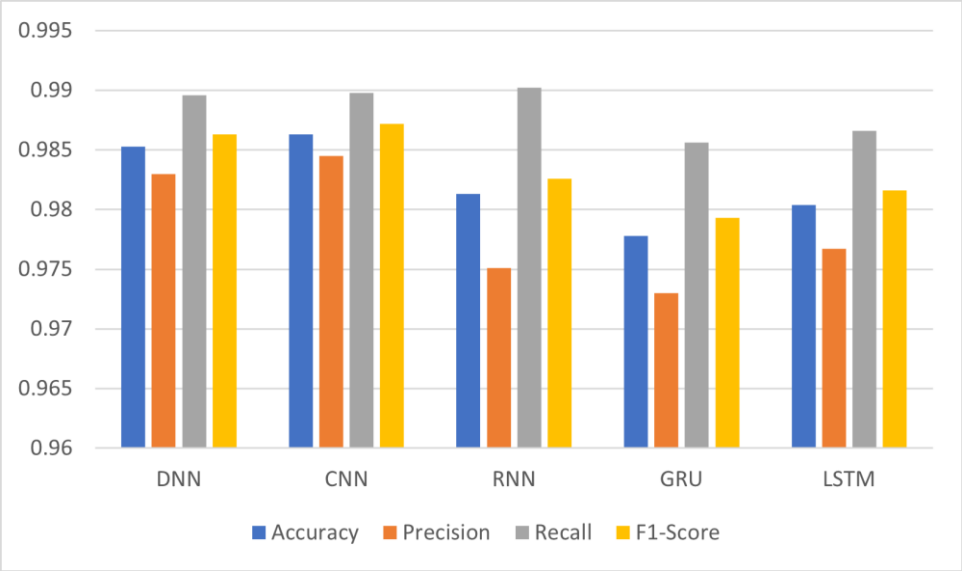

Figure 4.11. Result of deep learning algorithms with metrics classification.

Table 4.12. Result of Confusion Matrix in metrics of deep learning algorithms.

	DNN	CNN	GRU	RNN	LSTM
TP	14433	14460	14262	14296	14326
TN	16601	16604	16534	16611	16550
FP	287	260	458	424	394
FN	173	170	240	163	224

PART 6

CONCLUSION AND FUTURE WORKS

5.1. CONCLUSION

As explained in the previous chapters, the SDN structure is vulnerable to many attacks, especially those that target controllers that are located on the network and require an intrusion detection system to be located on the network. This system, called NIDS, must be designed to protect your network from a variety of attacks. In connection with this new environment, this system is built and designed to detect anomalies. In short, it relies on binary classification, the system is highly accurate, and one of the main issues researchers face when considering building an intrusion detection system is accuracy. As mentioned in the methodologies, classical and traditional algorithms are unable to deal with big data, and if these classical methods are used, accuracy will be low and performance will be poor. As a result, these approaches have relied on deep learning algorithms, which have a high ability and efficiency to deal with big data and produce high results and performance.

In this study, an anomaly detection system was built using deep learning algorithms (DNN, CNN, GRU, RNN, LSTM). NIDS assessment is based on several metrics (Accuracy, Accuracy, Recall, F1 Score) and other metrics such as confusion matrix for each classifier and ROC curve. These metrics are used to evaluate the performance of each classifier and determine which classifier achieves the best results.

NIDS is built on the TensorFlow framework and several libraries such as (numby, Pandas, Keras) and programs this system after performing data processing and feature selection methods and before entering the classifier. This approach was performed using Google's collaboration environment. These approaches also used the NSL-KDD dataset, which includes 12 features. These features are extracted from the 42 features

of the NSL-KDD dataset. completely reliable to compare and study the performance of structures. Furthermore, this dataset is considered recent by the researchers after updating and expanding it. Our approach depends on the binary classification in the evaluation. The classifiers have achieved a high degree of accuracy, especially the CNN algorithm, the rest of the classifiers also perform well. The results of our classifiers are almost close.

