DDOS ATTACK DETECTION BASED ON MACHINE LEARNING

2022
MASTER THESIS
COMPUTER ENGINEERING

AHMED SARDAR AHMED ISSA

Thesis Advisor
Assist.Prof.Dr. Zafer ALBAYRAK

DDOS ATTACK DETECTION BASED MACHINE LEARNING

Ahmed Sardar Ahmed ISSA

T.C.

Karabuk University

Institute of Graduate Programs

Department of Computer Engineering

Prepared as Master Thesis

Thesis Advisor

Assist.Prof.Dr. Zafer ALBAYRAK

KARABUK

January 2022

I certify that in my opinion the thesis submitted by AHMED SARDAR AHMED ISSA titled "DDOS ATTACK DETECTION BASED ON MACHINE LEARNING" is fully adequate in scope and in quality as a thesis for the degree of Master of Science.

Assist.Prof.Dr. Zafer ALBAYRAK .....
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. January 7, 2022

Examining Committee Members (Institutions)	Signature
Chairman : Prof.Dr. Ahmet ZENGİN (SAU)	.....
Member : Assist.Prof.Dr. Zafer ALBAYRAK (SUBU)	.....
Member : Assist.Prof.Dr. Muhammet ÇAKMAK (KBU)	.....

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

Ahmed Sardar Ahmed ISSA

ABSTRACT

M. Sc. Thesis

DDOS ATTACK DETECTION BASED ON MACHINE LEARNING

Ahmed Sardar Ahmed ISSA

Karabük University
Institute of Graduate Programs
The Department of Computer Engineering

Thesis Advisor:
Assist. Prof. Dr. Zafer ALBAYRAK
January 2022, 62 pages

Distributed denial-of-service (DDoS) attacks are almost always placed at the top of the hierarchy of attacks facing networks and Intrusion Detection Systems (IDS). For the reason that these attacks cause servers to fail, causing users to be inconvenienced when requesting service from those servers, as well as causing the company's reputation to suffer and revenue to be lost. Therefore, this study suggested a modern method that is constructed from two of the best deep learning algorithms. Therefore, this study suggested a modern method that is constructed from two of the best deep learning algorithms to detect DDoS attacks more accurately, namely the CLSTMNet. The CLSTMNet architecture was composed of seven layers that were compacted from the Convolutional Neural Networks (CNN) layers and the Long Short-Term Memory (LSTM) layers. The performance of CLSTMNet was compared with the performance of both CNN and LSTM by applying them to the NSL-KDD dataset. The performance evaluated utilizing four metrics: accuracy, precision, recall, and F1 score. Experimental results illustrated that CLSTMNet had outperformed compared to others

in all metrics. When compared to most previous work that used various machine learning algorithms, our model has the highest accuracy.

Key Words : DDoS, IDS, machine learning, deep learning, CNN, LSTM, NSL-KDD.

Science Code : 92403

ÖZET

Yüksek Lisans Tezi

DDOS ATTACK DETECTION BASED ON MACHINE LEARNING

Ahmed Sardar Ahmed ISSA

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi Zafer ALBAYRAK
Ocak 2022, 62 sayfa

Dağıtılmış hizmet reddi (DDOS) saldırıları, ağlara ve Saldırı Tespit Sistemlerine (STS) yönelik yapılan saldırıların başında gelmektedir. Bu saldırılar sunucuların hizmet dışı kalmasına, kullanıcıların bu sunuculardan hizmet almasında sıkıntı yaşamalarına; ayrıca firmaların itibar ve gelir kaybetmesine neden olmaktadır. Bu nedenle bu çalışma en doğru sonuçlar veren iki derin öğrenmesi algoritmasından oluşturulmuş bir yöntem önermektedir. Bu yöntem CLSTMNet olarak önerilmiştir. CLSTMNet mimarisi evrişimsel sinir ağları (CNN) katmanlarından ve uzun-kısa süreli (LSTM) algoritmasının performansı ile NSL-KDD veri setine uygulanarak karşılaştırılmıştır. Performans kriteri olarak doğruluk, kesinlik, hatırlama ve F1-score değerleri alınmıştır. Yapılan deneysel sonuçlarda CLSMNet yaklaşımının tüm metriklerde mevcut algoritma yaklaşımlarına göre en iyi performans değerlerine ulaştığı görülmüştür.

Anahtar Kelimeler : DDoS, IDS, makine öğrenimi, derin öğrenme, CNN, LSTM, NSL-KDD.

Bilim Kodu : 92403

ACKNOWLEDGMENT

First of all, I would like to give thanks to my advisor, Assist. Prof. Dr. Zafer ALBAYRAK, for his great interest and assistance in preparation of this thesis. Also, I would like to give thanks to my family for supporting me. Moreover, I want to thank everyone who has stood by me.

CONTENTS

Page

Page

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

SYMBOLS AND ABBREVITIONS INDEX

SYMBOLS

μ : mean

σ : standard deviation

ABBREVITIONS

IDS : Intrusion Detection System

DoS : Denial of Service

DDoS : Distributed Denial of service

NIDS : Network Intrusion Detection System

HIDS : Host Intrusion Detection System

CPU : Central Processing Unit

TCP : Transmission Control Protocol

ICMP : Internet Control Message Protocol

UDP : User Datagram Protocol

HTTP : Hypertext Transfer Protocol

SIDDOS : Sql Injection DDoS

CFS : Correlation-based Feature Selection

DCF : DDoS Characteristic Features

CSE : Consistency Subset Evaluation

AE : Auto Encoder

ANN : Artificial Neural Network

DNN : Deep Neural Network

CNN : Convolutional Neural Network

RNN : Recurrent Neural Networks

DT : Decision Tree

RF : Random Forest

NB : Naïve Bayes

SVM : Support Vector Machines

KNN : K-Nearest Neighbors

MAD : Mean Absolute Deviation

AIS : Artificial Immune Systems

LVQ : Learning Vector Quantization

PCA : Principal Component Analysis

DCA : Dendritic Cell Algorithm

GA : Genetic Algorithm

POD : Ping of Death

DL : Data Link Layer

ReLU : Rectified Linear Unit

FFN : Feedforward Network

MLP : Multilayer Perceptron

BP : Backward Propagation

SGD : Stochastic gradient descent

SCCE : Sparse Categorical Cross Entropy

ADAM : Adaptive Moment Estimation

LSTM : Long Short-Term Memory

PART 1

INTRODUCTION

1.1 OVERVIEW

Nowadays, the use of the Internet is growing tremendously. 4.66 billion is the number of people who utilized the Internet in January 2021 [1]. It indicates that the number of computers and systems connected to the outside world is significant, which introduces vital security concerns. Since there are no perfectly secured systems, security components such as Intrusion Detection Systems (IDS) have to be introduced. The IDS can be defined as a system for detecting attacks quickly in the absence of human help.

The significant challenge that has been a concern since the first IDS was introduced is the misclassification due to the low accuracy of detecting an attack and the inability to identify modern attacks [2]. Researchers have been working on this issue as it adds a burden to security analysts. Such a burden can lead analysts to ignore severe cyberattacks unintentionally. IDSs have to know all the types of attacks and deal with each type individually. One of the most important of these attacks is the Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks. DoS attacks happen after many resources (e.g., memory, CPU time) are consumed, which is because of generating a large number of requests to a target at the same time. Therefore, it makes the service unavailable to legal users. Rather than DoS attacks, DDoS attacks occur when more than one device or computer generate a huge amount of demand for service at the same time. These computers or devices, called botnets, are controlled by the attacker. Consuming network bandwidth and buffer memory are the prime objectives of an attacker in the case of DDoS attacks [3].

To solve the mentioned challenges, researchers have been working on improving DDoS detection systems or other IDS by introducing machine learning (ML) techniques. In the case of intrusion detection systems, machine learning algorithms rely on analyzing massive data sets to gather useful information, such that they can detect abnormal behavior on the network [4]. The information gathered from the data sets can be used to enhance detection systems. It can be achieved by training the algorithms; hence, allowing the security analysts to gain the desired level of satisfaction in regards to the misclassification. On the other hand, machine learning algorithms are mainly based on a data set, so they stay a little slow when the data set is very large and faces new attacks. Also, the ML algorithms aren't robust enough to learn [5]. Therefore, researchers have turned their attention to deep learning (DL), which has recently been used as an IDS and has possessed successes in this field [6].

In recent years, the notion of deep learning, which mimics the human brain, has arisen as a new area of study. DL is not just in the IDS field, but also speech recognition, image processing, and language translation are just a few of the fields where it has had effectiveness [7]. In spite of the existing DL model's being better than other models in terms of analyzing and speed, a DL algorithm alone can't properly invert multidimensional attribute interconnections [8]. Therefore, the current thesis hybridizes two of the most common deep learning algorithms, the Convolutional Neural Network (CNN) and the Long Short-Term Memory neural network (LSTM), generating a novel method called the CLSTMNet. CNN was utilized to auto-select features [9]. LSTM was utilized for prediction [10]. The CLSTMNet was designed to achieve high performance in detecting DDoS attacks by relying on its own architecture, which is composed of seven layers.

1.2 MOTIVATION

Today’s networks are not perfectly secured [2], as new technologies emerge, and continuous change in network infrastructure occurs, new security challenges appear. Therefore, to cope with these challenges, multiple layers of security have to be designed securely, i.e., an appropriately defense-in-depth infrastructure has to be deployed.

One of these security layers is the Network Intrusion Detection System. An IDS helps in notifying if there is an ongoing sophisticated attack. Alternatively, if an attack was conducted earlier and by whom, indicating that it also helps in identifying the adversary and its actions.

Intrusion Detection Systems have to be continuously enhanced so that they are up-to-date and can detect new attacks. However, although a large number of studies have aimed to improve intrusion detection, there are still challenges to building such systems with high efficiency. For this reason, the research on IDSs should be specific to one type of attack in order to know the accuracy of attack detection and which method is best for this type of attack. Therefore, this thesis will mainly focus on DDoS attack detection, benefiting from deep learning algorithms.

1.3 PROBLEM STATEMENT

Complex issues, such as image recognition and machine translation, can be solved by learning significant data and have shown a lot of success in the processing of large data sets by deep learning [11]. Deep learning algorithms have outperformed human specialists in several cases. Transferring the technology for task DDoS attack detection has an inspirational effect on us.

Like in other application areas, for using deep learning on network intrusion detection, two major tasks are involved: deep learning model construction, deep learning model training and evaluation. We have identified two problems that are specific to the designs of deep learning for network intrusion detection. One is that few training datasets are available and there are insufficient training data in the datasets, which may make network training and evaluation not effective. The second issue is that most existing deep learning models either have a low detection accuracy or computational complexity, which as a whole presents a compromised detection efficiency. Therefore, in this thesis we particularly target these two problems.

1.4 OBJECTIVE

The objective of this research is to create a novel deep learning method that detects DDoS attacks more accurately. Generating this method from the combination of two of the best deep learning algorithms and comparing their performance with that of those algorithms' performance by implementing them on the most challenging dataset, the NSL-KDD dataset. Furthermore, comparing the accuracy of this method with the accuracy of many state-of-the-art methods that were applied to the same dataset.

PART 2

LITERATURE REVIEW

In this part of the study, a review of the many previous published works that were relevant to the NSL-KDD dataset was demonstrated. The reason is that I want to make a comparison between my results and their results in Part Five. These works included various techniques for detecting DDoS attacks. Furthermore, there are 18 of these works, which are organized by year.

In 2013, the researchers [12] studied the NSL-KDD dataset, which solves some of the problems that are found in KDD cup99. The findings of the study showed that the NSL-KDD dataset is extremely beneficial for comparing various intrusion detection algorithms. The utilized methodology uses all of the dataset's 41 features to assess the potential for patterns of intrusion, but doing so takes time and impairs system performance. The dataset contains features that are unnecessary for the process, and other features that are irrelevant to the specific attack. In this case, the CFS Subset is used to simplify the dataset, making it easier to handle. With and without extracting features, it was completely obvious that Random Forest was superior to all of the other algorithms in both cases. This analysis demonstrated that Random Forest is capable of speeding up the training and testing methods for intrusion detection, which is critical for network applications such as network security. Random Forest was also capable of providing the highest testing accuracy possible in the case of reduced feature sets.

An ensemble of neuro-fuzzy classifiers presented by Boroujerdi and Ayat [13] in 2013 was a mix of complex classifiers with a simple and efficient boosting approach to accelerate DDoS attack detection. In terms of accuracy, this approach was superior to the standard machine learning methods used in intrusion detection systems, with lower

false alarm rates. Additionally, the design significantly increased computing efficiency by redistributing workload between classifiers while employing the capabilities of each classifier to recognize a certain type of attack while utilizing a special collection of features throughout the detection process.

In2015, Dhanabal and Shantharajah [14] conducted an analysis and evaluation of the NSL-KDD dataset in order to determine the effectiveness of different classifiers in anomaly detection in network traffic patterns. Additionally, they've studied the links between commonly used network protocol stacks and the various types of intrusion activities that hackers use to craft abnormal network traffic. A data mining tool known as WEKA was used to perform the analysis. The study concluded that by using figures and tables to visually examine the NSL-KDD dataset, researchers gained a solid knowledge of the dataset. It was also an important piece of information that the vast common of attacks employed the inherent shortcomings of the TCP protocol.

Yusof et al. [15] made it possible to create an effective IDS by employing a feature selection technique with ML in 2017. The utilized approach is a blending of two feature selection techniques: consistency subset evaluation (CSE) and DDoS characteristic features (DCF). Some feature selection techniques, like as CSE and DCF, are utilized to locate the most important features in the NSLKDD dataset for DDoS attacks. The full DDoS feature used is 1,2,3,4,5,6,7,8,10,14,23,29,30,32,33 and 36. The experiment results show that their proposed model outperforms these other four feature selection techniques (i.e. IG, chi-squared, gain ratio and CFS) in terms of accuracy and performance.

Kushwah and Ali [16] suggested an approach that was made up of an artificial neural network and a black hole optimization algorithm for detecting DoS assaults in clouds in 2017. The NSL-KDD dataset, which had 12,500 training samples and 2,597 test samples, was used for the studies. Ten trials were conducted. The highest level of accuracy was reached, at 96.30 percent.

In 2017, Igbe et al. [17] suggested an algorithm relying on the Artificial Immune System (AIS) called the Dendritic Cell Algorithm (DCA). This algorithm was utilized

for recognizing DDoS assaults, which was the cause of damaging the network many times. Incoming network traffic is classified into two categories: "regular" and "DoS/DDoS attack". Based on their results, their technique has great accuracy for detecting DoS and DDoS attacks.

Derakhsh et al. [18] for recognizing DDoS assaults on clouds offered a genetic algorithm (GA) as a feature selection in 2018. The researcher utilized a Bernoulli Naïve Bayes ML algorithm for classifying attacks. On the other hand, the achieved performance was low.

In 2018, Hoon et al. [19] published a paper that present new information about the performance of various learning methods in detecting DDoS attacks, and also how the parameters will actually effect the model's performance. All experiments were conducted by utilizing two data mining tools, H2O and WEKA. In the end, the researcher performed a comparison between many algorithms in terms of accuracy. The Distributed Random Forest (DRF) obtained a high accuracy rate.

In 2018, Idhammad et al. [20] suggested a method relying on three feature selection techniques and an ensemble classifier. The feature selection techniques were: information gain ratio, co-clustering, and entropy estimation. Also, the utilized ensemble classifier for classifying DDoS assaults was the Extra-Trees classifier. Three famous datasets were used to carry out a number of experiments in order to evaluate the method's efficacy: the UNSW-NB15 dataset, the UNB ISCX 12 dataset, and the NSL-KDD dataset. In the end, when compared to previous published work on detecting DDoS assaults, the experiment's results were satisfactory in terms of accuracy.

In 2019, researchers [21] suggested a semi-supervised ML method relied on hybridization of unsupervised and supervised techniques, and compared it with them in terms of the performance of detecting DDoS attacks. Unsupervised part consists of some estimation steps including clustering which reduces the false positive rates and increases the accuracy by reducing irrelevant data. In supervised part Random forest (RF) algorithm was used to accurately classify the DDoS attack data and it also reduces

the false positive rate of unsupervised part. By implementing it using the Python programming language, the accuracy obtained for this approach is 93% which is better compared to 81% supervised and 57% unsupervised.

In 2019, Mukhametzyanov et al. [22] suggested the neural network (NN) method that consists of only one hidden layer. The architecture of this method was composed of eleven neurons in the input layer, thirty-three neurons in a single hidden layer, and an output neuron. This NN model was applied on the analytical platform Deductor. During training, the accuracy of classification was 97.94 percent. However, during testing, the accuracy of classifying was 97.87 percent. Thus, the neural network model is good at recognizing DDoS attacks, which makes it an adequate model for the IDS.

