PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

YAHIA FARES UNIVERSITY OF MEDEA

Faculty of Science

Department of Mathematics and Computer Science

Graduation Project

Presented for obtaining the Master's Degree in Computer Science

Option: Systems Engineering and Web Technology

THEME

DDoS attacks detection using Deep Learning

Prepared By:

- GHENIA Feriel
- ZEROUAL Insaf

The jury consists of:

President	Mr. CHETOUANE Mohamed
Reviewer	Mr. BENAISSA Azzedine
Supervisor	Mrs. CHERGUI Nadjah

Class of 2023/2024

Thanks

(And their final prayer will be, 'Praise be to Allah, Lord of the Worlds.')

Praise be to Allah, out of love, gratitude, and appreciation for the beginning and the end.

We extend our deepest thanks and appreciation to everyone who contributed to the completion of this research. Special thanks to our supervisor, Professor **CHERGUI Nadjah**, for her guidance and corrections throughout this period. Her valuable advice, continuous support, and the time she dedicated to us have always been a source of inspiration. We truly appreciate her dedication and patience with us. She is indeed an exemplary model of a teacher.

We also express our sincere thanks to the teaching staff at the Faculty of Mathematics and Computer Science for all they have provided us during these five years.

We thank the members of the jury for their interest in our research through examining and reviewing our work and enriching it with their suggestions.

This project would not have been realized without the support of friends and family. Special thanks to everyone who offered us words of encouragement, sincere advice, or prayers that accompanied us throughout these years. Every support and assistance had a significant impact on achieving this accomplishment.

In conclusion, we affirm that this work is the fruit of effort, study, and continuous cooperation, and we are grateful to everyone who contributed to its success, directly or indirectly. Thank you all, and we ask Allah for success in our future endeavors.

الإهداء

"يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ"

اللّهم كمّا أنعمّتَ فَزِدْ و كما زدتّ فَبارك و كما باركتَ فَتمّ و كما أتممت فثبّت ... لم تكن الرحلة قصيرة ولا ينبغي لها أن تكون ،لم يكن الحلم قريبا ولا الطريق كان محفوفا بالتسهيلات ،لكن بفضل من الله فعلتها .. فالحمد لله حبا وشكرا وامتنانا .. الحمد لله الذي يسر لي البدايات وبلغني النهايات .. ممتنة للطيبين الذين أوقعو بقلبي أثرا يروى بالدعوات والذكر الحسن ،لمن حلو فأمطرو فأزهرو بخافقي جنانا..

الى أبي .. إلى من أحمل اسمه بكل فخر..إلى أعظم شخص .. داعمي الأول الذي ساندني وعلمني أن الحياة صراع وسلاحها العلم.. إلى من علمني أن أرتقي سلم الحياة بحكمة وصبر..

الى أمي .. إلى من هي في الحياة حياة إلى التي ينحني إليها الحرف حبا وامتنانا ..أأهديك أملا .. وليس في الأرض أمل كالذي أدركت حقيقته من عينيك ... أم أهديك نجاحا ونجاحي الحقيقي تحت قدميك ... فأي شيء في هذا اليوم أهديه إليك...

الى إخوتي .. نادية.. رضا ..مريم.. نور الدين.. عبدو .. وبشير

إلى من قيل فيهم ... سنشد عضدك بأخيك ... إلى من أرى جمال أيامي فيهم ..إلى من آمنوا بقدراتي داعمين لي دون كلل أو ملل إلى ذلك المستودع الكبير من القوة والحب شكرا من القلب لجعلكم الحياة ألطف بكلملتكم وأفعالكم بمبادراتكم اللامتوقعة.. شكرا لصانعي أيامي بلطفهم

إلى أولاد أختي (إسلام و الأمورة ملاك) وزوجها نبيل..

الى كل عائلة غنية.. من أعمامي ... أخوالي ... خالاتي.. و كل أولادهم و بناتهم ...

الى صديقاتي .. إنصاف ...رميسة .. زهراء .. آية.. سندس..

إلى رفاق الخطوة الأولى والخطوة ما قبل الأخيرة ... إلى من كن خلال السنين العجاف سحابا ممطرا ..فلا يؤنس الروح إلا روحا تألفها ..رعى الله قلوبا خلقت لتسعدنا..

أنا ممتنة إلى الذين حملوا أقدس رسالة في الحياة ..الذين مهدوا لنا طريق العلم و المعرفة.. إلى كل الأساتذة الذين كان لهم الفضل في مسيرتي الدراسية ووصولي الى هنا..

الى نفسي.. العظيمة الفتية التي تحملت كل العثرات وأكملت رغم الصعوبات..شكرا لنفسي التى صبرت واجتهدت إلى أن حققت .. فما ضاع جهد الأمس والله يجزي الحسن بالإحسان.. ومن قال أنا لها نالها ..

تم بحمد الله تخرجي و الروح من فرط السرور توهجت..

فريال..

الإهداء

(فَرِحِينَ بِمَآ ءَاتَىٰهُمُ ٱللَّهُ مِن فَضْلِهِۦ)

لم تكن الرحلة قصيرة ولا ينبغي لها أن تكون،
لم يكن الحلم قريبا ولا الطريق كان محفوفا بالتسهيلات،
لكن ها أنا اليوم أخطي خطوات تخرجي، أقطف ثمار جهدي وأرفع قبعتي بكل فخر،
فالحمد لله الذي يسر البدايات وبلغنا النهايات بفضله وكرمه.

أهدي هذا النجاح لنفسي أولا ثم إلى من سعى معي لإتمام هذه المسيرة.
إلى من لا ينفصل إسمه عن إسمي، ذلك الرجل العظيم الذي علمني الحياة وأعطاني دون مقابل، إلى فخري وصانع قوتي **"أبي الغالي"**
إلى من جعل الجنة تحت أقدامها، تلك التي يسرت لي الشدائد بدعائها، إلى القلب الحنون **"أمي العزيزة"**

إلى المؤنسات الغاليات، الى إخواتي حبيباتي **"رتاج، اسراء ودعاء"**
إلى عزوتي وسندي في هذه الحياة، **" أخي إسلام"** وصغير بيتنا **"إياد"**

إلى من ساندتني بكل حب عند ضعفي، وأزاحت عني طريق المتاعب، إلى بنت خالتي **"بشرى"**

إلى من يذكرني بقوتي، ويقف خلفي كظلي، إلى من زرع الثقة والإصرار داخلي **"خطيبي"**

إلى من رافقوني طوال سنوات الدراسة، وكانو لي السند والعون، إلى صديقاتي **"آية، رميسة وفاطمة"**

إلى الأستاذة **"شرقي نجاح"** التي أشرفت على هذه المذكرة
وأحب أن أختم الإهداء بصاحبة الفضل العظيم صديقة الرحلة والنجاح **"فريال"**

وأخيرا من قال أن لها "نالها" وأنا لها إن أبت رغما عنها أتيت بها
الحمد لله الذي توالى بمنه وكرمه لفرحة التمام والختام.

إنصاف

Abstract

The internet faces severe threats from distributed denial of service (DDoS) attacks, which significantly impact network security. DDoS attacks aim to disrupt services by flooding servers with excessive messages, rendering them inaccessible to legitimate users. Detecting and mitigating these attacks is crucial, demanding adaptive solutions. Our project addresses these challenges by leveraging deep learning for enhanced DDoS attack detection. Through advanced model training and feature engineering, we developed a hybrid Bi-LSTM-GRU model achieving remarkable accuracy of 99.81%. This underscores deep learning's efficacy in predictive analytics and emphasizes the importance of feature selection for optimizing model performance. Our work contributes to smarter, responsive electronic security systems, vital for combating evolving digital threats.

Key words: Distributed denial of service (DDoS), Network security, Deep learning, Attack detection, Feature engineering, Hybrid model, GRU, Bi-LSTM, Predictive analytics, Cybersecurity, Digital threats.

Résumé

L'internet est confronté à de serieuses menaces d'attaques par deni de service distribue (DDoS), qui ont un impact considérable sur la sécurité du réseau. Les attaques DDoS visent à perturber les services en inondant les serveurs de messages excessifs, les rendant inaccessibles aux utilisateurs légitimes. Détecter et atténuer ces attaques est essentiel et nécessite des solutions adaptatives. Notre projet répond à ces défis en tirant parti de l'apprentissage profond pour améliorer la détection des attaques DDoS. Grâce à une formation avancée sur les modèles et à l'ingénierie des attributs, nous avons développé un modèle hybride Bi-LSTM -GRU atteignant une précision remarquable de 99,81%. Cela confirme l'efficacité de l'apprentissage profond dans l'analyse prédictive et souligne l'importance de la sélection des attributs pour améliorer les performances du modèle. Notre travail contribue à créer des systèmes de cybersécurité plus intelligents et plus réactifs, ce qui est essentiel pour lutter contre l'évolution des menaces numériques.

Mots clés: Deni de service distribue (DDoS), Sécurité du réseau, Apprentissage Profond, Détection des attaques, Ingénierie des attributs, GRU, Bi-LSTM, Modèle hybride, Analyse prédictive, Cybersécurité, Menaces numériques.

ملخص

يواجه الانترنت تهديدات خطيرة من هجمات رفض الخدمة الموزعة (DDoS)، والتي تؤثر بشكل كبير على أمن الشبكة. تهدف هجمات DDoS إلى تعطيل الخدمات عن طريق إغراق الخوادم برسائل زائدة، مما يجعلها غير قابلة للوصول للمستخدمين الشرعيين. يعد اكتشاف هذه الهجمات والتخفيف من آثارها أمرًا بالغ الأهمية، ويتطلب حلولاً تكيفية. يعالج مشروعنا هذه التحديات من خلال الاستفادة من التعلم العميق لتحسين اكتشاف هجمات .DDoS ومن خلال التدريب المتقدم على النماذج وهندسة الميزات، قمنا بتطوير نموذج GRU-Bi-LSTM الهجين محققًا دقة ملحوظة تبلغ 99.81%. وهذا يؤكد فعالية التعلم العميق في التحليلات التنبؤية ويؤكد على أهمية اختيار الميزات لتحسين أداء النموذج. يساهم عملنا في إنشاء أنظمة أمان إلكترونية أكثر ذكاءً واستجابة، وهو أمر حيوي لمكافحة التهديدات الرقمية المتطورة.

الكلمات المفتاحية: هجمات رفض الخدمة الموزعة (DDoS)، أمن الشبكة، التعلم العميق، اكتشاف هجمات، هندسة الميزات، النموذج الهجين، GRU، Bi-LSTM، الأمن الإلكتروني، التحليلات التنبؤية، التهديدات الرقمية.

Contents

1 DDoS attacks background 16

1.1 Introduction 17

1.2 The Open Systems Interconnection (OSI) model 17

1.3 Packet and components 19

1.3.1 Packet Definition 19

1.3.2 Packet components 19

1.4 TCP and UDP Protocol 21

1.4.1 TCP 21

1.4.2 UDP 22

1.4.3 The differences between TCP and UDP 22

1.5 How a TCP connection works 22

1.5.1 Establish connection 22

1.5.2 Send packets of data 23

1.5.3 Close the connection 24

1.5.4 Detecting lost packets 24

1.6 Denial-of-service (DoS) attacks 25

1.7 Distributed denial of service (DDoS) attack 26

1.8 The Difference between DoS and DDoS Attacks 26

1.9 Types of DDoS attacks 27

1.9.1 Reflection-based attacks 27

1.9.2 Exploit-based attacks 27

1.10 Potential Impact of DDoS Attacks 28

1.11 Intrusion Detection System IDS 28

1.12 Conclusion 29

2 Literature review **30**

2.1 Introduction 31

2.2 Definition of Deep Learning 31

2.3 Taxonomies of the DL approach 33

2.4 Popular Deep Learning Algorithms 33

2.4.1 Artificial Neural Network (ANN) 33

2.4.2 Deep Neural Network (DNN) 34

2.4.3 Convolutional neural network (CNN) 35

2.4.4 Recurrent Neural Network (RNN) 36

2.4.5 Long Short-Term Memory (LSTM) 37

2.5 A Multilayer Perceptron (MLP) 38

2.6 A Gated Recurrent Unit (GRU) 38

2.7 A hybrid neural network (HNN) 39

2.8 Steps before applying the DL model 40

2.8.1 Data preprocessing 40

2.8.2 Data balance 42

2.8.3 Data splitting 42

2.9 Performance Metrics 44

2.9.1 The confusion matrix 44

2.9.2 The precision 45

2.9.3 The Recall 45

2.9.4 F1-Score 45

2.9.5 Accuracy 46

2.10 Related Works 46

2.11 Discuss the various factors 53

2.12 Conclusion 55

3 Research Methodology **56**

3.1 Introduction 57

3.2 Methodology 57

3.3 Diagram of the Proposed Approach 58

3.4 Dataset preparation 58

3.4.1 Extract a Balanced mini-dataset 59

3.4.2 Feature Selection and Refinement 60
3.4.3 Data preprocessing 65
3.4.4 Data Splitting 66
3.5 Description of the implemented Models 66
3.5.1 MLP Model 66
3.5.2 GRU Model 67
3.5.3 LSTM Model 68
3.5.4 BI-LSTM+GRU Model 68
3.5.5 CNN+GRU Model 69
3.5.6 BI-LSTM+CNN Model 70
3.6 Conclusion 71

4 Evaluation and tests **72**
4.1 Introduction 73
4.2 Tools and environment 73
4.2.1 Material used 73
4.2.2 Programming language 73
4.2.3 Modules used 75
4.2.4 Development environment 76
4.3 Evaluation and discussion 76
4.3.1 MLP Model 76
4.3.2 GRU Model 77
4.3.3 LSTM Model 78
4.3.4 BI-LSTM+GRU Model 79
4.3.5 CNN+GRU Model 80
4.3.6 BI-LSTM+CNN Model 80
4.4 Comparison of the proposed models 81
4.5 Discussion 82
4.6 Comparison of our results with related works 84
4.7 Conclusion 86

List of Figures

1.1 Seven Layers of OSI Model 19
1.2 Components of Packets 20
1.3 TCP Open Connection 23
1.4 TCP Send Packets 23
1.5 TCP Close Connection 24
1.6 TCP Packet Retransmission with RTO Calculation 25

2.1 Sub-Branches of AI 32
2.2 Deep Learning Neural Network Architecture 32
2.3 A taxonomy of DL techniques 33
2.4 Structure of ANN 34
2.5 Structure of CNN 35
2.6 Structure of RNN 36
2.7 Types Of RNN 37
2.8 Structure of LSTM 37
2.9 Data Processing Stages 42
2.10 The Confusion Matrix 45

3.1 The General Flow of The Process 58
3.2 The Distribution and statistic of DDoS attacks inside CICDDoS2019 59
3.3 Ratio circle for distributing data categories 60
3.4 MLP model Architecture 67
3.5 GRU model Architecture 67
3.6 LSTM model Architecture 68
3.7 BILSTM+GRU model Architecture 69

3.8 CNN+GRU model Architecture 69
3.9 BI-LSTM+CNN model Architecture 70

4.1 Report MLP 76
4.2 MLP model Accuracy/Loss curve in terms of Epochs 77
4.3 Report GRU 77
4.4 GRU model Accuracy/Loss curve in terms of Epochs 78
4.5 Report LSTM 78
4.6 LSTM model Accuracy/Loss curve in terms of Epochs 79
4.7 Report BI-LSTM+GRU 79
4.8 BI-LSTM+GRU model Accuracy/Loss curve in terms of Epochs 79
4.9 Report CNN+GRU 80
4.10 CNN+GRU model Accuracy/Loss curve in terms of Epochs 80
4.11 Report BI-LSTM+CNN 81
4.12 BI-LSTM+CNN model Accuracy/Loss curve in terms of Epochs 81
4.13 Bar chart of the accuracy of the models used in the study 84
4.14 Comparing our models with those of previous studies 85

List of Tables

1.1 Comparison of TCP and UDP 22

2.1 Data splitting 44

2.2 Analysis of Studies 53

3.1 Description of extracted features 65

4.1 The results of testing the proposed models 82

4.3 Compare the results of our best model with related works 86

List of Abbreviations

AI Artificial Intelligence

ANN Artificial Neural Network

Bi-LSTM Bidirectional Long Short-Term Memory

CNN Convolutional Neural Network

CSV Comma-Separated Values

DDoS Distributed Denial of Service

DL Deep Learning

DNN Deep Neural Network

DoS Denial of Service

FFNN Feedforward Neural Networks

GPU Graphics Processing Unit

GRU Gated Recurrent Unit

HNN hybrid Neural Network

HTM Hierarchical Temporal Memory

ICMP Internet Control Message Protocol

IDS Intrusion Detection System

IoT Internet of Things

IP Internet Protocol

LLC Logical Link Control

LSTM Long Short-Term Memory

MAC Media Access Control

MLP Multilayer Perceptron

ML Machine Learning

NLP Natural Language Processing

NSGA Non-dominated Sorting Genetic Algorithm

OSI Open Systems Interconnection

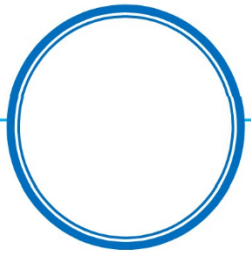
PCAP Packet Capture

RNN Recurrent Neural Network

SDN Software-Defined Networking

TCP Transmission Control Protocol

UDP User Datagram Protocol



General

Introduction

In light of the rapid development of digital technology and computing capabilities, networks have become the backbone of the infrastructure supporting various economic and social activities, making their security critically important. As communication and interaction over the Internet increase, cyber threats to networks have grown more complex and diverse. Among these threats, Distributed Denial-of-Service (DDoS) attacks have emerged as one of the most frequent and damaging. These attacks can completely paralyze digital services and critical systems, causing significant financial and reputational losses to organizations and individuals alike.

Deep learning, a subset of artificial intelligence, has gained significant importance in the field of network security. Unlike traditional machine learning methods, which are often limited by their inability to process unstructured data and their lack of accuracy in facing new attacks, deep learning techniques excel at analyzing large and complex datasets. By constructing and training artificial neural networks with multiple layers, deep learning models can learn from vast amounts of data, identifying intricate patterns and features automatically. This capability makes deep learning an ideal tool for detecting DDoS attacks, as it can distinguish subtle and varying patterns that separate attacks from legitimate traffic. By developing predictive models, deep learning provides a higher level of security and protection, enabling early detection and mitigation of potential threats.

Problem Statement

Distributed Denial-of-Service (DDoS) attacks pose a formidable challenge to network security due to their complexity and evolving nature. Unlike attacks that exploit specific vulnerabilities, DDoS attacks focus on overwhelming a network or system with a massive volume of traffic, thereby disrupting normal operations and making it difficult for legitimate users to access services. Detecting DDoS attacks is inherently difficult for several reasons:

1. **High Traffic Volume**: DDoS attacks generate an enormous amount of traffic, which can obscure legitimate traffic and strain the capacity of detection systems.
2. **Varied Attack Patterns**: DDoS attacks can vary widely in terms of their patterns, rates, and volumes. This variability makes it challenging to create a single detection method that is effective against all types of DDoS attacks.

3. **Evolving Attack Strategies**: Attackers continually update their tactics to bypass existing security measures. This requires detection methods to be adaptive and capable of recognizing new patterns of attack.
4. **Use of Compromised Devices**: Attackers often use a network of compromised devices, known as a botnet, to launch their attacks. This distributed nature makes it difficult to trace the source of the attack and filter out malicious traffic in real time.