5.2. FUTURE WORKS

Completely reliable to compare and study the performance of structures. Furthermore, this dataset is considered recent by the researchers after updating and expanding it. Our approach depends on the binary classification in the evaluation. The classifiers have achieved a high degree of accuracy, especially the CNN algorithm, the rest of the classifiers also perform well. The results of our classifiers are almost similar. Another method and direction for future research are to apply these approaches in other environments such as (IoT) and (CLOUD) and measure the effectiveness and performance of this model in these environments. Furthermore, applying different datasets and comparing the performance of each, or applying combined methods, is also one of the important directions in the development of these approaches.

REFERENCES

1. D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc.* ***IEEE***, vol. 103, no. 1, pp. 14–76, Jan. 2015, doi: 10.1109/JPROC.2014.2371999.(2015).

2. D. Kreutz, F. M. V. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," *HotSDN 2013 - Proc. 2013* ***ACM SIGCOMM Work****. Hot Top. Softw. Defin. Netw.*, pp. 55–60, 2013, doi: 10.1145/2491185.2491199.(2013).

3. T. N., FrappierMarc, and MammarAmel, "Intrusion Detection Systems: A Cross-Domain Overview," ***IEEE Commun****.* ***Surv. Tutorials***, vol. 21, no. 4, pp. 3639–3681, Oct. 2019, doi: 10.1109/COMST.2019.2922584.(2019).

4. K. Durkota, V. Lisý, C. Kiekintveld, K. Horák, B. Bošanský, and T. Pevný, "Optimal Strategies for Detecting Data Exfiltration by Internal and External Attackers," *Lect. Notes Comput.* ***Sci****. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10575 LNCS, pp. 171–192, 2017, doi: 10.1007/978-3-319-68711-7_10.(2017).

5. S. Osken, E. N. Yildirim, G. Karatas, and L. Cuhaci, "Intrusion detection systems with deep learning: A systematic mapping study," *2019* ***Sci****. Meet. Electr. Biomed. Eng. Comput. Sci. EBBT 2019*, Apr. 2019, doi: 10.1109/EBBT.2019.8742081.(2019).

6. J. F. Kremer and B. Müller, "Cyberspace and international relations: Theory, prospects and challenges," ***Cybersp. Int. Relations Theory, Prospect. Challenges***, vol. 9783642374814, pp. 1–284, Dec. 2014, doi: 10.1007/978-3-642-37481-4.(2014).

7. H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in IoT networks: A survey," ***J. Netw. Comput. Appl.***, vol. 154, p. 102538, Mar. 2020, doi: 10.1016/J.JNCA.2020.102538.(2020).

8. K. Kimani, V. Oduol, and K. Langat, "Cyber security challenges for IoT-based smart grid networks," ***Int. J. Crit. Infrastruct. Prot.***, vol. 25, pp. 36–49, Jun. 2019, doi: 10.1016/J.IJCIP.2019.01.001.(2019).

9 M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for internet of things data analysis: a survey," ***Digit. Commun. Networks***, vol. 4, no. 3, pp. 161–175, Aug. 2018, doi: 10.1016/J.DCAN.2017.10.002.(2018).

10. A. A. Ghorbani, W. Lu, and M. Tavallaee, “Network Intrusion Detection and Prevention,” vol. 47, 2010, doi: 10.1007/978-0-387-88771-5.(2010).

11. S. Sezer *et al.*, “Are we ready for SDN? Implementation challenges for software defined networks,” ***IEEE Commun. Mag.***, vol. 51, no. 7, pp. 36–43, 2013, doi: 10.1109/MCOM.2013.6553676.(2013).

12. H. Liu and B. Lang, “Machine learning and deep learning methods for intrusion detection systems: A survey,” ***Appl. Sci.***, vol. 9, no. 20, Oct. 2019, doi: 10.3390/APP9204396.(2019).

