People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
Mohamed El Bachir El Ibrahimi University of Bordj Bou Arreridj
Faculty of Mathematics and Computer Science
Department of Computer Science

THESIS

Presented for graduation

Master in Computer Science

Specialty: information and communication technology

SUBJECT:

Deep Learning-based Anomaly Detection in Network Traffic Patterns

Presented by:

HEDJAM Lidia

BELOUAHRI Aya

Publicly supported on: 20/June /2024

In front of the jury composed of:

President: Dr. MAZA Sofiane

Examiner: M. BENSEFIA Hassina

Supervisor: Dr. BEGHOURA Mohamed Amine

2023/2024

Dedication

I dedicate this work
To dear mother and dear father for their love and support
No words can fully express my deep thanks for them
May God grant them happiness and health.

To my dear brothers and sisters for their
encouragement and affection.

To my partner who worked diligently and tirelessly
with me to realize this project.

To all my family and friends.

To all my teachers during the years of study and especially
The computer Science teachers of
Bordj Bou Arreridj University.

HEDJAM Lidia

Dedication

I dedicate this modest work first of all
To my dearest parents, for their patience, love, support and encouragement throughout my years of study.
May Allah, the Almighty, preserve you and health and long life.

To my lovely sisters and my brothers
Thank you for always putting up with me.
I love you very much.

To all my family, far and near.

To all my best friends.

BELOUAHRI Aya

Acknowledgment

*First of all, we would like to thank **ALLAH** the Almighty for giving us the courage, patience and health to carry out this modest work.*

*Our thanks and gratitude go to **Dr. BEGHOURA Mohamed Amine** who guided us during our study and the realization of this project, with a constant interest in our work.*

*Our respect and gratitude also go to **the members of the jury** who did us the honor of judging this work by their availability, their relevant observations, and their valuable advice that we deeply appreciate and that will undoubtedly contribute to the improvement of this memory.*

*Much of the credit goes to **our parents** who supported and encouraged us at every stage of our lives. You enlighten our lives with your presence and nothing can ever replace you. May Allah preserve and protect you.*

*Our thanks and respect to all **our teachers** during the years of study. Without forgetting of course to thank deeply all those who contributed to the realization of this work.*

Thank you all.

Abstract

The anomaly in network traffic is a crucial issue that can cause significant losses in network security and performance. This prompted us to undertake this work to detect these anomalies accurately and promptly using deep learning techniques.

This thesis investigates the use of long short-term memory (LSTM) neural networks, one of the deep learning methods, to detect anomalies in network data flows. LSTMs are well suited to this task thanks to their ability to capture long-term temporal dependencies. Our approach is distinguished by its ability to detect complex and varied anomalies, thus improving the security and efficiency of computer networks. The results show a significant improvement over traditional methods.

Keywords

Anomaly detection, Network traffic, Network security, Network performance, Deep learning, Long short-term memory (LSTM), Neural networks, Temporal dependencies, Complex anomalies, Computer networks.

Résumé

L’anomalie du trafic réseau est un problème crucial qui peut entraîner des pertes importantes de sécurité et de performance du réseau. Cela nous a incité à entreprendre ce travail pour détecter ces anomalies avec précision et promptement en utilisant des techniques d’apprentissage profond.

Cette thèse étudie l’utilisation de réseaux de neurones de mémoire longue durée (LSTM), l’une des méthodes d’apprentissage profond, pour détecter les anomalies dans les flux de données réseau. Les LSTM sont bien adaptés à cette tâche grâce à leur capacité à capturer des dépendances temporelles à long terme. Notre approche se distingue par sa capacité à détecter des anomalies complexes et variées, améliorant ainsi la sécurité et l’efficacité des réseaux informatiques. Les résultats montrent une amélioration significative par rapport aux méthodes traditionnelles.

Mots Clés

Détection d'anomalies, Trafic réseau, Sécurité réseau, Performance réseau, Apprentissage profond, Réseaux de neurones, Mémoire longue durée (LSTM), Dépendances temporelles, Anomalies complexes, Efficacité des réseaux informatiques, Méthodes traditionnelles.

ملخص

يعد شذوذ حركة المرور على الشبكة مشكلة حاسمة يمكن أن تؤدي إلى فقدان كبير لأمن الشبكة وأدائها. دفعنا هذا إلى القيام بهذا العمل لاكتشاف هذه الحالات الشاذة بدقة وسرعة باستخدام تقنيات التعلم العميق.

تدرس هذه الأطروحة استخدام الشبكات العصبية للذاكرة طويلة المدى(LSTM) ، وهي إحدى طرق التعلم العميق، لاكتشاف التشوهات في تدفقات بيانات الشبكة. إن (LSTM) مناسبة تمامًا لهذه المهمة بفضل قدرتها على التقاط التبعيات الزمنية طويلة الأجل. يتميز نهجنا بقدرته على اكتشاف الحالات الشاذة المعقدة والمتنوعة، وبالتالي تحسين أمن وكفاءة شبكات الكمبيوتر. تظهر النتائج تحسنًا كبيرًا عن الأساليب التقليدية

الكلمات المفتاحية

شذوذ حركة المرور، أمن الشبكة، أداء الشبكة، تقنيات التعلم العميق، الشبكات العصبية، الذاكرة طويلة المدى التبعيات الزمنية، الحالات الشاذة المعقدة، كفاءة شبكات الكمبيوتر، الأساليب التقليدية.

Table of Contents

Abbreviations

LSTM	Long Short-Term Memory
DL	Deep learning
ML	Machine Learning
AI	Artificial Intelligence
LAN	Local Area Network
MAN	Metropolitan Area Network
IoT	Internet of Things
K-NN	K- Nearest Neighbors
ANN	Artificial neural networks
CNN	Convolutional Neural Networks
RNN	Recurrent Neural Network
GRU	Gated Recurrent Unit
TCN	Temporal Convolutional Network
CIC	Canadian Institute for Cyber security
DDoS	distributed denial-of-service
DoS	Denial of Service
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ICP	Ideal Customer Profile
DNS	Domain Name System
CSV	Comma separated value
DNN	Deep Neural Network

List of Figures

List of Tables

General Introduction

In today's digital landscape, the security of network traffic has emerged as a critical concern, demanding advanced techniques for timely and accurate anomaly detection. Traditional methods often struggle to adapt to the dynamic and intricate patterns exhibited by network traffic, resulting in high false alarm rates. This thesis focuses on addressing these limitations through the application of deep learning techniques for anomaly detection in network traffic.

Motivated by the urgent need to enhance anomaly detection methods amidst evolving cyber threats and complex network dynamics, this research leverages the power of deep learning, particularly Long Short-Term Memory (LSTM) networks. The goal is to revolutionize anomaly detection by developing advanced systems capable of accurately identifying a wide range of anomalies, from subtle deviations to sophisticated cyber-attacks. By providing actionable insights, these advancements aim to empower organizations to strengthen their cyber defenses, mitigate operational risks, and safeguard digital assets.

The primary objective of this thesis is to analyze and develop advanced anomaly detection techniques for sequential data in network traffic patterns using deep learning, specifically LSTM networks. The structure of this thesis unfolds as follows:

- **Chapter 1: Anomaly-Based Detection in Network Traffic Patterns**: This chapter analyzes traditional and advanced anomaly detection techniques, justifying the adoption of deep learning methods.
- **Chapter 2: Deep Learning Fundamentals**: Here, fundamental concepts of artificial intelligence, machine learning, and deep learning are discussed, with a focus on LSTM as a neural network model.
- **Chapter 3: Dataset Description**: This chapter provides a detailed description of the dataset used in the study, including its sources, preprocessing methods, and types of attacks simulated.
- **Chapter 4: Methodology**: The methodology is described, encompassing the architectural details of the LSTM models employed and the implementation phases.
- **Chapter 5: Comparative Results**: This chapter presents a comparative study of the model results with related works using the same dataset, offering an evaluation of experimental outcomes.

Chapter 1:
Anomaly-Based Detection in Network Traffic Patterns

1. Introduction

The network traffic analysis, especially in the context of Internet of Things (IoT), plays a crucial role in maintaining the performance and security of computer networks and IoT infrastructures through anomaly detection that involves identifying events that fall outside the normal range. In this chapter, we introduce the concepts of network security, anomaly detection, and its various method.

2. Network Traffic

Network traffic is the term used to describe the movement of data packets across networks. Everything from emails sent to websites visited to files downloaded can be included…etc. [1]

3. Types of Network Traffic anomalies

3.1. LAN and MAN anomalies

Local Area Network (LAN) anomalies refer to abnormal patterns or behaviors observed within a local network environment, affecting the communication and data transfer between devices and nodes within the network.

Metropolitan Area Network (MAN) anomalies refer to abnormal patterns or behaviors observed within a metropolitan network environment, affecting the communication and data transfer between interconnected LANs and WANs within a geographic area.

Their impact is on the network performance, increased latency, reduced network throughput, compromised network security, and potential network downtime. [2] [3]

3.2. BOTNET (IoT) anomalies

A botnet is a network of threatened computers, IoT devices, and other internet-connected devices that are controlled by malicious actors to conduct unauthorized and malicious activities. Botnet anomalies refer to abnormal patterns or behaviors associated with botnet activities within a network environment. The impact of BOTNET anomalies is on compromised network security, unauthorized access, data exfiltration and theft, distributed denial-of-service (DDoS) attacks and propagation of malware and viruses. [4] [5]

4. IoT Network

IoT means how the entities communicate with each other. Given that they are multiple, diverse and have low processing power requirements, Internet of Things devices are vulnerable to cyber-attacks and this has made security and privacy regulations necessary to prevent system threats and vulnerabilities. [6]

Detection of anomalies or outliers is one area IoT research, as errors resulting from malicious attacks or rare observations can cause anomalies, and damage the entire Internet of Things if not detected and addressed. [7]

5. What are anomalies?

The term anomaly or outlier is derived from a Greek word “anomolia” meaning uneven or irregular. [8]. They are unusual conduct or pattern in data specify a problem and does not conform to expected normal behavior [7] . The reasons of anomaly are different can be induced credit card fraud, cyber-intrusion…etc. [9]

As showing in the figure 1, there are two normal dataset zone, N1 and N2; the points that are far away from these zones (O1 O2 and the region points of O3) are the anomalies. [9]

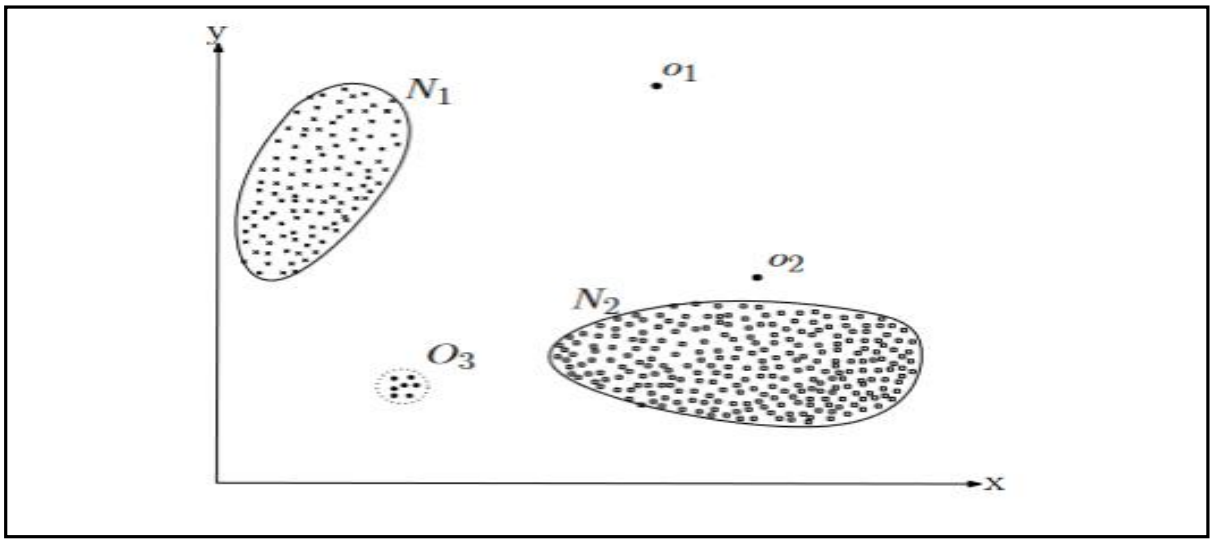


Figure 1: Example of anomaly in a 2D dataset

6. Anomaly detection

Anomaly detection, also called as "deviation detection", it is the problem of the incompatibility in data patterns or behaviors that can referred to as anomaly, outlier. [8]

Anomaly detection applications are various such as intrusion detection for cyber-security, fault detection in safety critical systems. A several anomalous detection techniques have been developed given its importance in maintaining system integrity and security. [9]

7. Anomaly detection techniques

Anomaly detection techniques can work in one of the three method based on the availability of labels: [9]

7.1. Supervised technique

This approach necessitates the training dataset that has determined instances for normal and abnormal category to produce a predictive model for these categories then compared data to the model to determine its class. [9]

7.2. Semi-Supervised technique

This approach assumes that the training dataset is labeled just for normal class and it is vastly applicable because of that, then using the model to find anomalies in the test data. [9]

7.3. Unsupervised technique

This technique is widely applicable because it does not require to training data. By using the unlabeled dataset as training data, semi-supervised approaches can be reformed to operate in an unsupervised method. [9]

8. Challenges in Network Traffic Anomaly Detection

- **High Volume and Velocity**: Analyzing vast amounts of high-speed network traffic data in real-time is a significant challenge. [12]
- **Complex and Variable Traffic Patterns**: Network traffic patterns are influenced by various factors, making it difficult to distinguish between normal and anomalous traffic. [13]
- **Variety of Traffic Types**: Differentiating between various types of network traffic (unicast, multicast, broadcast, etc.) presents challenges for anomaly detection. [14]
- **Dynamic Nature of Networks**: Constant changes in network devices, applications, and users require adaptive anomaly detection models. [15]

9. Requirements for Effective Network Traffic Anomaly Detection

- **Scalability**: Anomaly detection systems should be scalable to handle increasing network traffic volume and complexity. [14]
- **Real-time Analysis**: Anomaly detection systems should analyze network traffic in real-time or near real-time to detect and respond to anomalies promptly. [16]
- **Adaptability and Learning**: Anomaly detection systems should adapt and learn from new data to maintain accurate and up-to-date detection models.
- **Accuracy and Precision**: Anomaly detection systems should accurately distinguish between normal and anomalous traffic to minimize false positives and negatives. [17]
- **Integration and Compatibility**: Anomaly detection systems should integrate seamlessly with existing network infrastructure and security tools. [12]