In 2019, Verma et al. [23] suggested a hybrid method consisting of Mean Absolute Deviation (MAD) thresholding and RF to extract DDoS attacks from a legal request. The MAD was utilized as a feature selection technique, and RF was employed as a classifier. It had the ability to divide the DDoS attacks depending on their own system: UDP, TCP, and ICMP. Using MAD-RF, they achieved an accuracy of 98.226 percent, a detection rate of 98.066 percent, a false alarm rate of 0.019 percent, a precision of 98.34 percent, and an F1-score of 0.983 percent. As well, the utilized feature selection strategy was also found to perform better than the proposed strategy after comparing with them.

In 2019, Azizi and Hosseini [24] suggested a hybrid framework. The researchers separated processes on two sides to detect DDoS attacks quickly: client and proxy. Moreover, each side did its own work. The data collection, the feature extraction, and the divergence test were done on the client side. On the proxy side, the following algorithms were utilized: k-nearest neighbors (K-NN), decision tree (DT), random forest (RF), the naïve Bayes (NB), and multilayer perceptron (MLP). There were a variety of feature selections because each algorithm extracted a varied set of features. Also, each algorithm evaluated their features depending on their specified features. The researchers suggested using the KNIME platform to apply their experiments. RF achieved higher performance when applied to the employed datasets, the dataset made available by Alkasassbeh et al. [25] and the NSL-KDD dataset. Additionally, behavior-

based attacks were detected across a wide range of scenarios. In the end, the researchers utilized various classifier algorithms rather than utilizing only one, which improved the capability to recognize unknown attacks.

Das et al. [26] suggested a network IDS (NIDS) by employing an ensemble method for classification and also by minimizing the dataset features in 2019. When they carried out their experiments, they used the NSL-KDD dataset with a decreased number of features, down to 24, in order to better detect DDoS attacks. As a result of their domain expertise, they selected the most relevant features that can only have an impact on a DDoS attack and no other type of attack. The complete list of DDoS features that were used is as follows: 2, 3, 4, 5, 7, 8, 10, 13, 23, 24, 25, 26, 27, 28, 29, 30, 33, 34, 35, 36, 38, 39, 40, and 41 (in that order). They selected MLP, SMO, IBK, J48, and IBK classifiers from a variety of classification families in order to create a diverse set of classifiers. 10-folds of cross-validation in combination with voting for the majority serves to bind classifiers together. Using an NIDS with reduced feature set, they claim to be able to successfully detect 99.1% of DDoS attacks.

In 2020, Ma et al. [8] suggested a unique DL method to improve Internet of Things IDS (IoT-IDS). The suggested DL method was the CNN, depending on the attributed fusion mechanism and Cross Categorical Entropy loss function. Moreover, this CNN was comprised of multiple layers, and each layer had various neurons. TensorFlow was used in the implementation of the model, and experiments on the NSL-KDD dataset have been performed. The experiment demonstrated that the proposed methodology outperformed previous methods that used CNN, support vector machines (SVM), DT, Bayesian classifiers, KNN, and recurrent neural networks (RNN).

In 2020, Prathyusha and Kannayaram [27] suggested an approach relying on AIS to recognize and decrease DDoS assaults in cloud computing. This approach's initial contribution was to define and study the most relevant DDoS attack features. This methodology was very different from the traditional ones, as it determines and analyzes the probable features that may be used in a DDoS attack in the NSL-KDD dataset. The mechanism's evaluation depends on factors like the accuracy, precision, specificity, and sensitivity that apply to virtualized systems. The conclusion from this

method is that the simulation of the AIS can be an excellent strategy for improving classical defenses for DDoS attack detection in cloud computing. Considering this research results, this approach is moving ahead with the research that leads to the first stages of new research areas.

In 2020, Bhardwaj et al. [28] suggested an approach that uses two techniques: one for classification that was represented by the deep neural network and another one for feature selection that was represented by the auto encoder. They first created a naive system by putting the parameter values randomly, then they enhanced it by changing them. They implemented their work by applying it to the CICIDS2017 dataset and the NSL-KDD dataset. Finally, they evaluated their findings with the previous published work; the obtained accuracy was equal to 98.43% for the NSL-KDD dataset and 98.92% for the CICIDS2017 dataset.

In 2020, Bagyalakshmi and Samundeeswari [29] suggested two feature selection techniques: Principal Component Analysis (PCA) and Learning Vector Quantization (LVQ). The first one is for reducing dimensions, and the second one is for filtering. The number of features selected by PCA was equal to 21, and the number of features selected by LVQ was equal to 20, out of 42 features in the NSL-KDD dataset. Then the result was classified by SVM, DT, and NB. After that, the resulting classification capability of each model is measured and compared. The findings demonstrated that the LVQ-DT has the best performance.

PART 3

THEORETICAL BACKGROUND

3.1. INFORMATION SECURITY

Information security revolves around going through certain phases to strengthen the security posture in a system [30]. As a goal, information security attempts to protect Confidentiality, Integrity, and Availability (the CIA triad). Confidentiality means the information is not readable by those who are not authorized, where the goal of integrity is to protect data from being modified. Availability is the ability to access certain information when needed by those who are authorized.

3.1.1. Information Security Process

In the process of information security, there are three main categories, which are prevention, detection, and response. In order to have a secure system, each phase requires maintenance, analysis, and organizing strategies to move to the next phase.

3.1.1.1. Prevention

In this phase, security policies, awareness training, and access controls must be designed and conducted to prevent attacks [31]. These procedures have to be implemented early on as they are related to each other. Security policies are high-level security measures conducted by organizations to achieve desired security objectives. Moreover, security policies are based on three main categories, which are physical controls, logical controls, and administrative controls [32].

Awareness training is a very critical control [31]. Organizations always try to educate their employees to avoid being victims of cyberattacks. Awareness training programs highlight the importance of security, how to avoid being a subject for attacks, providing knowledge of best practices (passwords, email, remote work, secure browsing, etc.), how to report a security issue, and so on.

Access control [30] provides an identity and a specific level of authentication and authorization to each user. An identity is a unique identifier, and in order to use specific resources in a system, the identity has to be authenticated or validated by three main factors, "something you know, something you are and something you have". Based on the provided information, a certain level of authorization will be given.

3.1.1.2. Detection

This phase is an essential one, as defending the network against malicious attacks is one of the most critical procedures and must be handled by network administrators and security analysts [30]. One technology that can be used to discover intrusions is an intrusion detection system (IDS). As mentioned earlier, intrusion detection systems always have to be improved because no matter how secure the system is, there will exist attacks that are capable of compromising the system. An IDS is capable of detecting a conducted attack, for instance, by checking the signatures, and modified files and configuration. Nevertheless, when an attack or a breach occurs, the IDS alerts the network's administrators, then they have to follow a response plan, as will be discussed shortly.

3.1.1.3 Response

Organizations have always to be prepared for an incident to defend their systems [31]. It can be achieved by establishing an incident response strategy. A response plan must describe which procedures to be taken during an incident. Furthermore, for each type of incident, there must be a specific type of response depending on the threat level. During the response phase, several steps must be conducted, including containment, eradication, and recovery. These steps revolve around selecting a strategy to contain

an attack, gathering shreds of evidence to support incident response documentation, identifying attackers, and eradicating the incident impact on business operations.

3.1.2. Intrusion detection system IDS

Intrusion detection is a possible way of avoiding an attacker from performing a distributed denial of service (DDoS) attack in a secured network. An excellent intrusion detection system is the system can recognize a new DDoS speedily and without the need for human assistance. IDS can be divided into Host Intrusion Detection System (HIDS) and Network Intrusion Detection System (NIDS), each of which include the following.

- HIDS: A network device or workstation can implement this type of IDS. HIDS is capable of protecting one device from a DDoS attack, however it does not provide network monitoring.
- NIDS: is an IDS designed to protected network, and its used identify and classify whole network traffic from all nodes. Figure 3.1 illustrates the different between HIDS and NIDS structure.

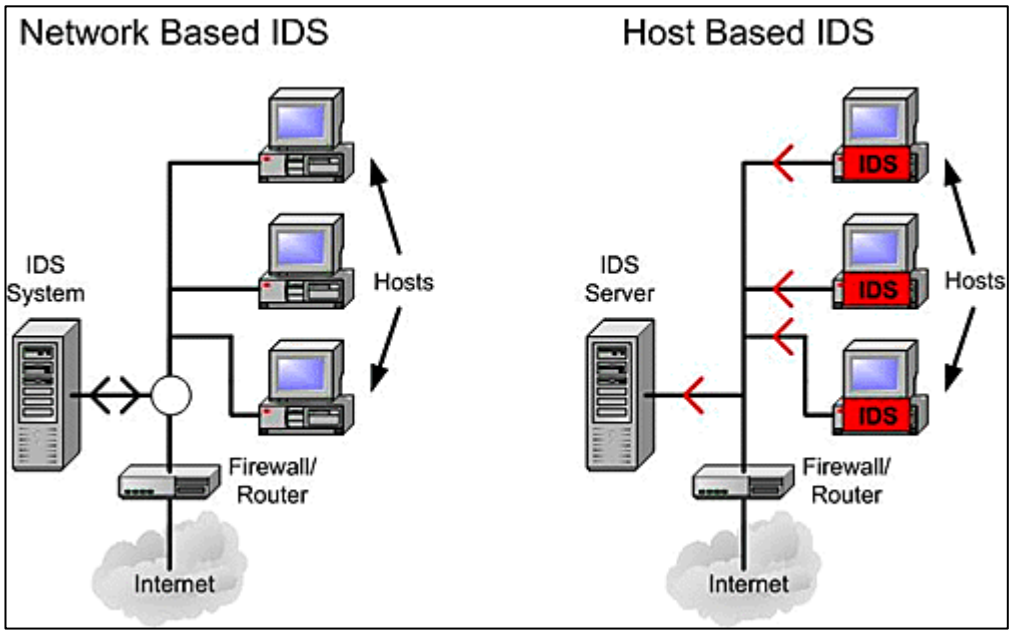

Figure 3.1. HIDS and NIDS structure [33]

IDS can be used to identify and classify network traffic either using signature-based or anomaly-based methods. Anomaly-based approach is used to compare network traffic flow with previous baseline data, and as a result, it requires training data in order to perform logically and effectively. While a signature-based (also known as misuse detection), each individual packet is exposed to a signature-based focus, which is then compared to a previously stored signature or identified attack for verification. For anomaly-based recognition, training data is required; for signature-based recognition, a previously-stored signature is required. In additional, the detection rates of signature-based IDS are high for known intrusion but, not able to detect the unknown attacks [34].

3.1.3. Attack classification

Network intrusions can compromise the integrity, confidentiality and availability of a system. The increasing variety of networks makes the intrusions become more and more advanced, non-repetitive and highly concealed. Some typical attacks are Denial of Service, Trojan, Worm, Backdoor and User to Root. A short description of each attack is given below.

- Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are the most common attacks in network communications. The attacks can flood a system or a network server by sending enormous volume of illegitimate network traffic packets, making the service of the system or the network server unavailable to the legitimate users and causing substantially economic losses to the service provider. The denial of service is a broad attack and can be launched in many ways.
- Trojan is a malicious application that can intercept network communications and control a system to execute unwanted operations. Most of Trojan applications utilize a shell technique to hide themselves to escape the detection of the anti-virus software. Therefore, they are often difficult to detect.
- Worm is similar to Trojan. It is also a malicious application. However, unlike Trojan, worm can self-replicate from one computer to another computer. The worm attack usually has various functions for different malicious purposes.

Some functions may just monitor the system processes, and some may only be activated when a computer performs certain critical operations.

- Backdoor is also a software and is commonly stealthily installed after a computer system or a web page background is compromised. It can be used to re-access the system or the web page background even if the related vulnerabilities have been fixed. Some types of backdoor attacks are even not launched by hackers, but are pre-installed to the systems by some unreliable manufacturers.
- •User to Root (U2R) is an operating-system-level attack. The most common U2R attack is buffer overflow. With buffer overflow, the system's access control can be illegitimately modified. For instance, the access control can be overwritten with a null or a large memory address that causes system disabled.

3.1.4. Distributed Denial of Service Attack

3.1.4.1. What is a DDoS attack?

When a Denial of Service (DoS) attack is launched, the attacker attempts to make network resources unavailable to legitimate users by flooding the host of service. DDoS (distributed denial of service) is same DoS attack but attack is launched from different sources. Generally, DoS attack is initiated from a single infected device or virtual machines utilizing an Internet connection whereas DDoS attacks are initiated from many different infected devices or virtual machines to overload the target systems. therefore, a DDoS attack gets made deadlier and very hard to deal with. Figure 3.2 shows a DoS and DDoS structure.

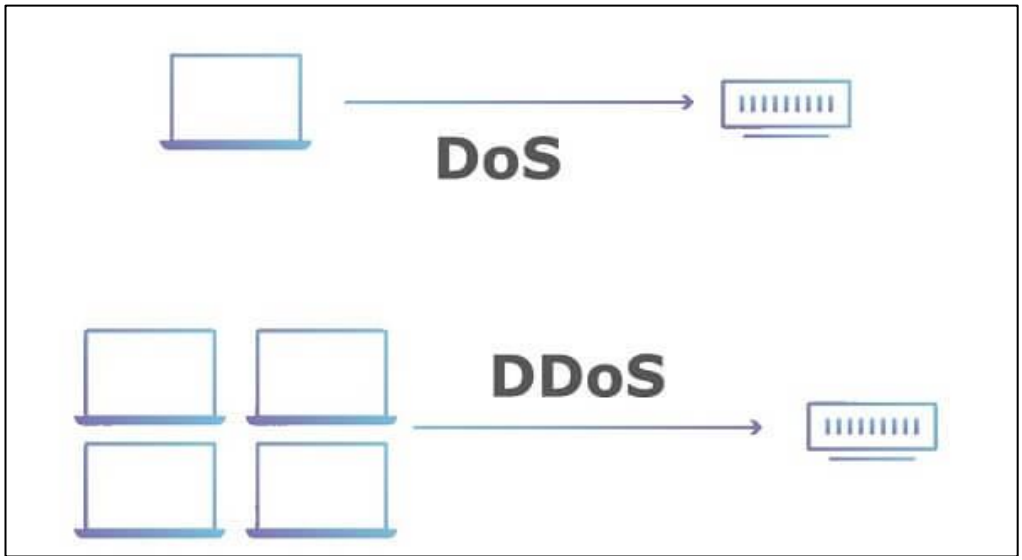

Figure 3.2. DoS and DDoS structure [35]

The term "weapon of mass destruction on the Internet" is often used to describe distributed denial of service (DDoS) attacks [36]. Even if an organization has implemented a typical security system, it will be virtually impossible to protect against a DDoS attack because of the large number of attacks in the same time and the attack is improved very fast. This is largely due to the fact that DDoS attacks try to simulate normal traffic but have increased exponentially. Because DDoS attacks simulate normal traffic, but have dramatically risen, this is mostly due to it. A distributed denial of service (DDoS) attack targeted GitHub, a platform for computer programmers, on February 28, 2018. The attack peaked at 1.35 Tbps [37]. Also in March 2018, NETSCOUT Arbor was hit by DDoS attacks that stopped at 1.7 Tbps [38]. In June 2020, Amazon platform for cloud computing services from Amazon Web Services (AWS), was hit by a massive DDoS attack earlier that year that peaked at 2.3 terabytes per second [37,38]. These are some of the biggest DDoS attacks in the world in recent years. This has led to huge losses in industry and government globally due to DDoS attacks in recent years [39]. These problems are caused by the devices interacting with remote applications, which allows malicious agent to control the devices. The main reasons for the increase in DDoS attacks are that implementing DDoS attacks is easy and simple, does not require a great deal of technological understanding on the part of the attacker, and there were many platforms and software that could be used to coordinate the attack [40]. In general, the attackers use many devices called botnet in the DDoS attacks quickly.

Figure 3.3 illustrates how the attacker puts authority on a system. That is by putting its control on the powerful server called the "Control Server" [41]. This server was rich in terms of memory size, bandwidth, CPU power, capacity, and capability. The attacker sends his commands via many PCs, which are known as botnets or agents. A botnet is a device or computer connected to the Internet that is controlled by a hacker and used to attack a victim remotely. The owner of this botnet does not realize he is a part of the attacking process or that there is malware on his computer. A DDoS attack is performed through proxies by the assaulter [42].