13. M. Casado, T. Koponen, S. Shenker, and A. Tootoonchian, “Fabric: A retrospective on evolving SDN,” *HotSDN'12 - Proc.* ***1st ACM Int****. Work. Hot Top. Softw. Defin. Networks*, pp. 85–89, 2012, doi: 10.1145/2342441.2342459.(2012).

14. F. Hu, Q. Hao, and K. Bao, “A survey on software-defined network and OpenFlow: From concept to implementation,” ***IEEE Commun****. Surv. Tutorials*, vol. 16, no. 4, pp. 2181–2206, Apr. 2014, doi: 10.1109/COMST.2014.2326417.(2014).

15. FeamsterNick, RexfordJennifer, and ZeguraEllen, “The road to SDN,” ***ACM SIGCOMM Comput****. Commun. Rev.*, vol. 44, no. 2, pp. 87–98, Apr. 2014, doi: 10.1145/2602204.2602219.(2014).

16. I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, “Towards software defined cognitive networking,” ***2015 7th Int. Conf. New Technol. Mobil. Secur. - Proc. NTMS 2015 Conf. Work.***, **Sep. 2015**, doi: 10.1109/NTMS.2015.7266528.(2015).

17. I. Ahmad, “Improving Software Defined Cognitive and Secure Networking,” Jul. 2020, Accessed: Aug. 31, 2021. [Online]. Available: ***https://arxiv.org/abs/2007.05296v1***.(2020).

18. M. P. Fernandez, “Comparing OpenFlow controller paradigms scalability: Reactive and proactive,” ***Proc. - Int. Conf. Adv. Inf. Netw. Appl***. *AINA*, pp. 1009–1016, 2013, doi: 10.1109/AINA.2013.113.(2013).

19. C. Fancy and M. Pushpalatha, “Performance evaluation of SDN controllers POX and floodlight in mininet emulation environment,” *Proc.* ***Int. Conf. Intell. Sustain. Syst. ICISS 2017***, pp. 695–699, Jun. 2018, doi: 10.1109/ISS1.2017.8389262.(2017).

20. A. Tsakountakis, G. Kambourakis, and S. Gritzalis, “Towards effective wireless intrusion detection in IEEE 802.11i,” *Proc. - 3rd Int. Work. Secur. Priv. Trust Pervasive Ubiquitous Comput.* ***SecPerU 2007***, pp. 37–42, 2007, doi: 10.1109/SECPERU.2007.18.(2007).

21. R. A. Kemmerer and G. Vigna, “Intrusion Detection: A Brief History and

Overview (Supplement to Computer Magazine),” ***Computer (Long. Beach. Calif).***, vol. 35, no. 04, pp. 27–30, Apr. 2002, doi: 10.1109/MC.2002.10036.(2002).

22. “Snort - Network Intrusion Detection & Prevention System.” ***https://www.snort.org/ (accessed Sep. 01, 2021).***(2021)**.**

23. C. Mishra and D. L. Gupta, “Deep Machine Learning and Neural Networks: An Overview,” ***IAES Int. J. Artif. Intell***., vol. 6, no. 2, p. 66, Jun. 2017, doi: 10.11591/IJAI.V6.I2.PP66-73.(2017).

24. O. Bark, A. Grigoriadis, J. Pettersson, V. Risne, A. Siitova, and H. T. Yang, “A deep learning approach for identifying sarcasm in text,” *undefined*, 2017.(2017).

25. Y. Kim, “Convolutional neural networks for sentence classification,” ***EMNLP 2014 - 2014 Conf. Empir.*** *Methods Nat. Lang. Process. Proc. Conf.*, pp. 1746–1751, 2014.(2014).

26. Y. Su and C.-C. J. Kuo, “On Extended Long Short-term Memory and Dependent Bidirectional Recurrent Neural Network,” Feb. 2018, Accessed: Oct. 28, 2021. [Online]. ***Available: http://arxiv.org/abs/1803.01686.*** (2021).