To effectively detect and mitigate DDoS attacks, it is crucial to identify and analyze a set of features that impact detection. Key features that are instrumental in distinguishing between legitimate traffic and DDoS attack traffic include:

- **Traffic Volume**: Anomalous spikes in traffic volume can indicate a potential DDoS attack.
- **Packet Rate**: A high rate of packets per second, particularly if they are from a wide range of sources, can signal an ongoing attack.
- **Protocol Distribution**: Unusual distributions of network protocols (e.g., a sudden increase in UDP traffic) can be indicative of a DDoS attack.
- **Session Duration**: Short-lived sessions that flood the network can be characteristic of certain types of DDoS attacks.
- **Port Usage**: Random or unusual port usage patterns may suggest a DDoS attack targeting various services.

Understanding these features and leveraging advanced analytical techniques, such as deep learning, can significantly enhance the ability to detect and mitigate DDoS attacks. By analyzing traffic data and identifying abnormal patterns early, we can develop robust defense mechanisms that ensure the availability and reliability of network services.

Challenges

- DDoS Attacks have different patterns, rates, and volumes, making it challenging to develop a single, effective detection method.
- Attackers continually update their strategies to evade detection, making it essential to develop methods that can adapt to these changes.

- The Sheer volume of traffic generated during a DDoS attack can overwhelm detection systems, making it challenging to filter out malicious traffic from legitimate traffic.
- Detection systems must function efficiently within constraints such as limited processing power and memory, ensuring timely and effective detection without overloading the network infrastructure.

Purpose and objectives

- The primary goal of our research is to determine the relevant features it identify DDoS attacks.
- Apply different deep learning models in order to enhance the detection rate
- Provide a comparative study based on features and the applied models.
- Ensure that detection methods operate efficiently, handling high volumes of traffic without significant resource utilization.
- Develop DL-based methods that can accurately detect DDoS attacks with high precision and recall.
- Develop methods that can adapt to evolving attack strategies and patterns.

Organization of Work

This work consists of four chapters beside a general introduction and general conclusion. A description of each chapter is presented in the following lines:

Chapter 01:

In this chapter, we will provide a comprehensive explanation of the concept of Distributed Denial of Service (DDoS) attacks and their importance in the field of information security. We will focus on its different types, in addition to the negative effects it can have on systems and networks. This chapter aims to clarify the basic background and provide a solid introduction to understanding the challenges we will face in developing DDoS attack detection models in subsequent chapters.

Chapter 02:

Within the scope of this chapter, we will provide a comprehensive overview of the different classifications of deep learning and the technical architectures available, focusing on the basic elements related to model analysis and evaluation. We will also review previous studies relatedto detecting DDoS attacks using deep learning, and discuss the various factors affecting these studies. Which contributes to a deeper understanding of the challenges facing efforts to detect and combat these attacks.

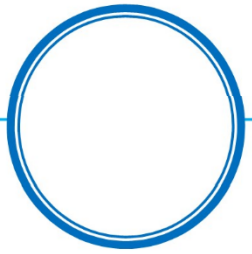
Chapter 03:

In the third chapter, we will focus on presenting the proposed models in detail, with special emphasis on the methodological steps followed in their application. We will also explain the steps for improving and modifying models, including selecting optimal features and tuning parameters.

Chapter 04:

In the last chapter we will present the work environment, as well as the tools used in our study. We also discuss the obtained results and comparing them with other works in the literature.

A general conclusion will end our thesis, which will contain the open views from this research work.



CHAPTER I

DDoS attacks background

1.1 Introduction

In the world of interconnected networks, DDoS attacks are among the most pressing and complex threats. In this section, we will explore the basic concepts of networking and DDoS attacks. We'll delve into understanding the layers that ensure information flows smoothly and securely, without going into detail about individual protocols. We shed light on the systems that manage data and organize its exchange, enabling us to appreciate the complexity and precision requiredto protect these networks. We will also focus on defining these attacks, their different types, and the negative effects they can have on network systems.

1.2 The Open Systems Interconnection (OSI) model

The Open Systems Interconnection (OSI) model, developed by the International Organization for Standardization (ISO) in 1984, is recognized as an effective methodology for grasping the details and complexities of network communication. It continues to be widely used to understand and standardize the functions of communication or computing systems, regardless of the underlying infrastructure or technologies employed. It is also utilized in diagnosing network problems and it aims to enable different systems to work together through the reliance on standardized protocols. This is achieved by dividing the broad process of network communications into seven distinct layers, each performing specific functions that facilitate network connectivity. Here is a brief overview of each layer, focusing on their roles and operations: [1]

- **Application Layer (Layer 7):** This is the top layer in the OSI model where users directly interact with software applications. It provides network services to end-users and applications such as web browsing, email, and file transfer. It operates to provide protocols that allow programs to send and receive information and present useful data to users. Examples of application layer protocols include HTTP for web browsing, SMTP for email transfer, FTP for file transfer, and DNS for domain name resolution.
- **Presentation Layer (Layer 6):** The presentation layer acts as a translator in the network, responsible for data encryption, compression, and translation to ensure that data is sent across the network in a readable format for the receiving system.
- **Session Layer (Layer 5):** The session layer manages and monitors communications between computers. It is responsible for establishing communication channels called

sessions. It oversees opening these sessions, ensuring their continuity during data transfer, and closing them upon completion of the connection.

- **Transport Layer (Layer 4):** At this layer, data from Layer 5 is segmented into parts and reassembled upon receipt, converting it into usable data. Services in the transport layer also include flow control and reliable data transfer at a rate compatible with the receiver's connection speed, as well as mechanisms for congestion control. Examples: The Transmission Control Protocol (TCP) provides reliable, orderly, and error-checked delivery of a stream of bytes. The User Datagram Protocol (UDP) is used for message transfer where error-checking and correction are not required, making it faster for broadcasting, video conferences, and games.
- **Network Layer (Layer 3):** At this level, segments are divided into network packets and reassembled upon receipt. This layer handles logical addressing and routing according to unique network addresses (IP addresses).
- **Data Link Layer (Layer 2):** The data link layer constructs, manages, and sends frames from and to the physical layer, controlling how data is placed on the physical medium. This layer consists of two sublayers: 1. Logical Link Control (LLC): Defines network protocols and handles error detection and correction. - Media Access Control (MAC): Uses MAC addresses to connect devices and determine permissions for data transmission and reception.
- **Physical Layer (Layer 1):** Concerned with sending and receiving the bitstream over a physical medium with bit rate control. It converts frames into electrical or optical signals and sends them across the physical medium, which includes devices, cables, opticalfibers, and wireless waves.

The OSI model can be seen in the figure 1.1 below:

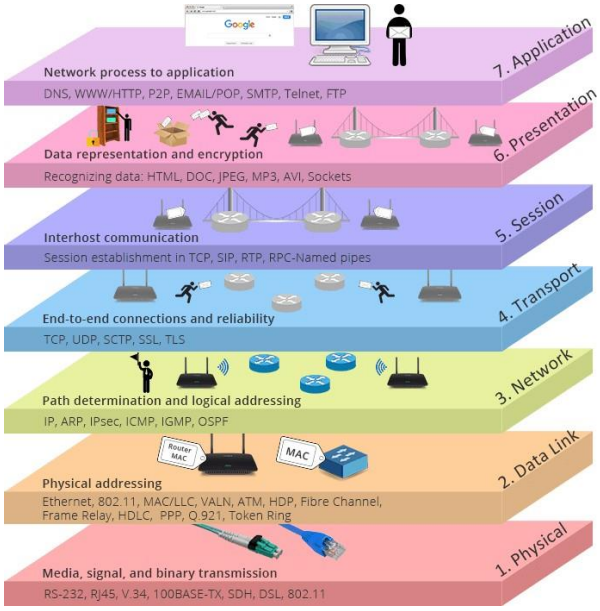


Figure 1.1: Seven Layers of OSI Model [2]

Each layer of the OSI model has a distinct role but works closely with the layers directly above and below it to provide seamless network connectivity. Understanding these layers aids in diagnosing network issues, detecting errors, and repairing them.

1.3 Packet and components

1.3.1 Packet Definition

A packet is a small data unit that is sent over computer networks that. when data is sent over the Internet, is divided into small parts known as packets for easier transportation and management. It is then reassembled at the receiving end to compose the original message. This system enables high bandwidth efficiency and facilitates network management and troubleshooting. [3]

1.3.2 Packet components

Here is a detailed explanation of the components of the packet as it is illustrated in the figure 1.2:

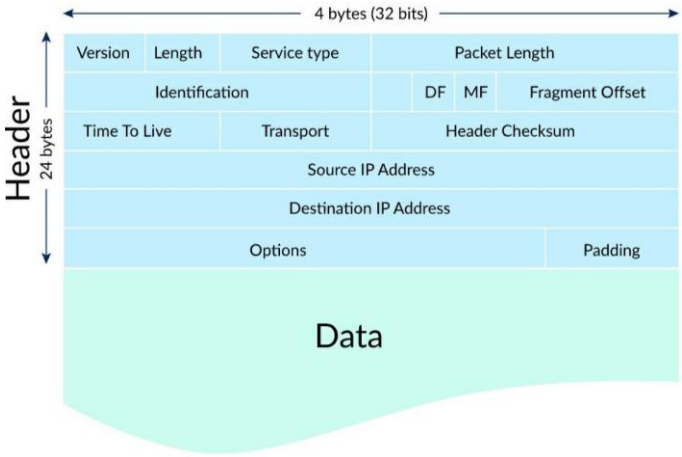


Figure 1.2: Components of Packets [4]

1. **Header:** This part of the packet contains all the information necessary to direct the packet and deliver it to the correct destination. It includes:

   1.1. **Version:** Indicates the version of the IP protocol used. For IPv4, this is always set to 4.

   1.2. **Internet Header Length (IHL):** Specifies the length of the header in 32-bit words. The minimum value is 5, which means a length of 20 bytes.

   1.3. **Type of Service (ToS):** known as Differentiated Services Code Point (DSCP), it determines datagram priority and routing request by low delay, high throughput, or highly reliable.

   1.4. **Total Length:** Specifies the size of the entire packet, including the header and data, in bytes.

   1.5. **Identification:** A unique identifier assigned to each packet that is used to reassemble fragmented packets at the destination.

   1.6. **Flags:** It consists of three control flags:

   - Bit 0: reserved; It must be zero.
   - Bit 1: Do not fragment (DF).
   - Bit 2: More parts (MF).

   1.7. **Fragment Offset:** Indicates the position of the fragment data relative to the beginning

of the original data.

1.8. **Time to Live (TTL):** Limits the lifetime of the packet. It decreases by one each time the packet passes through the router. If it reaches zero, the package is discarded to prevent it from being in circulation indefinitely.

1.9. **Protocol:** Specifies the next level protocol used in the data portion of the packet, such as TCP (6) or UDP (17).

1.10. **Header Checksum:** The checksum used to check for errors in the header at each hop.

1.11. **Source address:** The IP address of the device sending the packet.

1.12. **Destination address:** The IP address of the device receiving the packet.

1.13. **Options (variable):** Allows various options, such as security, recording path, and timestamp. This field is optional and not often used.

1.14. **Padding:** Ensures that the header is a multiple of 32 bits. This field is filled with zeros if necessary.

2. **Payload:** which follows the header, this is the actual content or actual data that the packet transmits. [5]

1.4 TCP and UDP Protocol

Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are fundamental transport layer protocols that facilitate data transmission across networks. While both are integral to network communication, they exhibit distinct operational characteristics and are suited for different applications.

1.4.1 TCP

TCP is a connection oriented protocol, ensuring reliable and ordered delivery of data streams over the network. It achieves this through a process known as the three-way handshake, which establishes a robust connection before any data transfer occurs. TCP's error-checking features and acknowledgment system provide a mechanism for the sender to retransmit lost or corrupted packets, there by guaranteeing data integrity. [6]

1.4.2 UDP

On the other hand, UDP is a connectionless protocol, designed for low-latency and loss-toleratingtransmissions. It eschews the overhead of establishing a connection, error-checking, and packetordering, resulting in a protocol that is faster but less reliable than TCP. UDP is particularly well-suited for real-time applications where speed is more critical than reliability, such as video streaming or online gaming. [6]

1.4.3 The differences between TCP and UDP

Feature	**TCP**	**UDP**
Connection	Connection-oriented	Connectionless
State	No need for pre-established connection	Requires pre-established connection
Reliability	Provides reliable data transfer	Does not guarantee packet delivery
Ordering	Maintains order of data packets	Does not ensure ordered delivery
Speed	Slower due to error correction	Faster due to lack of error correction

Table 1.1: Comparison of TCP and UDP

In summary, the choice between TCP and UDP depends on the requirements of the application in question. For tasks that demand high reliability and data integrity, TCP is the preferredprotocol. Conversely, for applications where speed is paramount, and occasional data loss is acceptable, UDP is the more suitable option.

1.5 How a TCP connection works

1.5.1 Establish connection

To open a TCP connection, the system follows a process known as a three-way handshake. This process ensures that both parties are ready to communicate and exchange data. This handshake takes place in 3 steps, which are as follows:

1. **SYN:** The initiating device sends a SYN packet with a serial number to the receiving device to initiate the connection.

2. **SYN-ACK:** The receiving device responds with a SYN-ACK packet indicating receipt of the first packet.
3. **ACK:** The sending device sends an ACK packet in response to a SYN-ACK packet confirming that the sending device is ready to communicate and exchange data. [7]

Below is figure 1.3, an illustration of the three-way handshake process:

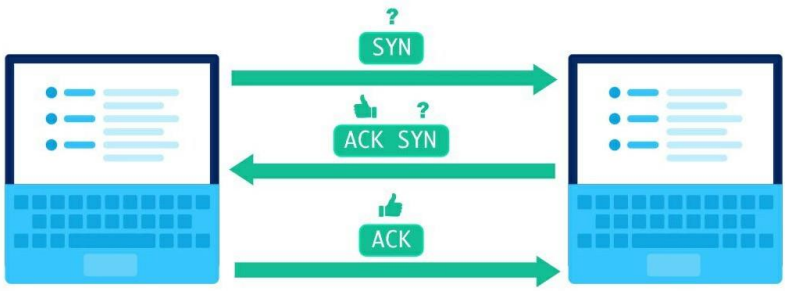


Figure 1.3: TCP Open Connection [7]

Once this handshake is complete, a trusted connection is established, and data exchange can begin with confidence that it will arrive correctly and sequentially.

1.5.2 Send packets of data

When a packet of data is sent over TCP, the recipient must always acknowledge what theyreceived. Figure 1.4 shows the process of sending a packet



Figure 1.4: TCP Send Packets [7]

The first computer sends a packet with data and a sequence number. The second computer acknowledges it by setting the ACK bit and increasing the acknowledgement number by the length of the received data. Those two numbers help the computers to keep track of which data was successfully received, which data was lost, and which data was accidentally sent twice. [7]

1.5.3 Close the connection

To close the TCP connection, the system follows the four-way handshake process, which takesplace in four steps:

1. **FIN:** A client that wants to terminate the connection sends a FIN packet to the server, indicating that the client has no additional data to send.
2. **ACK:** The server receives the FIN packet and responds with an ACK packet indicating that it received a connection termination signal from the client.
3. **FIN:** After the server finishes sending all the remaining data, it sends a FIN packet to the client.
4. **ACK:** The client receives the FIN packet and replies with an ACK packet. [7]

Below is the figure 1.5, which is an illustration of the four-way handshake process to close the connection:

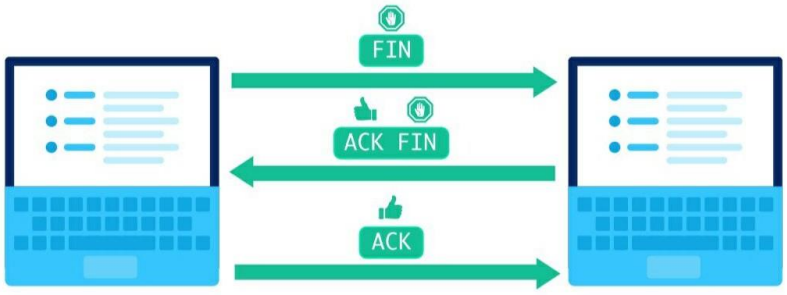


Figure 1.5: TCP Close Connection [7]

After completing these steps, the connection will be closed. This process ensures that both parties have completed their tasks and that the connection is closed without losing any data. [7]

1.5.4 Detecting lost packets

Detecting lost packets in TCP (Transmission Control Protocol) is a crucial part of ensuring reliable data transmission. Here's how TCP detects lost packets:

1. **Retransmission Time-Out (RTO):** TCP uses a timer for each packet sent. If the sender does not receive an acknowledgment (ACK) before the timer expires, it assumes the packet is lost and retransmits it. The RTO is calculated based on the Round-Trip Time

(RTT) of the network. Figure 1.6 shows the process of retransmitting a TCP packet with RTO calculation

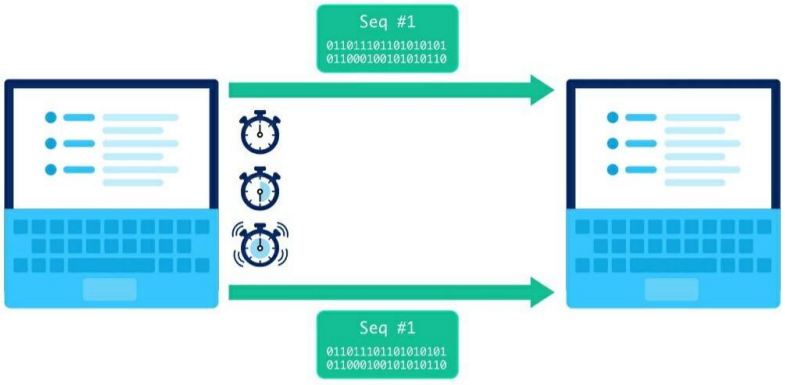


Figure 1.6: TCP Packet Retransmission with RTO Calculation [7]

2. **Duplicate ACKs and Fast Retransmit:** If the sender receives three duplicate ACKs for the same data, it indicates that a packet has been lost. TCP then performs a fast retransmit of the lost packet without waiting for the RTO to expire.
3. **Selective Acknowledgment (SACK):** This is an optional feature where the receiver can inform the sender about all the segments that have been received successfully. This allows the sender to retransmit only the missing segments.
4. **Recent Acknowledgment (RACK):** RACK is a newer algorithm that uses the time of arrival of ACKs to detect lost packets. It's more efficient in networks with reordering or those that use pacing. [8]

These mechanisms help TCP to maintain a reliable connection by ensuring that all packets are eventually delivered in the correct order.

1.6 Denial-of-service (DoS) attacks

Denial-of-service (DoS) attacks exploit internet to target Web services. This type of attack is intended degrade normal services for legitimate users by sending huge unwanted traffic to the victim (machines or networks) to exhaust services and connection capacity or the bandwidth employing a single server. Single source attacks are countered by rejection or blocking. [9]

1.7 Distributed denial of service (DDoS) attack

A distributed denial of service (DDoS) attack is a large-scale coordinated attack, launched using a large number of compromised hosts. At an initial stage, the attacker identifies the vulnerabilities in one or more networks for installation of malware programs in multiple machines to control them from a remote location termed as 'zombies'. The group of zombies usually formed the 'botnet'. At a later stage, the attacker exploits these botnets to send attack packets to the target machine(s). Depending on the intensity of attack packets and larger botnet used to attack, commensurate damage occurs in the victim network. If the attacker can exploit a large number of compromised hosts, a network or a Web server may be disrupted within a short time.[10]

1.8 The Difference between DoS and DDoS Attacks

A denial-of-service (DoS) attack overloads a server with traffic sent from a single source. A distributed denial-of-service (DDoS) which involves multiple computers or machines flooding a targeted resource with traffic. The following are some of the distinctions between Dos and DDoS attacks.