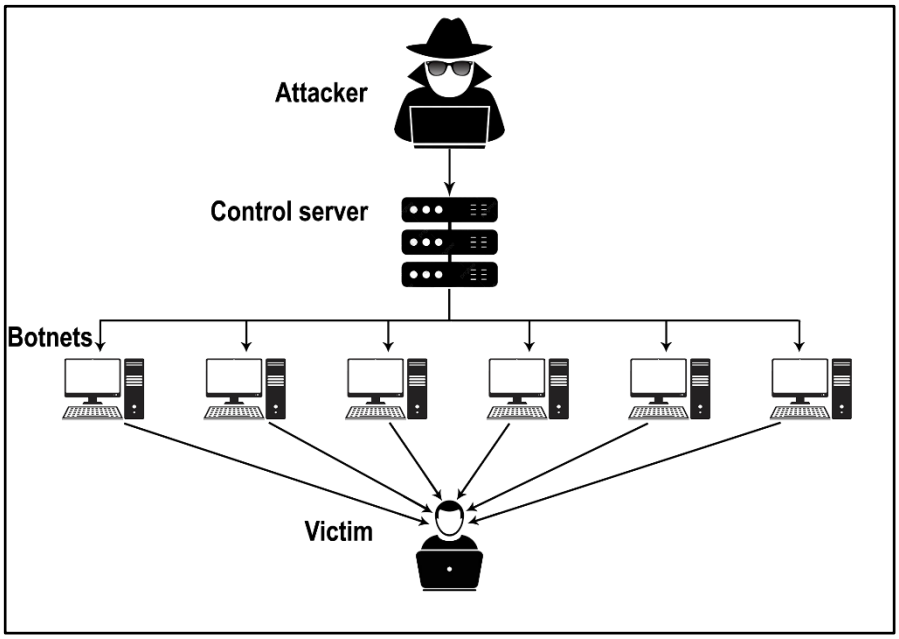

Figure 3.3. DDoS attack

3.1.4.2. DDoS attacks classification

DDoS attacks can be divided into two types its application layer and network layer [43] or can be divided into three types as follow [44]:

- Volume Based Attacks: UDP floods and other spoofed-packet floods are comprised in this category. Fully overwhelming the bandwidth of the victim side is a major attack purpose.

- Protocol Attacks: SYN floods, Smurf DDoS, Ping of Death, fragmented packet attack and other types of DDoS are covered. server and intermediary communications resources that can be consumed by this kind of attack.
- Application Layer Attacks: This kind include some advanced techniques such as SIDDOS, HTTP GET/POST floods, and attacks against Apache, Windows, and much other. The objective of these kind of attack is to overwhelm the web server with apparently legal and harmless queries.

3.1.4.3. Common DDoS attacks types

UDP Flood Attack

UDP flood is a form of Distrubuted DOS attack. Also, User Datagram Protocol (UDP) is connection less protocol. In this attack type, the host scans for applications related with the datagrams, when didnt found any of that, the host issues a "Destination Unreachable" packet back to the sender. A massive number of UDP packets is transmitted to the target machine from all of the workstations. [25]. The cumulative effect of this flood attacks is that the system will overloaded and thus not responding to legitimate traffic.

SYN flood

A SYN flood is a type of DDoS attack based on a gap in the TCP protocol called the three-way handshake. A three-way handshake will also be performed when a SYN request to establish a TCP connection with a host is followed by a SYN-ACK response from the host and then confirmed by an ACK response from the requester. A SYN flood occurs when the requester sends several SYN requests but the host's SYN-ACK does not respond or sends the SYN request from a faked IP address. As a result, we don't have three handshakes because the number three was overlooked. Finally, DoS will occur because the host will continue to wait for acknowledgement for each of the requests, rendering them unable to establish a new connection. Figure 3.4 illustrates SYN flooding attack.

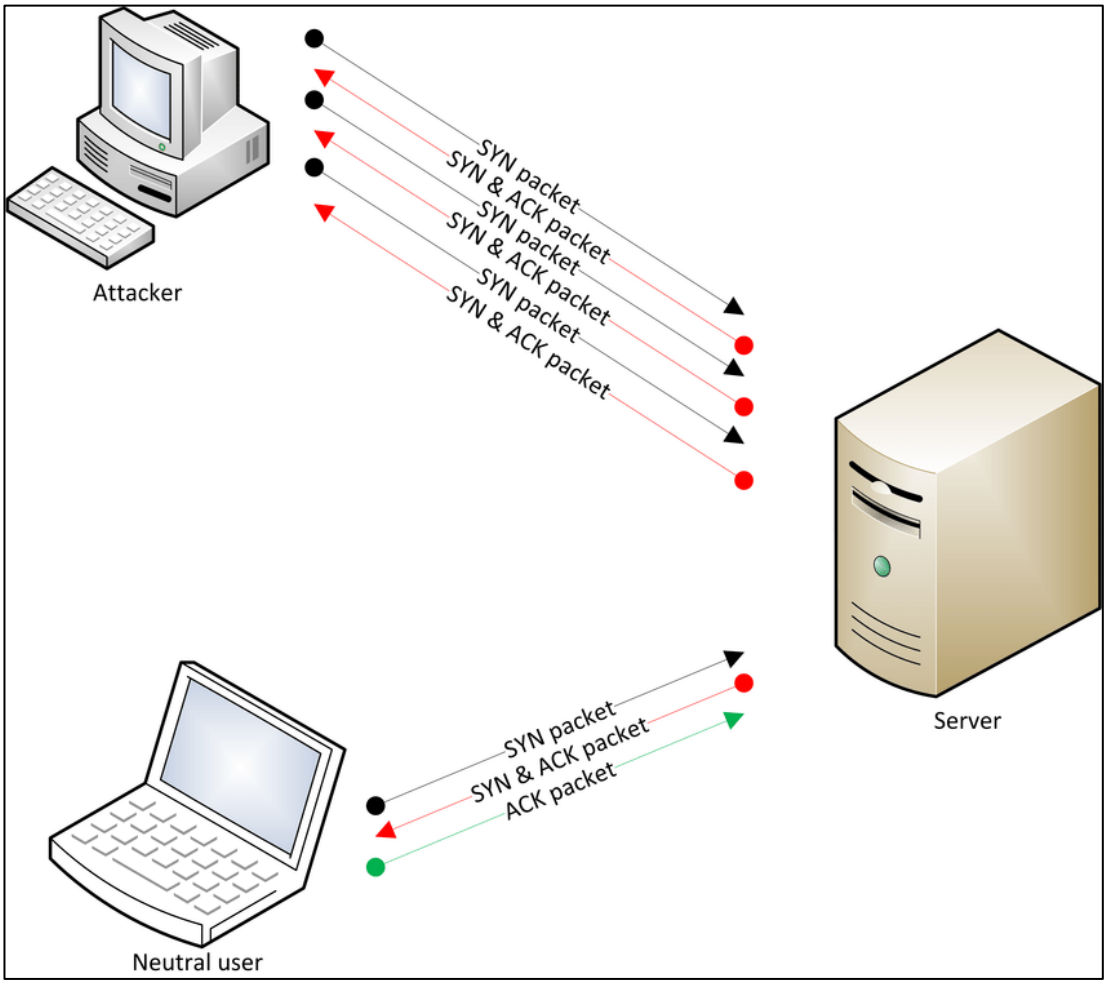

Figure 3.4. SYN flood [45]

Ping of Death

An attack when a harmful or malicious pings is sent repeatedly to a machine is called a “POD” attack. The length of IP packets with its header is 65,535 bytes. The Data Link Layer (DL) may place constraints on the maximum frame size. For example, on an Ethernet network, the frame limit is set at 1500 bytes. This situation would need numerous IP packets (known as fragments) to be fragmented into several packets and then reassembled by the destination host. The reassembled packet ends up being bigger than 65,535 bytes because of malicious modification of fragment content this is what happen in ping of death scenario. This might cause denial of service for genuine packets due to an overflow of storage buffers generated for the packet.

Smurf attack

The Smurf attack is a type of DDoS attack that occurs at the network layer. One of amplification shape of attack is the smurf attack that is broadcast networks address by using an ICMP request. ICMP is generally used for data interchange and to defining the operational status of the nodes.

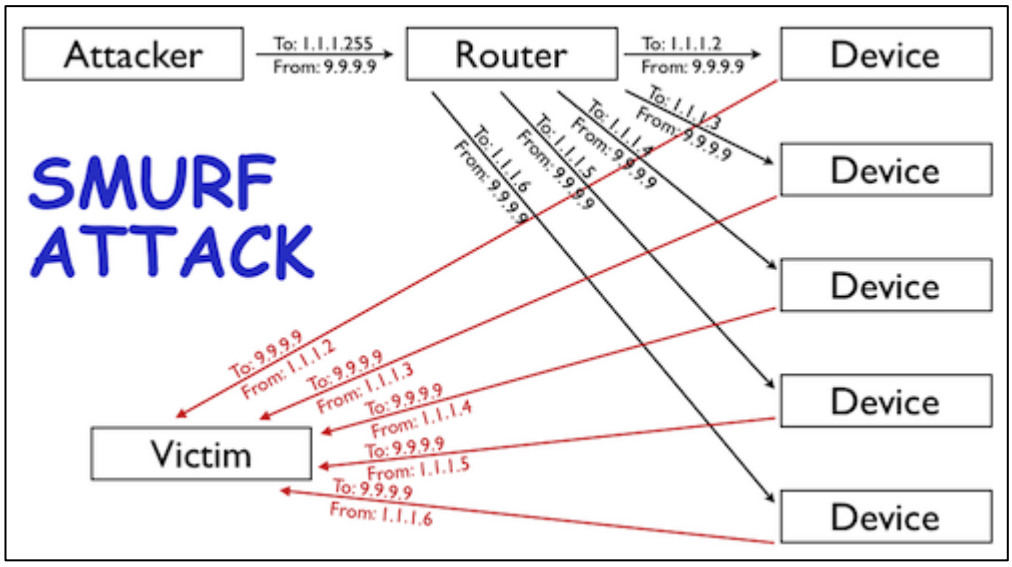

Figure 3.5. Smurf attack [46]

Fig. 2 explain how the IP address of the victim is spoofed by the attacker from ICMP request. because of the ICMP dose not use the handshaking protocol therefore the end node does not look at the source node if it is legitimate or not. when the router receive the request,the request will be forwarded immediatly to every device is connected in the network. After the victims receive these responses, the attackers guarantee their success. denial of access to services of the server will happend because of a big number of ICMP.

HTTP flood

This attack is a type of DDoS attack that occurs at the application layer. HTTP GET or HTTP POST request to attack a web server or an application. It is extremely difficult to detect and block an HTTP attack for the reason that launching an attack does not require reflection technology, malformed packets, or spoofing. Comparing with other

attacks, this attack to breakdown a targeted server need only a less bandwidth. It is very difficult to determine valid traffic because an HTTP flood attack uses the standard URL request, which makes it one of the most advanced unstable security challenges.

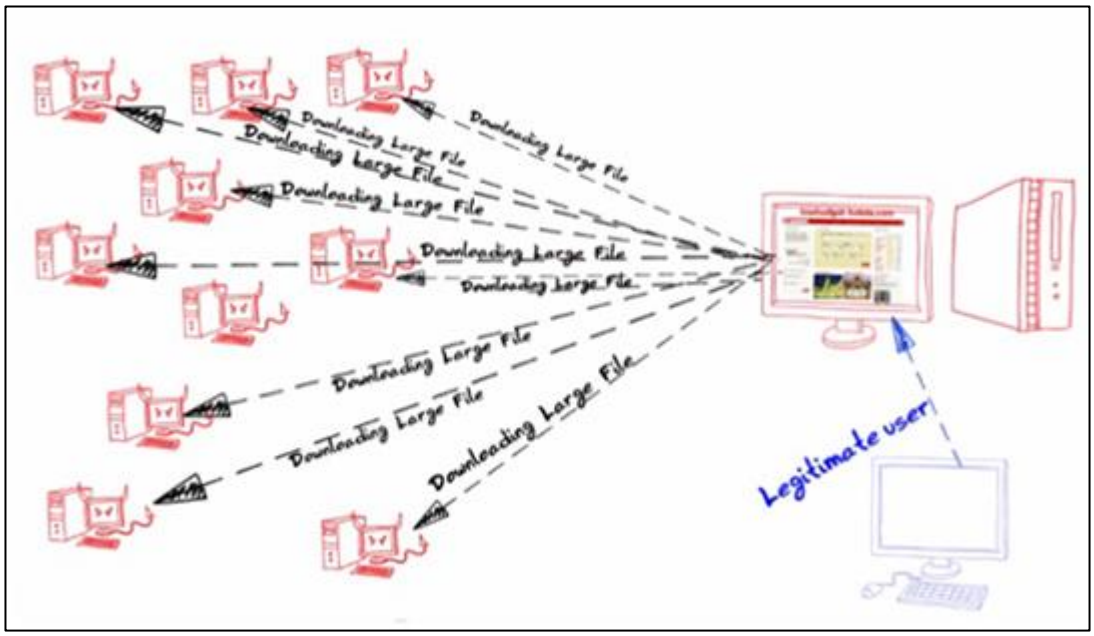

Figure 3.6. HTTP Get flood [47]

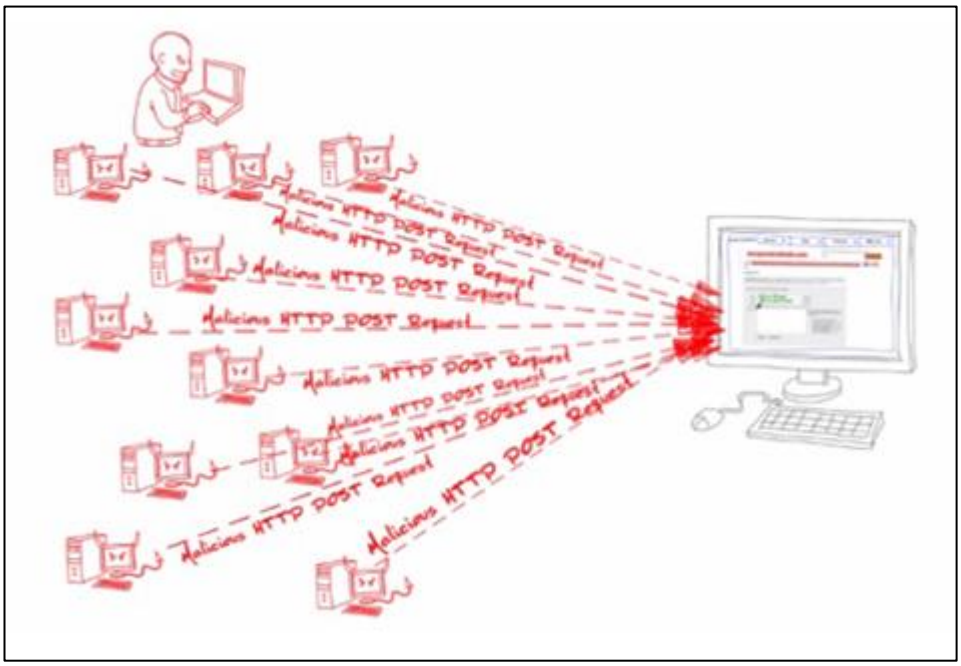

Figure 3.7. HTTP Post flood [47]

3.2. MACHINE LEARNING

In short, machine learning is just one of the several fields of artificial intelligence that personal computer users may take use of in order to allow their computers to improve by self-learning without being programmed or constantly amended. Machine learning's core and key notion is, to design a model that addresses a challenge, while developing a model with maintaining a sufficient measure of data, which leads to predictions that have significant effects. Therefore, machine learning systems have an unlimited number of models and algorithms that may employ a virtually limitless supply of approaches to learn, adapt, and refine their output from experience. Many other aspects of the business may benefit from machine learning, such as healthcare, educational, government, financial services, transportation and many more sectors. In order to tackle the problem of gene expression data, the binary classification task is utilized. A system which may produce a capability to reduce the variance between predicted values $\hat{y}i = (\hat{y}1, \hat{y}2, \hat{y}3, ..., \hat{y}n)$ and the required $yi= (y1, y2, y3, ..., yn)$ when the input is $xi= (x1, x2, x3, ..., xm)$, is a required system or model. It should be noted that the values m and n refer to total numbers of input instances and output instances, respectively, and that the variable i refers to the predicted numbers [48].

3.2.1. Supervised learning, unsupervised learning and semi-supervised

Supervised learning, unsupervised learning, and semi-supervised learning are the three primary forms of machine learning, which can be further split into subtypes [49].

3.2.1.1. Supervised learning

Supervised learning [50] is a technique used to train a machine using labelled datasets. As the name implies, it means that some of the labelled data is tagged with the correct answer. The way supervised learning works is by training the labeled data, one can predict unforeseen results. For instance, suppose that one wants to train a machine to predict how long it does take between from a location to another; specific data must be gathered and analyzed. Namely, such data may include weather conditions, holidays, time of the day, and route chosen. All these details are considered as inputs.

Naturally, if it is raining, one assumes that it will take a longer time to reach the desired location, but a machine needs statistics. Consequently, in order to create a dataset that can get trained, specific data such as the total time it takes from a start location and corresponding data that includes time, weather condition, route, and so one. Based on the given information, the machine will be able to see the relationship between different data and predict the time it takes to travel from a location to another.

3.2.1.2. Unsupervised Learning

Unsupervised learning [50] is a technique where a machine learning model does not need to be supervised. The model will instead try to discover the information by itself, which means that unsupervised learning deals with unlabeled datasets. In unsupervised learning, the machine can find all types of data patterns, and it also helps in identifying the features one needs to categorize the data.