10. Time Series Concept

A time Series is a sequence of data points listed in chronological order and taken at consecutive evenly spaced points in time. It refers to analysing change in data trends over a period of time using methods to extract statistics and other characteristics of the data. One of the Time Series applications is predicting the future value of an item based on its past value. It used in anomaly detection by analyzing and learning time series, then understanding the changes in historical data and detect normal or abnormal data points. [18]

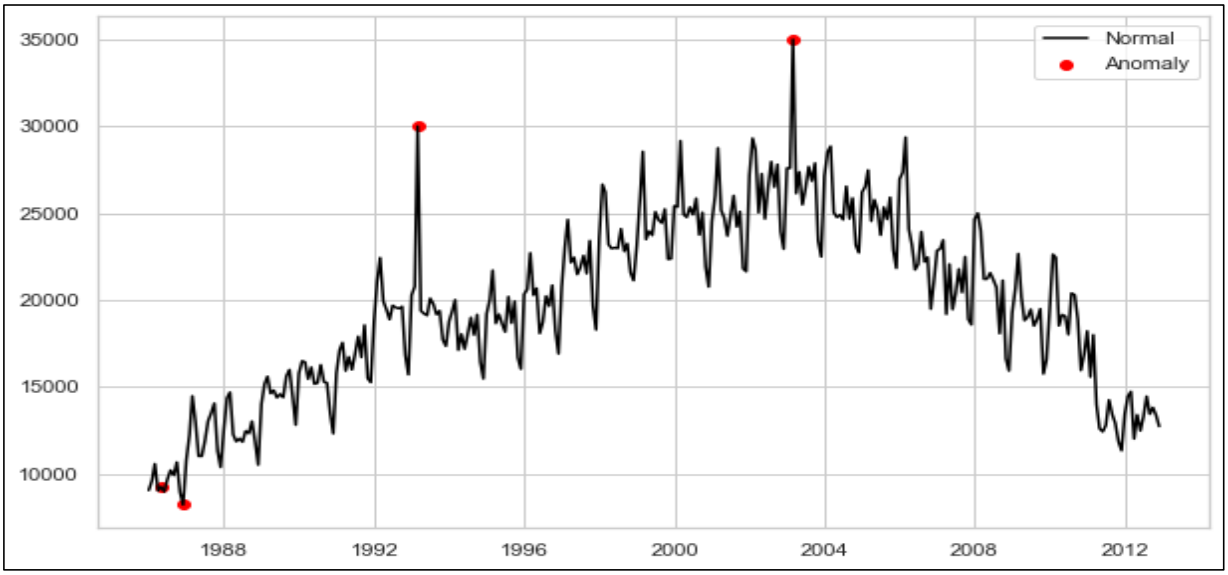


Figure 2: Anomaly Detection in Time Series [19]

In the figure 3, the algorithm firstly is very naive and cannot understand what counts as an anomaly. The more data it receives, the more differences it can detect and adjust itself. The true negatives means the contamination parameters are too high. On the other hand, if the red dots are not what it should be, the contamination parameters are set too low. [19]

11. Anomaly Detection Traditional Methods

Z- Score: method for anomaly detection identifies anomalies by measuring how many standard deviations an observation deviates from the mean of a dataset. Here's how it works: **Formula: Z = (X−μ) σ**. Where **X** is the data point, **μ** is the mean, and **σ** is the standard deviation. [20]

K-Nearest Neighbors (K-NN): It is a simple distance-based anomaly detection technique that can accurately detect anomalies. In dataset, the k-nearest-neighbors must be found for each data point. Based on these neighbors, an anomaly score is calculated. [21]

K-Means Clustering: attempts to reduce the total distance (Euclidean distance) between the data points and their assigned cluster centers. The algorithm iteratively assigns each data point to the nearest center and recalculates the center positions until convergence or a specified number of iterations. [22]

12. The limitations of traditional methods

- **Requires manual feature engineering**: Traditional machine learning methods often require human-crafted features that do not account for the underlying pattern and relationship in the data. It is a time-consuming, which would lead to subpar performance. [23]
- **High false positive rates**: Traditional methods may produce high false positive rates, incorrectly identifying normal data as anomalies or misclassifying instances. [23]

- **Less adaptive to dynamic and evolving environments**: Traditional models are static and do not adapt to changes in the data distribution or evolving patterns. They require frequent retraining and recalibration to maintain performance in dynamic environments. [23]
- **Computationally expensive, especially for large datasets**: Traditional algorithms may be computationally expensive and require significant computational resources, memory, and time to train and evaluate, especially on large-scale datasets. [23]
- **Scalability issues with very large datasets or high-dimensional spaces**: Traditional methods may face scalability issues when handling with extensive datasets, leading to increased computational complexity, memory usage, and potential performance degradation. [23]

13. Anomaly Detection Advanced Methods (Deep Learning Methods)

Neural Networks: include an interrelated layer of neurons that learn to operate with complex missions by the adaptation of connections weights between nodes during training. The common methods used are convolutional neural networks (CNNs). [25]

Recurrent Neural Networks (RNNs): technique that allow information to persist, making them suitable for sequential and time-series data. For anomaly detection in time-series data, Long Short-Term Memory (LSTM) is commonly used. Their advantages include capturing temporal dependencies and patterns, improved detection accuracy for dynamic and evolving environments. [25]

Convolutional Neural Networks (CNNs): they are specialized for processing grid-like data, such as images and they consist of convolutional layers, pooling layers, and fully connected layers. CNNs are widely used in computer vision tasks like image classification, object detection, and image segmentation. [4]

Autoencoder: is a type of neural network architecture designed to efficiently compress (encode) input data down to its essential features, then reconstruct (decode) the original input from this compressed representation.[26]

14. Related Work

In this study, a deep learning-based neural network is proposed that employs for the capturing of complex relationships and structures within the data. [44]

DNN features a more complicated hidden layer structure with many more levels. Well-designed deep neural networks can be trained to produce the necessary results with high accuracy scores, and they are used in various aspects such as computer vision, anomaly detection and all other deep learning domains. [76]

15. Why deep learning

Although conventional anomaly detection techniques are extensively employed across several fields, they frequently encounter difficulties in identifying intricate patterns, managing large-scale data, and adjusting to constantly changing and dynamic surroundings. [9]

Neural networks that simulate human decision-making are used in machine learning to solve real-world issues, utilizing some of the fundamental concepts of artificial intelligence.

Deep Learning concentrates even further on a subset of machine learning tools and techniques and uses them to solve nearly any problem requiring artificial or human intelligence. [27]

16. Difference between modern and traditional methods

The application of deep learning while expanding the number of data is the main distinction between it and machine learning. Deep learning techniques require a large amount of data; they perform poorly with little amounts of data. [28]

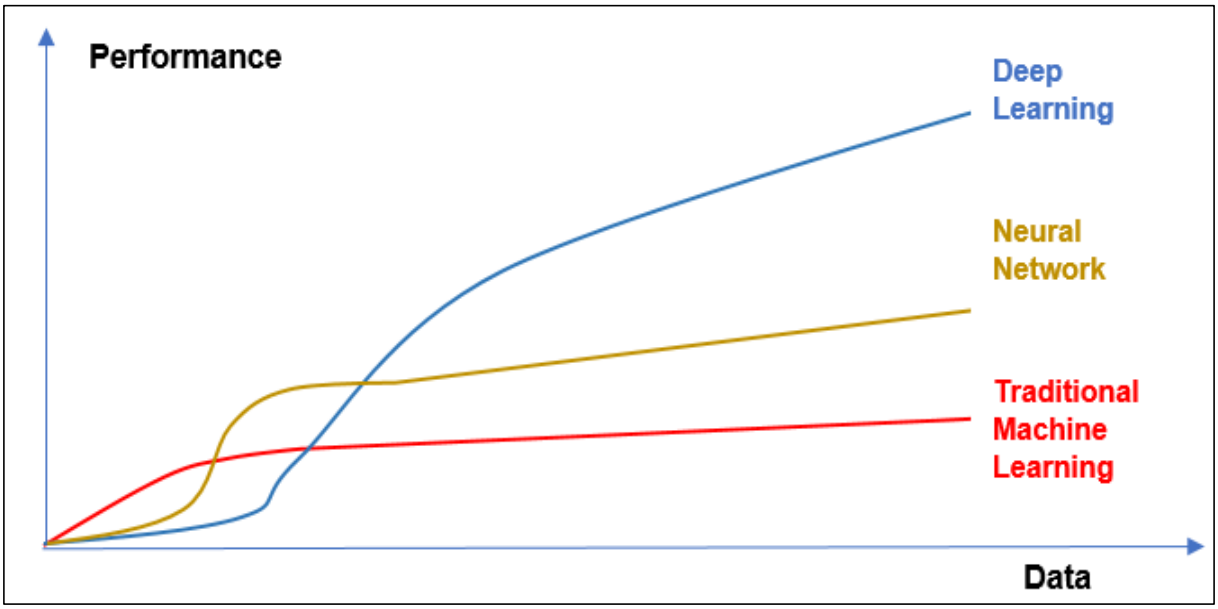


Figure 3: Advanced versus Traditional Methods

17. Conclusion

Traditional anomaly detection methods struggle with complex patterns, high-dimensional data, and dynamic environments, however Deep learning techniques excel at learning from raw data, capturing intricate patterns and temporal dependencies, making them more accurate and scalable for anomaly detection in various domains and real-world applications.

Chapter 2:
Deep Learning Fundamentals

1. Introduction

In today's technological landscape, Artificial intelligence stands out as a driving force of innovation, and within this field, machine learning emerges that enabling computers to learn from data and make decisions. Deep learning, an advanced aspect of machine learning, uses neural networks to detect intricate patterns in the data. Among these neural network architectures, Long Short-Term Memory networks (LSTMs) are particularly notable for their ability to capture long-term dependencies. In this chapter, we will explore the concepts of AI, machine learning (ML), and deep learning (DL), with a detailed focus on LSTM structures.

2. Artificial Intelligence (AI)

Artificial intelligence is the simulation of human intelligence by machines [30]. It exploits computer science and robust datasets to solve problem. It includes machine learning and deep learning sub-fields, which are commonly referenced when discussing artificial intelligence. These fields, which use AI techniques, aim to develop expert systems that can classify or predict things depending on the data they are given [29].Artificial intelligence techniques are the recent approaches for the anomaly detection in IoT. [10]

3. Machine learning (ML)

A branch of artificial intelligence that uses mathematical models to enable machines to learn from data. It is particularly the procedure used to extract pertinent data from a training data set. Finding a model's parameters that will maximize performance is the aim of this phase, especially when the model is carrying out the task that it has been given. [30]

Deep learning is a new field in machine learning that automatically learns datasets without the influence of rules or human knowledge. Other machine learning type models are defined by the presence or absence of human influence on the raw data. These models include supervised learning, unsupervised learning, semi-supervised learning, and others. For improved processing, enormous volumes of raw data are needed for this. [32]

4. Deep learning (DL)

It is a special subfield of machine learning; hat essentially consists of an artificial neural network, drawing inspiration from the structure and functionality of human brain. [32]

Neural network layers power deep learning by executing specific tasks based on input data. Neurons in the neural network can be configured by extensive data training. The product is a deep learning model that can process new data after it has been trained. Without human assistance, deep learning models gather data from various sources and evaluate it instantly. [33]

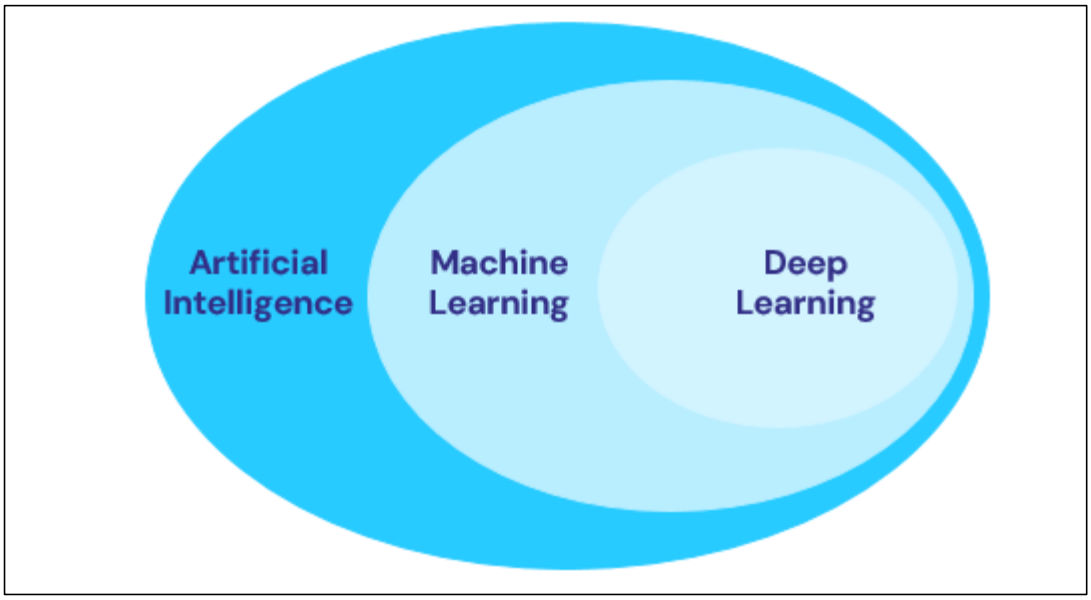


Figure 4: The relationship between AI, ML and DL

5. Deep learning architectures

Deep learning architectures often have three different kinds of learning models, include in the first type the architecture that is trained on completely labeled data; in the second, the architecture is trained on unlabeled data and attempts to construct a structure by extracting meaningful information. For the final model, a training dataset includes both labelled and unlabelled data.[34]