- **Traffic Volume:** Because a DDoS attack employs several distant computers (zombies or bots), it may transmit much higher amounts of traffic from various places at the same time, quickly overloading a server and eluding detection.
- **Manner of Execution:** A DDoS attack employs a command-and-control (C&C) server to coordinate numerous hosts infected with malware (bots), resulting in a botnet. A Dos attack, on the other hand, is often carried out from a single machine using a script or tool.
- **Tracing of Source:** Because a botnet is utilized in a DDoS attack, tracking the true origin is significantly more difficult than tracing the origin of a Dos attack.
- **Ease of Detection:** Because a Dos comes from a single location, it is simple to identify and block the connection. In reality, a competent firewall can achieve this. A DDoS attack, on the other hand, emanates from several locations, hiding its origins.
- **Speed of Attack:** A DDoS attack may be deployed much quicker than a Dos attack that starts in a single site. The increased assault pace makes detection more difficult. [11]

1.9 Types of DDoS attacks

DDoS attacks are of multiple types, which can be divided into two main parts:

1.9.1 Reflection-based attacks

Reflection-based attacks are one of the most common forms of DDoS attacks, where attackers exploit legitimate external mirror servers as part of their attacks Attackers deliberately manipulate the source IP address of packets to match the victim's IP address, in order to send them to reflector servers and confuse the victim with response packets. This leads to the depletion of its resources and hinders the efficiency of its services. In this category, TCP-based attacks consist of MSSQL and SSDP, whereas UDP-based attacks consist of CharGen, NTP, and TFTP.Various attacks, such as DNS, LDAP, NETBIOS, and SNMP, can be executed using either TCPor UDP. [12]

1.9.2 Exploit-based attacks

Exploit-based attacks rely on exploiting vulnerabilities in certain applications or protocols so that erroneous requests are sent on a large scale to target servers, overtaxing resources and disrupting services. These attacks are performed through the use of application layer protocols along with transport layer protocols, such as TCP and UDP. SYN Flood, UDP Flood, and UDP-Lag attack are examples of this type of DDoS attack. The following is a brief explanation of themechanism of each type:

- **UDP flood attack:** A UDP flood attack begins on the remote host by sending a large number of UDP packets. UDP packets are sent to random ports on the target device at a significantly high frequency. As a result, the entire network bandwidth is used, causing system crashes and decreased performance.
- **SYN flood:** SYN flood attacks drain server resources by taking advantage of the TCP three-way handshake. This attack begins by sending a series of SYN packets to the target device until the server experiences a system failure.
- **UDP-Lag attack** A UDP-Lag attack is a specific type of attack that causes a breakdown in communication between a client and a server. This tactic is mainly used in online gaming scenarios, when players aim to slow down or interrupt other players in order to

gain a competitive advantage over them. This attack can be carried out in two ways: either using a hardware key called a delay switch, or using software running on the network and consuming other users' bandwidth. [12]

1.10 Potential Impact of DDoS Attacks

The potential impact of DDoS attacks can be summarized as follows:

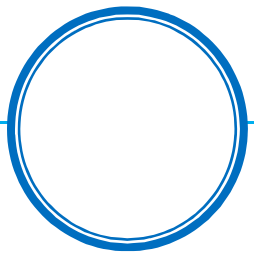
- Complete or partial disruption of services, causing services to fail to respond to legitimate inquiries and loss of visibility over resources.
- Exploiting the attack for criminal purposes and carrying out sabotage acts, which threatens to disrupt systems and destroy data.
- Negative effects on the confidence of investors and customers, which affects the company's reputation and public relations.
- The risk of losing competitiveness and declining the company's position in the market.
- DDoS attacks, as a result of stopping services, cause the loss of associated revenues and market share, thus placing the company in very large losses.
- The scope of their work is expanding, targeting all sectors and institutions, including small and large companies, governments, non-profit institutions, and even individuals.

1.11 Intrusion Detection System IDS

Intrusion Detection System IDS is a software or hardware designed to monitor a network for malicious activities or potential security attacks and report them to a system administrator. The information collected by an IDS is stored inside a security information and event management (SIEM) system. The IDS are only prevention systems that can monitor, detect, and alert about malicious or suspicious activities. They aren't designed to stop the attacks. There are two categories of IDS; signature-based and behavior-based, depending on how they analyze data ona system. Signature-based IDS analyzes the system for misused activities and behavior-based IDS analyzes the behavior of systems for anomalies. Both of them need analyses the packet elements. [13]

1.12 Conclusion

We have seen how DDOS attacks are not just a threat to cybersecurity and only, but rather a challenge to the stability of the entire internet infrastructure. By understanding the basic concepts covered, we lay the foundation for exploring solutions and strategies to counter theseattacks.



CHAPTER II

Literature review

2.1 Introduction

Since 2012, deep learning algorithms have become highly effective tools for addressing a wide range of problems with accuracy and speed, thanks to advancements in computing performance. These algorithms have been successfully applied in various fields such as facial recognition, medical research, and now extend to the field of cybersecurity. In this chapter, we will explore different deep learning algorithms, their classifications, along with presenting evaluation metrics for these algorithms. In addition, we will present and analyze previous research on the useof deep learning algorithms in detecting DDoS attacks.

2.2 Definition of Deep Learning

Deep learning is an innovative and influential development in the field of artificial intelligence. It originated from research conducted in the mid-20th century, specifically from the work of British mathematician Alan Turing, who was interested in creating machines capable of under- standing and acquiring knowledge. Deep learning relies on mathematical techniques that utilize multi-layered artificial neural networks to evaluate data and recognize patterns and features ina manner that mimics the human brain. [14]

Deep learning involves training neural networks that contain more than two hidden layers between the input and output layers. Historically, training these deep networks was a significant challenge due to the issues of exploding and vanishing gradients during the gradient descent process, which is used to train neural networks. However, scientists have managed to overcome these problems thanks to a range of modern advancements such as the use of ReLU activation functions, Long Short-Term Memory (LSTM) units, skip connections in residual neural net- works, and improvements in gradient descent techniques. As a result of these enhancements, itis now possible to effectively train very deep neural networks. The term “deep learning” now refers to the use of a sophisticated set of algorithmic and mathematical tools to train neural networks, regardless of their depth. [15]

The figure 2.1 shows subcategories of artificial intelligence.

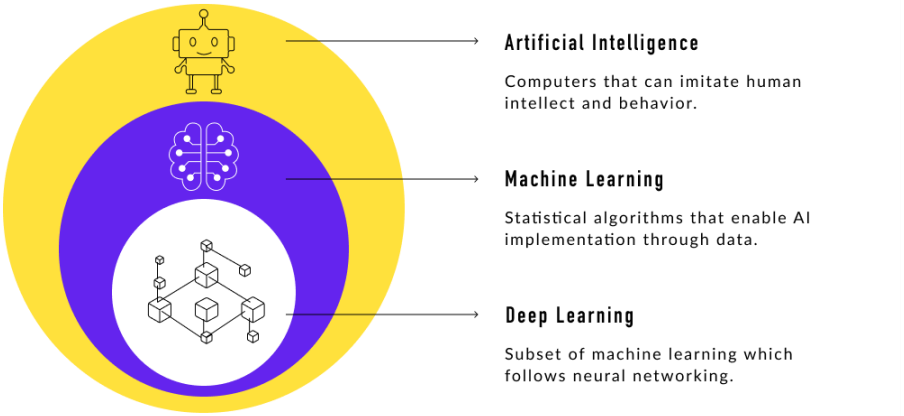


Figure 2.1: Sub-Branches of AI [16]

A deep learning neural network is made up of three main layers:

- **The input layer:** this layer receives the input data.
- **The hidden layer:** this layer performs mathematical calculations.
- **The output layer:** this layer returns the output data.

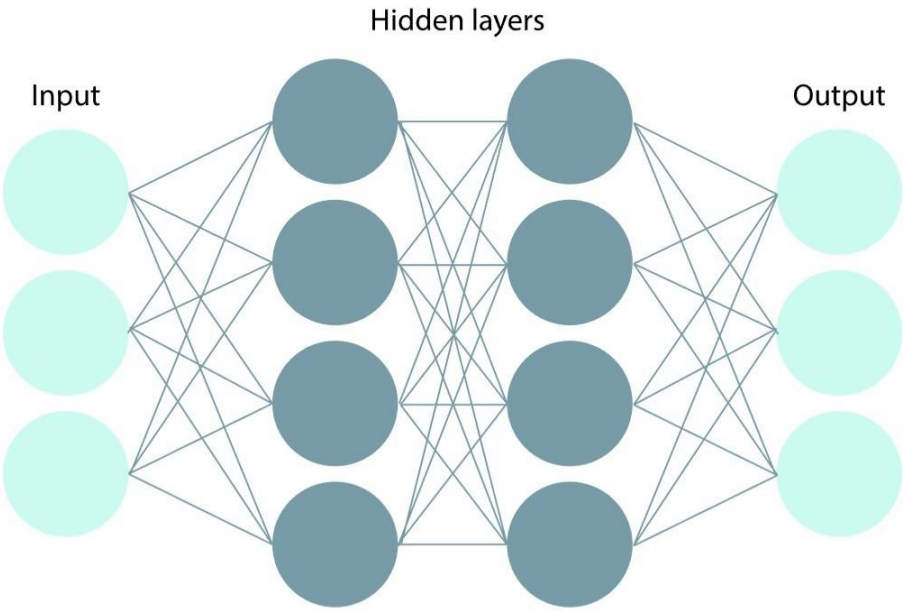


Figure 2.2: Deep Learning Neural Network Architecture [17]

2.3 Taxonomies of the DL approach

The various types of deep neural network techniques, which typically consider several layers of information-processing stages in hierarchical structures to learn. A typical deep neural net- work contains multiple hidden layers including input and output layers. The present taxonomy divides DL techniques broadly into three major categories: (i) deep networks for supervised or discriminative learning; (ii) deep networks for unsupervised or generative learning; and (iii) deep networks for hybrid learning combing both and relevant others [18], as shown in figure 2.3

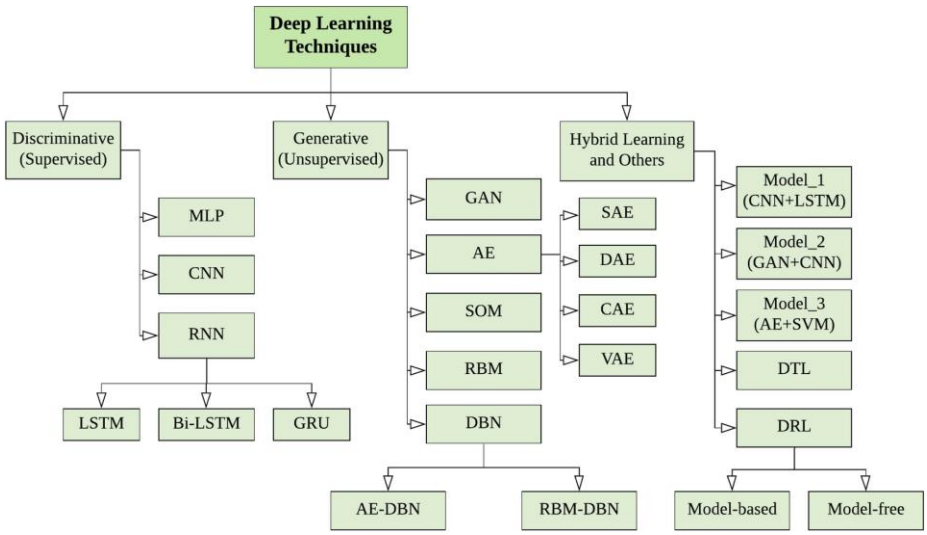


Figure 2.3: A taxonomy of DL techniques [18]

2.4 Popular Deep Learning Algorithms

2.4.1 Artificial Neural Network (ANN)

ANN stands for “Artificial Neural Network”. It is a computational model inspired by the structure and functioning of biological neural networks found in the human brain. ANN is the foundational concept behind many machines learning algorithms, including deep learning. It consists of interconnected processing nodes, also known as neurons or nodes, organized into layers “input layer, multiple hidden layers, and an output layer”. Every node in one layer is connected to every other node in the next layer. The network will be deeper by increasing the number of hidden layers.

In the training phase, ANN initially learns to identify patterns based on inputs given to the

input layer. During this phase, the output of ANN compares with the actual output. The difference between these two knows as an error. The aim is to minimize the error by adjusting the weight and bias of the interconnection which is known as backpropagation. With the process of backpropagation, the difference between the desired output and actual output produces the leasterror. [19]

Figure 2.4 shows the mechanism followed by the ANN model, showing how data is processed and flowed through the different layers to reach the targeted results.

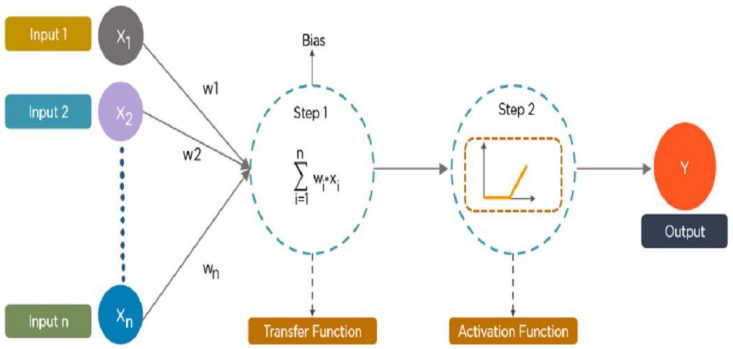


Figure 2.4: Structure of ANN [20]

2.4.2 Deep Neural Network (DNN)

A deep neural network (DNN), also called Deep Nets, is a neural network with a certain level of complexity. They can be thought of as stacked neural networks, or networks made up of several layers, usually two or more, which include inputs, outputs, and at least one hidden layer in between. These networks operate in a feed-forward manner, meaning that data progresses unidirectionally from input to output, preventing feedback loops. Hidden layers are pivotal, processing and transforming data through interconnected neurons, enabling the network to learn and extract meaningful features automatically. Links between layers are one way to go forwardand never touch a node again. DDNs are often used to handle unlabeled and unstructured data.However, training DNNs requires significant computational resources due to their multiparameterization, often requiring powerful GPUs and extensive memory. Techniques such as trainingback-propagation and regularization methods such as dropout and batch normalization enhanceits accuracy and generalizability. [21]

2.4.3 Convolutional neural network (CNN)

A Convolutional Neural Network (CNN) is a class of feedforward neural networks (FFNN) specifically designed to process and analyze high-dimensional data such as visual images. In- spired by the visual processing mechanism of the human brain, CNNs are well-suited for capturing hierarchical patterns and spatial dependencies within images. They utilize convolutional layers that apply small filters moving across the image to extract features and detect fundamental patterns such as edges and shapes. This approach significantly reduces the number of parameters compared to traditional FFNNs, thus decreasing computational complexity and enhancing performance in visual data processing. Although CNNs are commonly used for imageanalysis, they can also be adapted for other domains such as natural language processing, timeseries analysis, and speech recognition, making them a versatile and powerful tool in various applications. [15]

Figure 2.5 highlights the process carried out by the Convolutional Neural Network (CNN) model, showing how visual data is received and analyzed across multiple layers to extract features and accurately classify images.

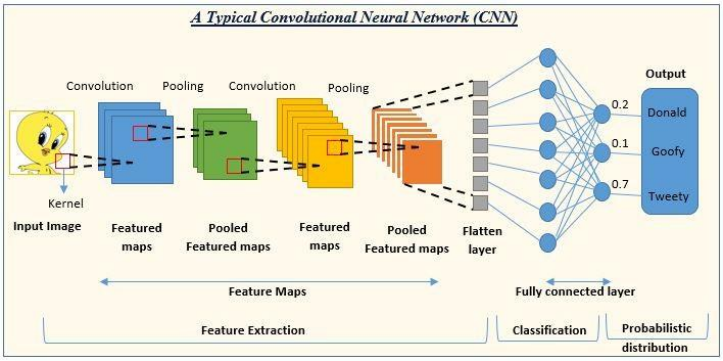


Figure 2.5: Structure of CNN [22]

- **Operation of convolutional neural networks (CNN)**

  CNNs use a series of layers, each of which detects different features of an input im- age. Depending on the complexity of its intended purpose, a CNN can contain dozens, hundreds or even thousands of layers, each building on the outputs of previous layers to recognize detailed patterns.

2.4.4 Recurrent Neural Network (RNN)

A recurrent neural network (RNN) which are formed from feedforward networks, is a type of artificial neural network which uses sequential data or time series data where the output from the previous step is fed as input to the current step. RNNs Are commonly used for such as language translation, natural language processing (NLP), speech recognition, and image captioning. While traditional deep neural networks assume that inputs and outputs are independentof each other, RNN distinguished by its hidden state. The state is also referred to as Memory State since it remembers the previous input to the network. Another distinguishing characteristic of recurrent neural networks is that they share the same weight parameter across each layerof the network. [23]

Figure 2.6 shows the procedures performed by a RNN model, showing how it processes time-series data and learns from long-term dependencies to generate predictions or outcomes.

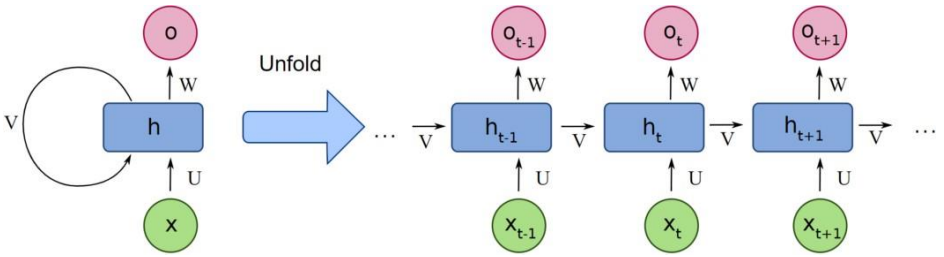


Figure 2.6: Structure of RNN [24]

- **Types Of RNN:**

  There are four types of RNNs based on the number of inputs and outputs in the network. [23]

  1. One to One
  2. One to Many
  3. Many to One
  4. Many to Many

Figure 2.7 shows the variety of types of recurrent neural network (RNN) models.