Table 3.1. Supervised ML vs Unsupervised ML

parameters	**Supervised ML**	**Unsupervised ML**
Process	Input and output data are given.	Only input data is given.
Input data	The machine is trained using labeled data.	The machine is not given unlabeled data.
Algorithms used	SVM, NN, Random Forest, Linear and Logistics regression, Classification trees.	Different categorized: K-means, Cluster algorithms, Hierarchical clustering, and so on.
Computational complexity	simple	complex
Use of data	Uses training data and relate input and output results.	Does not use output data.
Accuracy of results	Accurate and trustworthy.	Less accurate and trustworthy.
Real time learning	Learning is offline.	Real-time.
Number of class	Known	unknown

3.2.1.3. Semi-supervised

There is another type of learning, which is called semi-supervised learning [50]. This technique makes use of both supervised learning and unsupervised learning. It combines some of the labeled data with a massive amount of unlabeled data during the training phase.

3.2.2. Classification

Classification is employed in order to forecast the numerical replies. The main goal of this type of supervised machine learning is to develop a model that is capable of precisely classifying a given input vector into available and accessible classes based on the labels and training dataset. Most of the time, each sample will contain only one input class. The input space is divided into decision regions, referred known as decision surfaces and decision boundaries [50].

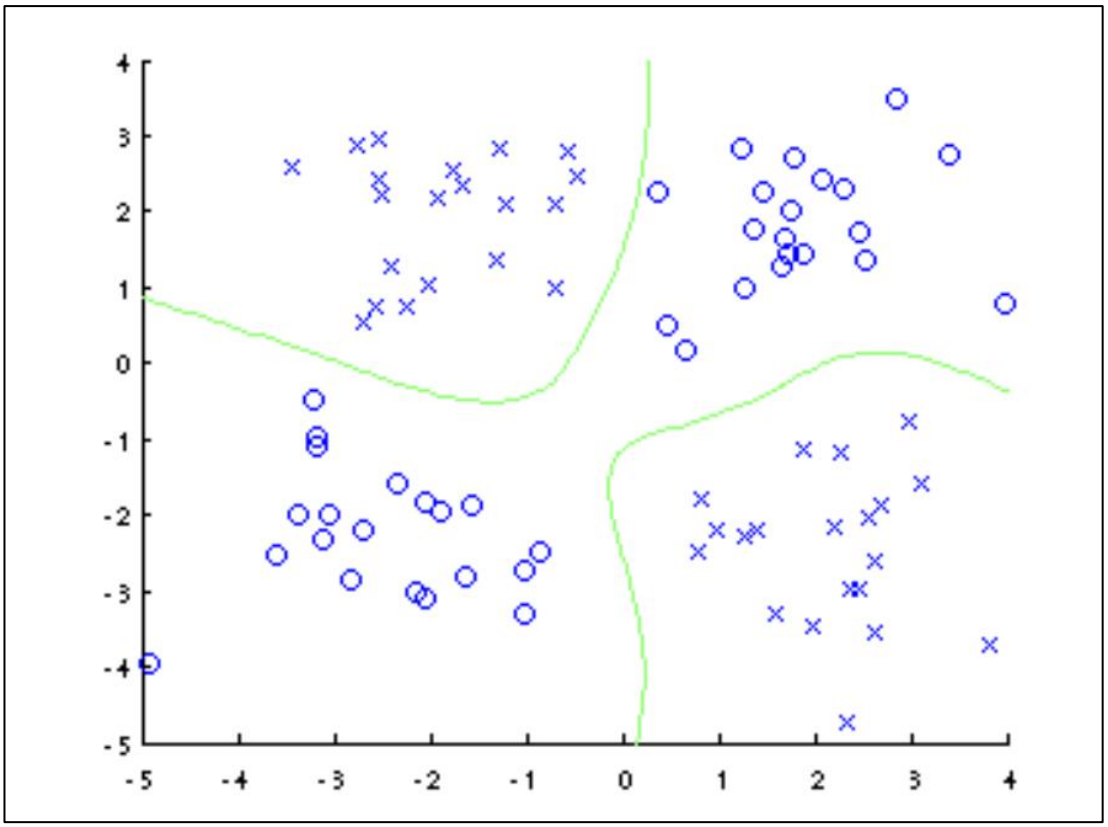

Figure 3.8. Decision Surface [48]

The decision surface in Figure 3.8. is demonstrated. This example has two separate points, O and X, that indicate the class labels. Also as a figure above illustrate, there are two lines that establish a boundary for display new data between the points. If it is not depicted in a highly charged location, the class to which it belongs will be obvious [48]. During the classification process, you have to go through the following steps in order: preprocessing, training, and classification.

Preprocessing: Data at this phase is prepared to be an input before handling. Although in the great majority of situations, the input data will be noisy, fragmentary, or inconsistent, the preprocessing is important because it is often necessary to clean up the raw data before meaningful processing can take place. While doing these duties, you could also work on various processes associated with the data, such as data cleansing, data transformation, feature extraction, and more [48].

Training: As the model go into this phase, data are trained to provide the highest level of accuracy feasible when making predictions.

Classification: this phase is beginning once the previous is complete, the given input is allocated to one of the learned models, which makes a choice based on previously-created decision rules.

The classification problem consists of three main parts. Firstly, the input data's class frequency and distribution of possible classes. Secondly, distinguishing features are defined by establishing the link between input and output. Thirdly, reducing the final cost of penalizing false predictions by determining the loss function. All of the categorization problems in it, based on the probability theory, may be encountered in everyday life. Probabilistic models are used to exhibit and depict uncertainty by creating and generating a vector representing the probability of each potential class [51].

The final learning model is applied for prediction, allowing the learnt model to provide insight into unknown data class (test data). Although this is true, all classification algorithm that are designed and developed want to store training data in their memory.

The classifier's performance may suffer when used on fresh data, as a result of the training process, but will remain accurate when used on training data. Overfitting is the term used to describe this condition. The capability to generalize the learning model's goal is required to construct an effective model for learning. For that reason, the learning model's goal is met when it can perform as well with testing data sets as it does with training data sets. Another possible cause for the emergence of overfitting is the model's level of difficulty to fit. The generalization of data with complex structures can be difficult and complicated for a learning model to accomplish. For the purpose of developing a model with perfect performance for the processing of fresh data, the number of training stages as well as the complexity of the model must be carefully considered [52].

3.2.3. Preprocessing

Data preparation is regarded to be one of the most critical phases in data mining techniques as well as machine learning. Preprocessing is useful in helping you deal with many sorts of data challenges and challenges when you have a big dataset and need to generate a finer result. Using the preprocessing approach ensures that raw data is preprocessed and upgraded to become appropriate, suitable, and pure, therefore improving accuracy [53].

3.3. DEEP LEARNING

Features plays a large role in machine learning systems that use traditional machine learning approaches. Domain knowhow and fine-tuning are required to pick such features, and they help to ensure the hit of traditional machine learning techniques. Representation learning is a group of techniques that are effective for classifying and predicting just about anything [54]. The learning methodology commonly referred to as deep learning takes on many different forms and is distinguished by its capacity to learn from many levels of representation via the composition of numerous non-linear transformations. The notion that the data is created by a number of underlying elements that are interrelated in a hierarchical structure is the basis for stacking many layers of transformations. The lowest layer is typically used when talking about the layers

nearest to the input. This is based on the concept that the layers closest to the input, or at the 'lowest' level, indicate low-level characteristics, such as gradients and edges, but the layers closest to the output, or at the high-level, include more advanced characteristics, such as faces and objects. Data size and methods of problem-solving are two of the most significant distinctions between machine learning and deep learning techniques [55]. Big data is frequently employed in deep learning, as opposed to machine learning. The second difference between machine learning and deep learning is that deep learning will solve a problem from start to finish, whereas machine learning has a split and handle methodology. In addition, deep learning lets us carry out parallel processing in sequential layers, whereas machine learning lets us carry out single layer processing. Deep Neural Networks (DNN) are one of the most widely used deep learning methods [56].

PART 4

METHODOLOGY

The methodology flowchart is illustrated in figure 4.1. In the first step, the utilized dataset will be presented. Then, the description of preprocessing techniques and which one was used. After that, it is the explanation of the deep learning methods that were applied to this dataset. In the end, it is the performance evaluation metrics that matter.

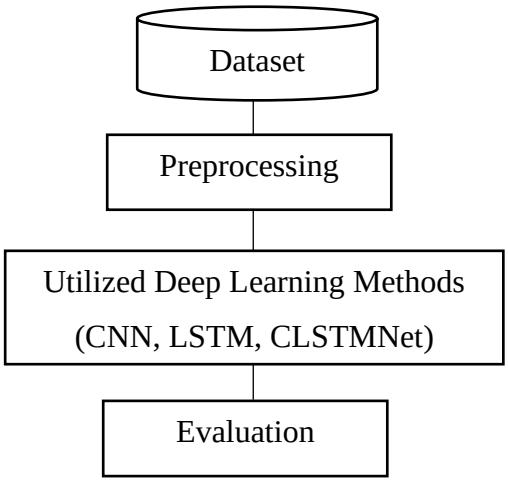

Figure 4.1. methodology model

4.1. DATASET

To examine the performance of our suggested model, the NSL-KDD dataset was utilized. The NSL-KDD dataset has been widely used in the area of evaluating IDS systems as well as detecting DDoS attacks. Many problems that were faced with the KDD '99 dataset were solved when it was upgraded to the NSL-KDD dataset. The main problems are the following: The dataset is imbalanced; some kinds of assaults are recognized so quickly and have many redundant records. Then, the redundant rows

were deleted and all kinds of assaults were redistributed to make it more suitable for evaluating algorithms. This dataset is comprised of 148514 records with forty-one columns (features) [57] where the names of the features are tabulated in table 5.1. In the end, the dataset was divided to 80% for training and 20% for testing.

Table 4. 1. NSL-KDD dataset attributes

Attribute No.	**Attribute Name**	**Attribute No.**	**Attribute Name**
1	Duration	22	Is_guest_login
2	Protocol_type	23	Count
3	Service	24	Srv_count
4	Flag	25	Serror_rate
5	Src_byte	26	Srv_serror_rate
6	Dst_byte	27	Rerror_rate
7	Land	28	Srv_rerror_rate
8	Wrong_fragment	29	Same_srv_rate
9	Urgent	30	Diff_srv_rate
10	Hot	31	Srv_diff_host_rate
11	Num_failed_logins	32	Dst_host_count
12	Logged_in	33	Dst_host-srv-count
13	Num_compromised	34	Dst_host_same_srv_rate
14	Root_shell	35	Dst_host_diff_srv_rate
15	Su_attempted	36	Dst_host_same+src_port_rate
16	Num_root	37	Dst_host_srv_diff_host_rate
17	Num_file_creations	38	Dst_host_serror_rate
18	Num_shells	39	Dst_host_srv_serror_rate
19	Num_access_files	40	Dst_host_rerro_rate
20	Num_outbound_cmds	41	Dst_host_srv_rerror_rate
21	Is_hot_login		

4.2. PREPROCESSING

Data preprocessing is often considered to be a key and essential component of data mining and ML. In the case of a huge dataset, preprocessing can be utilized to deal with multiple issues at once. Therefore, the outcome will be improved. At the same time, another advantage of applying data preprocessing is to guarantee that almost all requested data fits and is widespread appropriately [58]. There are many preprocessing techniques. One of them is *StandardScaler*. The *StandardScaler* eliminates averages and scales them to a unit variance; therefore, the features will be standardized. The below equation explains the StandardScaler technique, which calculates the standard score of a sample x.:

$$z = \frac{x-\mu}{\delta} \tag{4.1}$$

The mean of the training samples is represented by μ. Also, the standard deviation of the training samples is represented by δ. If with_mean is equal to false, the μ will be zero, and if with_std is equal to false, the δ will be one. It handles each feature independently depending on the training set instances. In the testing set, the mean and standard deviation that are stored through employing a transform will be utilized.

4.3. DEEP LEARNING METHODS

this section will explain the utilized deep learning methods in this thesis that applied on the NSL-KDD dataset. Moreover, all function and parameters will be mentioned and described. The subsections were sorted from the easiest one to more complicated. The utilized methods were CNN, LSTM, and CLSTMNet. However, before explain these methods, it is important to explain artificial neural network (ANN). Because of in all methods we have a fully connected layer is same ANN. In the last subsection we will mention the parameters and function employed for learning.

4.3.1. ARTIFICIAL NEURAL NETWORK

Using many layers of artificial neurons, the neural network may be trained to learn about the data. This is known as deep learning. The connections of a real neural network are used to derive this hypothesis. A real neural network can be represented in Figure 4.1. Dendrites receive and transmit input data or signals, whereas axons transfer data between cells via synapses.

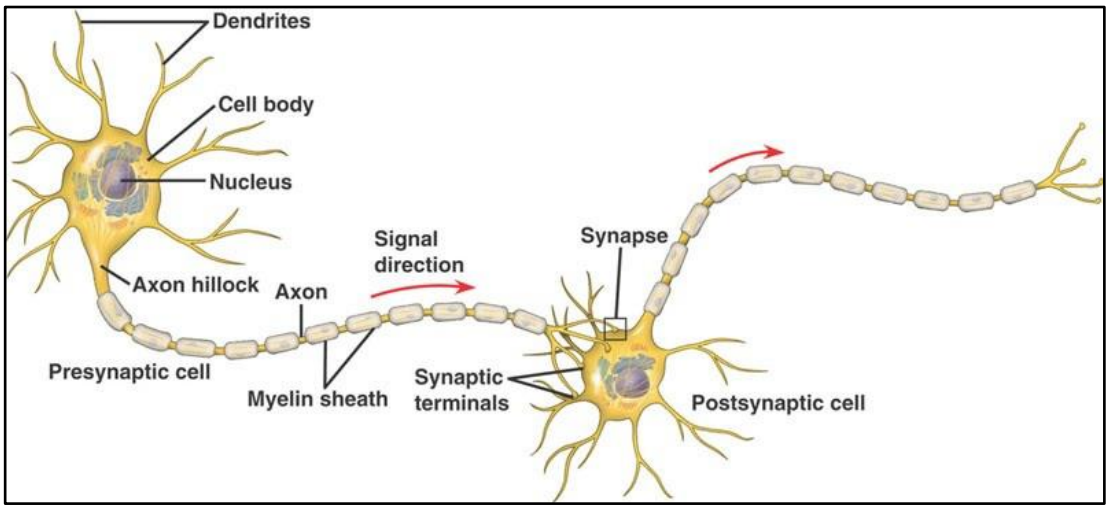

Figure 4.2. Real Neural Network [48]

The learning process of the artificial network is achieved by modifying the synaptic weights of the network during the adaptation phase. Due to the parallel distributed design of the network architecture, which is capable of learning, it is feasible to perform the complicated classification job in a fair amount of time [48].

4.3.1.1. Perceptron

A perceptron is a critical component of neural networks. At the same time, may be classified as a deep network and a feed-forward network that constructs the limits of a linear decision A perceptron in an Artificial Neural Network (ANN) that is shown in Figure 4.2, is a function which accepts vectors x ∈ R^n as input that are parameterized by a weight vector w ∈ R^n and a bias b and generates a scalar as output. On the other hand, The term "activation function" refers to $f: \mathbb{R} \longrightarrow \mathbb{R}$ [59]. There are many different activation functions, which will be explained later in the chapter.

