People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University May 8, 1945 guelma

Faculty of Mathematics, Computer Science and Material Sciences

computer science department

Master's degree thesis

Branch : Computer Science

Option : Informatic Systems

Theme

The detection of Botnet attacks in the Industrial Internet of Things (IIoT)

---

supervised by :

Dr. Mohamed Amine Ferrag

Presented by :

Nouar Ikrame

July 2022

Acknowledgments

Alhamdulillah who made it possible for me. Alhamdulillah, who enabled me to do this and gave me the ability. Then, I extend my highest expressions of respect and appreciation to my supervisor for leading me and driving me to this point, for the tips, and the information that he shared with us, thanks for Djallel Hamouda either for the help and all what he did for me.

My little family no words can be enough to say thank you for all the support, love and power that you gave me to continue my journey to the end, Father, Mother, Sisters and Brothers, thank you.

My second family who shared my happiness, weakness, and all the moments in this trip, who supported me, laughed and cried with me, my friends : Ines, Abdou.. My new friends that i have known recently but as if i have known you for too long, I met you by chance, and perhaps a coincidence is better than a thousand dates, Amel, Houssem, Tarek, Hamza, belkhir, and all JV-GAS team thank you for the amazing experience that you offered to me. at the end i want to say thanks you to my self for not giving up and staying strong.

RÉSUMÉ

L'Internet Industriel des Objets (IIoT) est un ensemble de périphériques interconnectés quotidiens dans un environnement industriel, dotés de processeurs légers et de cartes réseau, pouvant être gérés par des services web et/ou d'autres types d'interface. Les vulnérabilités des réseaux IIoT augmentent considérablement avec les cyberattaques complexes, telles que les botnets. Une attaque de botnet est une cyberattaque à grande échelle menée par des appareils infectés par des logiciels malveillants et contrôlés à distance. Elle transforme les appareils compromis en "robots zombies" pour le contrôleur du botnet. Un botnet peut lancer un certain nombre d'activités, telles que des attaques par déni de service distribué (DDoS), l'enregistrement de frappe, le phishing, le spamming, la fraude au clic, l'usurpation d'identité, ... etc. L'objectif de ce sujet est d'étudier et de proposer une méthode de sécurité pour détecter les attaques Botnet dans les systèmes IIoT.

Mots Clés : Cybersécurité, Apprentissage en profondeur, Botnet, Système de détection de botnet, Internet des objects industriel (IIoT)

ABSTRACT

The Industrial Internet of Things (IIoT) is a collection of daily interconnected devices in an industrial environment, equipped with light processors and network cards, which can be managed by web services and/or other types of interface. IIoT network vulnerabilities increase dramatically with complex cyberattacks, such as botnets. A botnet attack is a large-scale cyberattack carried out by remotely controlled, malware-infected devices. It turns compromised devices into "zombie robots" for the botnet controller. A botnet can initiate a number of activities, such as Distributed Denial of Service (DDoS) attacks, keylogging, phishing, spamming, click fraud, spoofing, etc. . The objective of this subject is to study and propose a security method to detect Botnet attacks in IIoT systems.

Keywords :Cyber Security, Deep learning, Botnet, Botnet detection system, Industrial Internet of things (IIoT)

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

GENERAL INTRODUCTION :

Enormous development of information and communication technologies currently offers essential facilities in terms of file transfer, messaging and many other forms of information exchange. This pushes the market to choose a new destination which is the internet of things (IOT) and industrial internet of things(IIOT). This field helps the economic progression and development. Those new axis opens a new door to cyber-attacks that threaten the stability of systems and businesses. The the vulnerability of systems is a real danger, and may cause a real critical situation. With the evolution of the strength and complexity of cyberattacks, the traditional methods are no more function, this is why the new researches focus on intrusion detection systems with some creative methods like Deep Learning methods a field of artificial intelligence (AI), in purpose to strengthen the detection systems.

The problem :

With the huge progress in the world of technologies and the the vulnerability of systems, the intrusion detection systems faces powerfull threats.
How to ensure the effectiveness of those systems against those malwares, and How to ensure that the used methods are effective and valid.

Proposed Solution :

For this reasons we are trying to study the existing works and propose a new security method based on Deep Learning techniques to detect Botnet attacks in IIoT systems, We have use a new data-set, which represents traffic real network containing several types of attacks,in simple definition our work represent in making a security network that stops hackers in their tracks, additionally make an easy system that help the administration to detect the attacks and to defend their networks.