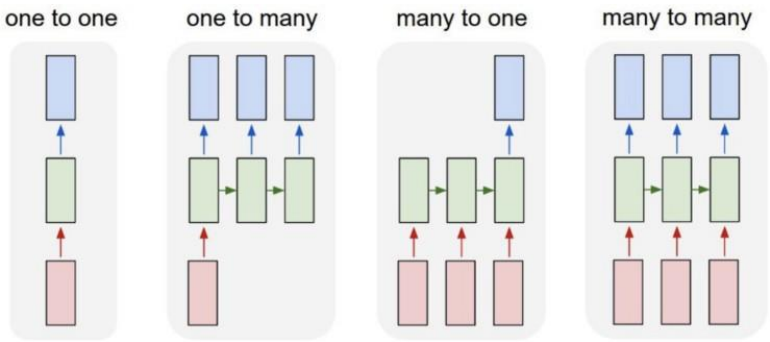


Figure 2.7: Types of RNN [23]

2.4.5 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a recurrent neural network (RNN). It has feedback connections, allowing them to exploit temporal dependencies across sequences of data, making it ideal for sequence prediction tasks. LSTM is designed to handle the issue of vanishing or exploding gradients, which can occur when training traditional RNNs on sequences of data. LSTM networks introduce memory cells, which have the ability to retain information over longsequences. Each memory cell has three main components: an input gate, a forget gate, and an output gate. These gates help regulate the flow of information in and out of the memory cell. LSTMs can also be used in combination with other neural network architectures. [25]

- **LSTM Architecture**

  LSTM resolves the vanishing gradient problem faced by RNN. LSTM works like an RNNcell, but the LSTM network architecture consists of three parts, as shown in the figure 2.8 below:

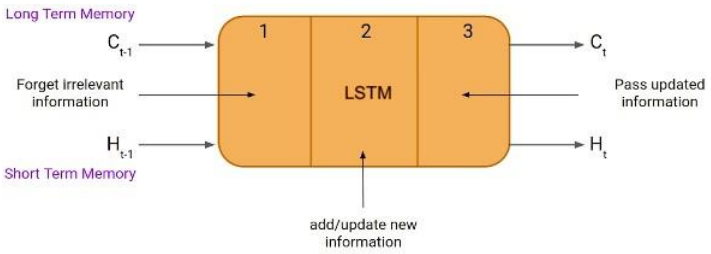


Figure 2.8: Structure of LSTM [26]

The memory cell is controlled by three gates: the input gate, the forget gate, and the output gate. These gates decide what information to add to, remove from, and output from the memory cell. The input gate controls what information is added to the memory cell. The forget gate controls what information is removed from the memory cell. The output gate controls what information is output from the memory cell. By using these gates, LSTM networks can selectively store, update, and retrieve information over long sequences. This makes them particularly effective for tasks that require modeling long-term dependencies, such as speech recognition, language translation, and sentiment analysis. [26]

- **Bidirectional LSTM (Bi-LSTM)**

Bidirectional LSTM (Bi-LSTM/ BLSTM) is recurrent neural network (RNN) that is able to process sequential data in both forward and backward directions. This allows Bi LSTM to learn longer-range dependencies in sequential data than traditional LSTMs, which canonly process sequential data in one direction. Bi-LSTM adds one more LSTM layer, which reverses the direction of information flow. [27]

2.5 A Multilayer Perceptron (MLP)

A multilayer perceptron (MLP) is a type of artificial neural network (ANN) that consists of multiple layers of nodes. The layers include an input layer, one or more hidden layers, and an output layer, each node, apart from the input nodes, has a nonlinear activation function. An MLP uses backpropagation as a supervised learning technique. Since thereare multiple layers of neurons, MLPs are notable for being able to distinguish data that is not linearly separable, making them useful for solving complex problems like image processing. They are typically used for any data that can be flattened into a 1D vector. MLP is widely used for solving problems that require supervised learning as well as re- search into computational neuroscience and parallel distributed processing. Applications include speech recognition, image recognition and machine translation. [28][29]

2.6 A Gated Recurrent Unit (GRU)

GRU is an advancement of the standard RNN i.e., recurrent neural network. It is designedto model sequential data by allowing information to be selectively remembered

or for- gotten over time. GRUs are very similar to Long Short-Term Memory (LSTM). But they have a simpler architecture than LSTM, with fewer parameters, which can make it easier to train and more computationally efficient. The main difference between GRU and LSTM is the way they handle the memory cell state. In LSTM, the memory cell state is maintained separately from the hidden state and it is updated using three gates: the input gate, output gate, and forget gate. In GRU, the memory cell state is replaced with a"candidate activation vector," which is updated using two gates: the reset gate and update gate. The basic idea behind GRU is to use gating mechanisms to selectively update the hidden state of the network at each time step. The gating mechanisms are used to controlthe flow of information in and out of the network. [30]

2.7 A hybrid neural network (HNN)

A hybrid neural network (HNN) is a computational model that integrates various neural network architectures or computational paradigms to leverage their unique strengths and capabilities. By combining different components, HNNs can address complex tasks thatare beyond the reach of individual models.

- **Methods for Constructing HNNs**

  1. **Integration of biological and artificial neural networks:** By combining biological neurons with artificial neurons, HNNs can facilitate research in neuroscience and bioengineering, as well as the development of neuroprosthetics and brain-computer interfaces.

  2. **Combination of connectionist and symbolic computation:** HNNs can integrate the learning and pattern recognition capabilities of connectionist models (such as ANN) with the explicit representation and manipulation of symbols in symbolic computation. This allows HNNs to handle tasks that require both statistical pattern processing and logical reasoning, such as natural language understanding and cognitive modeling.

  3. **Fusion of different types of neural networks:** HNNs may combine multiple types of neural networks, such as feedforward, recurrent, convolutional and spiking networks to exploit their complementary strengths in handling different aspects of data or tasks. This approach enables HNNs to achieve superior performance across a

wide range of applications from image and speech recognition to time-series prediction and control.

In summary, HNNs offer a flexible and powerful framework for building computational models that can tackle diverse problems by blending the capabilities of different neural network architectures and computational paradigms. [31][32]

2.8 Steps before applying the DL model

2.8.1 Data preprocessing

Data preprocessing stands out as a basic step that purifies and improves the data to make it more suitable for training DL algorithms, as it removes impurities, removes noise from the data panel, and addresses missing values. This process enables the model to excel with maximum performance and accuracy.

1. **Data normalization**

   Data normalization is an essential preprocessing step in machine learning and deep learning. It involves adjusting features of data so that they have a mean of zero and a standard deviation of one. This process is essential to train algorithms more efficiently and achievebetter performance.

   One of the simplest ways to normalize data is “Min-Max Scaling”, which is a basic normalization method that rescales feature values to a fixed range of [0, 1]. This technique is particularly useful when dealing with features that operate at different levels, ensuringthat each feature contributes proportionately to the final model.

   The mathematical formula for the Min-Max scale is expressed as follows:

$$X_{\text{norm}} = \frac{(X - X_{\text{min}})}{(X_{\text{max}} - X_{\text{min}})}$$

   where:

   - X represents the original feature value
   - X_{min} is the minimum feature value

- X_{max} is the maximum value of the feature
- X_{norm} is the normalized feature value

By implementing Min-Max Scaling, the transformed features will have a uniform distribution within the range [0, 1], which is essential in reducing the influence of outliers andenhancing the predictive accuracy of the overall model. [14]

2. Data standardization

Often referred to as Z-score, data standardization is one of the pivotal preprocessing techniques in DL. It comes into play when the feature inputs exhibit Gaussian distributions with different means and standard deviations, requiring fit on a uniform scale. This trans-formation normalizes all feature inputs to adhere to a standard normal distribution that has a mean of zero and a variance of unity.

The mathematical formula for Z-score normalization is given by:

$$z = \frac{(x - \mu)}{\sigma}$$

where:

- x is the original feature value
- μ is the average of the feature values
- σ is the standard deviation of the feature values

The result is a dataset where features share a uniform distribution, facilitating fair comparison and efficient integration into different deep learning models. [14]

Figure 2.9 shows the effect that normalization and standardization have on the data set.

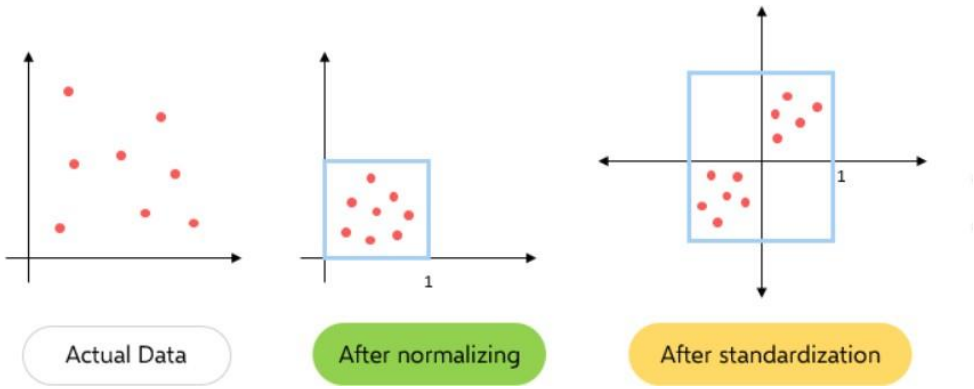


Figure 2.9: Data Processing Stages [33]

2.8.2 Data balance

An imbalanced data set is a set in which we notice a large number difference between the minority class and the majority class. This disparity can lead to poor performance of models trained on this data, especially when they have to apply it to new or unknown data. In order to obtain balanced data, we can use different strategies to modify the data set before training our predictive model, among these strategies are:

1. **Undersampling:** In this technique, samples from the majority class are randomly deleted to balance the classes. This reduces execution time and storage problems by reducing the number of data. However, this can result in missing information that may be important for building the model, so this method is not always ideal.

2. **Oversampling:** Oversampling increases the number of instances of the minority class by a random frequency in order to provide greater representation of the minority class in the sample. Theadvantage of oversampling over undersampling is that this method does not lead to any loss of information. But it increases the likelihood of modification occurring because it reproduces minority class events. [14]

2.8.3 Data splitting

To effectively train and evaluate deep learning models, data is divided into three main sets:

1. **Train set:** It is a collection of data used to train the model and help it learn hidden features

and patterns within the data. During each training session (epoch), the neural network is repeatedly provided with the same training data, allowing the model to continuously learn the data's features. It is crucial that the training set includes a diverse range of inputs to ensure the model is trained on all possible scenarios and can predict any unseen sample data that may appear in the future. This set is used to build the model by applying various model parameter settings, enabling the model to effectively learn patterns and relationships from the data. [34]

2. **Validation set:** It is separate from the training set and is used to monitor the model's performance during training. This combination provides insights that help fine-tune the model's distinct parameters and configurations. Essentially, the validation group acts as a critic, indicating whether the training is progressing correctly. The primary goal of a validation set is to prevent overfitting, which occurs when a model becomes overly specialized in classifying training samples but fails to accurately generalize and classify unseen data. [34]

3. **Test set:** It is an additional blind data set that is not used during the model selection and validation process, and is used to obtain an unbiased and accurate estimate of the overall model performance on unknown samples. Its goal is to provide a good estimate of the model's generalization performance to unknown future data coming from the same distribution. [34]

Here's a Table 2.1 for clarity:

Table 2.1: Data splitting

Data Set	**Function**	**Role in Model Development**	**Importance**
Training	Model learning	Parameter adjustment based on data features	Ensures model readiness for various scenarios
Validation	Performance monitoring	Hyperparameter tuning and model refinement	Prevents overfitting and ensures generalization
Testing	Final evaluation	Unbiased performance assessment	Measures model's real-world applicability

The size of the data partition varies based on the requirements and complexity of the specific model. Some Deep learning models require a larger amount of training data, while others may need a larger test set to ensure accurate evaluation. It is important to adjust the data segmentation according to the needs of the model to ensure its effectiveness and accuracy.

2.9 Performance Metrics

These are precise methods and standards used to evaluate and analyze the efficiency of computer models in data learning tasks. They provide indicators of the model's ability to predict or classify correctly. They assist us in conducting comparisons between several algorithms. They are also essential for guiding the model improvement process and ensuring optimal performance. [35]

2.9.1 The confusion matrix

Confusion matrix helps visualize if the model is confused in discrimination between the two classes, it is a "2×2" matrix, the labels of two rows and two columns are positive and negative to reflect the two class labels. The 4 elements of the matrix represent the 4 metrics which countthe number of correct and incorrect predictions made by the model:

- **True Positive (TP)**: the prediction and the actual value are positive.
- **True Negative (TN)**: the prediction and the actual value are negative.
- **False Positive (FP)**: the prediction is positive while the actual value is negative.
- **False Negative (FN)**: the prediction is negative while the actual value is negative.

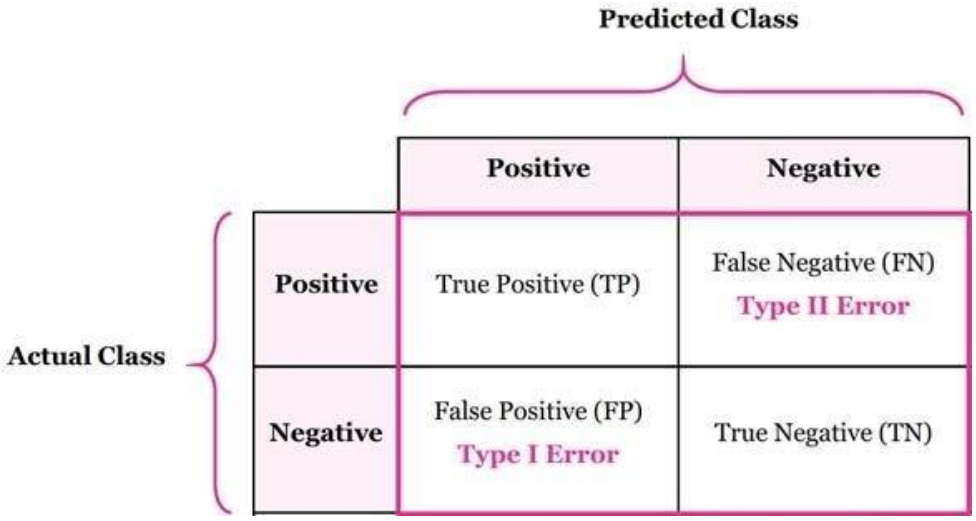


Figure 2.10: The Confusion Matrix [36]

2.9.2 The precision

Classification precision is the total number of correct predictions calculated by the total number of predictions made for a set of data, it allows to answer the following question: What proportion of positive identifications were actually correct?

$$\text{Precision} = \frac{TP}{TP + FP}$$

2.9.3 The Recall

Recall is a metric that quantifies the number of correct positive predictions made from all the positive predictions that could have been made. It allows us to respond to the following question: What proportion of real positive results were identified correctly?

$$\text{Recall} = \frac{TP}{TP + FN}$$

2.9.4 F1-Score

The F1-Score provides a way to combine precision and recall into a single measure to evaluate their correspondence.

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

2.9.5 Accuracy

Accuracy is a metric that generally describes model performance in all classes. It is useful when all classes are of equal importance. It is calculated as the ratio between the number of correct predictions and the total number of predictions.

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

2.10 Related Works

Recently, deep learning and machine learning algorithms have had great success in predicting DDoS attacks. In this section, we will present some research that sheds light on deep learning models and techniques that have been developed to detect these attacks and discuss the factors that influenced the effectiveness of the models used.

Below is a review of some of the leading research works using deep learning to detect DDoS attacks, highlighting those that have provided innovative methodologies in this context.

- Ortet Lopes et al. in [35], They presented an innovative IDS framework for detecting DDoS attacks, known as CyDDoS. This framework features the application of DNN alongside a core feature selection strategy conducted through meticulous analysis by five machine learning classifiers. After testing several feature sets, they found that these ten features provided the best performance: Min Packet Length, Init_Win_bytes_forward, Fwd Packet Length Min, ACK Flag Count, Fwd Packet Length Mean, Avg Fwd Segment Size, Total Backward Packets, Total Length of Fwd Packets, Flow Packets/s, and Flow IAT Min. This work aims to enhance model performance and reduce computationalresource requirements.

  The model's performance was evaluated using the CICDDoS2019 dataset. The results showed the proposed framework's effectiveness in detecting DDoS attacks compared tocontemporary intrusion detection systems.

- Xinlong et al. in [37], They proposed a hybrid deep learning model in a study titled "DDoS Attack Detection by Hybrid Deep Learning Methodologies" that combines HTM and LSTM techniques to detect malicious web traffic in the form of DDoS attacks. This model takes advantage of the capabilities of HTM in recognizing temporal patterns and

the flexibility of LSTM in dealing with long sequences to achieve high detection accuracy. The proposed model was applied to the CICDDoS2019 database. The results obtained showed the superiority of the proposed hybrid approach over current techniques used to enhance the efficiency of cybersecurity systems to detect DDoS attacks because it achieves a deeper understanding of data and temporal relationships.

- Zhou et al. in [38], put forward a collaborative approach to identifying DDOS attacks in a work titled "A Cooperative Detection of DDoS attacks based on CNN-Bi-LSTM in SDN". This approach is based on dividing the detection process into two distinct levels: the data level and the control level. At the data plane, the entropy of destination IP addresses is used, which is a metric that quantifies the diversity and spread of destination IP addresses within network traffic. This is done to detect any possible suspicious actions. At the control level, the CNN-Bi-LSTM model optimization is used to analyze 19 complex traffic features for deep detection. These features include: Src_ip, Dst mac, Dst ip, Packet Rate, PktCount, Flows, ByteCount, Protocol, Duration_sec, Tx kbps, Switch-id, Rx_kbps, TX Bytes, Tot_kbps, RX Bytes, Pktperflow, Port Number, Byteperflow, and Src mac.

  Reports with anomalies are sent to the controller responsible for running the deep detection module when a potential DDoS attack is detected. Experimental data shows that this model performs better than SVC-RF, Bi-LSTM, and RF models in terms of accuracy, detection rate, and false alarm rate. The model also reduced detection time by 13 seconds and reduced CPU usage by 12%. To obtain better performance, the task in future study will need to dynamically change the entropy threshold based on network conditions.

- Bastola et al. in [39], In this study, they proposed a model for detecting distributed denial of service (DDoS) attacks on SDNs using deep learning techniques. Different DDoS detection models such as LSTM, GRU, BLSTM and LSTM-GRU were created using the latest DDoS specific dataset namely CICDDoS2019. The different models were com- pared through cross validation with training and testing split in the ratio of 8:2. The LSTM-GRU hybrid model showed superiority over other models in terms of performancemetrics such as precision, positive precision, recall, specificity, and F-rate.

  The model is implemented on an SDN architecture, considering a standard Carnet topology and a linear topology of different sizes, and using the Python-based Ryu

controller. Traffic, including legitimate traffic and DDoS attacks, is generated and analyzed in real time. After extracting the feature values, they were fed into a detection model on the SDNRyu controller, where the traffic was classified as normal or a DDoS attack.

- E. Ataie et al. in [40], The study revolved around developing and testing a new deep learning model based on multi-layer perceptron (MLP) with the aim of detecting DDoS attacks on SDN networks using deep learning. The researchers worked to design a model capable of understanding complex patterns of data thanks to multiple hidden layers. To improve the model's performance, the researchers used feature selection techniques to identify the most important features, which contributed to increasing detection accuracy and reducing execution time.

  The model was tested on the CICDDoS2019 dataset, and the results showed that the proposed MLP model outperformed traditional algorithms and even deep algorithms in all performance metrics.

- Roopak et al. in [41], They proposed a study that uses a multi-objective optimization method known as non-dominated sorting algorithm (NSGA) was used to identify features in a pre-processed dataset. The study used a model combining CNN and LSTM to classify attacks, using the CICIDS2017 dataset and running experiments using a GPU. Itwas compared to many technologies such as MLP, SVM, RF and Bayes, and the results showed that the proposed model outperforms all these methods. Moreover, training timeis 11 times compared to other learning methods.

- Marcos V.O. Assis et al. in [42] proposed a defense system against DDoS attacks and intrusion attacks in a SDN environment, consisting of detection and mitigation modules. The detection module used deep learning-based GRU algorithm to analyze IP flow logs and detect attacks. The model was tested using CICDDoS2019 and CICIDS2018 datasets, and compared with seven other techniques: DNN, CNN, LSTM, SVM, LR, KNN, and GD. Detection and training times were not calculated, and the analysis was performed in an offline environment. However, the results show that GRU can detect attacks with high accuracy.

- Zhang et al. in [43], They proposed a study on DDoS attack detection in energy IoT, in which they presented a model based on Bi-LSTM neural network. The model uses an

edge detection framework using edge computing to reduce delay, extract network contextual information, and learn the temporal characteristics of attack traffic. Worked on original traffic dataset for energy IoT network. Simulation results showed that the proposed model outperformed traditional models such as RNN and LSTM in terms of accuracy, detection rate, and delay. However, the researchers point out the need to test the model inbroader environments that include energy IoT devices and use advanced feature selection techniques to improve performance.