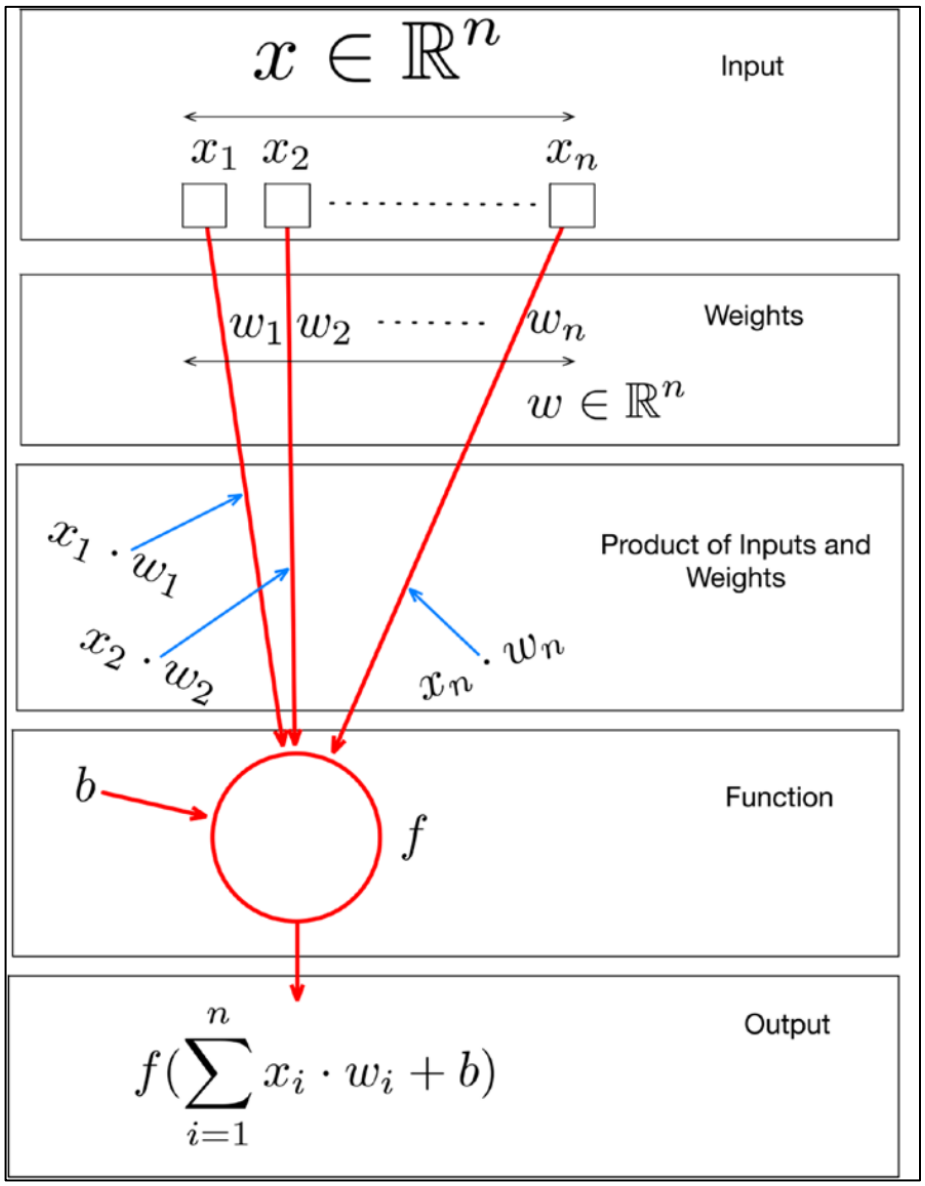

Figure 4.3. A Perceptron in Neural Network [59]

The linear function is the simplest activation function we start with. The rise and decrease in the value of f(x) is parallel to the rise and decrease in the value of x. The linear function overcomes the problem of a binary step function that only returns values of 0 and 1. The linear function range is (-infinity to infinity). However, this function is not designed to handle complicated situations. To resolve this issue, we implement non-linear functions therefore the model may learn iteratively.

$$f(x) = x \tag{4.2}$$

For these non-linear functions, the non-linear activation functions are necessary. A non-linear activation function with a finite number of possible values was published in the literature in the past. Activation functions such the Rectified Linear Unit ReLU function and Softmax function are often employed, especially because they are the most prevalent. The Softmax layer is often employed as an output layer in combination with the Cross Entropy loss function for multi-classification activities. The Softmax layer standardizes outputs of the preceding layer in order to be one. The preceding layer model's units represent the un-normalized score that the input belongs to a specific class. This layer has normalized by the Softmax, therefore the output value indicates the likelihood of each class [59]. The ReLU function will return 0 as an output if the input is less than0, while it will return the same input number if the input is higher than 0.

$$softmax(x) = \frac{e^{x1}}{\sum_{c=1}^{n} e^{xc}} \quad (4.3)\ [59]$$

$$ReLU(x) = \max(x, 0) \quad (4.4)\ [59]$$

ReLU functions are mathematically a lot simpler because both forward and backward passes through a ReLU are simple statements. There is an enormous benefit in situations when a network has a large number of neurons, because the training and assessment duration may be considerably reduced [59].

The perception can use a variety of techniques to anticipate the correct class label, including the choosing of the optimal weight vector from the training data set. The most commonly acknowledged method is the perception of the rule of learning. In this scenario, the perception-learning rule always reaches the optimal weight of infinite time when the classification matter is linearly separable. Weights can be generated randomly between (-1, +1) with this technique, for example. It should be noted that the perceptron is also applied to all training instances following that. Each epoch, weights are continually re-adjusted till the output is appropriately categorized [59].

4.3.1.2. Feedforward Network

Since a single perceptron is only capable of linear mapping, it is unable of dealing with complicated problems. Sophisticated problems can be solved using a more global technique, which is capable of inconsistent mappings. Perceptron is capable of constructing workable structures or even more complex ones to any depth we choose. We can also name this type of network a feedforward network (FFN). because of the direct connection between whole layers, FFN's fundamental structure may also be described as a multilayer perceptron (MLP). MLP has generally three primary layers: the input, hidden, and the output. Furthermore, unlike the input and output layers, the hidden layer may have multiple layers. Because of its hidden from outputs and inputs, this layer is known as the "hidden" layer. Figure 4.3 illustrate MLP structure. The work of learning the complicated model is handled by the hidden layer, which extracts features from input data.

A perceptron can be designed as a collection of perceptron placed on top of each other, in layers. A layer that comes after the last hidden layer or at end is known as the output layer. The term "hidden layer" refers to any layer preceding the output layer. The number of perceptron in a layer determines the width of the layer's width. The layer's number in the network represents the depth of the network. The idea of deep (as in deep learning) is derived from the relationship between width and depth. With the exception of the first layer, the output of each layer will be the input to the next layer. While the last layer of the network outputs likewise serves as the overall output. A prediction class is generated using input data. As we previous explained, a neural network may be described as a function $f_\theta: x \longrightarrow y$, with f θ being a neural network, and where f θ accepts the input $x \in \mathbb{R}^n$ and generates the output $y \in \mathbb{R}^m$ where θ is a neural network parameter $\theta \in \mathbb{R}^p$. θ is now a discrete quantity which has a single value of all the network weights for all the network perceptions. Selecting a design structure for a neural network is a difficult task because it is based on the number of layers, the number of perceptron for each layer and other things [59].

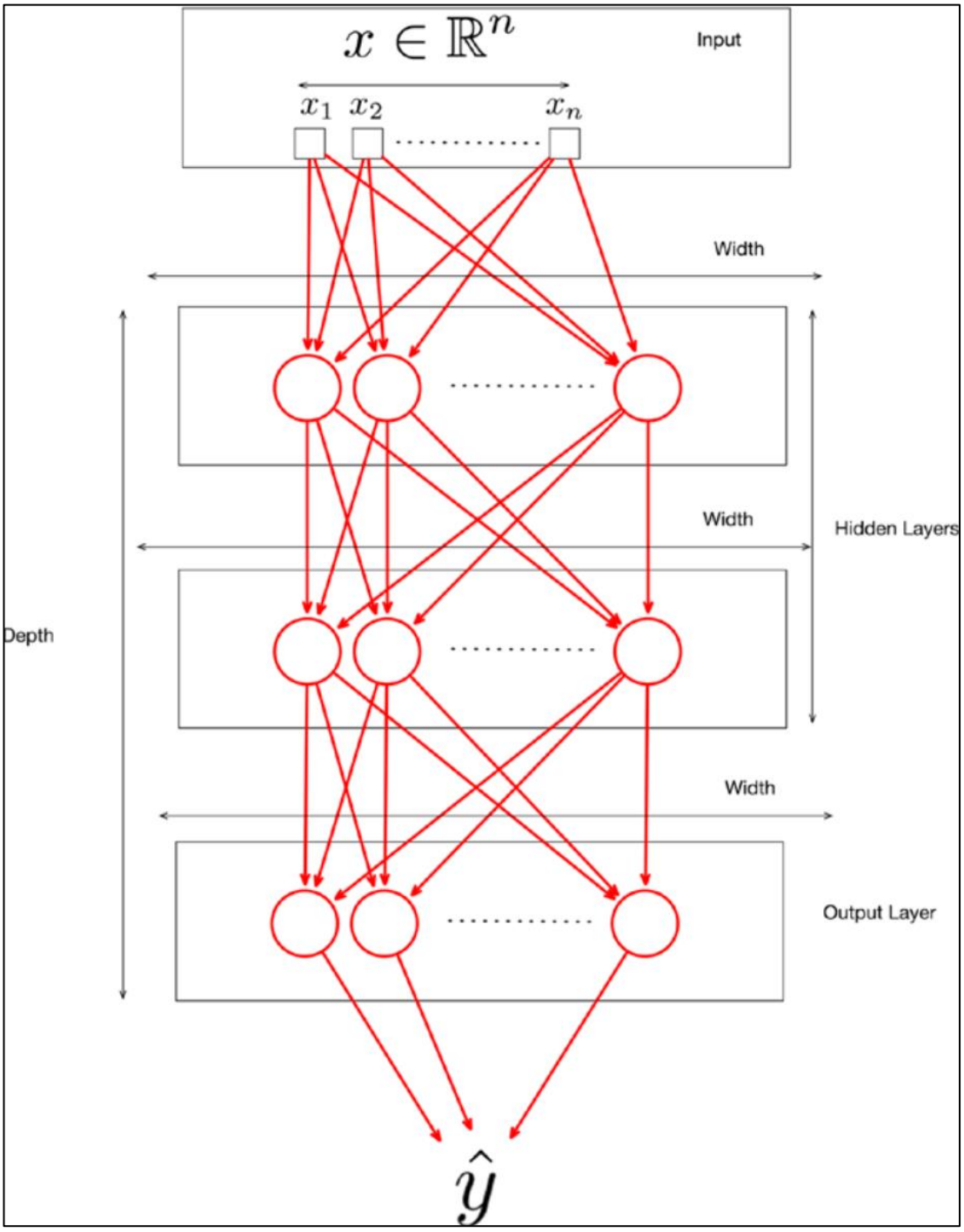

Figure 4.4. MLP Structure [59]

4.3.1.3. Backpropagation

During the training of artificial neural networks, backward propagation (BP) is a highly frequent and widely utilized optimization approach. Deep neural networks may learn by reducing the loss function and upgrading the parameters ($\theta = \{w^{(*)}, b^{(*)}\}$) of

the model during the training phase. Stochastic gradient descent (SGD) is one of the most frequent techniques used for upgrading and learning the parameters. When calculating the gradients of a loss function, the BP algorithm is employed, and the results are input into the SGD technique with the goal of upgrading the weights and biases. Back propagation can be divided into two phases, which are propagation and weights update [59].

Propagation: In this phase of BP, an input is provided, which then propagates throughout the entire network, reaching the output layer where the output is generated. loss function is used to calculate the difference between predicted results and actual results. In this research Sparse Categorical Cross Entropy (SCCE) loss function (4.5) is used, The intended output is y, while the prediction $f(x_i, \theta)$:

$$SCCE = -\sum_{i=1}^{n} y_i \log f(x_i, \theta) + (1 - y_i) \log\big(1 - f(x_i, \theta)\big) \quad (4.5)\ [59]$$

The error will move backward across the network while the weights wait for themselves to become current. All intermediate nodes between layers are therefore linked, and they will all contribute their error values to forward propagation as it passes through them. The propagation mechanisms, both forward and backward, wrapped the entirety of the network [59].

Weights updating: Backpropagation follows as a result of error estimation with consideration to the weights of the network, which determines the partial derivative of the loss function. The loss function can be reduced by employing some techniques based on optimization, as well as the gradient that is acquired and improved dramatically when the weights are updated. An interesting and instructive method is how the overall network cost changes, as well as how the weight update affects the performance and behavior of the entire network. A given method that is commonly used to loss function for executing optimization can be identified as an SGD algorithm. When using this technique, the loss function gradient weights can be updated in the reverse way. The SGD method significantly aids the discovery and enhancement of local optima by often and strongly updating the value and reducing overshoot by converging on the global minimum [48].

An algorithm known as Adaptive Moment Estimation (ADAM) [60] is employed in this thesis as an optimizer for the procedure of learning a neural network. The equation uses exponential moving average and adds bias correction when estimating the gradient mean (m_t) (4.6) and element-wise squared gradient (v_t) (4.7) [60]. The first and second order moment biases (β_1 , β_2) are computed and implemented at time step t. According to the ADAM algorithm, the value of parameters w_{t+1} at time $t+1$ are changed using (4.8) equation.

$$\hat{m_t} = \frac{m_t}{1-\beta_{1,t}} \quad (4.6)\ [60]$$

$$\hat{v_t} = \frac{v_t}{1-\beta_{2,t}} \quad (4.7)\ [60]$$

$$w_{t+1} = w_t + \frac{\eta}{\sqrt{\hat{v_t}+\epsilon}}\hat{m_t} \quad (4.8)\ [60]$$

To prevent the denominator from becoming zero, a smoothing factor ϵ is used. The learning rate, on the other hand, is represented by η.

4.3.2. CONVOLUTIONAL NEURAL NETWORK

The CNN works exceptionally well at identifying basic patterns in data, which enables it to be used to generate more complicated patterns in higher layers. CNNs are a unique kind of multilayer neural networks. It utilizes the backpropagation method, like most other neural networks. CNN's architecture distinguishes it from the competition. the architecture of convolutional NN is typically made up of an input layer, many hidden layers, and an output layer [61]. The CNN's hidden layers are composed of convolutional, pooling, and fully connected layers [58] as seen in Figure 4.4.

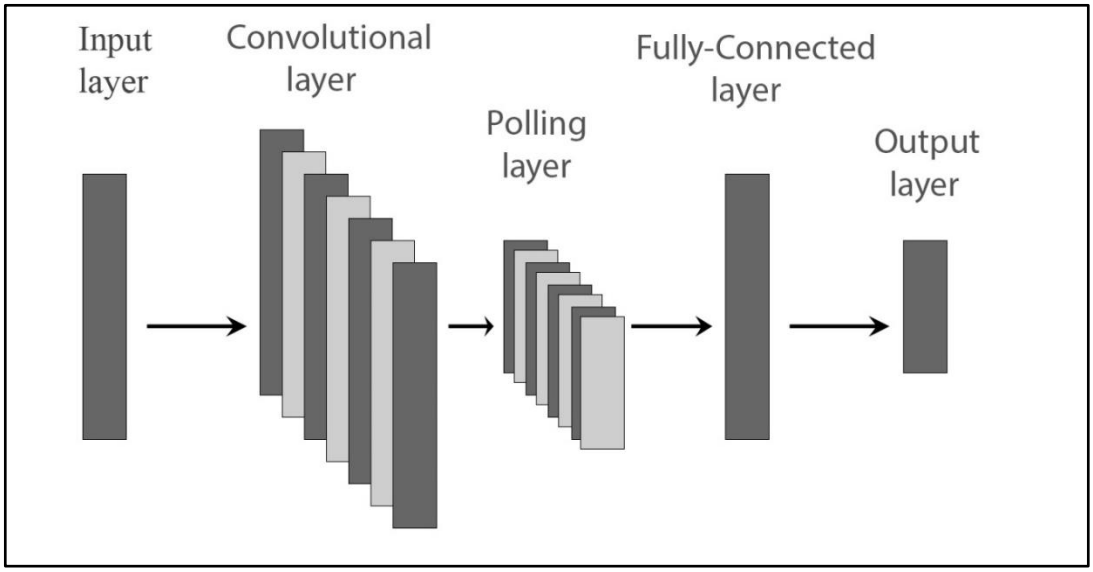

Figure 4.5. The General Structure of CNN [59]

4.3.2.1. Convolution Layer

The convolution layer takes the incoming data and applies a filtration to all of it, essentially multiplying it with the kernel with the purpose of extracting and creating the better output data [58]. The convolution process is conceptualized as a one-dimensional process with a specified input $I(t)$ and a kernel $K(a)$ The process to calculate the convolution may be summarized as follows:

$$s(t) = \sum_a I(t+a) \cdot k(a) \quad (4.9)\ [59]$$

The core of the process is that the kernel is a considerably smaller collection of multiple points of data than the data input, but when the input is equal to the kernel, the convolution process output is greater. A random input and a random kernel are fed into the convolution algorithm, which subsequently yields the greatest possible result when the kernel is equal to a specific part of the data input. This operation can be explained as shown in the Figure 4.5.