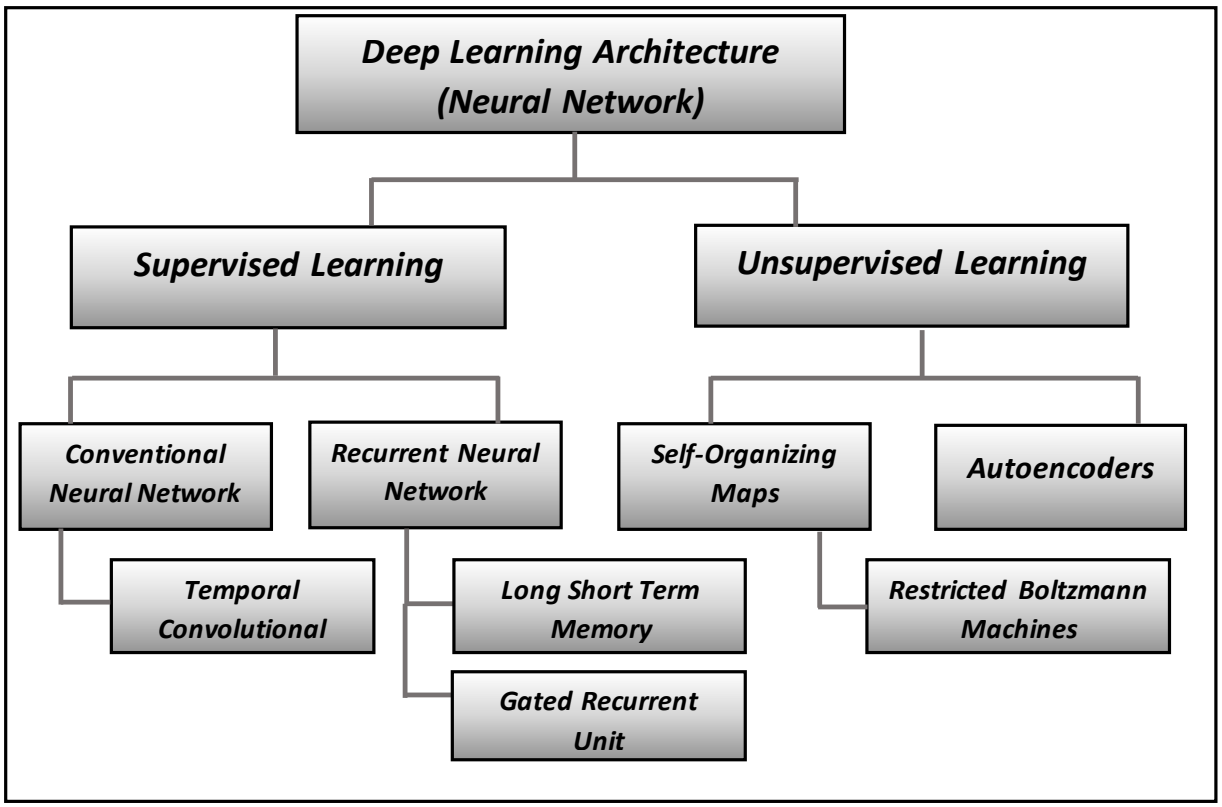


Figure 5: Deep Learning Architecture

6. Neural Network

Neural network or artificial neural network is the center of deep learning algorithms. It simulates how real neurons communicate with one another, drawing inspiration from the human brain, [29].

An input layer, one or more hidden layers, and an output layer make up connected node layer of an artificial neural network. Every node has a threshold and weight attached to it. A node is activated and sends data to the following layer of the network if its output exceeds a given threshold value, if not, no data is transferred to the network's subsequent tier. Training data is essential for neural networks to learn and become more accurate over time. [29]

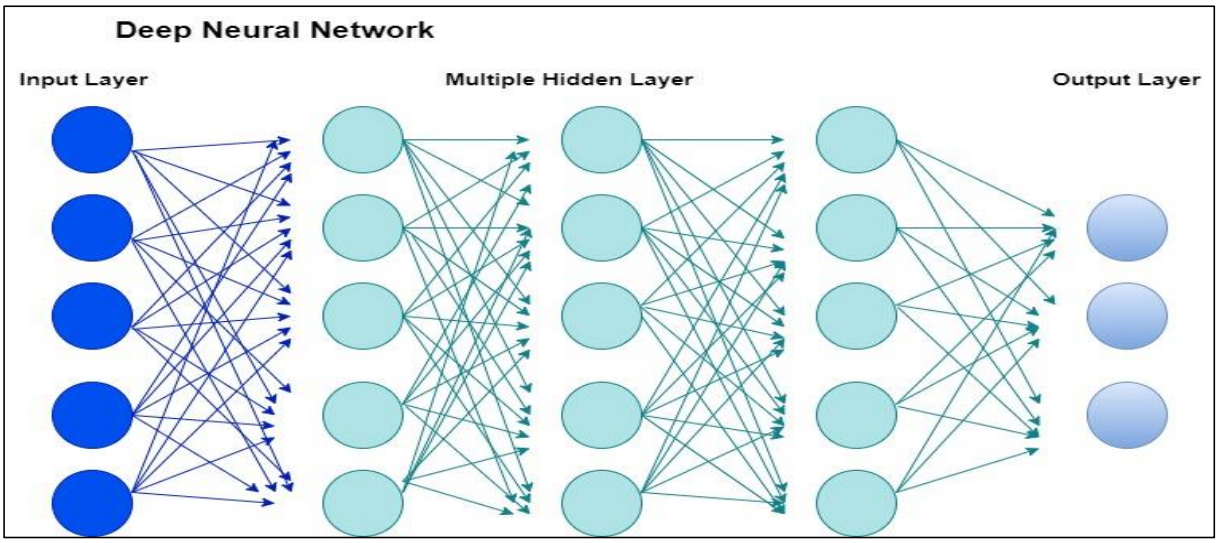


Figure 6: Deep Neural Network

7. Artificial neural network layers

The input data in an artificial neural network is assumed to be completely activated neurons (neurons can partially activate depending on the activation function, but in this example, each neuron either outputs either a 1 or a 0). Artificial neural networks are made up of these interconnected nodes layers. [37]

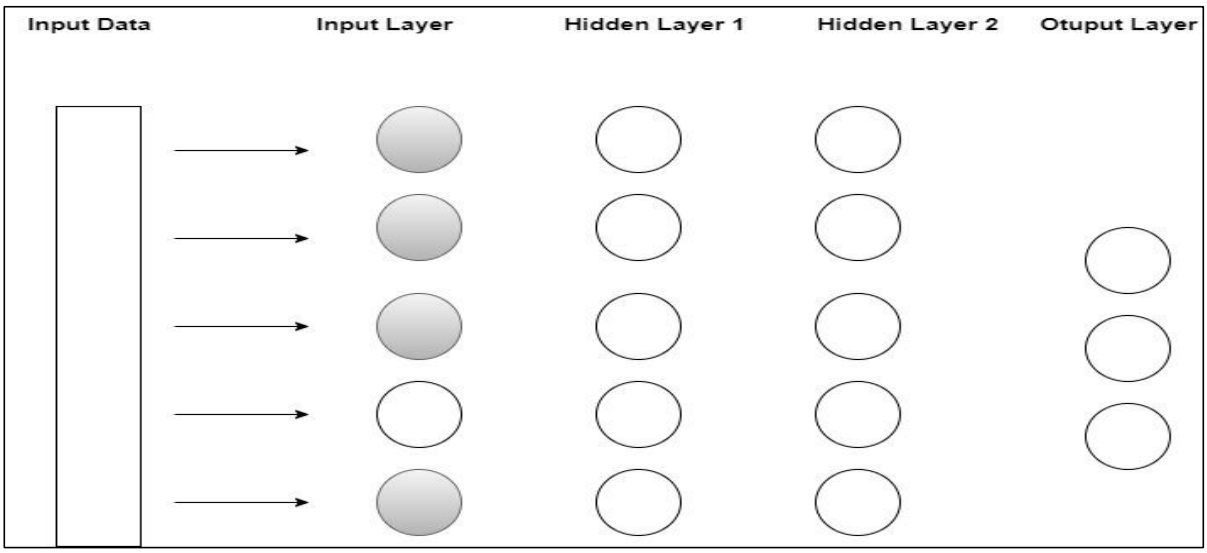


Figure 7: The input data runs through the input layer, and selective nodes fire

"**The input layer**" creates an output that is connected to the first hidden layer by taking all of the inputs. The hidden layer now receives the inputs from the outputs of the nodes that activate in the input layer and fresh data flows. [37]

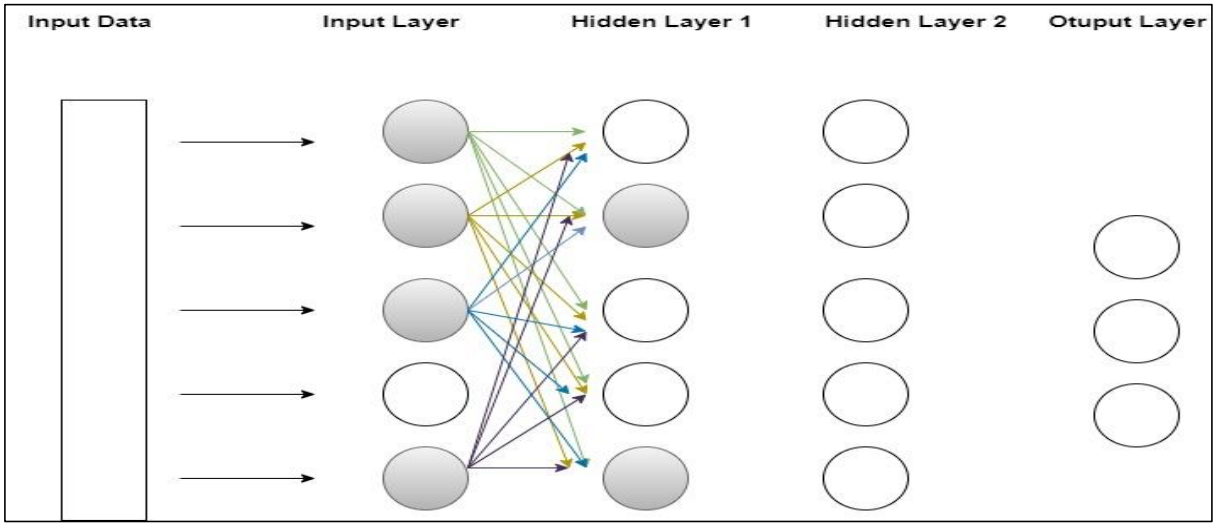


Figure 8: The first hidden layer

"**The first hidden layer**" receives the outputs from the input layer's activated neurons. Selective neurons fire in response to these outputs, which are now the inputs of the subsequent layer. [37]

Although the activation function, weight and other parameters for "**hidden layer 1**" are different from those of the input layer, the data processing is identical. This layer receives data, and its output serves as the input for the following hidden layer. In this example, the input from the preceding layer causes only two nodes to become active. [37]

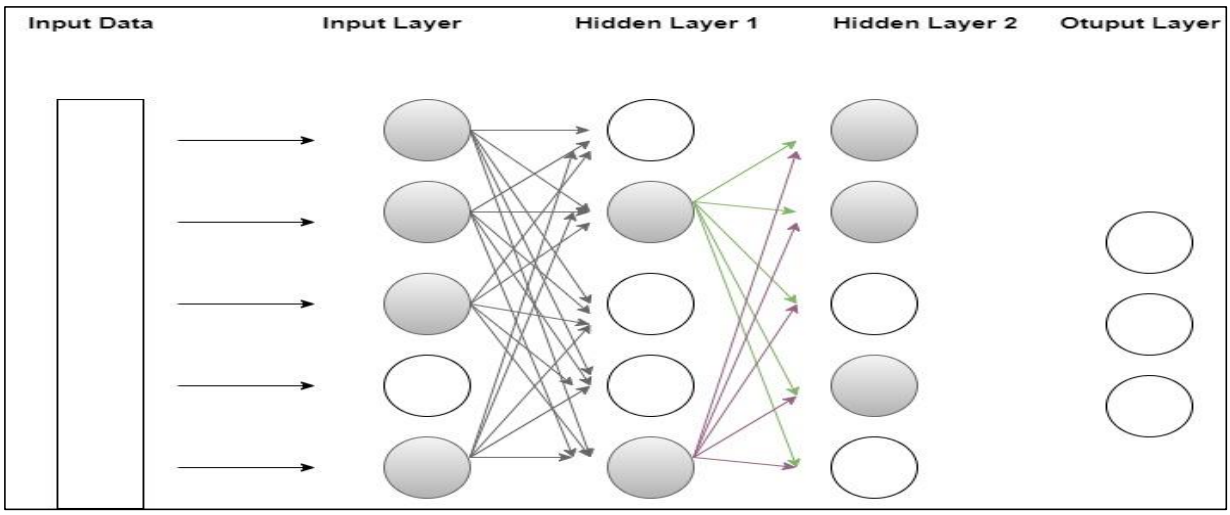


Figure 9: The second hidden layer

After the data processing by "**hidden layer 2**" transfers it to a new layer known as "**the output layer**," where only one of the layer's nodes will be activated. In this example, the output layer's first node is activated. [37]

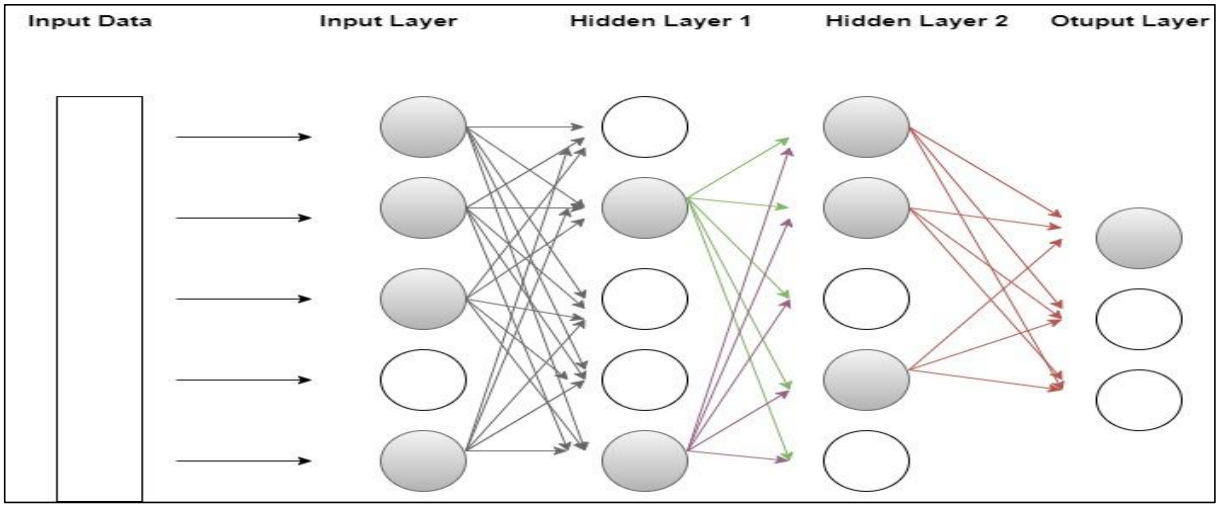


Figure 10: The Output Layer

8. Long Short-Term Memory Networks

LSTMs are one of deep learning model special a type of Recurrent Neural Network that makes the information permanent and can identify long-term dependencies in sequential data. [38]