- Haider et al. in [44], In this research, a framework based on pooling CNN is developed to detect DDoS attacks in SDNs. This framework is able to take advantage of centralized control and programming in SDNs, placing it at the control layer for efficiency and low cost. The CICIDS-2017 dataset was used to evaluate the model, which includes net-work flow attributes and contains benign data and multiple types of attack traffic. The model showed its superiority over traditional models such as RNN and LSTM in terms ofaccuracy, detection rate and time.

In Seeking a deeper understanding of the techniques used to detect DDoS attacks using a deep learning approach, we analyze previous literature and evaluate the factors that contributed to the accuracy of each model. Below Table 2.2 reviews the strengths and weaknesses of the different models used.

Study	**Strengths**	**Weaknesses**
Ortiz Lopez et al. (2021)	- Applying a basic feature selection strategy contributes to improving model performance and reducing the need for computational resources. - The ability of the model to train quickly in just 5 seconds.	Low number of epochs (5) may lead to non-optimal convergence.

Study	**Strengths**	**Weaknesses**
Xinlong et al. (2022)	- Combining two technologies to take advantage of the advantages of each technology. - Using HTM to recognize temporal patterns and LSTM to deal with long sequences.	The lack of detail about model parameters makes it difficult to under- stand how the model works and the impact of those parameters on its performance.
Zhou et al. (2023)	- Using CNN and Bi-LSTM techniques to analyze data spatially and tempo- rally, allowing a deeper understanding of patterns and analyses. - Applying a collaborative approach to analyzing data at multiple levels con- tributes to improving detection efficiency and reducing excessive use of system resources.	- The lack of data about model parameters makes it difficult to under- stand how the model works and the impact of those parameters on its performance. - Dynamically updating the entropy threshold can be a challenge to improve model performance under changing network conditions.
Pastula et al. (2021)	- The combination of LSTM and GRU provides powerful temporal patternrecognition - Using the CICDDoS2019 dataset enhances the model's ability to recognize updated attack variants	The lack of detailed model parameters makes it difficult to accurately evaluate the model and understand how those parameters affect model performance.

Study	**Strengths**	**Weaknesses**
E. Ataie et al. (2022)	- Using MLP allows understanding complex patterns in data - The variety of hidden layer settings and optimization parameters provides opportunities to improve model performance and adapt it to specific ap- plication needs -Use techniques to select the most important features	- The complexity of optimal set- tings of hidden layers and optimization parameters may negatively impact model reproducibility, making it difficult to replicate results and model applications in different environments. - Overfitting must be checked so as not to negatively impact the model's ability to generalize to unknown data.
Roopak et al. (2020)	- Use the NSGA method to select better features - The ability of CNN and LSTM to extract spatial and temporal patterns from data	- The training time of the proposed model is longer, which is a drawback in some applied cases - The NSGA algorithm requires additional computational resources to implement effectively, especially when dealing with large and complex data sets. - Using a GPU to execute experiments can be challenging in environments where these resources are not avail- able.

Study	Strengths	Weaknesses
MarcosV.O. As- sis et al. (2021)	- The GRU model provides enhanced efficiency in sequence learning and is considered effective in recognizingtemporal patterns - The accuracy is higher in CICD- DoS2019, which shows that it's may be more distinct or contain clearer pat- terns that make it easier for the model to recognize attacks.	CICIDS2018 may be more complex or diverse, making it difficult for the model to achieve the required efficiency.
Zhanget al. (2022)	- Applying the Bi-LSTM model provides accurate analysis of temporaldata	- The original traffic dataset for the energy IoT network may be insufficient to represent all possible DDoS attack scenarios in this particular domain. - Overfitting may have occurred due to the use of limited training data, which reduces its ability to generalize to new scenarios outside of the training data. - The complexity of the proposed model makes it difficult to understand and maintain this model effectively in real IoT environments.

Study	Strengths	Weaknesses
Haideret al. (2020)	- Utilizing CNN technology for effective pattern recognition and spatial data processing. - Benefiting from centralized control in SDN - Designing optimal model parameters: where ReLU and Sigmoid activation functions have aided in im- proving the learning and classification process, while the use of 100 epochs has provided sufficient training for the model without overgeneralization.	Evaluating the model using the CICIDS-2017 dataset, the results may not reflect performance in different network environments

Table 2.2: Analysis of Studies

This analysis will help to identify research paths that may enhance the development of detection technologies in the future.

2.11 Discuss the various factors

Upon reviewing the previous literature, we find that there are multiple factors affecting the effectiveness of using deep learning techniques in detecting DDoS attacks, which include:

- **Data Quality:** Accurate and diverse data are essential for training effective models. In- complete or biased data can lead to inaccurate results.
- **Dataset Diversity:** Datasets covering a wide range of attack scenarios help build stronger and more generalizable models.
- **Feature Selection:** The choice of features plays a crucial role in model accuracy. Studies

that carefully selected features achieved better performance compared to those that used all features indiscriminately.

- **Optimization and Configuration:** Optimal selection of model parameters and optimization techniques can significantly improve performance.
- **Computational Complexity:** More complex models may offer higher accuracy but require greater computational resources and may be less efficient in real-time.
- **Overfitting:** Models that overfit to training data may fail to generalize the results to new data.
- **Techniques Used:** Choosing the appropriate technique plays a crucial role in the effectiveness of cyberattack detection, where each model is characterized by distinctive features and advantages. DNNs possess a tremendous ability to acquire and depict complex characteristics. CNNs elucidate precise analysis of spatial data, while LSTM and Bi-LSTM efficiently handle temporal data. GRUs provide enhanced efficiency in sequential learning. As for MLPs, they excel in classifying high-level features. Hybrid models that combine various techniques can offer better performance by leveraging the strength of each model in a particular data pattern. Therefore, choosing the suitable model contributes to enhancing the system's ability to detect attacks effectively, ensuring a rapid response and accurate analysis of cyber threats.
- **Evaluation and Verification:** Comprehensive evaluation using diverse test datasets ensures that the model performs well under different conditions.
- **Updates and Maintenance:** Models need periodic updates to adapt to new attack patterns and changes in the network environment.

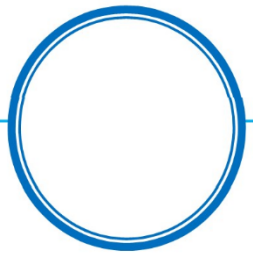
Together, these factors help create improved and more productive approaches to meeting increasing cyber challenges, develop advanced detection techniques for DDoS attacks and ensure the secure and reliable continuity of electronic services.

The review of prior literature demonstrates that the application of deep learning techniques in detecting DDoS attacks extends beyond merely achieving high accuracy. It encompasses the ability to assimilate complex data and adapt to new patterns of attacks. These studies highlight the importance of selecting suitable data, utilizing well-considered features, and implementing strategies to prevent overfitting to achieve outstanding performance. Moreover, it has been

proven that integrating multiple models is a successful approach to increase detection accuracy and reduce false positive alerts.

2.12 Conclusion

In conclusion of this review, we reviewed the different structures of deep learning techniques, as well as the model evaluation procedures in detail. Previous research related to detecting DDoS attacks using deep learning was also reviewed, showing that the use of these techniques greatly enhances the ability to address complex problems with high effectiveness, and makes it a crucial step towards developing more efficient solutions for data analysis and achieving advanced results in various fields, including Detect and counter DDoS attacks effectively and accurately.



CHAPTER *III*

Research Methodology

3.1 Introduction

The aim of this study is to determine the relevant features to detect DDoS attacks and propose a DL model in order to enhance the detection rate in literature. In this chapter, we present the methodology of our research based on the observations analyzed from previous research.

3.2 Methodology

The proposed approach to detecting DDoS attacks is based on the scientific foundations and previous studies mentioned in the previous chapter based on the CICDDoS2019 dataset.

First, we have recovered the KAGGLE CICDDOS2019 data collection, which includes two main files with different types of DDoS attacks. In order to address the problem of huge data and lack of balance, we extracted a balanced and uniform data set in one CSV file, while ensuring that all types of attacks are included to ensure comprehensiveness and accuracy of the analysis.

On this standardized data, we extracted features and then cleaned the lines and normalized the data to ensure its integrity and suitability for analysis. Next, we divided the data set into three distinct subsets: the training set, the validation set, and the test set, ensuring data diversity and comprehensiveness in each set. The test set includes all samples of DDoS type plus portmap type which does not appear in training and validation sets. This will enhance the model's ability and performance in dealing with new types of attacks.

After the partitioning process, we implemented 14 distinct deep learning models for various techniques CNN, MLP, LSTM, GRU, and Bi-LSTM, which also included hybrid models integrating these algorithms. Each model was evaluated using recognized performance indicators:Accuracy, Precision Positivity (Precision), Recall, and F1-Score.

The model evaluation method was an element of the continuous improvement cycle, not the end result. The process of analyzing and evaluating the models in an iterative cycle was necessary to ensure optimal results were achieved.

3.3 Diagram of the Proposed Approach

Figure 3.1 provides a detailed overview of the steps taken in implementing our proposed strategy.

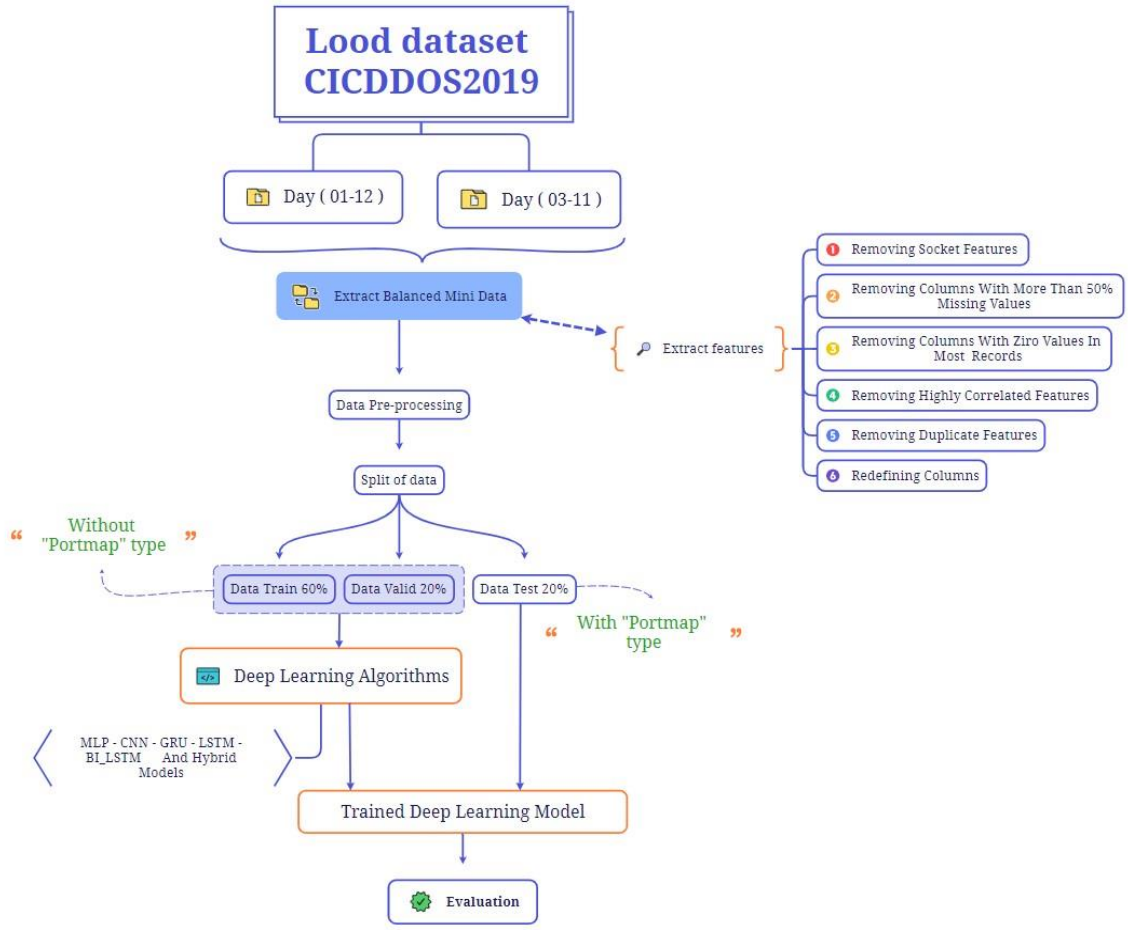


Figure 3.1: The General Flow of The Process

3.4 Dataset preparation

we have implemented the proposed model using the CICDDoS2019 dataset, which was developed and published by the Canadian Cybersecurity Institute in collaboration with the University of New Brunswick. These institutions have provided high-quality datasets that have been used by the research community for several years. This dataset is used to evaluate DDoS attack detection systems. It contains a wide range of fine-grained recordings of network traffic, containing more than 113,000 samples of normal traffic and more than 70 million malicious flows distributed across 13 categories of DDoS attacks, including the most common DDoS attacks.

Data were collected in two different sections: the training section and the test evaluation section. The training set was recorded on 12 January 2019, from 10:35 to 17:15 and contains 12 different DDoS attack types, with each attack type placed in a separate PCAP file. The test data was generated on March 11, 2019, from 09:43 to 17:35 and includes 7 different DDoS attacks. Also, each attack type is recorded in a separate PCAP file.[12]

The distribution of the different types attacks in the dataset is detailed in Figure 3.2.

Attacks				Training set No. of packet flows	Testing set No. of packet flows
DDoS Attacks	Reflection Attacks	TCP based attacks	MSSQL	4522492	5787453
			SSDP	2610611	-
		TCP/UDP based attacks	SNMP	5159870	-
			DNS	5071011	-
			NETBIOS	4093279	3657497
			LDAP	2179930	1915122
			PORTMAP	-	186960
		UDP based attacks	CharGen	-	-
			NTP	1202642	-
			TFTP	20082580	-
	Exploitation Attacks	TCP based attacks	SYN Flood	1582289	4891500
		UDP based attacks	UDP Flood	3134645	3867155
			UDP-Lag	366461	1873
Total Attack Flow				**50063112**	**20364525**

Figure 3.2: The Distribution and statistic of DDoS attacks inside CICDDoS2019 [40]

3.4.1 Extract a Balanced mini-dataset

The electronic data collection is largely unbalanced, which makes harmful traffic flow much higher than the normal flow. We used the technology of random samples for the prevailing records to obtain a small and balanced data set to ensure learning and non -biased classification of input patterns. We got a set of data in one CSV file with the aim of forming a comprehensive data collection that contains a variety of sufficiently for all different types of traffic and regular traffic. We followed the following steps:

1. **Selection of normal data**: We have identified all the regular traffic available in the two files, collected at a rate of about 113 thousand traffic.
2. **Selection of offensive data**: We have identified offensive movements with the same number of regular movements. We have confirmed that the specified attack data includes all types of DDOS attacks available in equal proportions and are chosen in random format.
3. **Fair distribution**: To ensure optimal balance, we ensured that the selected attack data

included all DDoS attack types available in the original dataset, ensuring the comprehensiveness of the small dataset and allowing models to learn from a variety of attacks.

Figure 3.3 is a relative circle that reflects the delicate balance between legitimate and illegitimate traffic categories in our dataset. The circuit also highlights the different types of attacks involved and shows the proportions of each type.

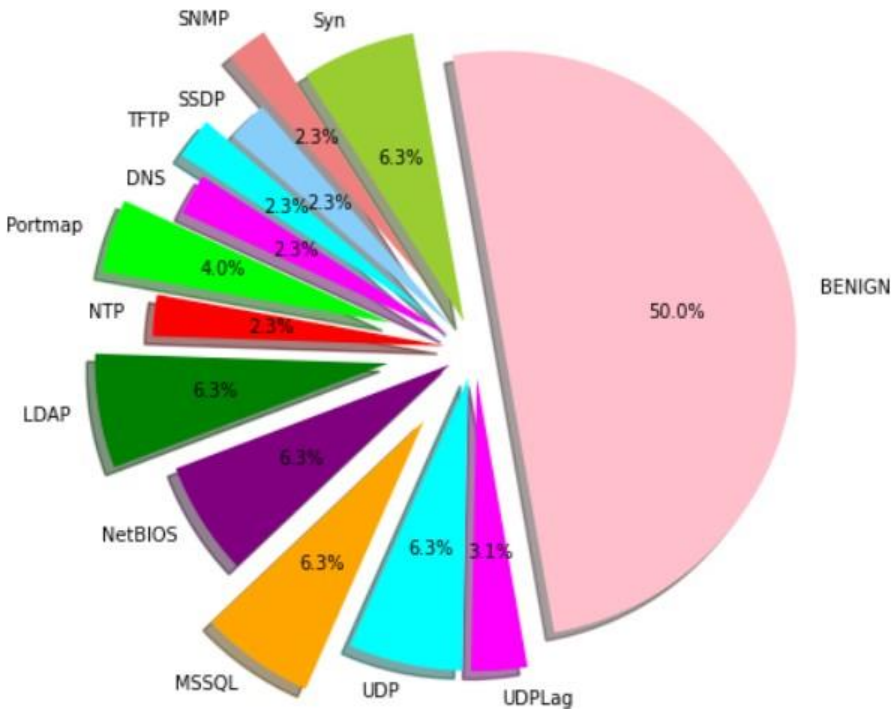


Figure 3.3: Ratio circle for distributing data categories

This balanced and integrated approach to the dataset contributes to building models that are better able to generalize and enhances their accuracy in testing real scenarios.

3.4.2 Feature Selection and Refinement

1. **Remove irrelevant features:** Remove irrelevant features: Features like “Unnamed: 0”, “Flow ID” and “Inbound” don’t add any information related to the issue at hand. Features such as the “timestamp” IP address and port number information are associated with the IP socket or feature ID, DDoS attacks are launched from different IP addresses that look like normal connections, so the IP source cannot be important information in order to detect these attacks. Hence, we only retain the relevant features and remove the unrelated

features manually. The number of features has been reduced to 79. [35] [45]

2. **Removing Columns with More Than 50% Missing Values:** We removed columns that contained more than 50% missing values. This helps to eliminate incomplete data that could affect the model's accuracy. By removing these columns, we maintain a clean dataset free from incomplete data that could lead to bias or confusion during the training process.

3. **Removing Columns with Zero Values in Most Records:** We removed columns that contained zero values in a significant proportion of the records. Columns that are mostly zero do not add any useful information to the model and can lead to unnecessary complexity. By removing these columns, we ensure that the features used in the model carry significant information.

4. **Removing Highly Correlated Features:** After calculating the absolute correlation matrix among all features, we removed features that were highly correlated (more than 95%). Highly correlated features may carry redundant information, which increases the model's complexity without adding extra value. By eliminating these features, we improve the model's ability to learn from the underlying data without repetition.

5. **Removing Duplicate Features:** We removed features that were completely duplicated. Duplicate features do not add any new value to the model but increase the data size and the model's complexity. By removing duplicate features, we maintain a simple and effective dataset, which helps improve the model's performance and reduce the training time.

6. **Redefining Columns:** After implementing all the purification and removal steps, we rearranged the remaining columns. This includes renaming columns if necessary and ensuring that the columns are consistent and organized in a way that facilitates the model's handling. Rearranging helps enhance the readability and understanding of the data, contributing to the ease of the training process and analysis of the results.