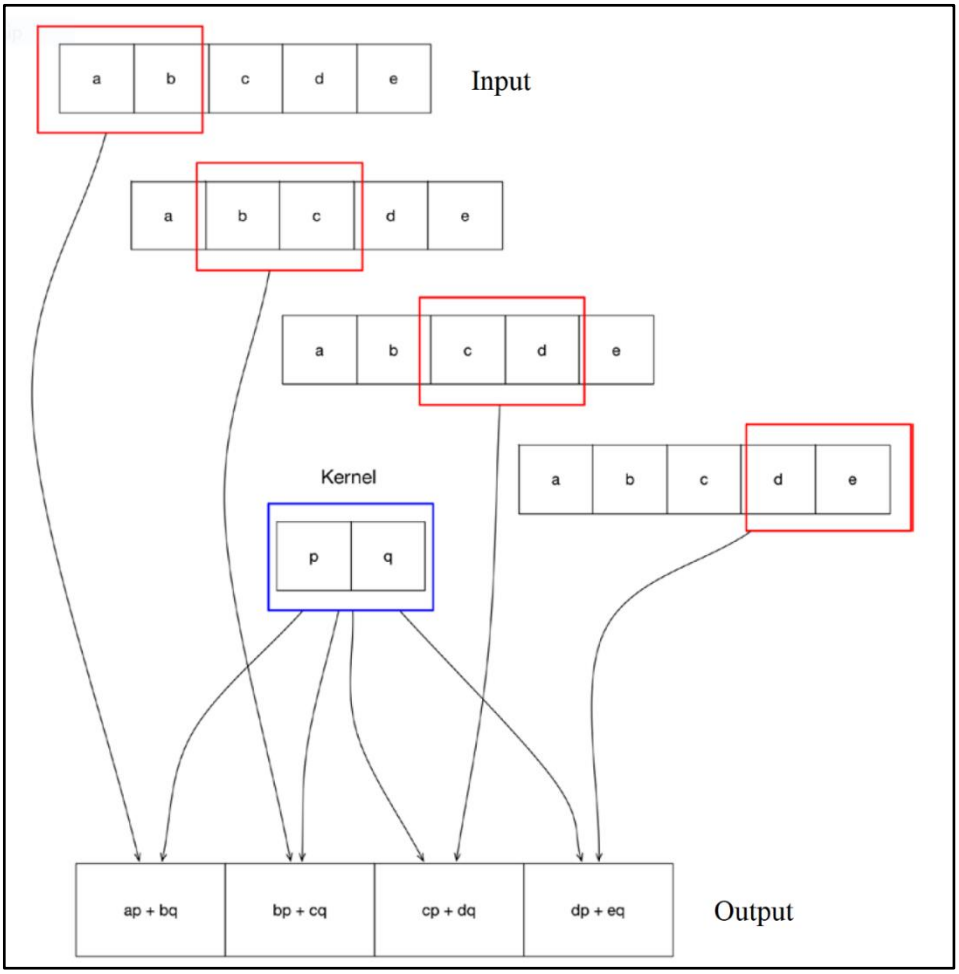

Figure 4.6. Convolution operation [59]

4.3.2.2. Pooling Layer

The subsampling layer is also known as the pooling layer. Reducing the dimensionality is the goal. Typically, the pooling layers come after the convolutional layers, when the spatial scale of the output is reduced and down-sampled. The pooling operation may be done to both efficiently compute the complexity of the network and reduce the parameters number. Max-pooling seems to be the most widely used pooling method, and hence, it is mostly found in this layer. A technique of selecting the biggest element inside small region in the certain pooling region is known as "max-pooling". when the stride is set to two, the max-pooling layer output will be halved [58].

4.3.2.3. Fully Connected Layer

Multilayer Perceptron (MLP) is a last type of neural network that was described in the previous section of this thesis, Feedforward Network.

4.3.2.4. The Utilized CNN Architecture

The suggested CNN consists of five layers, which are applied to the NSL-KDD dataset. The structure begins with the input layer, which is the data-preprocessing output, and follows the convolutional layer to max-pooling layer to flatten layer. This makes it flatten after max-pooling it by 2. In the end, the fully connected layer represents the output layer. The CNN employed architecture, with its functions and parameters, was demonstrated in Figure 4.7.

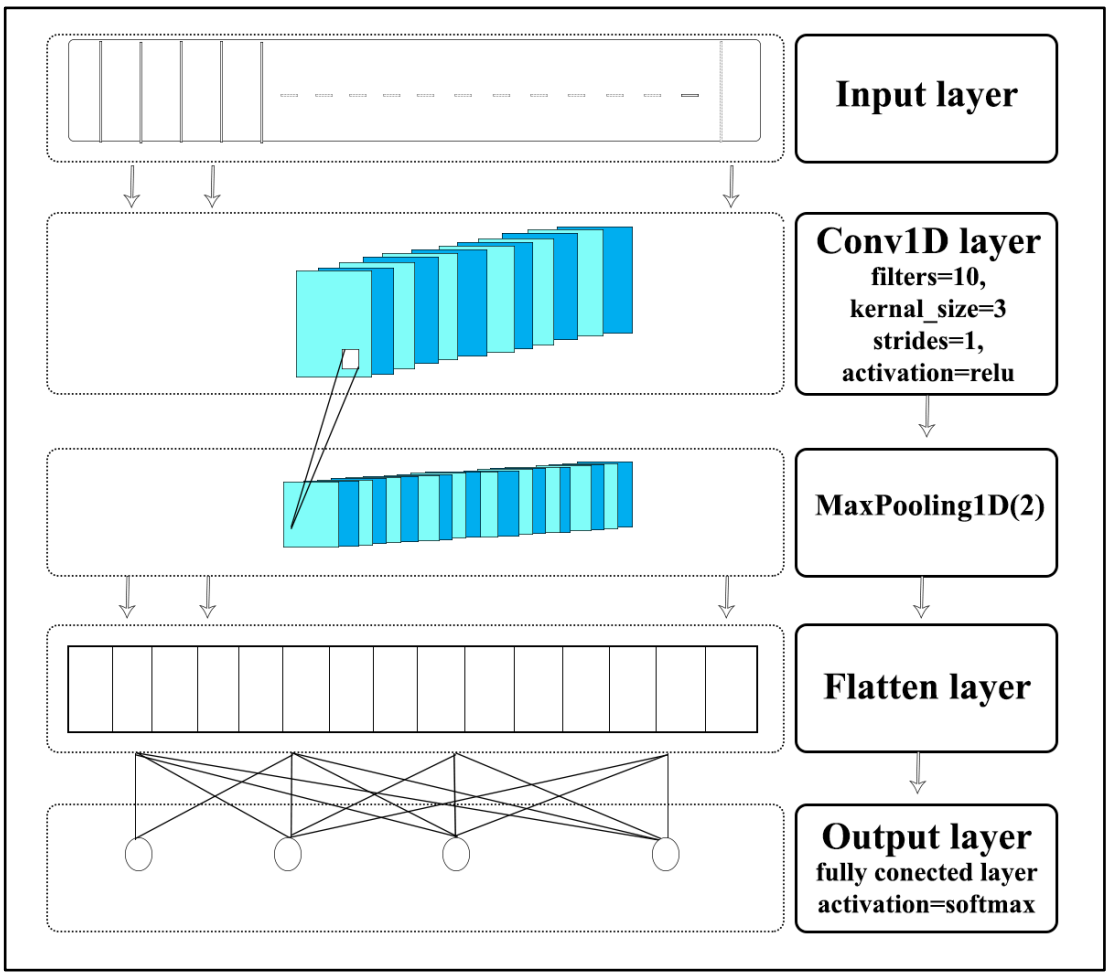

Figure 4.7. The utilized CNN architecture

4.3.3. LONG SHORT-TERM MEMORY

Before describing what an LSTM is, Recurrent Neural Networks (RNN) have to described as the LSTM is a variant of the RNN. The main idea behind RNNs is to benefit from the sequential information [62]. In regular Neural Networks, the assumption is that inputs and outputs are independent of each other, and in many case scenarios, this is a bad idea. The reason is that if one wants to predict a particular value, information about the previous one is essential to have. The RNN is called recurrent due to its ability in performing computations based on the given information in the "memory", including information about what has been considered in the calculation so far.

LSTM [63] is an RNN variant and can learn from long term dependencies. As the name implies (Long Short-Term Memory), this algorithm is capable of remembering given information for long periods. It operates by performing three main step processes called gates; Forget gate, Input gate and Output gate. A complete overview of LSTM is shown in Figure 4.6.

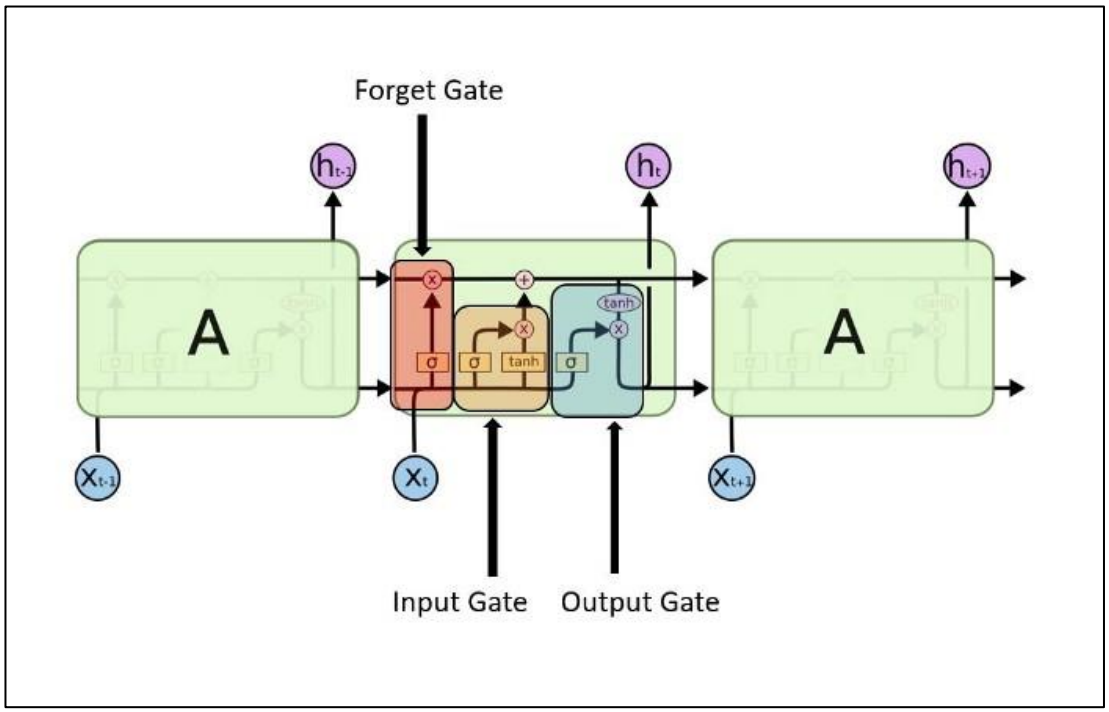

Figure 4.8. LSTM with its Gates [62]

4.3.3.1. Forget gate

It is what makes a decision about how much of the past to remember. It determines what information to remove from the cell in a particular timestamp which decided by the *sigmoid* function (or a squashing function which limits the output to a range between 0 and 1 in order to predict the probability). As shown in Figure 4.7, it checks the previous state h_{t-1} and the given input x_t, then it decides whether to delete or keep the information by outputting a number between 0 (delete this) and 1 (keep this) for each number in the cell state C_{t-1}. The forget gate equation is represented in 4.10.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4.10}$$

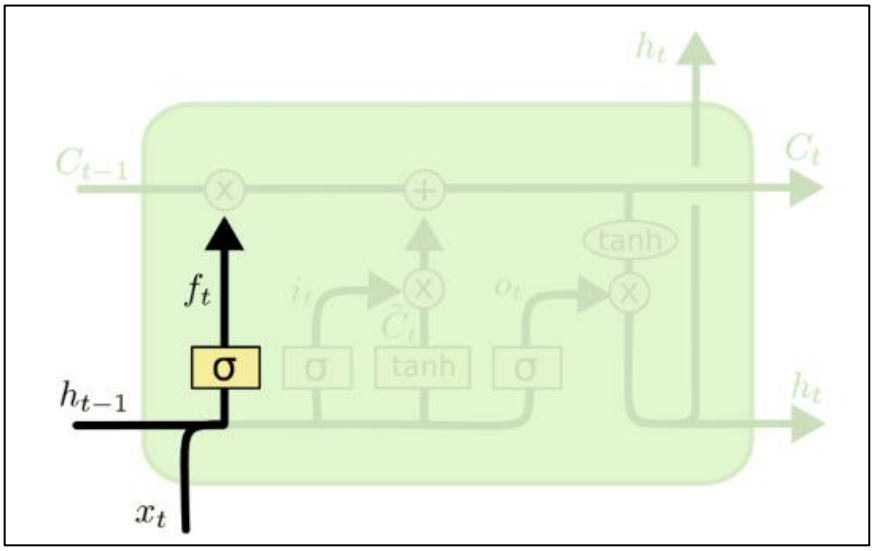

Figure 4.9. Forget gate [62]

4.3.3.2. Update gate/input gate

It is what decides how much of a particular piece of information to add to the current state. In this gate, just as in the forget gate shown in equation 4.11, the *sigmoid* function decides which value will go through and which will not. In addition, a *tanh* function, as shown in equation 4.12, is used to weight the values passed to determine their importance, represented by a specific value from -1 and 1. LSTM update gate/input gate is shown in Figure 4.8.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (4.11)$$

$$\hat{C}_t = tanh(W_X \cdot [h_{t-1}, x_t] + b_C) \quad (4.12)$$

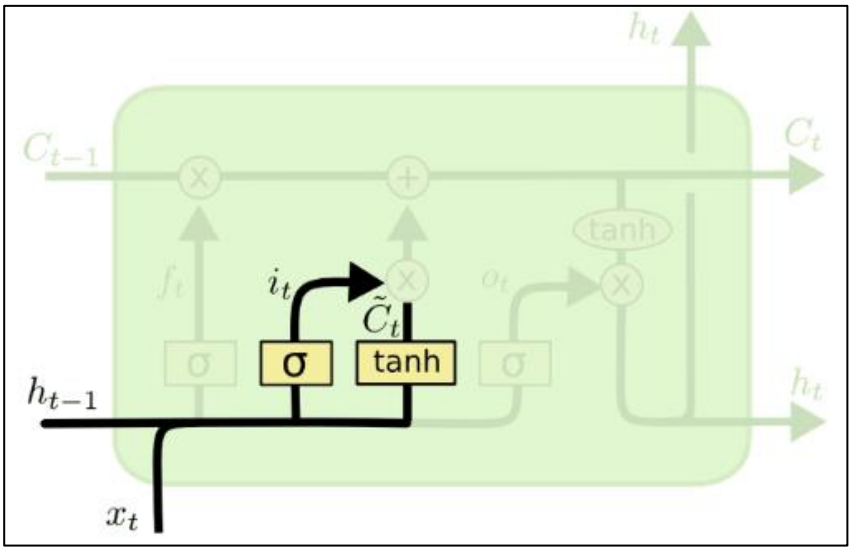

Figure 4.10. Update/input gate [62]

4.3.3.3. output gate

Finally, the output gate shown in 4.9, is the gate used to decide which piece of information will make it to the output. The sigmoid and the *tanh* functions are used for the same purpose as in both forget and input gates. Both of the equations are represented in 4.13 and 4.14, respectively.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (4.13)$$

$$h_t = o_t * tanh(C_t) \quad (4.14)$$

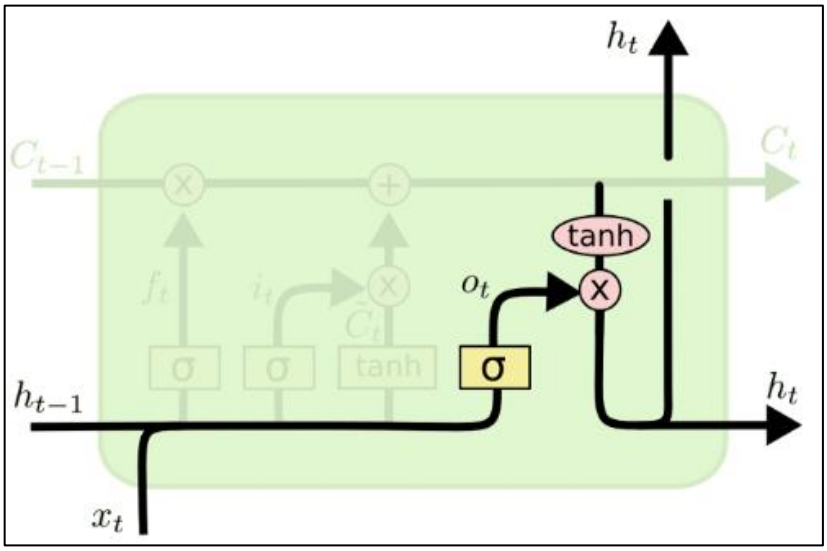

Figure 4.11. Output gate [62]

4.3.2.4. The Utilized LSTM Architecture

The proposed LSTM was comprised of three layers, which were applied to the same dataset, the NSL-KDD dataset. The employed algorithm was so simple that it begins with the input layer, which is the data-preprocessing output, and follows the LSTM layer to the output layer, which is the fully connected layer. In Figure 4.12, the LSTM-utilized architecture, with its functions and parameters, was illustrated.