LSTM has developed into a potent tool in deep learning and artificial intelligence that is facilitating advances across a wide range of domains. Python programmers can use the Keras library to implement the LSTM Model. [38]

9. LSTM Architecture

LSTM functions similarly to an RNN cell at a high level. This is how the LSTM network operates internally, as seen in the architecture below, the LSTM network design is composed of three components, each of which serves a distinct function. [38]

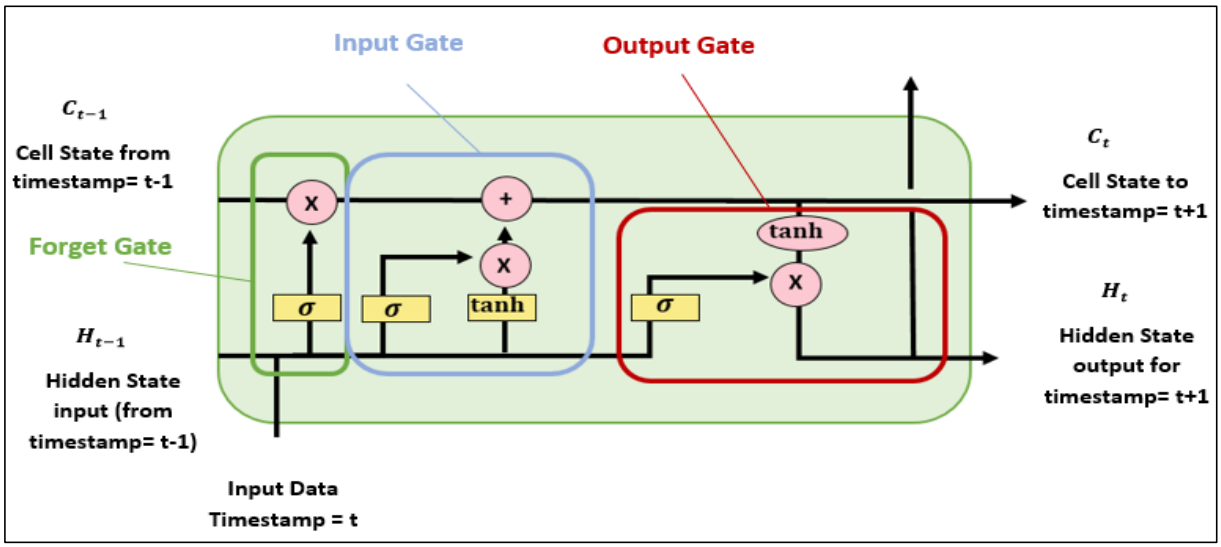


Figure 11: LSTM network Diagram [38]

10. The functions of LSTM architecture's gates

Forget Gate: In The first decision made in an LSTM neural network cell is whether to retain or forget the data from the previous time step. This is the forget gate equation (1): [38]

$$\boldsymbol{f_t = \sigma(X_t \times U_f + H_{t-1} \times W_f)} \quad (1)$$

X_t : Input to the current timestamp.

U_f : Weight associated with the input

H_{t-1} : The hidden state of the previous timestamp

W_f: It is the weight matrix associated with the hidden state. [38]

Then, we apply a sigmoid function that make $\boldsymbol{f_t}$value between **0** and **1**. $\boldsymbol{U_f}$ helps to transform the current input X_t into a suitable format that can be combined with the transformed hidden state $\boldsymbol{H_{t-1}}$ and the same for $W_f$to the hidden state.

The cell state of the preceding timestamp is then multiplied by this $\boldsymbol{f_t}$, as indicated in equations (2) and (3): [38]

$$\boldsymbol{C_{t-1} \times f_t = O} \quad \boldsymbol{if} \quad \boldsymbol{f_t = 0} \quad \text{(Forget everything)} \quad (2)$$

$$\boldsymbol{C_{t-1} \times f_t = C_{t-1}} \quad \boldsymbol{if} \quad \boldsymbol{f_t = 1} \quad \text{(Forget nothing)} \quad (3)$$

One important element that is employed to regulate the information flow via the gates is the sigmoid function. A mathematical function that transfers input values to the range between **0** and **1** is called the sigmoid function, represented by the symbol "**σ**" in equation (4):

$$\boldsymbol{\sigma(x) = \frac{1}{1+e^{-x}}} \quad (4)$$

Input Gate: The input gate is employed to measure the importance of the new data that the input contains. The input gate represent in equation (5): [38]

$$\boldsymbol{i_t = \sigma(X_t \times U_i + H_{t-1} \times W_i)} \quad (5)$$

X_t : Input at the current timestamp.
U_i : Weight matrix of input.
H_{t-1} : A hidden state at the previous timestamp.
W_i : Weight matrix of input associated with hidden state. [38]

Again, we have applied the sigmoid function over it. As a result, the value of $\boldsymbol{i}$ at timestamp $\boldsymbol{t}$ will be between **0** and **1**. [38]

New Information: The new information that added to the cell state is a function of the hidden state at t-1 and input $\boldsymbol{X_t}$ at $\boldsymbol{t-1}$, processed through a ***tanh*** activation function that makes the value of new information will be between -1 and 1. This process is represented in equation (6): [38]

$$\boldsymbol{N_t = \tanh(X_t \times U_c + H_{t-1} \times W_c)} \quad (6)$$

The value of new information will range from **-1** to 1 as a result of the ***tanh*** function. Information is added to the cell state at the current timestamp if the value of $\boldsymbol{N_t}$ is positive, and subtracted from the cell state if it is negative. [38]

However, the N_t will not be added directly to the cell state. Here comes the updated equation: [38]

$$C_t = f_t \times C_{t-1} + i_t \times N_t \quad (7)$$

In this case, C_{t-1} represents the cell state at the current timestamp, and the other values are those that we previously calculated.

Output Gate: The output gate's equation is as follows and it is similar to the two gates before it: [38]

$$O_t = \sigma(X_t \times U_O + H_{t-1} \times W_O) \quad (8)$$

Because of the sigmoid function, its value will be between **0** and **1**. We will now use O_t and ***tanh*** of the updated cell state to determine the current hidden state. As seen below: [38]

$$H_t = O_t \times \tanh(C_t) \quad (9)$$

11. LSTMs Applications in Network Traffic Analysis

Given the relevance of LSTMs to sequence data, they have been successfully applied to various tasks in network traffic analysis, including:

Anomaly Detection: LSTM can learn the normal patterns of network traffic and identify anomalies that may represent malicious network activities or other unusual events. [40]

Traffic Prediction: LSTMs can be trained to predict future network traffic based on historical data, allowing network administrators to predict and prepare for changes in network load, peak traffic hours, or potential congestion. [40]

Network Intrusion Detection: LSTMs can determine patterns associated with different network attacks types and making them useful for intrusion detection systems that aim to protect networks from cyber threats. [40]

12. Conclusion

The integration of AI, ML, DL and artificial neural networks marks a transformative advancement in technology and provides robust tools for data-driven learning, predictive modeling, and addressing intricate challenges. Long Short-Term Memory (LSTM) networks stand out with their sequential architecture, facilitating a deeper understanding and easier design of advanced AI models.

Chapter 3:
Dataset Description

1. Introduction

The main objective of this section is to offer a thorough IoT attack dataset that will foster the creation of security analytics applications for IoT operations. In this section, we will describe the procedures and materials used in the CICIoT2023 dataset creation, which is important for understanding the creation and collection of this dataset and thus evaluate the performance of deep learning model in identifying IoT network traffic as either malicious or benign using this dataset.

2. IoT Lab and Typology

Creating IoT security data that can effectively foster real-world applications presents a challenge specially the establishment of an extensive network comprising IoT devices and topologies. [41]

The Canadian Institute for Cybersecurity (CIC) has emerged as an important unit within the cybersecurity domain and renowned for its different contributions such as the generation of the CICIoT2023 datasets used and the establishment of the IoT lab equipped with a dedicated network infrastructure, including racks and storage areas made to arrange effectively network and IoT equipment. [41]

Understanding different IoT topologies is essential for designing, deploying, and managing IoT networks effectively, as it defines the layout of cables, devices, and routing paths in a network.[42] The IoT topology used to generate the CICIoT2023 dataset included a variety of tools, software applications and 105 IoT devices such as cameras, sensors and microcontrollers, of these, 67 IoT devices were directly implicated in the conducted attacks, while 38 Zigbee and Z-Wave devices were attached with five hubs. [43]

3. Typology Parts

Two components make up this topology:

The first Part:

The connectivity is facilitated by a desktop Windows 10 PC, which is connected to seven Raspberry Pi devices via a VeraPlus access point, which is connected to a Cisco switch. During the trials, the Raspberry Pi devices are responsible for carrying out the malicious behaviors and attacks. A Gigamon Network Tap connects to the Cisco switch, collects the all IoT traffic then transmits it to two network monitors. The network monitors are in charge of recording and storing the traffic in real-time using the network protocol analyzer Wireshark. [43]

A network tap is a hardware device that intercepts a network connection and duplicates the information for monitoring and analysis purposes. The integration of these network taps ensures that they do not interfere with regular network activities, add delay or degrade network performance. Between the attacking and legitimate devices, this gadget contains two monitoring ports, two networks, and one port that connects to the attackers and the other to the victims. We can record traffic entering and leaving the IoT network by using monitor ports. [43]

The second part:

Five gateways and base stations are linked to a Netgear Unmanaged Switch to enable communication with IoT devices that use protocols like Z-Wave and Zigbee. Moreover, the switch is connected to another VeraPlus controller. In turn, this controller is connected to two more Zigbee/Z-Wave hubs and a large number of gadgets that have been identified as attack targets. [43]

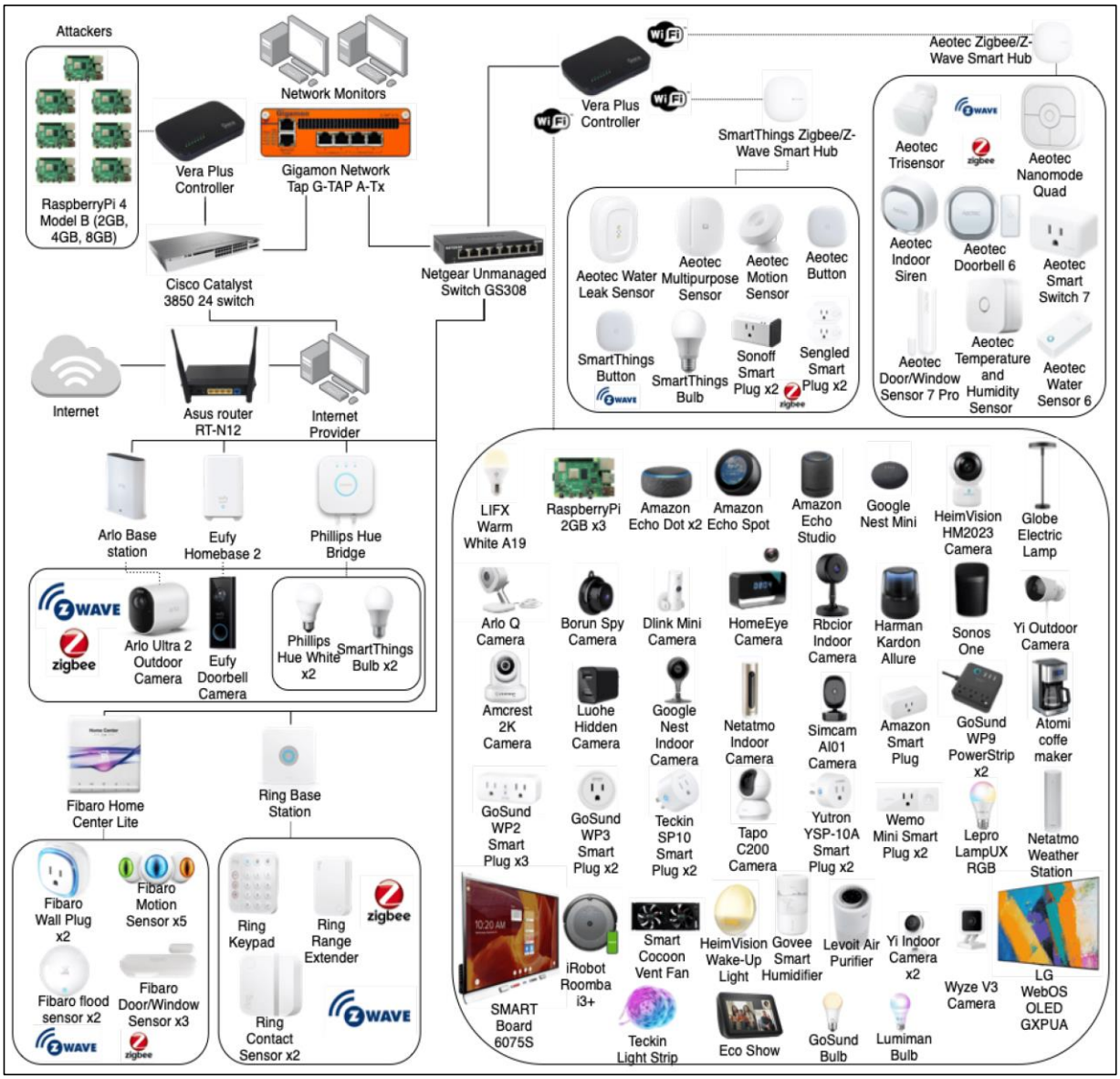


Figure 12: IoT network typology

4. Dataset

We use an IoT attack dataset "CICIoT2023", using a large topology contained seven attacks types, which are DDoS, Denial of Service (DoS), Recon, Web-based, Brute Force, Spoofing, and Mirai. These attacks are spanning across 105 IoT tolls targeting other IoT devices. The purpose of this dataset is to support the creation of security analytics software that has been designed for the Internet of Things (IoT). [43] [44]

5. Attacks Description

The attacks are initiated and executed by malicious IoT devices that targeting vulnerable IoT device. The next table provides a detailed list of the features used to analyze network traffic in the CICIoT2023 dataset. Each feature is described with its name, meaning and with the corresponding number of generated rows. These characteristics are essential for capturing network traffic behavior and detecting anomalies. [44]

Table 1: Attack types and number of rows

	Attack	Rows
DDoS	ACK Fragmentation	285 104
	UDP Flood	5 412 287
	SlowLoris	23 426
	ICMP Flood	7 200 504
	RSTFIN Flood	4 045 285
	PSHACK Flood	4 094 755
	HTTP Flood	28 790
	UDP Fragmentation	286 925
	UCP Fragmentation	452 489
	TCP Flood	4 497 667
	SYN Flood	4 059 190
	SynonymousIP Flood	3 598 138
DoS	TCP Flood	2 671 445
	HTTP Flood	71 864
	SYN Flood	2 028 834
	UDP Flood	3 318 595
Recon	Ping Sweep	2262
	OS Scan	98 259
	Vulnerability Scan	37 382
	Port Scan	82 284
	Host Discovery	134 378
Web-Based	Sql Injection	5245
	Command Injection	5409
	Backdoor Malware	3219
	Uploading Attack	1252
	XSS	3846
	Browser Hijacking	5859
Brute Force	Dictionary Brute Force	13 064
Spoofing	Arp Spoofing	307 593
	DNS Spoofing	178 911
Mirai	GREIP Flood	751 682
	Greeth Flood	991 866
	UDP Plain	890 576

6. Attack Types

In this part, we focuses on understanding and categorizing various cyber threats and methods employed in this research, this figure shows a mind map of the different types of attack included in the CICIoT2023 dataset. Each branch of the map represents a category of attack, subdivided into specific subtypes.