27. K. Kalkan, L. Altay, G. Gür, and F. Alagöz, “JESS: Joint Entropy-Based DDoS Defense Scheme in SDN,” ***IEEE J. Sel. Areas Commun***., vol. 36, no. 10, pp. 2358–2372, Oct. 2018, doi: 10.1109/JSAC.2018.2869997.(2018).

28. P. Van Trung, T. T. Huong, D. Van Tuyen, D. M. Duc, N. H. Thanh, and A. Marshall, “A multi-criteria-based DDoS-attack prevention solution using software defined networking,” ***Int. Conf. Adv. Technol. Commun.***, vol. 2016-January, pp. 308–313, Jan. 2016, doi: 10.1109/ATC.2015.7388340.(2016).

29. K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, “Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments,” ***Comput. Networks***, vol. 62, pp. 122–136, Apr. 2014, doi: 10.1016/J.BJP.2013.10.014.(2014).

30. M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, “SPHINX: Detecting Security Attacks in Software-Defined Networks,” Feb. 2015, doi: 10.14722/NDSS.2015.23064.(2015).

31. Y. Fu, F. Lou, F. Meng, Z. Tian, H. Zhang, and F. Jiang, “An intelligent network attack detection method based on RNN,” ***Proc. - 2018 IEEE 3rd Int. Conf. Data Sci.*** *Cyberspace, DSC 2018*, pp. 483–489, Jul. 2018, doi: 10.1109/DSC.2018.00078.(2018).

32. B. Zhang, Y. Yu, and J. Li, “Network intrusion detection based on stacked sparse autoencoder and binary tree ensemble method,” ***2018 IEEE Int. Conf.***

Commun. Work. ICC Work. *2018 - Proc.*, pp. 1–6, Jul. 2018, doi: 10.1109/ICCW.2018.8403759.(2018).

33. Q. Niyaz, W. Sun, and A. Y. Javaid, "A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN)," ***ICST Trans. Secur. Saf.***, vol. 4, no. 12, p. 153515, Nov. 2016, doi: 10.4108/eai.28-12-2017.153515.(2016).

34. M. Gupta and S. K. Shrivastava, "Intrusion Detection System based on SVM and Bee Colony," ***Int. J. Comput. Appl.***, vol. 111, no. 10, pp. 27–32, Feb. 2015, doi: 10.5120/19576-1377.(2015).

35. I. Kumar, N. Mohd, C. Bhatt, and S. K. Sharma, "Development of IDS Using Supervised Machine Learning," ***Adv. Intell. Syst. Comput.***, vol. 1154, pp. 565–577, 2020, doi: 10.1007/978-981-15-4032-5_52.(2020).

36. S. K. Rajwar, D. I. Mukherjee, and D. P. K. Manjhi, "Machine Learning Methods for Network Intrusion Detection," ***SSRN Electron. J.***, Sep. 2018, doi: 10.48550/arxiv.1809.02610.(2018).

37. Q. Niyaz, W. Sun, A. Y. Javaid, and M. Alam, "A deep learning approach for network intrusion detection system," ***EAI Int. Conf. Bio-inspired Inf. Commun. Technol.***, 2015, doi: 10.4108/EAI.3-12-2015.2262516.(2015).

38. M. Maithem and G. A. Al-sultany, "Network intrusion detection system using deep neural networks," ***J. Phys. Conf. Ser.***, vol. 1804, no. 1, p. 012138, Feb. 2021, doi: 10.1088/1742-6596/1804/1/012138.(2021).

39. Supriya Shende, "Long Short-Term Memory (LSTM) Deep Learning Method for Intrusion Detection in Network Security," ***Int. J. Eng. Res.***, vol. V9, no. 06, Jul. 2020, doi: 10.17577/IJERTV9IS061016.(2020).

40. A. Prabhakaran, V. Kumar Chaurasiya, S. Singh, and S. Yadav, "An optimized deep learning framework for network intrusion detection system (NIDS)," *2020* ***Int. Conf. Eng. Telecommun***. *En T 2020*, Nov. 2020, doi: 10.1109/ENT50437.2020.9431266.(2020).