After implementing these systematic steps, the number of features was reduced to 39. Using these final features, we can build more accurate and efficient models to analyze data and detect attacks. Table 3.1 shows a brief explanation of each of the remaining features:

No.	Feature	Description
1	Protocol	Protocol used in the communication (e.g., TCP, UDP, ICMP), which determines how data is transferred over the network.
2	Flow Duration	Duration of the flow in microseconds, representing the total time of the connection between the source and destination.
3	Total Fwd Packets	Total number of packets sent from the source to the destination during the flow.
4	Total Backward Packets	Total number of packets sent from the destination to the source during the flow.
5	Total Length of Fwd Packets	Total length of packets sent from the source to the destination, reflecting the amount of data sent.
6	Fwd Packet Length Max	Maximum length of packets sent from the source to the destination.
7	Fwd Packet Length Min	Minimum length of packets sent from the source to the destination.
8	Fwd Packet Length Std	Standard deviation of the length of packets sent from the source, reflecting the variation in packet size.
9	Bwd Packet Length Max	Maximum length of packets sent from the destination to the source.
10	Bwd Packet Length Min	Minimum length of packets sent from the destination to the source.
11	Bwd Packet Length Mean	Mean length of packets sent from the destination to the source.

No.	Feature	Description
12	Flow Bytes/s	Number of bytes transferred in the flow per second, reflecting the data transfer rate.
13	Flow Packets/s	Number of packets transferred in the flow per second, reflecting the packet transfer rate.
14	Flow IAT Mean	Mean inter-arrival time between packets in the flow, which can reflect the pattern of data transfer.
15	Flow IAT Std	Standard deviation of the inter-arrival time between packets in the flow, reflecting the variability in intervals.
16	Flow IAT Max	Maximum inter-arrival time between packets in the flow.
17	Flow IAT Min	Minimum inter-arrival time between packets in the flow.
18	Fwd IAT Mean	Mean inter-arrival time between packets sent from the source to the destination.
19	Bwd IAT Total	Total inter-arrival time between packets sent from the destination to the source.
20	Bwd IAT Mean	Mean inter-arrival time between packets sent from the destination to the source.
21	Bwd IAT Std	Standard deviation of the inter-arrival time between packets sent from the destination to the source.
22	Bwd IAT Max	Maximum inter-arrival time between packets sent from the destination to the source.
23	Bwd IAT Min	Minimum inter-arrival time between packets sent from the destination to the source.
24	Fwd Header Length	Length of the header of packets sent from the source, reflecting control information.

No.	Feature	Description
25	Bwd Header Length	Length of the header of packets sent from the destination,reflecting control information.
26	Source Port	The port number on the source device that It is a part of thetransport layer protocol (TCP or UDP).
27	Bwd Packets/s	Rate of packets sent from the destination to the source persecond.
28	Max Packet Length	Maximum length of a packet in the flow.
29	Packet Length Std	Standard deviation of packet lengths in the flow, reflectingthe variation in packet sizes.
30	Packet Length Variance	Variance in packet lengths in the flow.
31	ACK Flag Count	Number of packets with the ACK flag set, reflecting the ac-knowledgment rate in the connection.
32	URG Flag Count	Number of packets with the URG flag set, reflecting the rateof urgent packets.
33	CWE Flag Count	Number of packets with the CWE flag set, reflecting the rateof packets with control flags.
34	Down/Up Ratio	Ratio of downstream to upstream data flow, which can indicate download and upload patterns.
35	Init_Win_bytes_forward	Initial window size for forward direction, reflecting the amount of data that can be sent without waiting for an ac- knowledgment.

No.	Feature	Description
36	Init_Win_bytes_backwar	Initial window size for backward direction, reflecting the amount of data that can be received without sending an ac-knowledgment.
37	act_data_pkt_fwd	Number of packets with actual data sent from the source.
38	min_seg_size_forward	Minimum segment size sent from the source.
39	Fwd Packets/s	Rate of packets sent from the source to the destination persecond.

Table 3.1: Description of extracted features

3.4.3 Data preprocessing

This preprocessing includes several steps to ensure data quality and improve performance whentraining deep models. Here we review these steps:

1. **Remove rows with more than 5% missing values:** Rows with missing values greater than 5% of the number of columns are removed, ensuring that the remaining data is completer and more reliable.
2. **Data cleaning:** The original data contains many lost values (NAN) and endless values, as well as repeated rows. We have removed all the lost and unlimited values without replacing them with any alternative value, with the aim of maintaining data integrity and not modifying it. As well as deleting repeated rows to ensure that there is no unwanted repetitions.
3. **Normalize input data:** Data features have different value ranges, and using original data without normalization may lead to classification errors and increased training time. So, we normalize the data so that the values are between zero and one. This helps the model learn patterns faster and more effectively.
4. **Encoding tagged data:** We focused on a binary classification of traffic into normal or malicious. We considered all DDoS classes as attack classes, while we classified normal

traffic as benign. We coded the text values of the labels into binary values: 1 for normal movement and 0 for attack movement.

These steps help to improve the performance of deep models when training and reduce the possibility of classification errors.

3.4.4 Data Splitting

We first split the database into two separate groups: a training set and a test set, with only the portmap attack included in the test data to evaluate the models' ability to recognize attacks for which they were not previously trained. This split was performed using the "train test split" function from the Scikit-Learn library, where we designated 20% of the data for the test set and the remaining 80% for the training set. To ensure data stability each time we evaluate the models, we saved the training and test sets in CSV files.

Next, we split the remaining training set again into two groups: a training set and a validation set, using the same "train_test_split" function. In this step, we allocated 20% for the validation set and the remaining 80% for the training set. To ensure an equal distribution of categories in each set, we used the "stratify" parameter.

This process ensures that the test data was not visible during the training and validation processes, ensuring that it did not affect the development of the model subjectively.

After completing the previous stages, the data is ready to apply our proposed models and analyze their performance.

3.5 Description of the implemented Models

After a careful review of the deep learning techniques and a careful analysis of related works in the previous chapter, we implemented 14 innovative models to address current challenges, they vary between standard models and hybrid models. Below we will review some of these models, with a brief explanation of the parameters of each.

3.5.1 MLP Model

The MLP model enables us to extract and analyze complex features from the data, aiding in the detection of suspicious patterns indicative of DDoS attacks. By utilizing MLP, we can

process data efficiently and effectively, providing a rapid and reliable response to security threats. Figure 3.4 explains the composition of our MLP model.

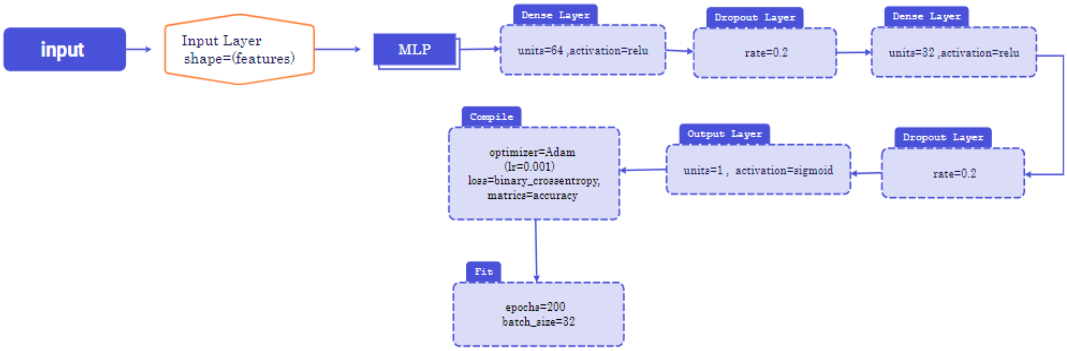


Figure 3.4: MLP model Architecture

3.5.2 GRU Model

We implemented the GRU model as part of our strategy to detect DDoS attacks, to leverage its ability to capture temporal dependencies in the data. This allows us to detect suspicious temporal patterns that may indicate the presence of DDoS attacks. Figure 3.5 explains the composition of our GRU model.

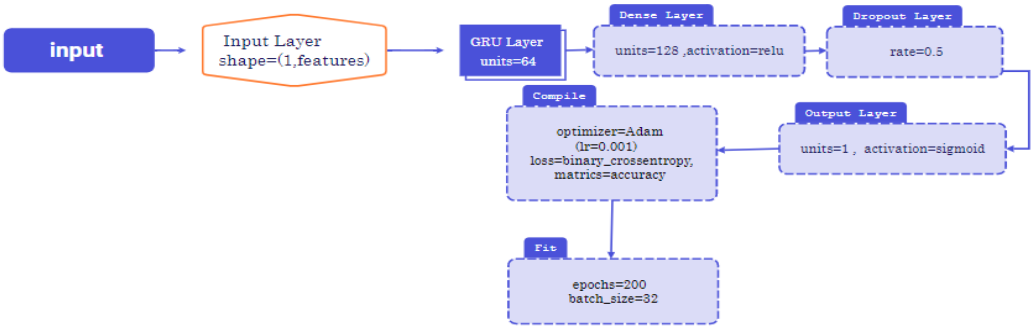


Figure 3.5: GRU model Architecture

3.5.3 LSTM Model

We have implemented an LSTM model to detect DDoS attacks due to its effectiveness in understanding and learning the temporal context of the data. Its ability to maintain memory units over time allows for learning complex temporal patterns and dependencies in the data. With this capability, we can comprehensively and efficiently analyze network traffic flow and identify any abnormal changes indicative of DDoS attacks. Figure 3.6 explains the composition of our LSTM model.

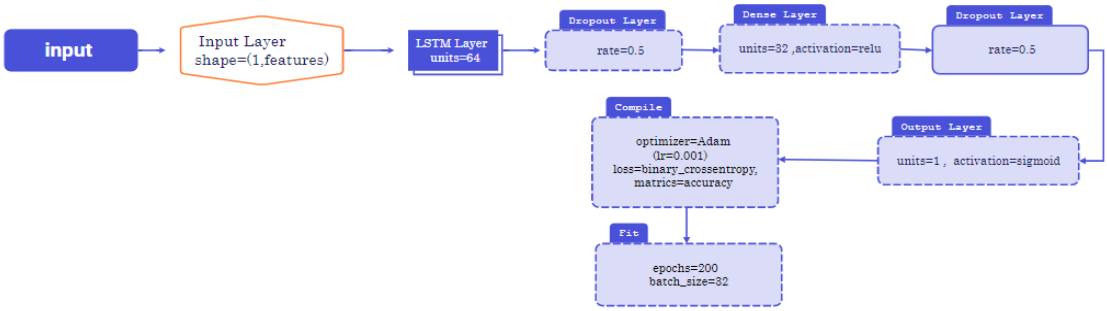


Figure 3.6: LSTM model Architecture

3.5.4 BI-LSTM + GRU Model

We implemented the hybrid model combining GRU and Bi-LSTM to leverage the combined advantages of both techniques. This hybrid model benefits from the efficiency and fast training of GRU in processing temporal data, along with the ability of Bi-LSTM to learn from temporal context in both directions, any of the preceding data (past) and subsequent data (future) within a chronological sequence. This integration enhances the model's accuracy in understanding temporal changes and identifying malicious patterns more effectively. Figure 3.7 explains the composition of our hybrid Bi-LSTM +GRU model.

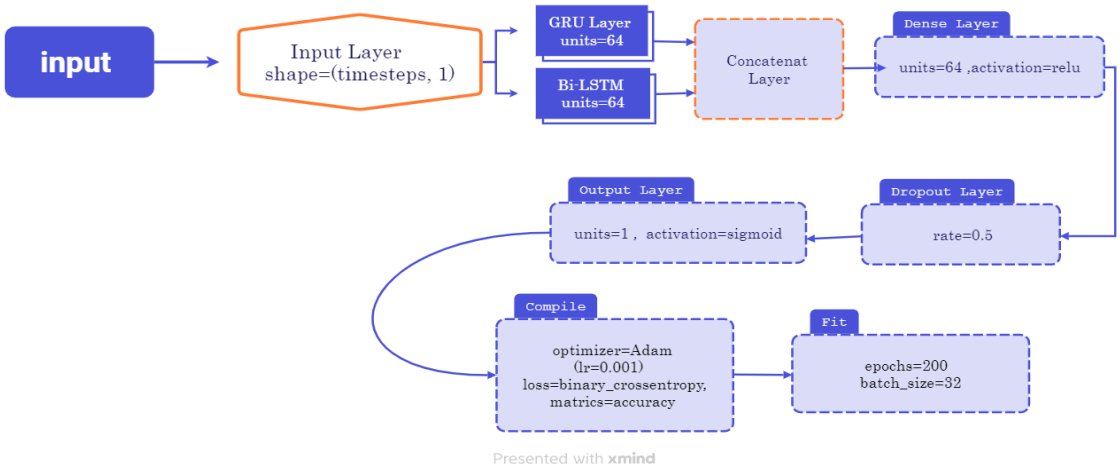


Figure 3.7: BI-LSTM + GRU model Architecture

3.5.5 CNN+GRU Model

This model allows us to take advantage of CNN's ability to extract spatial features from data, such as traffic patterns, and combine it with GRU's ability to process the temporal sequence of these features. This integration enhances the model's ability to understand complex temporal and spatial patterns in the data. Figure 3.8 explains the composition of our hybrid GRU+CNN model.

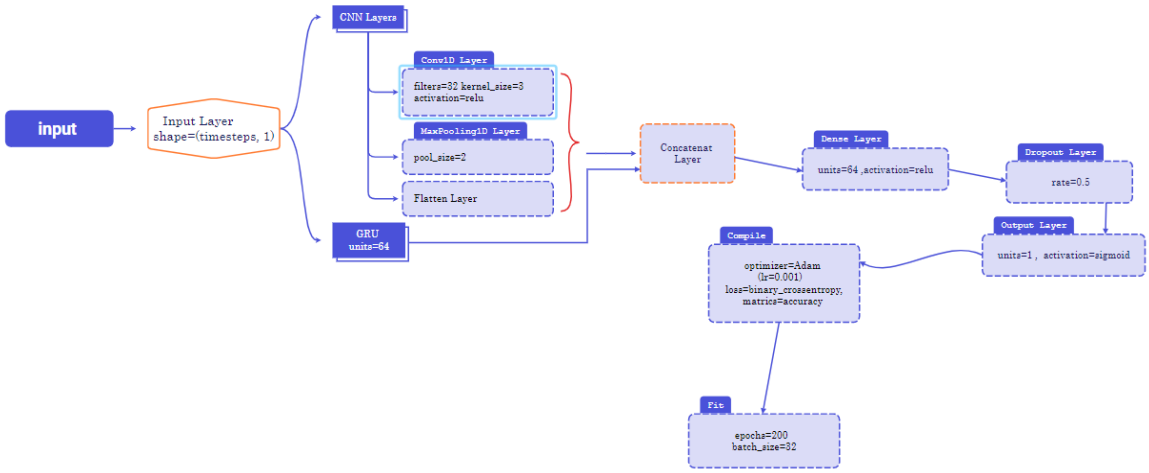


Figure 3.8: CNN+GRU model Architecture

3.5.6 BI-LSTM + CNN Model

This combination allows the power of both CNN and Bi-LSTM to be used to enhance the model's ability to detect complex temporal and spatial patterns in the data. Compared with the model combining CNN and GRU, the current model using Bi-LSTM has the advantage of understanding the temporal context in both directions (past and future). Figure 3.9 explains the composition of our hybrid GRU+CNN model.

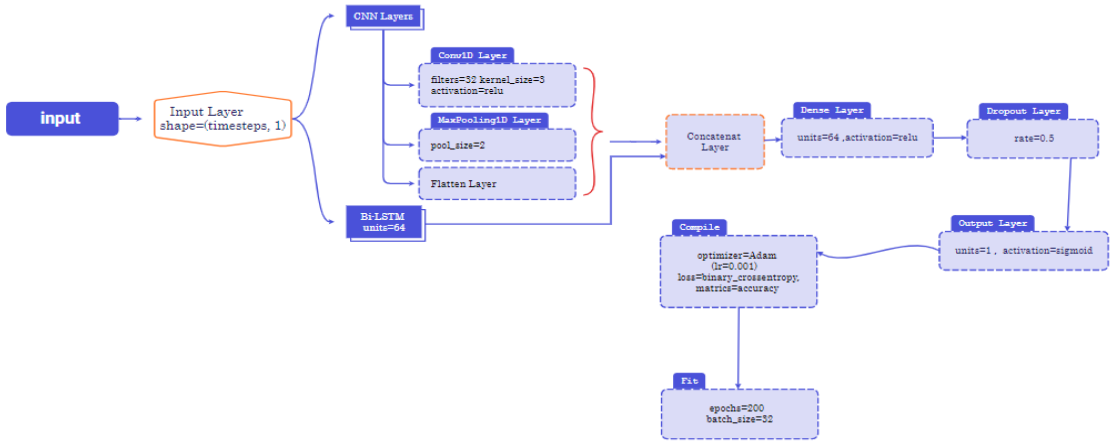


Figure 3.9: BI-LSTM + CNN model Architecture

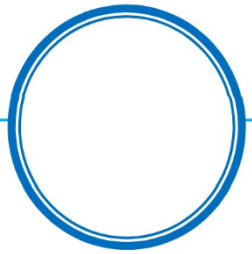
In our pursuit of excellence during the training process, we relied on several strategies to ensure optimal performance. Among these, we utilized **EarlyStopping** technique that halts training automatically when the model's performance on the validation set ceases to improve for a specified number of epochs. This prevents overfitting and reduces unnecessary training time.

Additionally, we employed **Model Checkpoint**, which allow us to save the model's state at specific intervals during training. This enables us to resume training from the last saved state in case of any interruptions, ensuring that progress is not lost and facilitating continuous model improvement.

By adopting these methods, we were able to enhance the efficiency of our training and achieve high performance for our proposed models.

3.6 Conclusion

The models proposed in this chapter represent part of our efforts to improve the efficiency of detecting DDoS attacks using DL. In the next chapter, we will continue evaluating the performance of these models and discuss the results in more detail.



CHAPTER IV

Evaluation and Tests

4.1 Introduction

After an in-depth research journey and meticulous experimentation throughout Chapter Three, where we showcased our innovative deep learning-based models to tackle the challenges of Distributed Denial of Service (DDoS) attacks, in this chapter, we will review the development environment used to build our models and present the results we have achieved. We will focus on the accuracy and efficiency demonstrated by the models. Additionally, we will compare ourmodels with each other and with relevant previous works.

4.2 Tools and environment

4.2.1 Material used

Our work was executed using an **ASUS TUF Gaming F15** laptop. The device comes with a random-access memory capacity of 8 gigabytes to deal with applications with high efficiency, and an Intel(R) Core (TM) i5-10300H processor that ensures super-fast response speeds of up to 2.50 GHz. Additionally, the device operates on a 64-bit version of Windows 11, which providesan easy and reliable user environment.

4.2.2 Programming language

1. **Python**

   Python is a widely used, interpreted, object-oriented, high-level programming language with dynamic semantics. It was developed by Guido Van Rossum in the early nineties atthe University of Amsterdam in the Netherlands. Its name was inspired by the popular TV series Montu Python's Flying Circus. It debuted in February 1991 with its first release, Python 0.9.0.

   Python is known for its easy syntax and rich libraries. This makes it an easy language to learn and easy to use in a variety of applications (console applications, desktop applications, web applications, mobile applications, in addition to the areas of machine learningand deep learning). [14][46]

2. **Features of Python**

   - **Open source:** Python is an open-source software, which means that its source code

is publicly available and can be used by anybody without any fee.