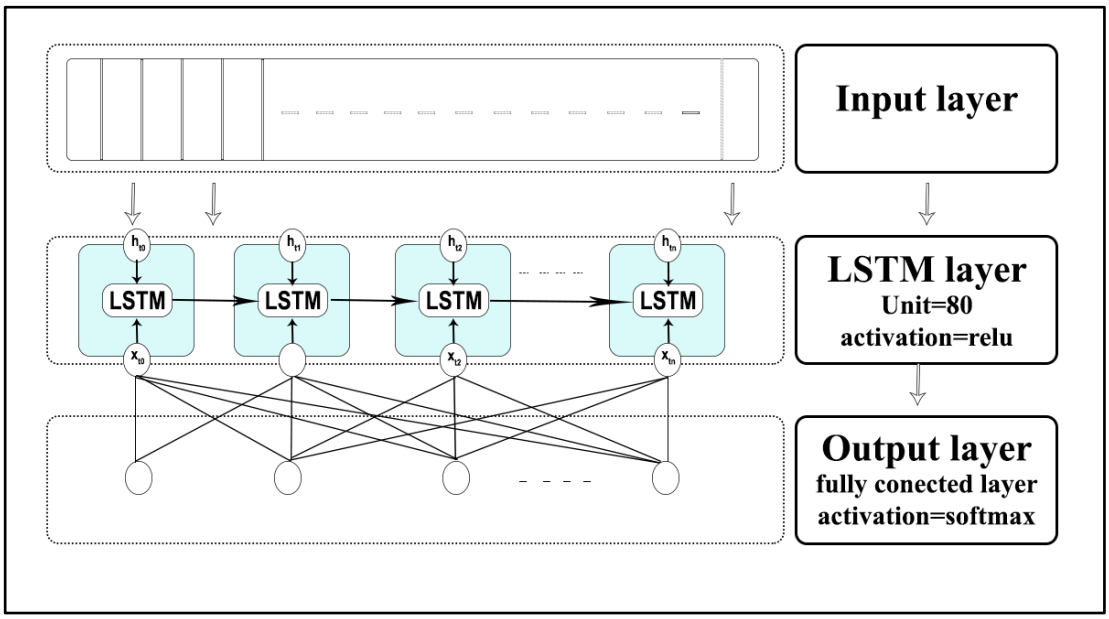

Figure 4.12 The utilized LSTM architecture

4.3.4. CLSTMNet

The suggested method is a combination of LSTM and CNN, and it was comprised of seven layers, which were applied to the mentioned dataset, the NSL-KDD dataset. This method was built to detect DDoS attacks with the best possible accuracy. Taking the advantages of two of the best deep learning algorithms, the CNN is the best feature selection because of its architecture, and the LSTM for predicting based on built-in memory blocks [9]. Also, it takes advantage of his own architecture, which is composed of the input layer, two convolution layers, two max-pooling layers, an LSTM layer, and an output layer that is a fully connected layer [64].

The CLSTMNet architecture diagram, with its functions and parameters for each layer, is mentioned in figure 4.13.

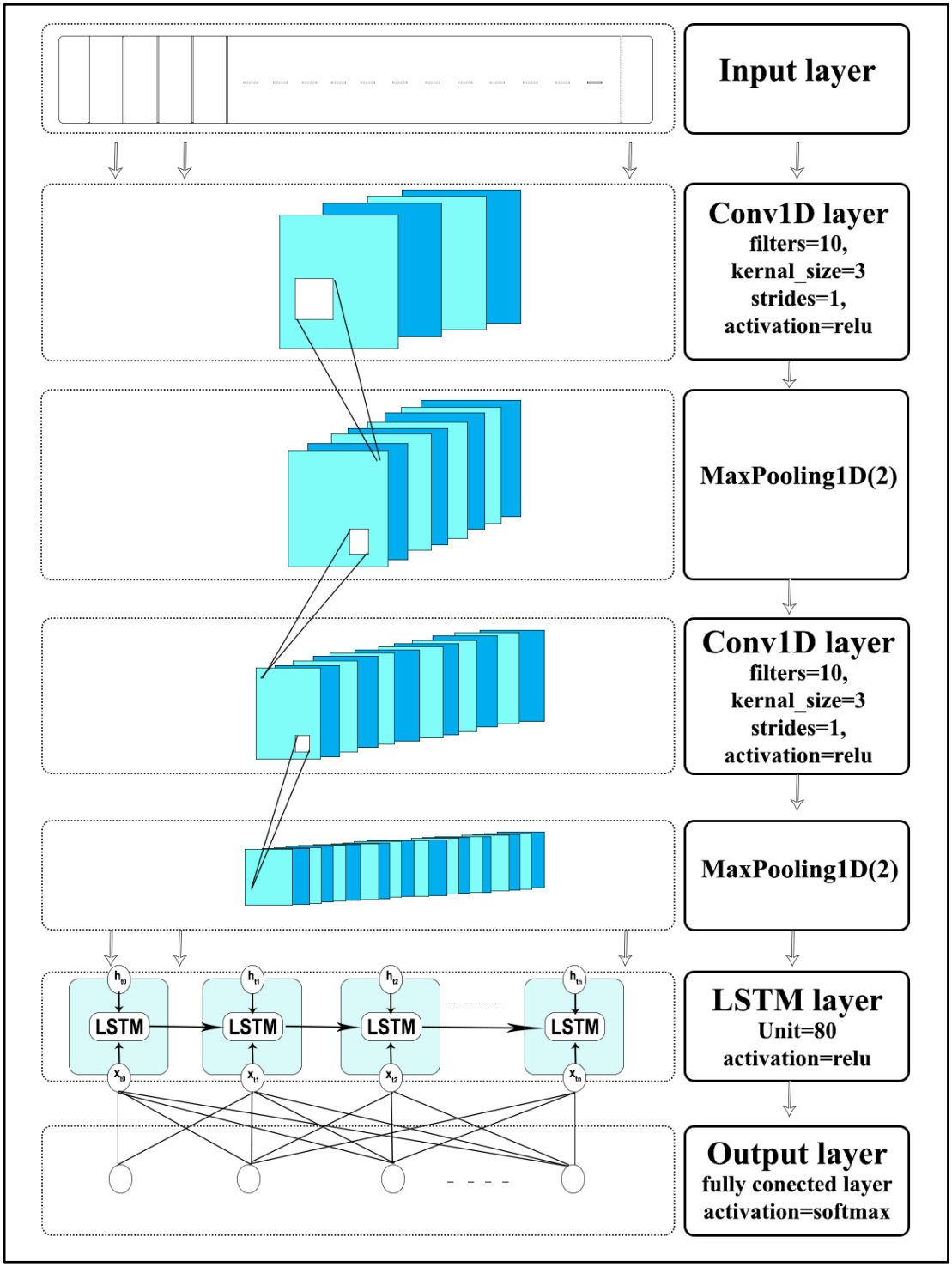

Figure 4.13. CLSTMNet structure

4.3.5. LEARNING

This subsection explains the functions and parameters that were used by CNN, LSTM, and CLSTMNet for learning. the weight initializer that were employed for all methods

is the glorot_uniform function [65]. Moreover, this initializer function is useful for obtaining samples from a distribution of uniform within the bounds (*-b,b*). Where *b* is calculated as:

$$b = \sqrt{\frac{6}{(\text{fan\_in} + \text{fan\_out})}} \tag{4.15}$$

The fan_in and fan_out represent the number of input and output units of the weight tensor, respectively. Therefore, for updating weight the Adaptive Moment Estimation ADAM is employed which is an optimization algorithm for stochastic gradient descent with a learning rate of 0.0001. Furthermore, because the learning rate manages the updating weight by reducing loss, it is its most effective hyper parameter. In the same time, the error is calculated using the Sparse Categorical Cross-entropy loss function. In order to train the networks, 500 epochs have been used. The term "epoch" refers to one period of time in which all of the samples have been trained. In addition, the batch size utilized is 32.

4.4. PERFORMANCE MEASUREMENT

The performance evaluation metrics are important for evaluating the deep learning methods in the last phase of the thesis methodology. The utilized performance evaluation metrics are: accuracy, precision, recall, and F1 score. These performance evaluation metrics were utilized in the testing phase when applying them to the NSL-KDD dataset. The performance evaluation metrics depend on the four expected results. The expected results are: true positive (TP) is the accurate detection of DDoS attacks; true negative (TN) is the proper recognition of normal records or other types of attacks; false positives (FP) is the incorrect detection of DDoS assaults; and false negatives (FN) is the incorrect recognition of normal records or other types of attacks.

Accuracy: a measure of how well a prediction matches the actual outcome [66].

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{4.16}$$

Precision: Precision is a result of dividing true recognition of DDoS attacks into all true detections, therefore knowing the performance of the system to identify DDoS attacks from other attacks or normal streams [67].

$$Precision = \frac{TP}{TP+FP} \tag{4.17}$$

Recall: recall is equal to dividing the number of predicted attacks by the actual attacks [68]. Therefore, it indicates how many DDoS attacks are truly predicted. It is also called the "true positive rate."

$$Recall = \frac{TP}{TP+FN} \tag{4.18}$$

F1 score: it is a balance between precision and recall, and it is between 0 and 1 [69].

$$F1\ score = \frac{2TP}{2TP+FP+FN} \tag{4.19}$$

PART 5

RESULTS AND DISCUSSION

It is in this chapter that you will find a description of the programming language used with the platform and libraries that are utilized. Moreover, you will find a discussion of the results acquired from our model, which was utilized in this thesis. In addition, it also includes a comparison between CNN, LSTM, and CLSTMNet in terms of accuracy, recall, precision, and F1 score; and a comparison of our model's results to previously published studies in terms of accuracy. All of the trials in this study were carried out on Intel (R) Core (TM) i5-4200U CPU @ 1.60GHz (4 CPUs) and 8192 MB RAM.

5.1. PYTHON

Python is efficient high-level and object-oriented programming language. Despite the complexity of Python with so many libraries to learn, its syntax is rather straightforward, and the ideas are not too difficult to understand. Therefore, it is simple to code and understand. Guido Van Rossum created Python, which was released in 1991. A wide range of machine learning, artificial intelligence and computation libraries are available from Python. Such as NumPy, SciPy, Scikit Learn, TensorFlow, Keras, Theano and more other [70]. The Keras library from Python was used to create and train suggested models, and it was executed on TensorFlow's framework.

5.1.1. TensorFlow

TensorFlow is a free and open-source framework that may be used for high-performance numerical computing. The TensorFlow is a flexible and extensible architecture that makes it possible to run computation easily on many platforms

(Tensor Processing Unit, Graphics Processing Unit, Central Processing Unit), on desktops, in data centers, and on mobile and other devices. Initially, the TensorFlow was created by Google Brain team members and researchers in the Google's AI division in Google technology company. TensorFlow is a versatile framework that can handle many scientific applications thanks to which the computational core can serve a wide range of scientific fields. It is a great choice for training DNN, since each single element of the network can be fine-tuned to provide infinite variety [71].

5.1.2. Keras

In contrast, open-source NN library means that the source code is available to the public. This library may be used with many frameworks like TensorFlow or Theano, all while also offering the ability to run and execute. The Keras was built to be accessible, modular, and extendable in order to better enable rapid experiment with deep learning. Activation functions, normalization techniques, and optimization algorithms are among the several algorithms available in the Keras library. Choosing this library provides excellent benefits: quick execution, comprehensive documentation, and a nice development environment [72].

5.2. RESULTS AND DISCUSSION

In the present thesis, to demonstrate that our method has the best performance in detecting DDoS attacks, we compared it with traditional CNN and LSTM using four metrics: accuracy, precision, recall, and F1 score. Then, we compared it with previously published studies in terms of accuracy. The comparison was made between (CLSTMNet, LSTM, and CNN) based on five experiments for each method. Moreover, we extracted from them the mean, median, and standard deviation in order to facilitate the comparison between them in terms of performance. As it is demonstrated in tables 5.1, 5.2, and 5.3.

Table 5.1 shows the performance of the CNN per execution

Execution NO.	**Accuracy**	**Precision**	**Recall**	**F1 score**
1	97.67	97.94	**97.92**	97.92
2	97.80	93.77	83.67	83.72
3	97.74	83.80	83.65	83.72
4	**97.83**	84.16	83.55	83.85
5	97.75	**98.23**	97.78	**98.00**
Mean	97.76	91.58	89.31	89.44
Median	97.75	93.77	83.67	83.85
Standard deviation	0.061	7.160	7.793	7.776

Table 5.2 shows the performance of the LSTM per execution

Execution NO.	**Accuracy**	**Precision**	**Recall**	**F1 score**
1	97.55	83.12	81.30	82.17
2	98.57	83.87	83.66	83.75
3	98.23	79.19	83.91	81.10
4	**98.97**	**84.19**	**84.39**	**84.28**
5	92.93	67.38	59.96	61.95
Mean	97.25	79.55	78.64	78.65
Median	98.23	83.12	83.66	82.17
Standard deviation	2.471	7.092	10.513	9.421

Table 5.1 shows the performance of the CNN per execution. As it is demonstrated from that table, the max for all metrics is bold, which indicates the highest rate. For more explanation, the maximum accuracy was achieved in execution number four, which had a 97.83% accuracy. Also, the mean accuracy was 97.76%, the median was 97.75%, and the SD was equal to 0.061. Moreover, execution number five holds a maximum precision of 98.23%, and the mean, medium, and SD of precision for all executions were 91.58%, 93.77%, and 7.160, respectively. In addition, the maximum recall was in the 1st execution, which was equal to 97.92%. The mean of five executions for this metric was 89.31%, the median was 83.67%, and the F1score was 7.793. Furthermore, the maximum F1 score was reached in execution number five. The highest was 98%, the mean was 89.44%, the median was 83.85%, and the F1 score was 7.776.

The performance of LSTM was demonstrated in table 5.2. The table shows the 4th execution had a maximum accuracy, precision, recall, and F1score of 98.97%, 84.19%, 84.39%, and 84.28%, respectively. On the other hand, the LSTM's mean metrics were

97.25% accuracy, 79.55% precision, 78.64% recall, and 78.65% F1 score. Also, the median metrics for the five executions were 98.23% accuracy, 83.12% precision, 83.66% recall, and 82.17% F1 score. Finally, the SD for recall and F1 score were too much, as it is clear from the table, while the SD for accuracy and precision were normal.

Table 5.3 shows the performance of the CLSTMNet per execution

Execution NO.	**Accuracy**	**Precision**	**Recall**	**F1 score**
1	99.21	92.01	99.10	94.36
2	**99.31**	**99.18**	**99.18**	**99.18**
3	99.11	99.03	98.99	99.01
4	99.19	84.75	84.78	84.77
5	99.20	84.71	84.79	84.75
Mean	99.20	91.94	93.37	92.41
Median	99.20	92.01	98.99	94.36
Standard deviation	0.071	7.188	7.835	7.250

Table 5.4. Comparison between CLSTMNet and many state-of-the-art methods in the term of accuracy

NO	Name	Year	Accuracy %	Algorithm
1	Our proposed model	current	99.20	CLSTMNet
2	S. Revathi, Dr. a. Malathi [12]	2013	99.10	RF
3	Boroujerdi and Ayat [13]	2013	96.38	ensemble of neuro-fuzzy classifier
4	Dhanabal and Shantharajah [14]	2015	99.10	J48
5	Yusof et al. [15]	2017	91.70	DCF + CSE
6	Kushwah and Ali [16]	2017	96.30	ANN + black hole optimization algorithm
7	Igbe et al. [17]	2017	98.60	DCA
8	Derakhsh et al. [18]	2018	82.44	GA
9	Hoon et al. [19]	2018	93.26	DRF
10	Idhammad et al. [20]	2018	98.23	semi-supervised
11	Anjum and Shreedhara[21]	2019	93.26	semi-supervised
12	Mukhametzyanov et al. [22]	2019	97.94	NN
13	Verma et al. [23]	2019	98.27	MAD+RF
14	Hosseini and Azizi [24]	2019	98.90	hybrid technique
15	Das et al [26]	2019	99.10	Ensemble technique
16	Ma et al. [8]	2020	92.99	CNN
17	Prathyusha and Kannayaram [27]	2020	96.70	AIS
18	Bhardwaj et al. [28]	2020	98.43	AE+DNN
19	Bagyalakshmi and Samundeeswari [29]	2020	98.74	LVQ+DT

Table 5.3 represents the performance of our proposed method (the CLSTMNet). The maximum metrics among all five executions were in the 2nd execution, as was clear from the table, which were 99.31% accuracy, 99.18% precision, 99.18% recall, and 99.18% F1 score. Additionally, the mean of all metrics for all execution were the following: accuracy = 99.2%, precision = 91.94%, recall = 93.37%, and F1 score = 92.41. Furthermore, the median of accuracy is the same as the mean, and the medians of the other three metrics were equal to 92.01% precision, 98.99% recall, and 94.36% F1 score. Moreover, the SD of accuracy was closest to zero while the SD of other metrics was closest to seven, as is obvious from the table.