41. N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," ***IEEE Trans. Emerg. Top. Comput. Intell.***, vol. 2, no. 1, pp. 41–50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.(2018).

42. A. O. Alzahrani and M. J. F. Alenazi, "Designing a network intrusion detection system based on machine learning for software defined networks," ***Futur. Internet***, vol. 13, no. 5, 2021, doi: 10.3390/FI13050111.(2021).

43. T. M. Nam *et al.*, "Self-organizing map-based approaches in DDoS flooding detection using SDN," ***Int. Conf. Inf. Netw.***, vol. 2018-January, pp. 249–254, Apr. 2018, doi: 10.1109/ICOIN.2018.8343119.(2018).

44. A. Abubakar and B. Pranggono, “Machine learning based intrusion detection system for software defined networks,” ***Proc. - 2017 7th Int. Conf. Emerg. Secur. Technol. EST 2017***, pp. 138–143, Oct. 2017, doi: 10.1109/EST.2017.8090413.(2017).

45. A. Prakash and R. Priyadarshini, “An Intelligent Software defined Network Controller for preventing Distributed Denial of Service Attack,” ***Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2018***, pp. 585–589, Sep. 2018, doi: 10.1109/ICICCT.2018.8473340.(2018).

46. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, “Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks,” ***2018 4th IEEE Conf. Netw. Softwarization Work.*** *NetSoft 2018*, pp. 462–469, Sep. 2018, doi: 10.1109/NETSOFT.2018.8460090.(2018).

47. I. I. Kurochkin and S. S. Volkov, “Using GRU based deep neural network for intrusion detection in software-defined networks,” ***IOP Conf. Ser. Mater. Sci.*** *Eng.*, vol. 927, no. 1, Sep. 2020, doi: 10.1088/1757-899X/927/1/012035.(2020)**.**

48. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, “Deep learning approach for Network Intrusion Detection in Software Defined Networking,” ***Proc. - 2016 Int. Conf. Wirel. Networks Mob. Commun. WINCOM 2016 Green Commun. Netw.***, pp. 258–263, Dec. 2016, doi: 10.1109/WINCOM.2016.7777224.(2016).

49. J. Hussain and V. Hnamte, “A Novel Deep Learning Based Intrusion Detection System : Software Defined Network,” ***2021 Int. Conf. Innov. Intell. Informatics, Comput. Technol. 3ICT 2021***, pp. 506–511, Sep. 2021, doi: 10.1109/3ICT53449.2021.9581404.(2021).

50. Y. Hande and A. Muddana, “Intrusion Detection System Using Deep Learning for Software Defined Networks (SDN),” ***Proc. 2nd Int. Conf. Smart Syst. Inven. Technol. ICSSIT 2019***, pp. 1014–1018, Nov. 2019, doi: 10.1109/ICSSIT46314.2019.8987751.(2019).

51. A. H. Janabi, T. Kanakis, and M. Johnson, “Convolutional Neural Network Based Algorithm for Early Warning Proactive System Security in Software Defined Networks,” ***IEEE Access***, vol. 10, pp. 14301–14310, 2022, doi: 10.1109/ACCESS.2022.3148134.(2022) **.**

52. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” ***IEEE Symp***. *Comput. Intell. Secur. Def. Appl. CISDA 2009*, Dec. 2009, doi: 10.1109/CISDA.2009.5356528.(2009)**.**

53. “NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB.” https://www.unb.ca/cic/datasets/nsl.html (accessed May 07, 2022).(2022).

RESUME

Mahmood Radhi Hadi graduated from primary education. He completed his secondary education in Qutaibah in Diwaniah, then obtained a bachelor’s degree from Baghdad / Network Engineering in 2018. He moved to Karabük / Turkey in 2012 He started his master's education in the Department of Computer Engineering at Karabük University.