Figure 13: Dataset Overview

- **DDoS and DoS attacks**: aimed to endanger the availability of IoT operations, the attacks are executed :
- ACK fragmentation: Small packets used to compromise the network operations; these packets are sent and treated by routers, firewalls…etc. [46]
- Slowloris: uses HTTP requests, targeted web server and focusing on the applications layer [47]
- ICMP/HTTP/UDP/TCP flood: based on overwhelming a targeted device with different packets types. [48] [49]
- RST-FIN flood: degrades networking capability by sending RST-FIN packets.[50]
- RSH-ACK flood: degrades server operation using PUSH and ACK requests. [51]
- UDP fragmentation: refers to a special UDP flood that consumes more bandwidth while reducing the number of packets. [52]
- ICP fragmentation : uses identical fragmentation IP packets. [53]

- SYN flood: type of TCP flood, sends SYN packets to a targeted server. [54]
- Synonymous IP flood: an extensive number of manipulated TCP-SYN packets (source and destination address). [55]

- **Mirai attacks:** targets vulnerable IoT devices by infecting them with malware to form a botnet that can be used to launch large-scale distributed denial-of-service (DDoS) attacks overwhelm target servers or networks, rendering them inaccessible to legitimate users. [41] Some of most popular variations are GREIP GREETH and UDP Plain.
- **Spoofing attacks:** these attacks allow to malicious actors to masquerade as legitimate systems, granting them unauthorized access to network traffic. These attacks primarily target system access, data theft, and malware dissemination. [56], among the most prevalent spoofing techniques are:

- ARP spoofing: the transmission of falsified ARP (Address Resolution Protocol) messages, aiming to associate the MAC address of the attacker's device with the IP address of another legitimate device within the network.
- DNS spoofing: DNS entries in a DNS server's cache are manipulated, diverting users to fraudulent or malicious websites instead of their intended destinations.

- **Gathering Information from the IoT Topology**: involves collecting comprehensive data about the target. Various methods are employed for these attacks:

- Ping Sweep: or ping scan, it is a reconnaissance attack utilized to detect active hosts on a network. [57]
- OS Scan: (operating system) type of reconnaissance attack aims to determine the operating system type and version running on a targeted host. [58]
- Vulnerability Scan: A network security assessment technique that employs automated tools to identify vulnerabilities in a computer system or network [59]
- Port Scan: A reconnaissance attack that identifies open and active ports on a targeted host. [60]
- Host Discovery: Another reconnaissance attack used to identify active hosts on a network, often serving as the initial step in numerous cyber-attacks. [61]

- **Web-based attacks:** targeting web services operating on IoT devices, include:

- SQL and command Injection: These attacks target web applications by injecting malicious SQL code and commands into the application's input fields, to gain unauthorized access to databases.
- Backdoor Malware: The installation of malware on a targeted system allows the attacker to obtain unauthorized access.
- Uploading Attack: Exploiting vulnerabilities in a web application's file upload functionality to execute an attack.
- Cross-Site Scripting (XSS): Permits an attacker to inject malicious code, enabling the theft of sensitive information.
- Browser Hijacking: A cyber-attack wherein an attacker modifies a web browser's settings. [41]

- **Brute force attacks:** attempts to gain unauthorized access through repeated login attempts. [43]

7. Features Description

Table 2 lists all features used and their description, which are available in csv formats: [44]

Table 2: Description of Features extracted

Feature	Description
Flow Duration	Duration of the packet's flow
Header Length	Length of the packet's header
Protocol Type	Type of protocol (e.g., IP, UDP, TCP)
Time-to-Live (TTL)	Time-to-Live value of the packet
HTTP	Indicates if the application layer protocol is HTTP
HTTPS	Indicates if the application layer protocol is HTTPS
DNS	Indicates if the application layer protocol is DNS
Telnet	Indicates if the application layer protocol is Telnet
SMTP	Indicates if the application layer protocol is SMTP
SSH	Indicates if the application layer protocol is SSH
IRC	Indicates if the application layer protocol is IRC
TCP	Indicates if the transport layer protocol is TCP
UDP	Indicates if the transport layer protocol is UDP
DHCP	Indicates if the application layer protocol is DHCP
ARP	Indicates if the link layer protocol is ARP
ICMP	Indicates if the network layer protocol is ICMP
IP	Indicates if the network layer protocol is IP
LLC	Indicates if the link layer protocol is LLC
Total Packet Length	Summation of packet lengths in the flow
Min Packet Length	Minimum packet length in the flow
Max Packet Length	Maximum packet length in the flow
Average Packet Length	Average packet length in the flow
Packet Count	Number of packets in the flow
Rate	Packet transmission rate in the flow
Outbound Rate (Srate)	Outbound packet transmission rate in the flow
Inbound Rate (Drate)	Inbound packet transmission rate in the flow
FIN Flag Count	Count of packets with FIN flag set in the flow
SYN Flag Count	Count of packets with SYN flag set in the flow
RST Flag Count	Count of packets with RST flag set in the flow
PSH Flag Count	Count of packets with PSH flag set in the flow
ACK Flag Count	Count of packets with ACK flag set in the flow
ECE Flag Count	Count of packets with ECE flag set in the flow
CWR Flag Count	Count of packets with CWR flag set in the flow
CK Packet Count	Number of packets with ACK flag set in the flow
SYN Packet Count	Number of packets with SYN flag set in the flow

8. Conclusion

The CICIoT2023 dataset are used to enhance security analytics in IoT. The dataset creation process cover the CIC IoT Lab, device configurations, attack execution, and data collection methods. This dataset serves as a crucial resource for evaluating the deep learning algorithms performance thus improving IoT security and developing effective defense strategies.

Chapter 4:
Methodology

1. Introduction

Our study aims to develop a robust system for identifying deviations from normal network behavior. In this section, we will introduce the methodology used in our research to detect anomalies in network traffic using the CICIoT2023 dataset with LSTM model. We will explain the overall design for the suggested system.

2. Model Architecture

Our methodology for addressing anomaly detection in network traffic patterns follows a structured approach. We begin with preprocessing our training and test inputs to ensure data consistency. Next, we employ the LSTM model known for its ability to handle sequential data and capture complex temporal dependencies. Following model training, we evaluate the performance using metrics such as accuracy, precision, recall, and F-score. This architecture integrates robust data preprocessing, LSTM modeling, and performance evaluation to effectively detect anomalies in network traffic.

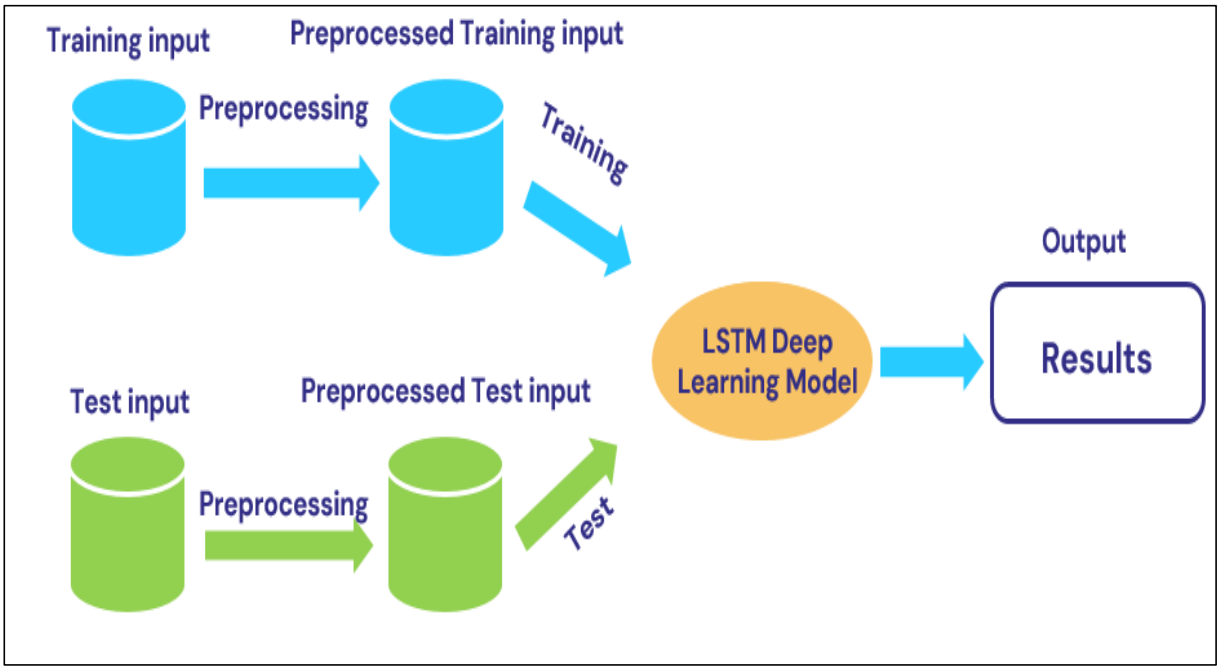


Figure 14: The general Architecture

3. Model Implementation

For the implementation, we used a DELL computer with the following features:

- Processor: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz.
- Installed memory (RAM):8GB.
- System Type: operating system 64bits.
- Microsoft Windows 10.

In preparing of the implementation environment, we was installed the small bootstrap version of Anaconda "**Miniconda**" for Windows 10 operating system. We used the version Miniconda 1.9.7. This is the installation: https://docs.anaconda.com/free/miniconda/index.html

Miniconda

It is a free installer for the conda package. It serves as a compact bootstrap version of the full Anaconda distribution, containing only conda, Python, the packages required by both conda and Python, as well as a limited number of additional useful packages. These supplementary packages typically include tools like pip, zlib, and a few others that are commonly needed in development environments.[66] [63]

Visual studio code

It is a free code editor developed by Microsoft and designed for the development of applications dedicated to machine learning and data science. It has extensions for other languages (PHP, C++, Java, Python…). [64]

Python Language

An advanced programming language with a simple structure. It is an effective language for scripting and rapid application development. Notable libraries in this domain include Pandas, Numpy, and Scikit-Learn, which are commonly employed for various data-related operations (loading, reorganization and processing). [65]

TensorFlow

It is an open source software library for numerical calculation using data flow charts. It is one of more than a dozen of machine intelligence libraries developed by big companies, and is probably one of the newest ones. [66]

Keras

It is an interface for creating and training Deep Learning models in Python. It has been developed to enable rapid experimentation and offers many advantages, including the ability to go from idea to result in record time. Keras supports both convolutional networks (CNNs) and recurrent networks (RNNs). [67]

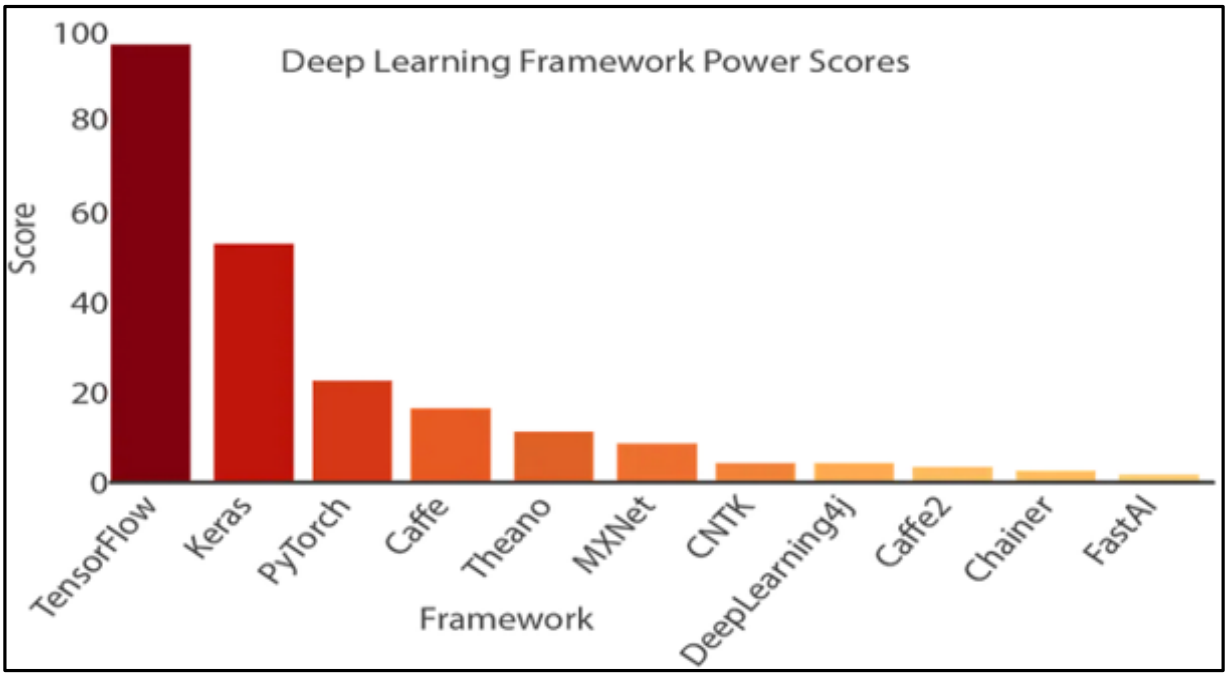


Figure 15: The 10 most popular Deep Learning frameworks [78]

3.1. Libraries Importation Phase

A several libraries have been utilized to offer fundamental capabilities for preprocessing, manipulating, and preparing data before entering it into our LSTM model:

Pandas: it is a Python library used for analyzing, cleaning, exploring, and manipulating datasets. It has a reference to both "Panel Data", and "Python Data Analysis. [68]

NumPy: an open source library for t Python, it is an acronym for Numerical and Python and used for scientific programming in Python, particularly for data science, engineering, mathematics or science. [69]

Scikit-learn: For the Python programming language. Generally used in machine learning projects and offered different algorithms, it is created for the Python numerical and scientific libraries Numpy and SciPy. [70]

Keras: An open source-learning construct vastly used in the fields of artificial intelligence. It offers a first-class interface for neural networks written in Python that can be used with other widely used libraries for deep learning, such as TensorFlow, Microsoft Cognitive Toolkit (CNTK) or Theano. [71]

Matplotib: A complete library for producing Php visualizations called Matplotlib makes simple and difficult things possible. [71]

OS: A common Python library for communicating with the operating system is called OS; it offers functions to work with files and folders.