By making the comparison between the results achieved from the three methods (CLSTMNet, LSTM, and CNN) as evidenced by the three tables above, we find the minimum, the maximum, the mean, and the median of all the metrics we utilized that CLSTMNet outperforms the other algorithms. On the other hand, the accuracy comparison between CLSTMNet and many state-of-the-art methods is illustrated in table 5.4. Furthermore, all of the mentioned methods were experimentally tested on the same dataset, the NSL-KDD dataset. As shown in the table, the current method outperformed the other methods in terms of accuracy. This study demonstrates to the researcher that the CLSTMNet was able to more actively detect DDoS attacks than traditional deep learning and machine learning techniques. This is because of its architecture and hybridization of two of the best deep learning algorithms, CNN and LSTM. Because of its architecture, the CNN layer is used for feature selection, and the LSTM layer is used as a predictor based on built-in memory blocks.

PART 6

CONCLUSION

This study proposed a novel method, the CLSTMNet method, that is a hybridization of two of the best deep learning algorithms to solve the biggest obstacle facing networks, the DDoS attacks. The two DL algorithms are: CNN was employed as a feature selector because of its architecture, and the LSTM was employed as a predictor because of its built-in memory block. Moreover, the architecture utilized for CLSTMNet made it more powerful to detect DDoS attacks, which were comprised of seven layers. At the same time, functions and parameters proposed in this study for learning enhanced the CLSTMNet performance. The famous NSL-KDD dataset was utilized for applying each of the CLSTMNet, LSTM, and CNN. All three methods were executed five times with 500 epochs each, utilizing the Python programming language. Also, all the experiments were applied on the TensorFlow platforms. The CLSTMNet method was compared with both the CNN and the LSTM methods depending on four metrics: accuracy, precision, recall, and F1 score. The experimental findings demonstrated that the CLSTMNet had the best performance among all the others: accuracy = 99.20%, precision = 91.94%, recall = 93.37%, and F1 score = 92.4. In the end, the CLSTMNet achieved high accuracy when compared with the accuracy of many state-of-the-art methods in terms of detecting DDoS attacks. The present study can contribute to this method being applied not only to detecting DDoS attacks but also to detecting all attacks. It can also be utilized in many different fields.

For future work, we recommend applying the CLSTMNet method to the various datasets. Also, we suggest changing the used architecture from sequential to parallel and adding a voting technique to it.

REFERENCES

1. Internet: Johnson, J., "Internet Users in the World 2021 | Statista", **https://www.statista.com/statistics/617136/digital-population-worldwide/** (2021).

2. Liu, H. and Lang, B., "Machine learning and deep learning methods for intrusion detection systems: A survey", ***Applied Sciences (Switzerland)***, 9 (20): (2019).

3. Banitalebi Dehkordi, A., Soltanaghaei, M. R., and Boroujeni, F. Z., "The DDoS Attacks Detection through Machine Learning and Statistical Methods in SDN", Journal of Supercomputing, ***Springer US***, 2383–2415 (2021).

4. Naveen Bindra and Manu Sood, "Detecting DDoS Attacks Using Machine Learning Techniques and Contemporary Intrusion Detection Dataset", ***Automatic Control And Computer Sciences***, 53 (5): 419–428 (2019).

5. Otoum, Y., Liu, D., and Nayak, A., "DL-IDS: a deep learning–based intrusion detection framework for securing IoT", ***Transactions On Emerging Telecommunications Technologies***, (September 2020): (2019).

6. Obaid, K. B., Zeebaree, S. R., and Ahmed, O. M., "Deep Learning Models Based on Image Classification: A Review", ***International Journal Of Science And Business***, 4 (11): 75–81 (2020).

7. Yuan, X., He, P., Zhu, Q., and Li, X., "Adversarial Examples: Attacks and Defenses for Deep Learning", ***IEEE Transactions On Neural Networks And Learning Systems***, 30 (9): 2805–2824 (2019).

8. Ma, L., Chai, Y., Cui, L., Ma, D., Fu, Y., and Xiao, A., "A Deep Learning-Based DDoS Detection Framework for Internet of Things", ***IEEE International Conference On Communications***, 2020-June: (2020).

9. Tasdelen, A. and Sen, B., "A hybrid CNN-LSTM model for pre-miRNA classification", ***Scientific Reports***, 11 (1): 1–9 (2021).

10. Donahue, J., Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T., "Long-term Recurrent Convolutional Networks for Visual Recognition and Description", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2625–2634 (2015).

11. Simonyan, K. and Zisserman, A., "VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION", (2015).

12. S. Revathi, D. a. M., "A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection", ***International Jornal Of Engineering Research And Technology***, 2 (12): 1848–1853 (2013).

13. Boroujerdi, A. S. and Ayat, S., "A robust ensemble of neuro-fuzzy classifiers for DDoS attack detection", ***Proceedings Of 2013 3rd International Conference On Computer Science And Network Technology, ICCSNT 2013***, 484–487 (2014).

14. Dhanabal, L. and Shantharajah, S. P., "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms", ***International Journal Of Advanced Research In Computer And Communication Engineering***, 4 (6): 446–452 (2015).

15. Yusof, A. R. A., Udzir, N. I., Selamat, A., Hamdan, H., and Abdullah, M. T., "Adaptive feature selection for denial of services (DoS) attack", ***2017 IEEE Conference On Applications, Information And Network Security, AINS 2017***, 2018-Janua: 81–84 (2017).

16. Kushwah, G. S. and Ali, S. T., "Detecting DDoS attacks in cloud computing using ANN and black hole optimization", ***2nd International Conference On Telecommunication And Networks, TEL-NET 2017***, 1–5 (2017).

17. Igbe, O., Ajayi, O., and Saadawi, T., "Denial of service attack detection using dendritic cell algorithm", ***2017 IEEE 8th Annual Ubiquitous Computing, Electronics And Mobile Communication Conference, UEMCON 2017***, 2018-Janua (October): 294–299 (2017).

18. Derakhsh, A. M., Daneshjoo, P., and Delara, C., "Using Genetic Algorithm to Improve Bernoulli Naïve Bayes Algorithm in Order to Detect DDoS Attacks in Cloud Computing Platform", ***International Journal Of Science And Engineering Investigations***, 7 (March): (2018).

19. Hoon, K. S., Yeo, K. C., Azam, S., Shunmugam, B., and De Boer, F., "Critical review of machine learning approaches to apply big data analytics in DDoS forensics", ***2018 International Conference On Computer Communication And Informatics, ICCCI 2018***, (1): 2–6 (2018).

20. Idhammad, M., Afdel, K., and Belouch, M., "Semi-supervised machine learning approach for DDoS detection", ***Applied Intelligence***, 48 (10): 3193–3208 (2018).

21. Anjum, M. and Shreedhara, K. S., "Performance Analysis of Semi-Supervised Machine Learning Approach for DDoS Detection", ***INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN TECHNOLOGY***, 6 (2): 144–147 (2019).

22. Mukhametzyanov, F., Katasev, A. S., Akhmetvaleev, A. M., and Kataseva, D.

V., "The neural network model of DDoS attacks identification for information management", ***International Journal Of Supply Chain Management***, 8 (5): 214–218 (2019).

23. Verma, P., Tapaswi, S., and Godfrey, W. W., "An Adaptive Threshold-Based Attribute Selection to Classify Requests Under DDoS Attack in Cloud-Based Systems", ***Arabian Journal For Science And Engineering***, 45 (4): 2813–2834 (2019).

24. Hosseini, S. and Azizi, M., "The hybrid technique for DDoS detection with supervised learning algorithms", ***Computer Networks***, 158: 35–45 (2019).

25. Alkasassbeh, M., Al-Naymat, G., B.A, A., and Almseidin, M., "Detecting Distributed Denial of Service Attacks Using Data Mining Techniques", ***International Journal Of Advanced Computer Science And Applications***, 7 (1): 436–445 (2016).

26. Das, S., Mahfouz, A. M., Venugopal, D., and Shiva, S., "DDoS Intrusion Detection Through Machine Learning Ensemble", ***Proceedings - Companion Of The 19th IEEE International Conference On Software Quality, Reliability And Security, QRS-C 2019***, 471–477 (2019).

27. Prathyusha, D. J. and Kannayaram, G., "A cognitive mechanism for mitigating DDoS attacks using the artificial immune system in a cloud environment", ***Evolutionary Intelligence***, (0123456789): 1–12 (2020).

28. Bhardwaj, A., Mangat, V., and Vig, R., "Hyperband tuned deep neural network with well posed stacked sparse autoencoder for detection of ddos attacks in cloud", ***IEEE Access***, 8: 181916–181929 (2020).

29. Bagyalakshmi, C. and Samundeeswari, E. S., "DDOS attack classification on cloud environment using machine learning techniques with different feature selection methods", ***International Journal Of Advanced Trends In Computer Science And Engineering***, 9 (5): 7301–7308 (2020).

30. Lapiedra, J., "The Information Security Process Preve ntion, Detection and Response", (2000).

31. Chapple, M. and Seidl, D., "CompTIA CySA+ Study Guide: Exam CS0-001", 24 (2017).

32. Ozalp, A. N., Albayrak, Z., and Zengin, A., "Expansion of Wireless Networks using IEEE 802.3af Protocol in Protected Areas", (2017).

33. "HIDS (Host-Based Intrusion Detection System) - Bauman National Library", **https://en.bmstu.wiki/index.php?title=HIDS_(Host-Based_Intrusion_Detection_System)&mobileaction=toggle_view_mobile** (2021).

34. Avalappampatty Sivasamy, A. and Sundan, B., "A dynamic intrusion detection system based on multivariate Hotelling's T2 statistics approach for network environments", ***Scientific World Journal***, 2015: (2015).

35. Internet: Cloudflare, "What Is a Denial-of-Service (DoS) Attack? | Cloudflare", **https://www.cloudflare.com/ru-ru/learning/ddos/glossary/denial-of-service/** (2021).

36. Whitman, M. E., "Principles of Information Security", Cengage Learning, 4. Ed., ***Cengage Learning***, (2016).

37. Internet: Cloudflare, "Famous DDoS Attacks | Biggest DDoS Attacks | Cloudflare", **https://www.cloudflare.com/learning/ddos/famous-ddos-attacks/** (2021).

38. Internet: Cimpanu, C., "AWS Said It Mitigated a 2.3 Tbps DDoS Attack, the Largest Ever", **https://www.zdnet.com/article/aws-said-it-mitigated-a-2-3-tbps-ddos-attack-the-largest-ever/** (2021).

39. Anstee, D., Chui, C. F., Bowen, P., and Sockrider, G., "WORLDWIDE INFRASTRUCTURE SECURITY REPORT, Arbor Networks Inc.", Westford, MA, USA, (2017).

40. Lima Filho, F. S. De, Silveira, F. A. F., De Medeiros Brito Junior, A., Vargas-Solar, G., and Silveira, L. F., "Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning", ***Security And Communication Networks***, 2019: (2019).

41. Tuan, T. A., Long, H. V., Son, L. H., Kumar, R., Priyadarshini, I., and Son, N. T. K., "Performance evaluation of Botnet DDoS attack detection using machine learning", ***Evolutionary Intelligence***, 13 (2): 283–294 (2020).

42. Beitollahi, H., Deconinck, G., Beitollahi, H., and Deconinck, G., "ConnectionScore: A Statistical Technique to Resist Application-layer DDoS Attacks", ***Journal Of Ambient Intelligence And Humanized Computing***, 5 (3): 425–442 (2014).

43. Ajeetha, G. and Madhu Priya, G., "Machine Learning Based DDoS Attack Detection", ***2019 Innovations In Power And Advanced Computing Technologies, I-PACT 2019***, 1: 1–5 (2019).

44. Yusof, M. A. M., Ali, F. H. M., and Darus, M. Y., "Detection and Defense Algorithms of Different Types of DDoS Attacks Using Machine Learning", ***Lecture Notes In Electrical Engineering***, 488: 370–379 (2018).

45. Luckner, M., "Conversion of decision tree into deterministic finite automaton for high accuracy online SYN flood detection", ***Proceedings - 2015 IEEE Symposium Series On Computational Intelligence, SSCI 2015***, (December 2015): 75–82 (2015).

46. "What Is a Smurf Attack? – HeelpBook", **https://www.heelpbook.net/2014/what-is-a-smurf-attack/** (2021).

47. Alam, M. F., "Application Layer DDoS A Practical Approach & Mitigation Techniques", ***Apricot 2014***, 55 (2014).

48. Daumé III, H., "A Course in Machine Learning", 5: (2012).

49. Alabadi, M. and Albayrak, Z., "Q-Learning for Securing Cyber-Physical Systems: A survey", (2020).

50. "Supervised vs Unsupervised Learning: Key Differences", **https://www.guru99.com/supervised-vs-unsupervised-learning.html** (2021).

51. Brindha, S., Prabha, K., and Sukumaran, S., "A survey on classification techniques for text mining", ***ICACCS 2016 - 3rd International Conference On Advanced Computing And Communication Systems: Bringing To The Table, Futuristic Technologies From Arround The Globe***, 2 (i): 1–5 (2016).

52. Zhang, C., Vinyals, O., Munos, R., and Bengio, S., "A Study on Overfitting in Deep Reinforcement Learning", (2018).

53. Zulkepli, F. S., Ibrahim, R., and Saeed, F., "Data pre-processing techniques for publication performance analysis", ***Lecture Notes On Data Engineering And Communications Technologies***, 5: 59–65 (2018).

54. Altunay, H. C., Albayrak, Z., Özalp, A. N., and Çakmak, M., "Analysis of Anomaly Detection Approaches Performed Through Deep Learning Methods in SCADA Systems", (2021).

55. Ayturan, Y. A., Ayturan, Z. C., and Altun, H. O., "Air Pollution Modelling with Deep Learning: A Review", ***International Journal Of Environmental Pollution And Environmental Modelling***, 1 (3): 58–62 (2018).

56. Lecun, Y., Bengio, Y., and Hinton, G., "Deep learning", ***Nature***, 521 (7553): 436–444 (2015).

57. Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A., "A detailed analysis of the KDD CUP 99 data set", ***IEEE Symposium On Computational Intelligence For Security And Defense Applications, CISDA 2009***, (July): (2009).

58. Zeng, H., Edwards, M. D., Liu, G., and Gifford, D. K., "Convolutional neural network architectures for predicting DNA-protein binding", ***Bioinformatics***, 32 (12): i121–i127 (2016).

59. Ketkar, N. and Santana, E., "Deep Learning with Python", ***Springer***, (2017).

60. Kingma, D. P. and Ba, J. L., "Adam: A method for stochastic optimization", ***3rd International Conference On Learning Representations, ICLR 2015 - Conference Track Proceedings***, 1–15 (2015).

61. Can Altunay, H. and Albayrak, Z., "Network Intrusion Detection Approach Based on Convolutional Neural Network", ***European Journal Of Science And Technology Special Issue***, (26): 22–29 (2021).

62. Internet: Gudikandula, P., "Recurrent Neural Networks and LSTM Explained | by Purnasai Gudikandula | Medium", **https://purnasaigudikandula.medium.com/recurrent-neural-networks-and-lstm-explained-7f51c7f6bbb9** (2021).

63. Internet: Thakur, D., "LSTM and Its Equations", **https://medium.com/@divyanshu132/lstm-and-its-equations-5ee9246d04af** (2021).

64. ISSA, A. and Albayrak, Z., "CLSTMNet: A Deep Learning Model for Intrusion Detection", ***Journal Of Physics: Conference Series***, 1973 (1): 012244 (2021).

65. Glorot, X. and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks", ***Proceedings Of The Thirteenth International Conference On Artificial Intelligence And Statistics***, 249–256 (2010).

66. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L., "Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions", Journal of Big Data, ***Springer International Publishing***, (2021).

67. Bhuyan, M. H., Bhattacharyya, D. K., and Kalita, J. K., "Network Anomaly Detection: Methods, Systems and Tools", ***IEEE COMMUNICATIONS SURVEYS & TUTORIALS***, 16 (1): (2014).

68. Macías, S. G., Gaspary, L. P., and Botero, J. F., "ORACLE: Collaboration of Data and Control Planes to Detect DDoS Attacks", (2020).

69. De, V., Rios, M., Inácio, P. R. M., Magoni, D., Freire, M., and Freire, M. M., "Detection of reduction-of-quality DDoS attacks using Fuzzy Logic and machine learning algorithms", ***Computer Networks***, 186: (2021).

70. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., and Dubourg, V. and others, "Scikit-learn: Machine Learning in Python", ***The Journal Of Machine Learning Research***, 12: 2825–2830 (2011).

71. Rampasek, L. and Goldenberg, A., "TensorFlow: Biology's Gateway to Deep Learning?", ***Cell Systems***, 2 (1): 12–14 (2016).

72. Erickson, B. J., Korfiatis, P., Akkus, Z., Kline, T., and Philbrick, K., "Toolkits and Libraries for Deep Learning", ***Journal Of Digital Imaging***, 30 (4): 400–405 (2017).

RESUME

Ahmed Sardar Ahmed ISSA graduated first from Zozan elementary school. He completed high school education in Zakho High School, after that, he obtained bachelor degree from University of Duhok/College of Science/Computer Department in 2011. Then in 2019, he started his master education in Karabük University/Faculty of Applied Science/Computer Engineering Department.