- **Easy to learn:** Python has a gentle learning curve and is designed to be readable (syntax closer to English) and written quickly (shorter code than C, C++ and Java). Making it accessible to beginners while remaining powerful for experienced developers.
- **High level language:** which is closer to human language, refers to a more advanced level of programming notion compared to machine language, such as assembly languages. Python, along with C, C++, Perl, and Java, is classified as a high-level programming language that incorporates low-level optimization techniques.
- **Portability:** High-level languages are cross-platform, meaning they are capable of running on all major hardware and software platforms. Python can be used on Linux, Windows, MacOs, Android, IOS.
- **Object-oriented:** Python supports object-oriented programming, which allows developers to create applications with reusable components and modules. Such as classes, inheritance, objects and overloading.
- **Interactive:** Python has an interactive mode that allows developers to test pieces of code quickly and efficiently. This is useful for mathematical programming.
- **Interpreted:** Python is an interpreted language, meaning they take source code as input and immediately compile each statement into portable byte-code before executing it. There is no requirement for compiling or linking.
- **Extendable:** Python is sometimes called a “glue” language, indicating that it can function in a multilingual setting. It is simple to add a new built-in function or modules written in C/C++/Java code to the Python interpreter.
- **Libraries:** Web services, databases, networking, graphical user interfaces, numerical packages, and three-dimensional graphics, among others.
- **Supports:** Assistance provided by the online Python community. [47]

4.2.3 Modules used

The Python language is known for its simplicity in syntax and rich libraries that make it easy to learn and use in different fields. To implement our experiment, we used the following modules:[48]

1. **Numpy:** It is the basic package for scientific computing in Python. In general, scientific computing, linear algebra is used to reshape vectors into matrices, normalize data, randomly partition data, and perform other mathematical operations.
2. **Pandas:** It is an open-source library for the Python programming language that allows data manipulation and analysis. In particular, data structures, digital signage processing, and time series are proposed. It allows us to analyses, read and manipulate training games.
3. **Tensorflow:** It is an open-source framework developed by researchers at Google to power machine learning, deep learning, and facilitate analytics. Provides the basic abstractions and elements necessary to develop machine learning solutions with high iteration speed.
4. **Keras:** It is one of the leading high-level neural network APIs written in Python. It operates on the basis of the automatic training platform "TensorFlow". Provides powerful and concise building blocks for building deep training networks.
5. **Scikit-learn:** Also known as **sklearn**, is a powerful Python library used for machine learning tasks such as classification, regression, clustering, and dimensionality reduction. It provides various tools for model evaluation and performance metrics (confusion matrix as well as calculating precision, recall and F1 score).
6. **Seaborn:** It is a statistical graphic photography library in the programming language Python, built on the Matplotlib Library. Seaborn provides a high-level interface to create graphics such as bar charts, pie charts, etc.
7. **Matplotlib:** It is a 2D graphics package for Python, designed to track and visualize data in the form of graphs. It is a set of command style functions that make matplotlib work like MATLAB. e.g., creates a figure, creates a plotting area in a figure… etc.

4.2.4 Development environment

Google Colaboratory

The work was carried out with the use of the Google Colaboratory, also known as Google Colab, is a component of the Google toolbox and was constructed on Jupyter Notebooks. It's an interactive cloud service that gives you free access to computational resources like GPUs and TPUs and lets you build and run Python programs straight from your browser without any prior setup. Colab is popular in domains like data science and machine learning because it facilitates sharing and teamwork. [49][50]

This tool provides 12 hours per day for the following tools:

- **RAM:** 12.7 GB available.
- **Disk:** 107.7 GB available.

4.3 Evaluation and discussion

Following the implementation of the models presented in the previous chapter, we now present the results achieved for each:

4.3.1 MLP Model

Classification Report:

	precision	recall	f1-score	support
0	0.9986	0.9898	0.9942	13770
1	0.9913	0.9988	0.9951	16132
accuracy			0.9946	29902
macro avg	0.9950	0.9943	0.9946	29902
weighted avg	0.9947	0.9946	0.9946	29902

Figure 4.1: Report MLP

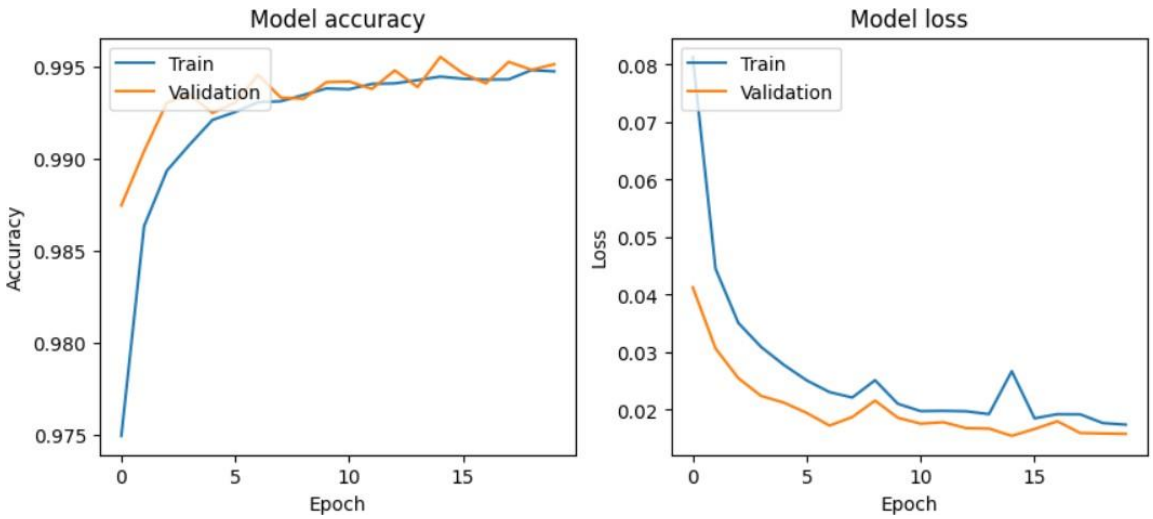


Figure 4.2: MLP model Accuracy/Loss curve in terms of Epochs

4.3.2 GRU Model

Classification Report:	precision	recall	f1-score	support
0	0.9979	0.9946	0.9962	13770
1	0.9954	0.9982	0.9968	16132
accuracy			0.9965	29902
macro avg	0.9966	0.9964	0.9965	29902
weighted avg	0.9965	0.9965	0.9965	29902

Figure 4.3: Report GRU

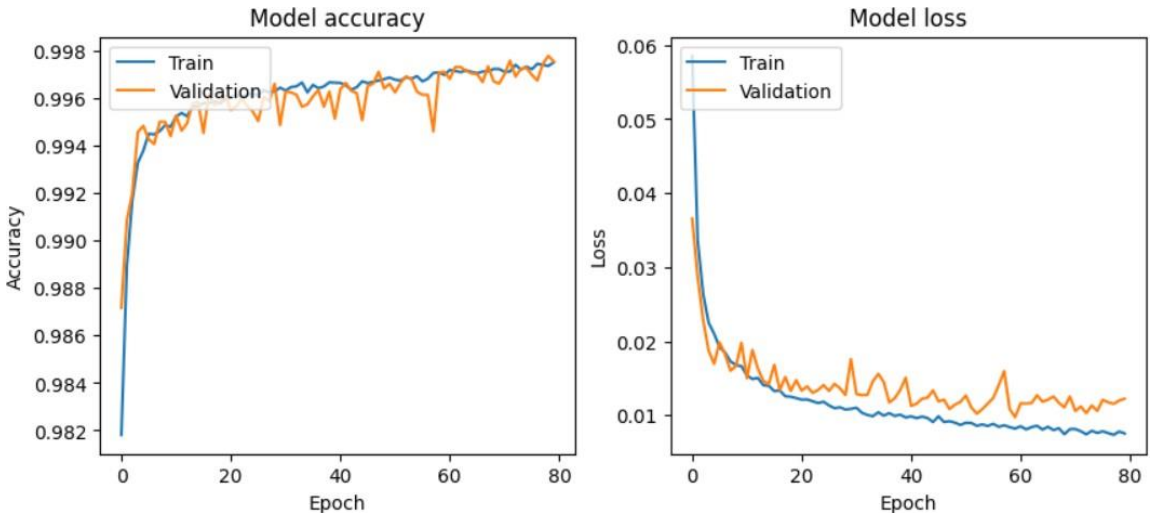


Figure 4.4: GRU model Accuracy/Loss curve in terms of Epochs

4.3.3 LSTM Model

Classification Report:

	precision	recall	f1-score	support
0	0.9980	0.9920	0.9950	13770
1	0.9932	0.9983	0.9958	16132
accuracy			0.9954	29902
macro avg	0.9956	0.9952	0.9954	29902
weighted avg	0.9954	0.9954	0.9954	29902

Figure 4.5: Report LSTM

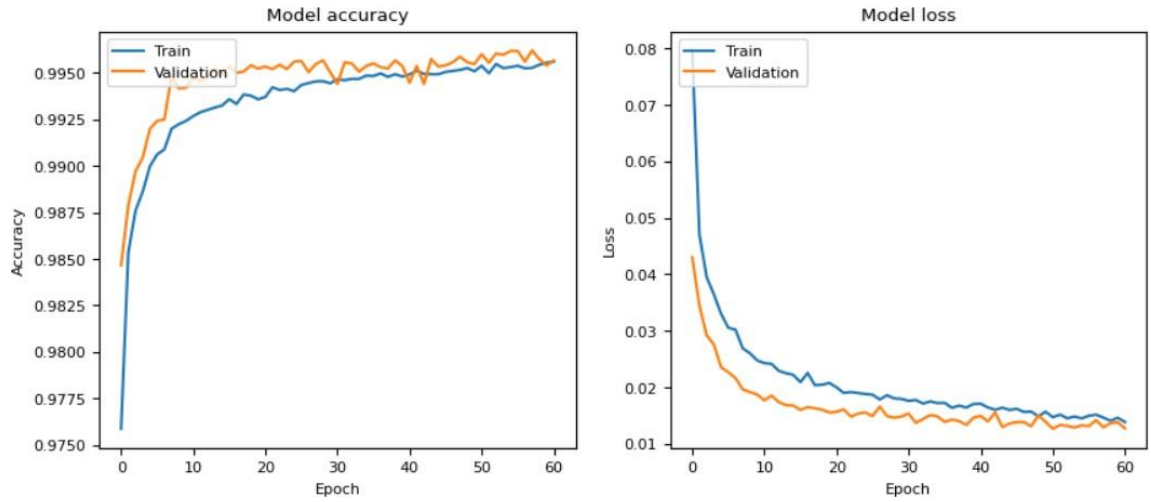


Figure 4.6: LSTM model Accuracy/Loss curve in terms of Epochs

4.3.4 BI-LSTM +GRU Model

Classification Report:	precision	recall	f1-score	support
0	0.9991	0.9967	0.9979	13770
1	0.9972	0.9993	0.9982	16132
accuracy			0.9981	29902
macro avg	0.9981	0.9980	0.9980	29902
weighted avg	0.9981	0.9981	0.9981	29902

Figure 4.7: Report BI-LSTM +GRU

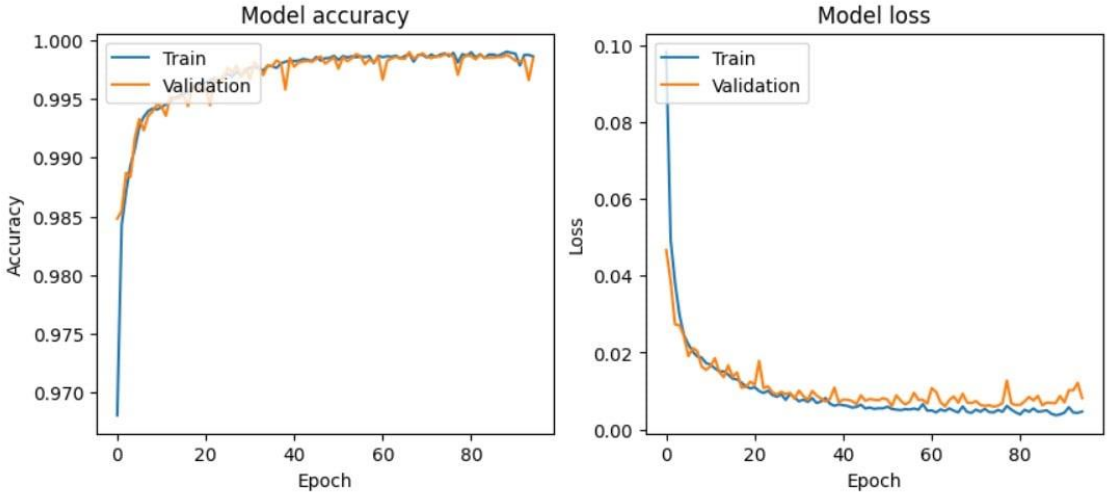


Figure 4.8: BI-LSTM +GRU model Accuracy/Loss curve in terms of Epochs

4.3.5 CNN+GRU Model

Classification Report:

	precision	recall	f1-score	support
0	0.9992	0.9959	0.9976	13770
1	0.9965	0.9993	0.9979	16132
accuracy			0.9978	29902
macro avg	0.9979	0.9976	0.9977	29902
weighted avg	0.9978	0.9978	0.9978	29902

Figure 4.9: Report CNN+GRU

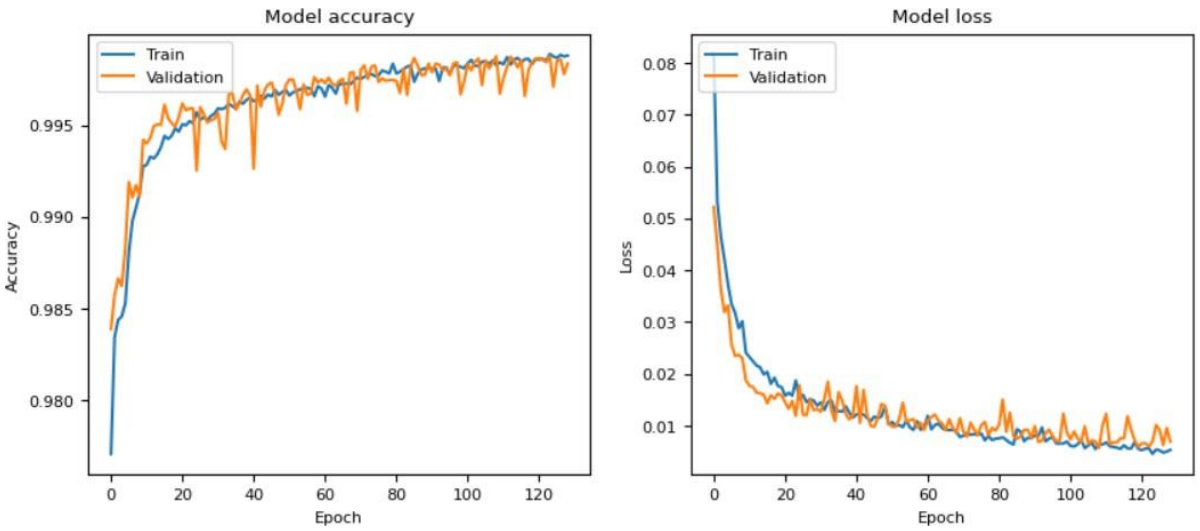


Figure 4.10: CNN+GRU model Accuracy/Loss curve in terms of Epochs

4.3.6 BI-LSTM +CNN Model

Classification Report:

	precision	recall	f1-score	support
0	0.9992	0.9959	0.9975	13770
1	0.9965	0.9993	0.9979	16132
accuracy			0.9977	29902
macro avg	0.9978	0.9976	0.9977	29902
weighted avg	0.9977	0.9977	0.9977	29902

Figure 4.11: Report BI-LSTM +CNN

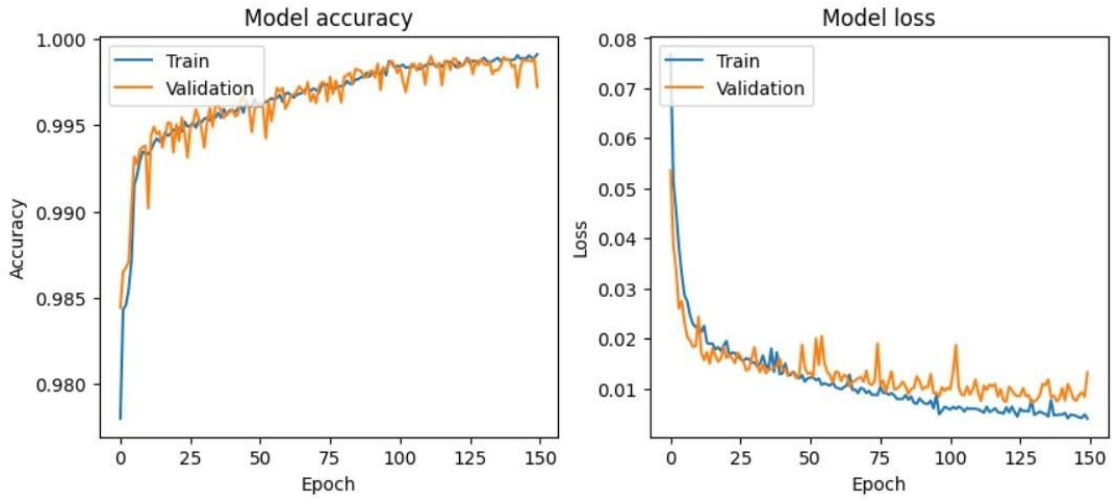


Figure 4.12: BI-LSTM +CNN model Accuracy/Loss curve in terms of Epochs

The models presented in detail so far are a selection from our comprehensive model set. Resultsfor the remaining models will be presented in Table 4.1.

4.4 Comparison of the proposed models

To facilitate comparison, Table 4.1 presents the results obtained from applying the models, including the accuracy, recall, and F1 values for each model.

Model	Accuracy	Precision	Recall	F1-Score
CNN	99.58%	99.58%	99.57%	99.57%
LSTM	99.54%	99.56%	99.52%	99.54%
MLP	99.46%	99.50%	99.43%	99.46%
BI-LSTM	99.65%	99.65%	99.63%	99.64%
GRU	99.65%	99.66%	99.64%	99.65%
LSTM +CNN	99.68%	99.68%	99.67%	99.67%
CNN+MLP	99.56%	99.58%	99.53%	99.56%
BI-LSTM +CNN	99.77%	99.78%	99.76%	99.77%
CNN+GRU	99.78%	99.79%	99.76%	99.77%
LSTM +MLP	99.67%	99.68%	99.66%	99.67%
BI-LSTM +MLP	99.66%	99.67%	99.64%	99.65%
MLP+GRU	99.56%	99.59%	99.53%	99.56%
LSTM +GRU	99.59%	99.60%	99.57%	99.58%
BI-LSTM +GRU	**99.81%**	**99.81%**	**99.80%**	**99.80%**

Table 4.1: The results of testing the proposed models

4.5 Discussion

The results presented in the table above highlight the exceptional performance of the different models used in detecting DDoS attacks. The Bi-LSTM +GRU model emerged as the most effective, achieving an exceptional accuracy of 99.81%, confirming its high ability to recognize complex temporal patterns in data and providing a more comprehensive

representation of attack patterns. The slight advantage in performance could be due to GRU's efficiency in learning from the data, which may have contributed to its ability to generalize well to unseen attack types in the test set.

The BI-LSTM +CNN model, with a high accuracy of 99.77%, demonstrates the power of combining convolutional neural networks and bidirectional long short-term memory networks to capture complex patterns in data.

Independent models such as CNN, LSTM, MLP, and GRU also show a high accuracy of more than 99.5%, which is commendable. This reflects their individual strength in learning from the dataset, with CNN superior at extracting spatial features and LSTM and GRU at capturing temporal dependencies.

In addition, the mixed models CNN+MLP, LSTM+MLP, BI-LSTM+MLP, and MLP+GRU provide outstanding performance, with accuracy ranging between 99.56% and 99.67%. These results indicate that combining MLP, a powerful deep learning model used for classification and prediction, with LSTM, BI-LSTM, and GRU, which excel at handling time sequences, improves the system's ability to detect DDoS attacks with high accuracy.