3.2. Data Pre-Processing Phase:

Data pre-processing is the more important step in data analysis and deep learning. It involves cleaning, transforming, and organizing raw data to ensure its quality and suitability for further analysis. This step includes:

Dataset loading and Split: To start the work, the dataset must be loaded, so the read_csv function of the pandas library is used to import data from CSV files, we split randomly the data in training, test and validation dataframes.

Binary Conversion: The “Replace” function is used to convert the labels ('BENIGN' and 'Anomaly') to binary values (0 and 1), to facilitate the analysis and prepare the data for training.

Category Conversion: The “Replace” function is used to group different label categories into eight more general categories in dataframes and thirty-four subcategories, to facilitate analysis.

Normalization: The “MinMaxScaler” function in the Scikit-learn library is used to put data on the same scale; we normalize numerical data between an interval (by default, between 0 and 1). Moreover, the “StandardScaler” function used to prepare the input data to be more acceptable for the process of the network training, which can lead to faster convergence and better performance.

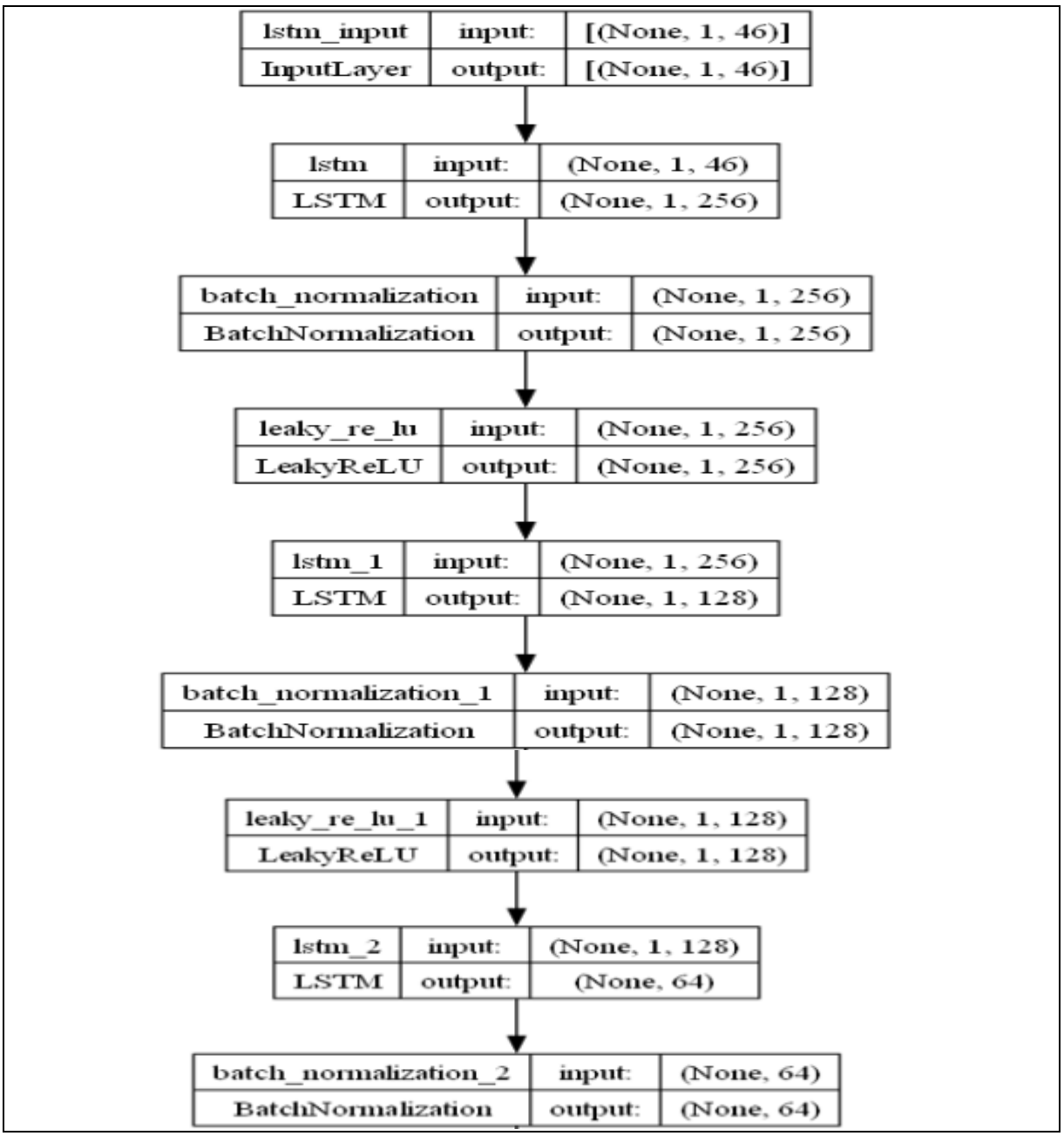
3.3. Learning phase

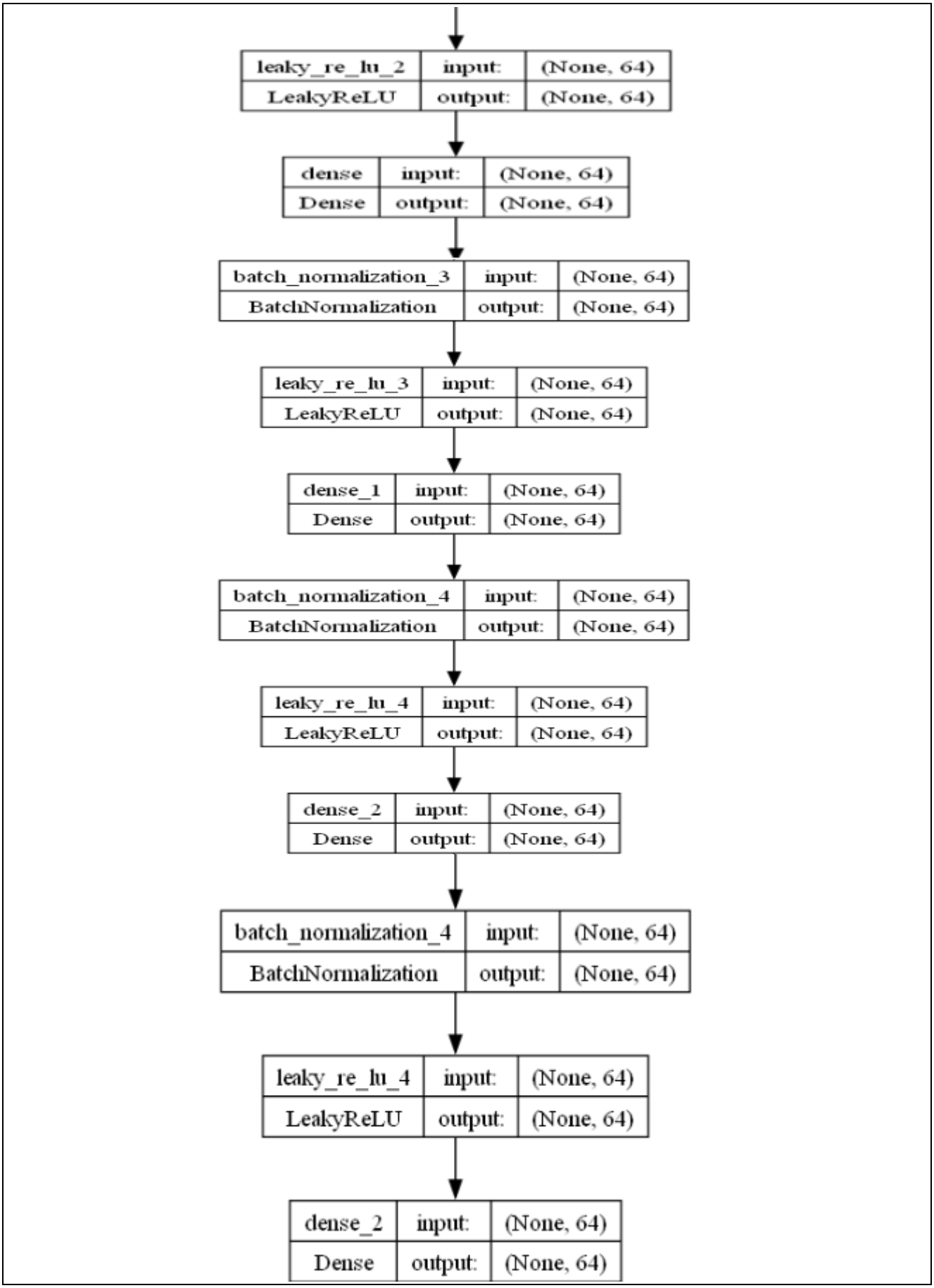
In this phase, we focus on implementing and training the LSTM model to detect anomalies in network patterns. Initially, we implemented a binary classification model with two classes: benign and malicious to distinguish normal and abnormal network traffic. We then extended it to include eight classes, covering the benign class and seven malicious classes, and further expanded it to encompass 34 sub-categories of the malicious classes. We assess the model's performance using accuracy, recall, F1 score and precision to determine its effectiveness in differentiating normal and abnormal network behavior.

3.3.1. LSTM Architecture

The architecture defines a sequential model using Keras for a classification task with three distinct methods. The model starts with an LSTM layer with 256 units and returns the sequences, followed by batch normalization and Leaky ReLU activation. Then, a second LSTM layer with 128 units is added, also accompanied by batch normalization and Leaky ReLU activation. A third LSTM layer with 64 units is then incorporated, followed by normalization and Leaky ReLU activation. The model also includes several dense layers with 64 units each, each followed by batch normalization and Leaky ReLU activation.

Finally, the output layer varies according to the classification method used: for the first method, it is adapted for two classes (binary); for the second method, it manages eight classes; and for the third method, it is designed for 34 classes. This flexibility allows the model to adapt to different classification needs by simply modifying the last output layer. The next figure shows the model architecture:




leaky_re_lu_2 | input: | (None, 64)
LeakyReLU | output: | (None, 64)
dense | input: | (None, 64)
Dense | output: | (None, 64)
batch_normalization_3 | input: | (None, 64)
BatchNormalization | output: | (None, 64)
leaky_re_lu_3 | input: | (None, 64)
LeakyReLU | output: | (None, 64)
dense_1 | input: | (None, 64)
Dense | output: | (None, 64)
batch_normalization_4 | input: | (None, 64)
BatchNormalization | output: | (None, 64)
leaky_re_lu_4 | input: | (None, 64)
LeakyReLU | output: | (None, 64)
dense_2 | input: | (None, 64)
Dense | output: | (None, 64)
batch_normalization_4 | input: | (None, 64)
BatchNormalization | output: | (None, 64)
leaky_re_lu_4 | input: | (None, 64)
LeakyReLU | output: | (None, 64)
dense_2 | input: | (None, 64)
Dense | output: | (None, 64)

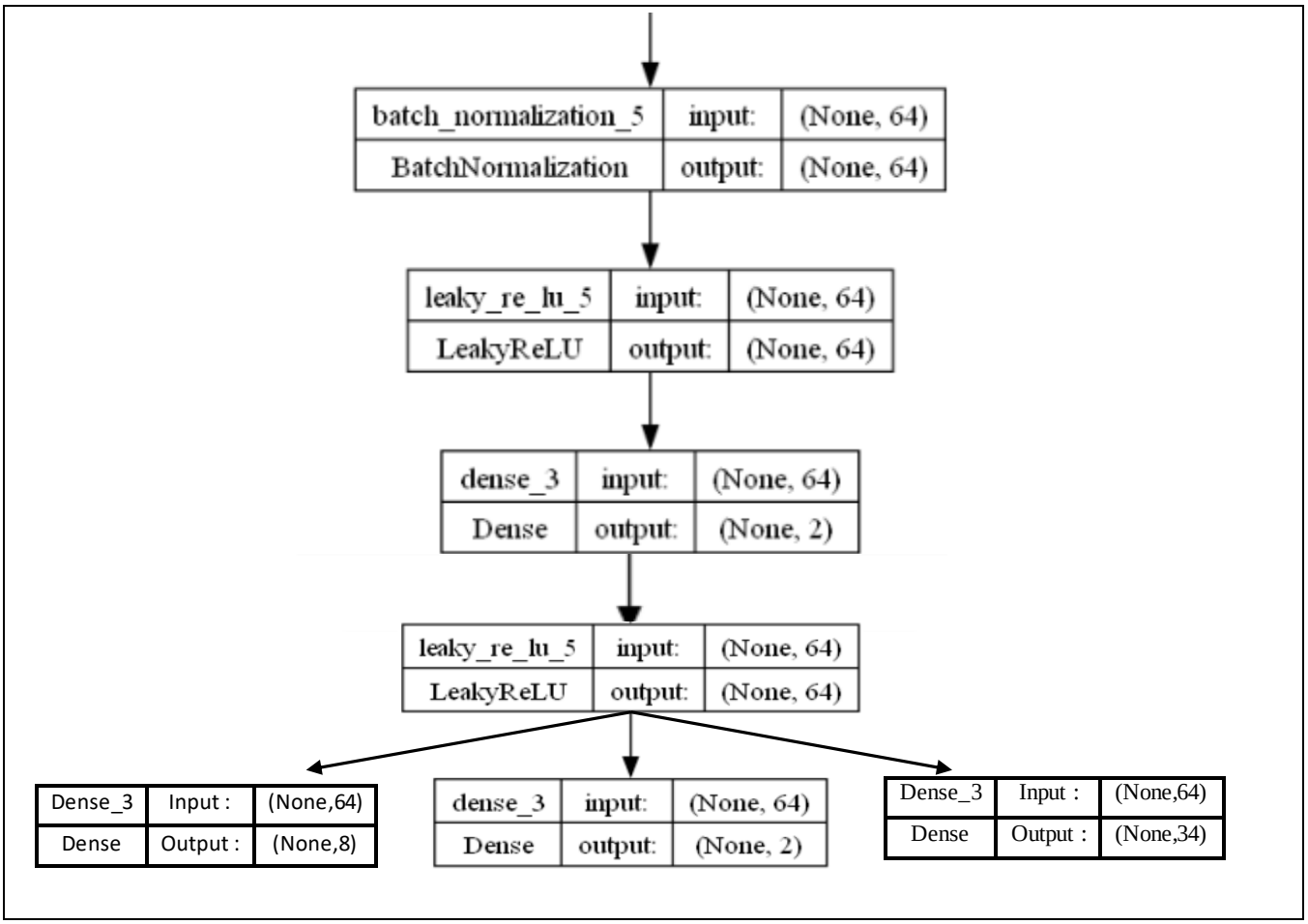


Figure 16: The LSTM Architecture for different Classes

3.4. Evaluation Phase

3.4.1. Performance Evaluation

Accuracy: This metric represent the proportion of correct classifications compared to all predictions made by the model. It evaluates the classification models by depicting the proportion of correct predictions in a given dataset, as per the following expression. [74]

$$\text{Accuracy} = (TN+TP) / (TP+TN+ FP+FN) \quad (10)$$

Recall: Also known as sensitivity or true positive rate, it measures the ratio of correctly identified positive instances to the total number of actual positive instances. The question that it answers: what proportion of actual positive results was identified correctly? [75] Mathematically is defined as:

$$\text{Recall} = TP/ (TP+FN). \quad (11)$$

Precision: it quantifies the ratio of correctly identified positive instances to the total number of instances classified as positive by the model by answering the following question: What proportion of positive identifications was actually correct?, and it can be defined as follows: [75]

$$\text{Precision} = TP/ (TP+FP) \quad (12)$$

F1 score: serves as a comprehensive measure of a model's performance, combining both precision and recall. A score close to 1.0 indicates that the model effectively classifies different categories in a precise and balanced manner. [74]

$$\text{F1 score} = 2(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall}) \quad (13)$$

Where:

TP (True Positives): The number of positive instances that are correctly predicted as positive by the model.

TN (True Negatives): The number of negative instances that are correctly predicted as negative by the model.

FP (False Positives): The number of negative instances that are incorrectly predicted as positive by the model.

FN (False Negatives): The number of positive instances that are incorrectly predicted as negative by the model

These metrics collectively provide insights into the performance of the deep learning models under evaluation, facilitating informed decision-making regarding their suitability for specific tasks or applications.

3.4.2. Loss Function Evaluation

Cross-Entropy Loss Function: In classification tasks, we work with probability predictions for categorization purposes. This means that a neural network's output must fall between 0 and 1. The cross-entropy loss function is a type of loss function that is able to calculate the error between a label representing the actual class and a predicted probability. On the other hand, the forecast $\mathbf{y}$ can have continuous values between 0 and 1. The cross-entropy loss between the ground truth vector $\hat{\mathbf{y}}$ and the prediction vector $\mathbf{y}$ can be calculated as follows in the next equation: [77]

$$L(\theta) = -\sum_{i=0}^{N} \hat{\mathbf{y}}_i . \log(\boldsymbol{y}_i) \quad (14)$$

Where:

$\hat{y}_i$: Entries in the ground truth label.

y_i: Entries in the Prediction vector.

N: The total number of instances.

4. Conclusion

In this chapter, we have given a thorough description of our implementation approach, including the architecture of the model we have used, and we have clarified which particular layers and parameters are included. Moreover, the different metrics used to judge and measure our model's performance and effectiveness have been identified.

Chapter 5
Comparative Results

1. Introduction

In this chapter, we will delve into the comparative analysis of two distinct models, our LSTM model and DNN deep learning model using CICIoT2023 dataset. By this comparison, we aim to evaluate the performance of these two models and the efficacy of intrusion detection methodologies across different network environments.

2. Model Evaluations

In the evaluation process, we have compared DNN and LSTM models, which have been efficiently used in various domains especially in the cybersecurity. Each model was applied to each classification types (binary and multi-class). We prepared the data to be adapted then we built the LSTM model. After the results, we have compared it with DNN model, the same dataset have been used to ensure a fair comparison.

Table 2 shows the performance results of these two models with different classification methods.

Table 3: Comparison Models performance using recorded metrics

	Metrics	**LSTM**	**DNN[41]**
2 classes	Accuracy	0.990705947	0.994422814
	Precision	0.858798652	0.947579486
	Recall	0.993836165	0.933277496
	F1 Score	0.914920945	0.940305998
8 classes	Accuracy	0,984299330	0.991147043
	Precision	0,678296950	0.679434746
	Recall	0,769432183	0.906642708
	F1 Score	0.669385722	0.69726491
34 classes	Accuracy	0.978476165	0.986118011
	Precision	0.652028624	0.665295126
	Recall	0.730245200	0.730245200
	F1 Score	0.657618742	0.672346883

2.1. Analysis of LSTM Model Performance

In this study, we used accuracy, precision, recall, and F1 score as measures to assess how well a Long Short-Term Memory (LSTM) model performed on datasets with different numbers of classes. The LSTM model demonstrated strong reliability and consistency in binary classification (two classes) with an accuracy of 99.07% and an F1 score of 91.49%. The model also demonstrated a solid balance between precision (85.87%) and recall (99.38%).

The model's accuracy was 98.42% when it was used to classify eight classes; its precision was 67.82%, recall was 76.94%, and its F1 score was 65.76%. The model achieved an accuracy of 97.84%, precision of 65.20%, recall of 73.02%, and F1 score of 65.76% for a more complicated thirty-four-class categorization. These outcomes demonstrate how well the model performs at various classification levels.

2.2. Models Comparison and Discussion

In comparing the LSTM and DNN models across all three types of classification tasks and various metrics (accuracy, precision, recall, and F1 score), DNN consistently outperforms LSTM in terms of accuracy and precision.

In binary classification, DNN shows significantly higher precision and slightly better overall metrics. In multi-class classification tasks (8 classes and 34 classes), DNN continues to exhibit higher precision and generally achieves better overall metrics compared to LSTM. However, LSTM performs competitively in recall across these tasks, especially in the multi-class scenarios.

Overall, DNN model excels the LSTM model in situations where accurately recognizing positive occurrences is critical, while the LSTM model performs better in scenarios where correctly identifying positive instances is crucial due to its greater precision and recall.

3. Experimental Results

This section presents the results obtained from the LSTM model evaluation for three different types of classification according to training and validation accuracy during epochs in the form of curves.

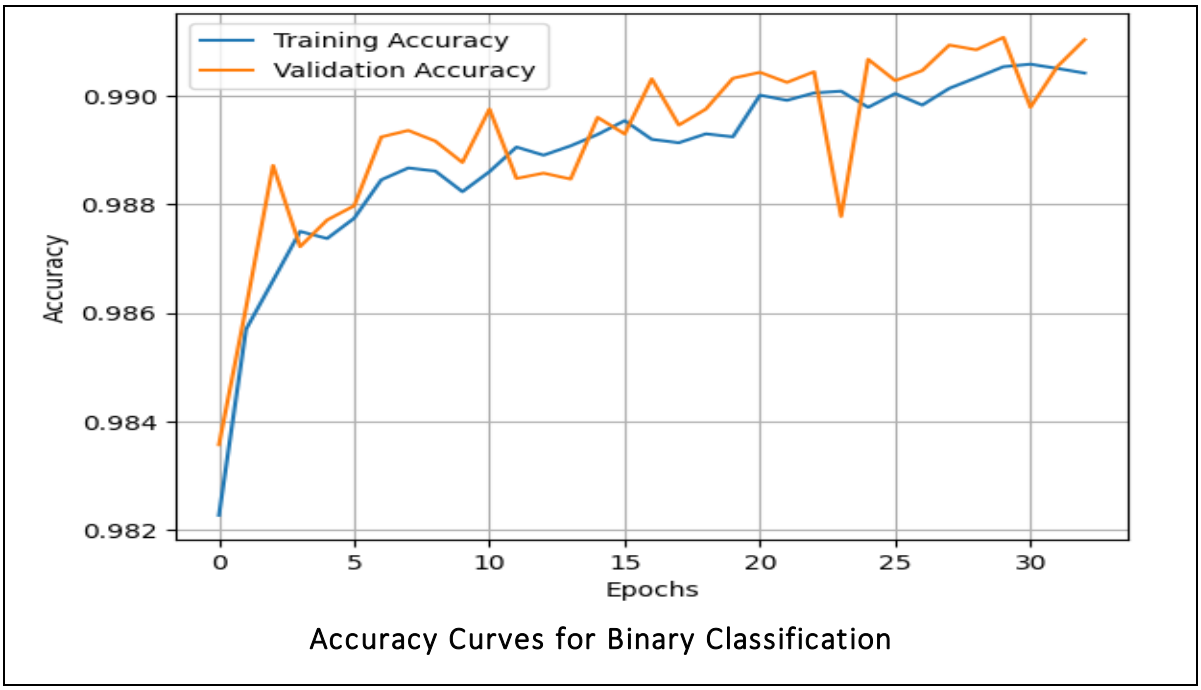


Figure 17: Accuracy Curves for Binary Classification

In the first methods used, the accuracy curve shows an initial increase from 0.9823 at epoch 0 to 0.9857 at epoch 1, then a continuous progression stabilizing around 0.9905 towards the end. After the 20's, the accuracy remains stable around 0.990 with minor variations. The validation curve follows a similar trend, starting at 0.9836 at epoch 0 and increasing to 0.9861 at epoch 1, with less variability thereafter. It stabilizes around 0.990-0.991 towards the end, with a peak at 0.9911 at the time 32.

This stabilization indicates that the model learns well without significant overlearning, achieving optimal performance.

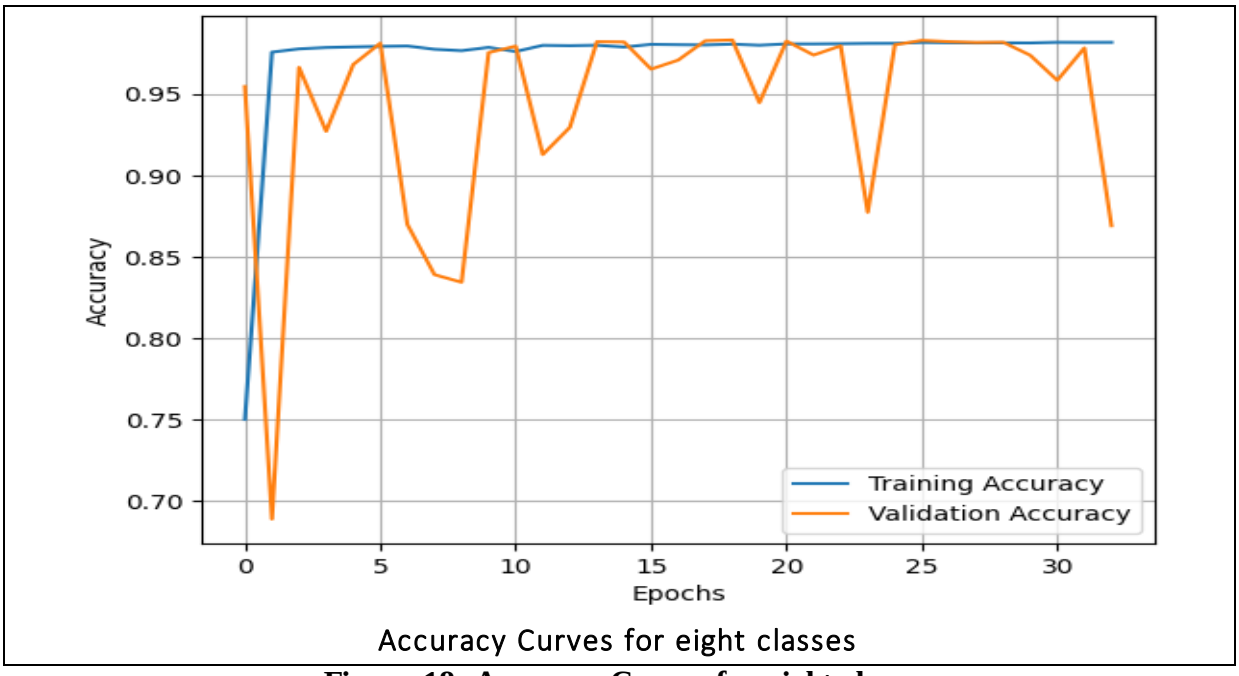


Figure 18: Accuracy Curves for eight classes

In the second method of eight classes, the accuracy curve shows an initial increase from 0.75 at epoch 0 to 0.9857 at epoch 1, then a continuous progression stabilizing around 0.9905 towards the end with minor variations. The validation curve starts decreases rapidly from 0,95 at epoch 0 to 0,65 at epoch 1. It then increases directly to 0.99 after second epoch and seeing an instability throughout the epochs between 0.84 to 0.99.This instability indicates that the model learns less compared to the first mode.