The hybrid LSTM+CNN model outperforms the standalone CNN and LSTM models, reinforcing the idea that combining different types of neural networks can lead to better performanceby taking advantage of their respective strengths, even though they are the models that achievethe highest accuracy, they require larger computational resources, which may be a limiting factor in real-world applications.

The chart below is a bar chart showing the accuracy of the models that have been suggested:

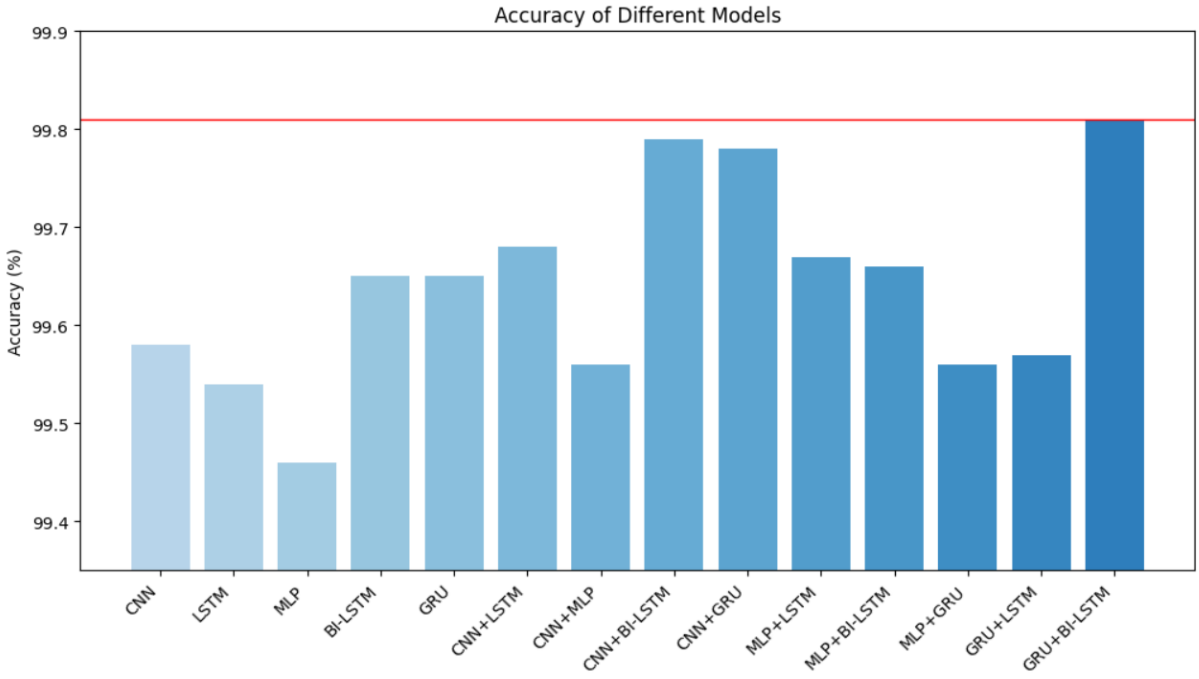


Figure 4.13: Bar chart of the accuracy of the models used in the study

The analyzed data indicates that models incorporating **GRU** or **Bi-LSTM** techniques, either individually or by combining them with other models, have resulted in high accuracy in detecting DDoS attacks. These results enhance the understanding that **Bi-LSTM** and **GRU** technologies possess exceptional capabilities in capturing the temporal and sequential characteristics of data. When combined, they achieve optimal accuracy, suggesting that the integration of **Bi-LSTM** and **GRU** can provide a robust model for efficiently detecting attacks. This discovery is an important step toward developing more precise and effective security systems and opens the door for further research to explore how these techniques can be enhanced and applied in various scenarios.

4.6 Comparison of our results with related works

Figure 4.14 provides a comparison between the accuracy achieved by the models of previous studies and our developed models, showing the differences in performance for each model separately, in order to highlight the significant improvements we have achieved.

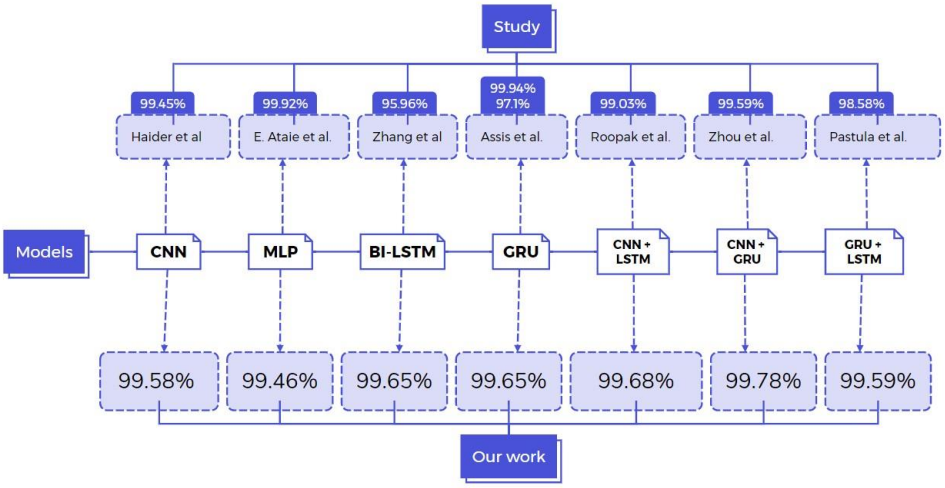


Figure 4.14: Comparing our models with those of previous studies

Table 4.3 presents a demonstration of our model achieving the highest (BI-LSTM+GRU) accuracy compared to the accuracy achieved in previous research, illustrating the progress we have made in DDoS attack detection.

Study	Year	Study Focus	Approach Used	Dataset Used	NO Features	Accuracy	Precision
Ortiz Lopez et al.	2021	Feature Extraction	DNN	CICDDoS2019	10	99.62%	99.65%
Xinlong et al.	2022	Attack Detection	HTM and LSTM	CICDDoS2019	-	97.74%	97.20%
Zhou et al.	2023	Feature Extraction, Attack Detection	Bi-LSTM -CNN	SDN-dataset	19	99.59%	-
Pastula et al.	2021	Attack Detection	LSTM-GRU	CICDDoS2019	-	98.58%	-
E. Ataie et al.	2022	Feature Extraction, Attack Detection	MLP	CICDDoS2019	-	99.92%	99.92%
Roopak et al.	2020	Attack Detection	LSTM and CNN	CICIDS2017	-	99.03%	99.26%
Assis et al.	2021	Attack Detection	GRU	CICDDoS2019 and CICIDS2018	-	99.94% 97.1%	-
Zhang et al.	2022	Feature Extraction, Attack Detection	Bi-LSTM	Energy IoT network	-	95.96%	-
Haider et al.	2020	Attack Detection	CNN	CICIDS-2017	-	99.45%	99.57%
Our methodology with 10 features (Ortiz Lopez et al)	2024	Test Feature, Attack Detection	BI-LSTM +GRU	CICDDoS2019	10	98.00%	98.89%
Our Study	**2024**	**Feature Extraction, Attack Detection**	**BI-LSTM +GRU**	**CICDDoS2019**	**39**	**99.81%**	**99.81%**

Table 4.3: Compare the results of our best model with related works

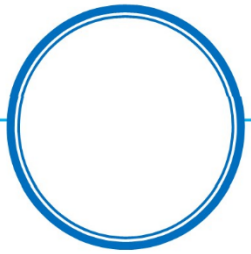
Table 4.3 provides a thorough documentation of the superiority of our model in detecting DDoS attacks, compared to most accuracy reported in previous academic research. This progress is evidence of the sober scientific approach and meticulous methodology we have followed.

By applying the features of (Ortiz Lopez et al) study to our model, we note that the accuracy achieved using the 39 features we extracted shows significant improvements compared to the features used in this study. This reinforces the importance of the features we have chosen and emphasizes the critical role they play in improving the model's ability to accurately detect attacks.

The experimental context in which our model was developed, in terms of data, features and parameters, makes direct comparison with other studies infeasible. However, the results achieved give a strong indication of the qualitative contributions of our study in enhancing the capabilities of detecting attacks, both known and new, and herald broad possibilities for future applications that enhance the stability and security of electronic infrastructures.

4.7 Conclusion

At the end of this chapter, we find ourselves having presented our proposed models and evaluated their performance in the development environment world. The results have been showcased with precision and reliability, accompanied by an analytical examination of the performance. We hope that the understanding of these results will contribute to enhancing future strategies and directing forthcoming research in this field.



General Conclusion

Distributed denial-of-service attacks (DDoS) are one of the most urgent cybersecurity threats in the midst of the digital revolution. These attacks necessitate a sophisticated technological response that goes beyond traditional detection and prevention methods because to their complexity and diversity. The existing techniques have proven inadequate against attackers using new strategies, necessitating the creation of precise and adaptable deep learning models.

This project presents attempts to bridge current gaps and provide advanced technological solutions in this field. By implementing several models such as CNN, LSTM, GRU, Bi-LSTM, and MLP as well as applying hybrid models that combine them, we aim to build systems that can accurately identify DDoS attacks and can adjust to novel attack patterns. Our contributions are aggregated in several stages, from processing and analyzing the CICDDoS2019 dataset, through extracting features and selecting the most effective models, to evaluating and continuously improving their performance.

The BI-LSTM +GRU hybrid model performed exceptionally well out of all the models that were created, exceeding the others in terms of efficiency in identifying both known and novel attacks with an accuracy rate of 99.81%. This discovery improves our capacity to counteract cyberthreats and offers a strong basis for creating more potent remedies. But science is by its very nature a discipline that seeks innovation and is always evolving. Consequently, in order to fully realize the potential of these models, we are dedicated to continuous development.

In the future, we look forward to the possibility of integrating our proposed model (BI-LSTM +GRU) into actual usage environments, to be an innovative and available technological solution that helps in early detection of DDoS attacks and supports the stability and security of the digital space. It provides valuable tools for cybersecurity professionals and organizations alike.

Bibliography

[1] Sumit Kumar, Sumit Dalal, and Vivek Dixit. “The osi model: overview on the seven layers of computer networks”. In: International Journal of Computer Science and Information Technology (2014), pp. 461–466.

[2] Unknown. TCP/IP vs. OSI: What’s the Difference Between the Two Models? 2024. URL: https://community.fs.com/article/tcpip-vs-osi-whats-the-difference-between-the-two-models.html(visited on 03/15/2024).

[3] What Is a Packet? URL: %5Curl%7Bhttps://www.cloudflare.com/learning/network-layer/what-is-a-packet/%7D (visited on 03/15/2024).

[4] Khan Academy. IP Packets. 2024. URL: https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:routing-with-redundancy/a/ip-packets(visited on 03/15/2024).

[5] IPv4 Header Format. URL: %5Curl%7Bhttps://www.educba.com/ipv4-header-format/%7D(visited on 03/15/2024).

[6] GeeksforGeeks. Differences between TCP and UDP. 2023. URL: https://www.geeksforgeeksorg/differences-between-tcp-and-udp/(visited on 03/16/2024).

[7] Khan Academy. Transmission Control Protocol (TCP). 2021. URL: https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:transporting-packets/a/transmission-control-protocol--tcp(visited on 03/16/2024).

[8] GeeksforGeeks. Packet Loss Detection Algorithms. 2022. URL: https://www.geeksforgeeks.org/packet-loss-detection-algorithms/(visited on 03/16/2024).

[9] K. Munivara Prasad, A. Rama Mohan Reddy, and K. Venugopal Rao. DoS and DDoS Attacks: Defense, 2014. URL: https://globaljournals.org/GJCST_Volume14/3–DoS-and-DDoS-Attacks-Defense-Detection.pdf (visited on 03/02/2024).

[10] Various Authors. DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance. CRC Press, Taylor & Francis Group, 2016. URL: https://el.newoutlook.it/download/book/DDoS.pdf.

[11] Zenarmor. DoS and DDoS Attacks: What are Their Differences? 2024. URL: https://www.zenarmor.com/docs/network-security-tutorials/dos-vs-ddos-attacks (visited on 05/02/2024).

[12] Iman Sharafaldin et al. CIC-DDoS2019 Dataset. 2019. URL: https:// www. unb. ca/cic/datasets/ddos-2019.html(visited on 04/03/2024).

[13] Hung-Jen Liao et al. “Intrusion detection system: A comprehensive review”. In: Journal of Network and 36.1 (2013), pp. 16–24.

[14] Korteby Amira Amarsetti Asmaa. “La detection de la maladie du cancer a` partir des donnees des puces à ADN à lùaide de l’apprentissage profond”. MA thesis. Medea: Universite Dr. Yahia fares, 2021.

[15] A. Burkov. The Hundred-page Machine Learning Book. Andriy Burkov, 2019 ISBN: 9781999579500. URL: https://books.google.dz/books?id=ZF3KwQEACAAJ.

[16] SITECH. AI vs Machine Learning vs Deep Learning. 2024. URL: https://www.sitech.me/blog/ai-vs-machine-learning-vs-deep-learning(visited on 03/19/2024).

[17] “Machine learning algorithms (article)”. In: (2023). URL: https://www.khanacademy.org/computing/ap-computer-science-principles/data-analysis-101/x2d2f703b37b450a3:machine-learning-and-bias/a/machine- learning-algorithms (visited on 03/19/2024).

[18] Iqbal H. Sarker. “Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions”. In: SN Computer Science 2.420 (2021). DOI: 10.1007/s42979- 021- 00815- 1. URL: https://link.springer.com/article/10.1007/s42979-021-00815-1(visited on 03/19/2024).

[19] I.N. da Silva et al. Artificial Neural Networks: A Practical Course. Springer International Publishing, 2016. ISBN: 9783319431628. URL: https://books.google.dz/books?id=DL_mDAAAQBAJ.

[20] Jitendra Sharma. Introduction to Supervised Deep Learning Algorithms! 2021. URL: https://www.analyticsvidhya.com/blog/2021/05/introduction-to-supervised-deep-learning-algorithms/(visited on 03/20/2024).

[21] V. Sze et al. Efficient Processing of Deep Neural Networks. Synthesis Lectures on Computer Architecture. Morgan Claypool Publishers, 2020.

[22] Saily Shah. Convolutional Neural Network: An Overview. 2022. URL: https://www. analyticsvidhya.com/blog/2022/01/convolutional-neural-network- an-overview/ (visited on 03/21/2024).

[23] Li Fei-Fei, Justin Johnson, and Andrej Karpathy. Lecture 10: Recurrent Neural Networks. https://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10. pdf. 2017.

[24] Debasish Kalita. A Brief Overview of Recurrent Neural Networks (RNN). 2024. URL: https://www.analyticsvidhya.com/blog/2022/03/a- brief- overview- of-recurrent-neural-networks-rnn/(visited on 03/24/2024).

[25] GeeksforGeeks. Deep Learning — Introduction to Long Short Term Memory. 2023. URL: https://www.geeksforgeeks.org/deep- learning- introduction- to-long-short-term-memory/(visited on 03/29/2024).

[26] Shipra Saxena. What is LSTM? Introduction to Long Short-Term Memory. 2024. URL: https://www.analyticsvidhya.com/blog/2021/03/introduction- to-long-short-term-memory-lstm/(visited on 03/29/2024).

[27] GeeksforGeeks. Deep Learning — Introduction to Long Short Term Memory. 2023. URL: https://www.geeksforgeeks.org/deep- learning- introduction- to-long-short-term-memory/(visited on 03/29/2024).

[28] NordVPN. Multilayer perceptron definition - Glossary. 2024. URL: https://nordvpn.com/cybersecurity/glossary/multilayer-perceptron/ (visited on05/01/2024).

[29] Wikipedia. Multilayer perceptron. 2024. URL: https://en.wikipedia.org/wiki/Multilayer_perceptron(visited on 05/01/2024).

[30] Unknown. “Gated Recurrent Unit Networks”. In: GeeksforGeeks (2023). URL: https://www.geeksforgeeks.org/gated-recurrent-unit-networks/ (vis- ited on 05/22/2024).

[31] Stefan Wermter and Ron Sun. An overview of hybrid neural systems. Springer, 2000.

[32] Yuhua Chen, Subhash Kak, and Lei Wang. “Hybrid neural network architecture for on-line learning”. In: arXiv preprint arXiv:0809.5087 (2008).

[33] What’s the difference between Normalization and Standardization? URL: %5C%5Curl% 7B[3](https://www.someka.net/blog/how- to- normalize- data- in-excel/)%7D(visited on 05/05/2024).

[34] Yun Xu and Royston Goodacre. “On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning”. In: Journal of Analysis and Testing 2 (Oct. 2018), p. 14. DOI: 10.1007/s41664-018-0068-2.

[35] Ivandro Ortet Lopes et al. “Towards effective detection of recent DDoS attacks: A deep learning approach”. In: Security and Communication Networks 2021 (2021), pp. 1–14. URL: https://www.hindawi.com/journals/scn/2021/5710028/.

[36] Manisha Sirsat. Neo: Generalizing Confusion Matrix Visualization to 2019. URL: https: //manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html(visited on 05/03/2024).

[37] Li Xinlong, Chen Zhibin, et al. “DDoS attack detection by hybrid deep learning methodologies”. In: Security and Communication Networks 2022 (2022), pp. 1–8. URL: https: //www.hindawi.com/journals/scn/2022/7866096/.

[38] Hongwei Zhou and Jie Ling. “A Cooperative Detection of DDoS attacks based on CNN-BiLSTM in SDN”. In: Journal of Physics: Conference Series. Vol. 2589. 1. IOP Publishing. 2023, p. 012001.

[39] Suwan Babu Bastola, Subarna Shakya, and Susmita Sharma. “Distributed denial of service attack detection on software defined networking using deep learning”. In: Proceedings of the 2017 2021, pp. 13–16.

[40] M. Bastam E. Ataie M. Vasou Jouybari. “An MLP-based Deep Learning Approach for Detecting DDoS Attacks”. In: Journal of Electrical Engineering (TJEE) 52.3 (2022), pp. 195–204.

[41] Monika Roopak, Gui Yun Tian, and Jonathon Chambers. “An intrusion detection system against ddos attacks in iot networks”. In: 2020 10th annual computing and communication workshop anIEEE. 2020, pp. 0562–0567.

[42] Marcos VO Assis et al. “A GRU deep learning system against attacks in software defined

networks". In: Journal of Network and Computer Applications 177 (2021), p. 102942.

[43] Yiying Zhang et al. "A BiLSTM-based DDoS attack detection method for edge computing". In: Energies 15.21 (2022), p. 7882.

[44] Shahzeb Haider et al. "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks". In: Ieee Access 8 (2020), pp. 53972–53983.

[45] Mahmoud Said Elsayed et al. "Ddosnet: A deep-learning model for detecting network attacks". In: 2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and MultimediaIEEE. 2020, pp. 391–396.

[46] Python Institute. About Python - Python Institute. https://www.pythoninstitute. org/about-python. 2021. (Visited on 06/03/2024).

[47] Python Tutorial. https://www.w3resource.com/python/python-tutorial.php. Accédé le 2024-06-09.

[48] Rais Kaouther Saradouni Melissa. "Détection de virus l'hépatite C basé sur les modèles de Machine Learning". MA thesis. Medea: Universite Dr. Yahia fares, 2023.

[49] Google Colaboratory. https://colab.research.google.com/. (Visited on06/06/2024).

[50] Frequently Asked Questions - Colaboratory. https://research.google.com/colaboratory/faq.html. (Visited on 06/06/2024).