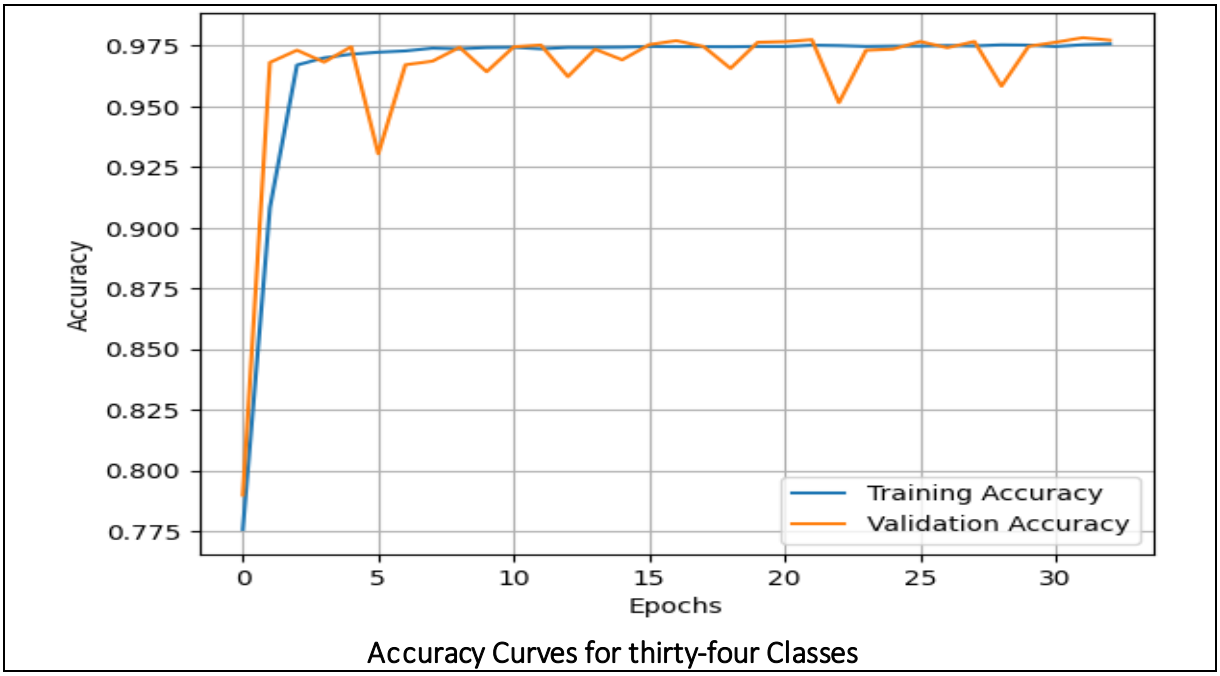


Figure 19: Accuracy Curves for thirty-four classes

The last method of 34 classes, the accuracy starts from 0.7757 at epoch 0, increases rapidly to 0.96 at epoch 1, and then stabilizes around 0.975 after epoch 16. The validation curve follows a similar trend, reaching 0.9681 at epoch 1, but shows more variability. After epoch 5, validation drops to 0.9305 then rises, with notable peaks, reaching a maximum of 0.9783 at epoch 32. These trends indicate effective learning initially, followed by potential over-learning, before performance stabilization.

4. Conclusion

In conclusion, the evaluation and performance analysis of our model demonstrate its effectiveness in anomaly detection and the comparison between LSTM and DNN deep learning models highlights the strengths and weaknesses of each approach, emphasizing the importance and the impact of model selection on network security outcomes.

General Conclusion

In conclusion, this thesis delves into the critical realm of anomaly detection in network traffic, recognizing its pivotal role in safeguarding the integrity and security of digital communication networks. As traditional methods falter in coping with the escalating complexity and volume of network data, the exploration of advanced techniques, particularly leveraging deep learning, emerges as imperative. Through the lens of Long Short-Term Memory (LSTM) networks, this research endeavors to overcome the limitations of conventional approaches and enhance the accuracy and timeliness of anomaly detection.

By delineating the challenges, elucidating the fundamentals of deep learning, describing the dataset, and presenting implementation details and experimental findings, this thesis strives to contribute to the evolving landscape of network security.

Looking forward, future efforts should focus on developing AI-driven anomaly detection systems that not only identify anomalies but also provide actionable insights and explanations to cybersecurity analysts. Moreover, expanding hybrid methods that combine LSTM with complementary models holds promise for achieving even greater performance gains. Ultimately, the culmination of this endeavor aims to furnish insights, methodologies, and advancements that fortify the resilience of network infrastructures against evolving cyber threats.

References

[1] P. Pecha, Science-of-network-anomalies, 14 MAY, 2021.

[2] Cisco Systems. "Local Area Network (LAN) Technologies and Management". Cisco Networking Academy, 2015.

[3] A. S. Tanenbaum et Wetherall, D. J. , "Computer Networks". Pearson., 2011.

[4] J. Nazario, "Defense and Detection Strategies against Internet Worms". Artech House., 2009.

[5] D. Z. C. &. L. W. Dagon, "Modeling Botnet Propagation Using Time Zones". Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS), 2006.

[6] A long short-term memory based approach fordetecting cyber attacks in IoT using CIC-IoT2023dataset, Akinul Islam Jony, Arjun Kumar Bose Arnob, 2024..

[7] Exploring the Use of Data-Driven Approaches for Anomaly Detection in the Internet of Things (IoT) Environment, Eleonora Achiluzzi, Menglu Li, Md Fahd Al Georgy, and Rasha Kashef, 2022.

[8] G. S. Madhuri, Statistical Approaches to detect Anomalies, 09 2020..

[9] K. VIPIN , A. BANERJEE, V, CHANDOLA , Anomaly Detection : A Survey, United states, 2009..

[10] I. B. A. A. P. L. Mustafa Al Samara, A Survey of Outlier Detection Techniques in IoT: Review and Classification, 2023.

[11] K. Hewelt, Data cleaning techniques: Strategies for reliable data analysis, 2024.

[12] A. A. Ghorbani, W. Lu, and M. Tavallaee, "Network Intrusion Detection and Prevention," 2010..

[13] M. Ahmed, A. Mahmood, and J. Hu, "A Survey of Network Anomaly Detection Techniques," Nov. 2015.

[14] S. Sigari, "Unlocking Cyber Security and Threat Intelligence: The Powerful Art of Data Science in Safeguarding Organizations," Mar. 12, 2023.

[15] R. U. Rehman, "Intrusion Detection Systems with Snort: Advanced IDS Techniques Using Snort, Apache, MySQL, PHP, and ACID.".

[16] O. Bonaventure, "Computer Networking: Principles, Protocols and Practice," Oct. 30, 2011.

[17] Chandola, V., Banerjee, A., & Kumar, V. (2009). "Anomaly detection: A survey". ACM Computing Surveys (CSUR)..

[18] S. A. Suman Kalyan Adari, Beginning Anomaly Detection Using Python-Based Deep LearImplement Anomaly Detection Applications with Keras and PyTorch,, 2024.

[19] Détection des anomalies des séries temporelles, 2024.

References

[20] Grubbs, F. E. (1969). "Procedures for detecting outlying observations in samples". Technometrics, 11(1), 1-21.

[21] F. Angiulli and C. Pizzuti, “Fast outlier detection in high dimensional spaces,” in European Conference on Principles of Data Mining and Knowledge Discovery. Springer, 2002, pp. 15–27.

[22] MacQueen, J. (1967). "Some Methods for classification and Analysis of Multivariate Observations". In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability.

[23] Cortes, C., & Vapnik, V. (1995). "Support-vector networks". Machine Learning.

[24] Liu, F. T., Ting, K. M., & Zhou, Z. (2008). "Isolation forest". In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining.

[25] Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). "A Critical Review of Recurrent Neural Networks for Sequence Learning".

[26] Dave Bergmann, Cole Stryker, What is an autoencoder?,23 November 2023.

[27] w, I., Bengio, Y., & Courville, A. (2016). "Deep Learning". MIT Press.

[28] J. Moolayil, Learn Keras for Deep Neural Networks, Canada, 2019.

[29] J. Chen, What Is a Neural Network?, Investopedia, Feb 2024.

[30] What is artificial intelligence (AI)?, APRIL 2024.

[31] m. A, I, YAKOUB et A, OUHAB, La détection de Covid-19 par l'apprentissage, ADRAR: UNIVERSITE AHMED DRAIA- ADRAR, . , 2021.

[32] S. Alla et Suman Kalyan Adari, «Beginning Anomaly Detection Using Python-Based Deep Learning,» 2019. [En ligne]. Available: https://link.springer.com/book/10.1007/978-1-4842-5177-5. [Accès le 20 Avril 2024]..

[33] A. S et S,K, ADARI, Beginning Anomaly Detection Using Python-Based Deep Learning,..

[34] «Qu'est ce que le deep learning,» OCI, 2024..

[35] S. Madhavan et M. Tim Jones, Deep learning architectures, IBM Developer, 24 January 2021 .

[36] W. S. MCCULLOCH et WALTER, PITTS, "A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY", Chicago,USA: Bulletin of Mathematical Biophysics..

[37] S. Alla et Suman Kalyan Adari, «Beginning Anomaly Detection Using Python-Based Deep Learning,» 2019. [En ligne]. Available: https://link.springer.com/book/10.1007/978-1-4842-5177-5. [Accès le 20 Avril 2024]..

References

[38] S. Saxena, «What is LSTM? Introduction to Long Short-Term Memory, Analytics vidhya,» JAN 2014. [En ligne]. [Accès le AVRIL 2024]..

[39] J. Z. a. X. He, «NTAM-LSTM models of network traffic prediction,» 2022. [En ligne]..

[40] Yan Tian, Kaili Zhang, Jianyuan, Xianxuan, Bailin Yang LSTM-based traffic flow prediction with missing data, Nov 2018.

[41] E. C. P. Neto, Sajjad Dadkhah , Raphael Ferreira,, Alireza Zohourian,, Rongxing Lu et Ali A. Ghorban, CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment, Switzerland.: Antonio Puliafito, 26 June 2023.

[42] «WizzDev,» 10 June 2022. [En ligne]. Available: https://wizzdev.com/blog/overview-of-iot-network-topologies/. [Accès le 23 April 2024].

[43] A long short-term memory based approach fordetecting cyber attacks in IoT using CIC-IoT2023dataset, Akinul Islam Jony, Arjun Kumar Bose Arnob, 2024.

[44] E. C. P. Neto, Sajjad Dadkhah , Raphael Ferreira,, Alireza Zohourian,, Rongxing Lu et Ali A. Ghorban, CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment, Switzerland.: Antonio Puliafito, 26 June 2023.

[45] Lamping, U.; Warnicke, E. Wireshark user's guide. Interface 2004, 4, 1..

[46] Kumari, P.; Jain, A.K. A Comprehensive Study of DDoS Attacks over IoT Network and Their Countermeasures. Comput. Secur. 2023, 127, 103096. [CrossRef].

[47] Duravkin, I.; Loktionova, A.; Carlsson, A. Method of slow-attack detection. In Proceedings of the 2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology, Kharkov, Ukraine, 14–17 October 2014.

[48] Harshita, H. Detection and prevention of ICMP flood DDOS attack. Int. J. New Technol.

[49] Acharya, A.A.; Arpitha, K.; Kumar, B. An intrusion detection system against UDP flood attack and ping of death attack (DDOS) in MANET. Int. J. Eng. Technol. (IJET) 2016.

[50] Cebeloglu, F.S.; Karakose, M. A cyber security analysis used for unmanned aerial vehicles in the smart city. In Proceedings of the 2019 1st International Informatics and Software Engineering Conference (UBMYK), Ankara, Turkey, November 2019.

[51] Chen, E.Y. Detecting TCP-based DDoS attacks by linear regression analysis. In Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, Athens, Greece, 21 December 2005.

[52] Kaufman, C.; Perlman, R.; Sommerfeld, B. DoS protection for UDP-based protocols. In Proceedings of the 10th ACM Conference on Computer and Communications Security, Washington, DC, USA, 27–30 October 2003.

[53] Gilad, Y.; Herzberg, A. Fragmentation considered vulnerable. ACM Trans. Inf. Syst. Secur. (TISSEC) 2013, 15, 1–31. [CrossRef].

References

[54] Bogdanoski, M.; Suminoski, T.; Risteski, A. Analysis of the SYN flood DoS attack. Int. J. Comput. Netw. Inf. Secur. (IJCNIS) 2013, 5, 1–11. [CrossRef].

[55] Raptis, G.E.; Katsini, C.; Alexakos, C. Towards Automated Matching of Cyber Threat Intelligence Reports based on Cluster Analysis in an Internet-of-Vehicles Environment. In Proceedings of the 2021 IEEE International Conference on Cyber Security and Resili.

[56] van der Merwe, J.R.; Zubizarreta, X.; Lukˇcin, I.; Rügamer, A.; Felber, W. Classification of spoofing attack types. In Proceedings of.

[57] Al-Jarrah, O.; Arafat, A. Network intrusion detection system using neural network classification of attack behavior. J. Adv. Inf. Technol. 2015, 6. [CrossRef.

[58] Orebaugh, A.; Pinkard, B. Nmap in the Enterprise: Your Guide to Network Scanning; Elsevier: Amsterdam, The Netherlands, 2011.

[59] deRito, C.; Bhatia, S. Comparative Analysis of Open-Source Vulnerability Scanners for IoT Devices. In Intelligent Data Communication Technologies and Internet of Things; Springer: Singapore, 2022; pp. 785–800.

[60] Bhuyan, M.H.; Bhattacharyya, D.K.; Kalita, J.K. Surveying port scans and their detection methodologies. Comput. J. 2011, 54, 1565–1581. [CrossRef].

[61] Wolfgang, M. Host Discovery with nmap. Explor. Nmap's Default Behav. 2002.

[62] S. Yaras et Murat Dener, IoT-Based Intrusion Detection System Using New Hybrid Deep Learning Algorithm, March 2024.

[63] April 2024. [En ligne]. Available: https://docs.anaconda.com/free/miniconda/index.html..

[64] «Wikipédia,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Visual_Studio_Code. [Accès le APRIL 2024].

[65] Afafe Lahreche, Deep learning for Arabic letters recognition,, 2019..

[66] C. Huyen, "TensorFlow for Deep Learning Research", Prepared..

[67] O. G. Yalçın, The Important Deep Learning Libraries to Look out for in 2021 and Why TensorFlow Should be Your Choice, Jan 2021.

[68] «W3school,» [En ligne]. Available: https://www.w3schools.com/python/pandas/pandas_intro.asp. [Accès le APRIL 2024]..

[69] NumPy : the most used Python library in Data Science, March 2023.

[70] H. e. a. 2. Hussain, Specht and Lee 2003,, Mirkovic and Reiher 2004,, Farraposo, Boudaoud et al. 2005,, HAMOUDA 2020, , Cheng, Xu et al. 2022, , Driss, Almomani et a. 2022, , Hazratifard, Gebali et al. 2022, , Luo 2022, Yang and Shami 2022 et Cheng, P., e.

[71] K. M. Azim et Dahdi Abderrezzak, Système de Détection d'Intrusion dans Les Réseaux D'Internet des Véhicules, 2023..

References

[72] Shafique, K.; Khawaja, B.A.; Sabir, F.; Qazi, S.; Mustaqim, M. Internet of things (IoT) for next-generation smart systems: A review.

[73] Neto, E.C.P.; Dadkhah, S.; Ghorbani, A.A. Collaborative DDoS Detection in Distributed Multi-Tenant IoT using Federated.

[74] Mhammedi Ayemn, Yakoub Imane, "La détection de Covid-19 par l'apprentissage profonde (Deep learning)", 2020/2021.

[75] Classification: précision et rappel, lien:https://developers.google.com/machinelearning/crash-course/classification/precision-and-recall?hl=fr

[76] Bharath K, 'introduction to Deep Neural Networks', Jul 2023

[77] Loss Functions in Deep Learning | MLearning.ai, Artem Opermann,2021

[78] Cheng, P., et al. (2022). "TCAN-IDS: intrusion detection system for internet of vehicle using temporal convolutional attention network