CHAPITRE 1
STATE OF THE ART

1.1 Introduction

After the explosion in the technology market and the new sector that appears, the Internet of Things(IOT) and Industrial Internet of Things (IIOT) are considered the new revolution in the field. In addition the Deep Learning is another level of artificial intelligence that we need to know about. To fully understand, we will have the necessary information in this chapter.

1.2 Internet of Things & Industrial Internet of Things

1.2.1 Internet of things

Internet of Things (IoT) consists of a group of connected devices that communicate with each other and with the user by exchanging information and data via the internet [7]. The IoT intelligence system, its flexibility in application, and expansion represent the main reasons for the immense growth of IoT in later a long time, as shown in Fig. 1.1. The effect of IoT has come to the most noteworthy levels on the concepts of applications in human activities. It is expected that the trend in this field

will be greater in the coming years, as its economic impact may reach between 900 billion US dollars and 2.3 trillion dollars annually until 2025. IoT applications are seeing a wide spread at the level of all segments, fabricating, farming, wellbeing, security, cities, buildings, and environment, among others. All these applications' fields have common fundamental characteristics : various intelligent devices that are interacting and coordinating to reach an objective using wired and wireless connections [32].

1.2.2 Industrial Internet of Things :

Industrial Internet of Things (IIoT), also called industry 4.0, is a category of IoT. As shown in Fig. 1.2 IIoT is used as a new IoT application paradigm in industrial systems to maximize the efficiency of business processes, Quality of Service (QoS), and reliability [14].

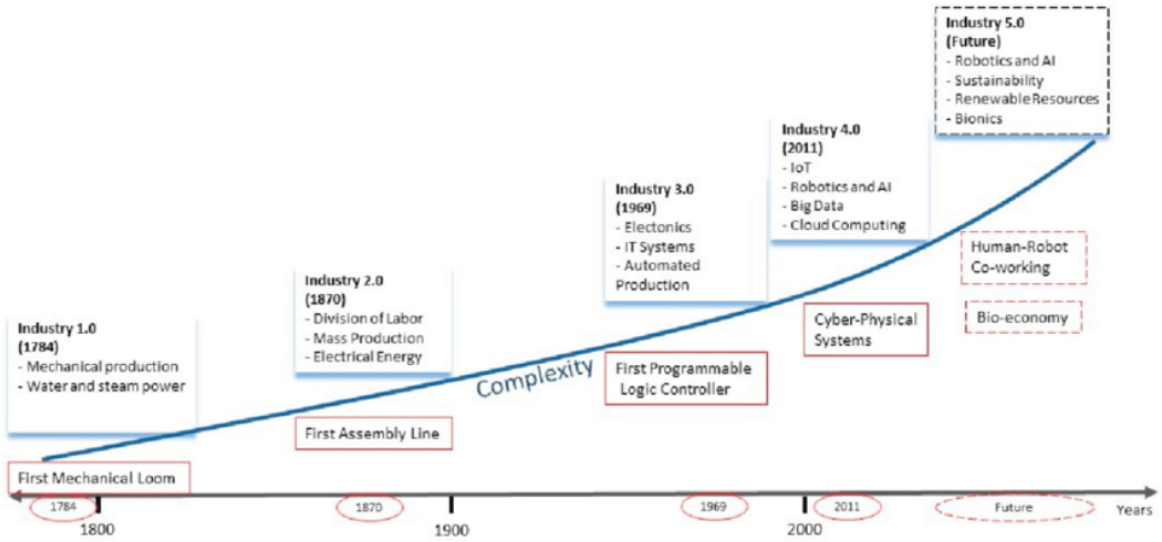


FIGURE 1.1 – Revolution of smart industries[28].

However, IIoT confronts challenges that make it particular and distinct from IoT's systems and services. The particularity resides in the need to integrate specific components such as data acquisition systems (SCADA), supervisory control, and programmable logic controllers (PLC). PLC and SCADA systems are interconnected with an

industrial network to build the infrastructure of Operational Technology (OT). The latter focuses on the fields of energy production facilities, industrial floor, and energy distribution networks, with the related strict requirements such as safety, continuous operation, real-time operation, etc. The potentialities of IIoT technology impose integration challenges of these OT systems with the classical enterprise IT systems at many levels, from enterprise management to cybersecurity [32]. Industrial technology is not limited to manufacturing or factories. The capabilities of the technology and its cyber-physical control maturity extend its utilization farther of the traditional factory environments and achieve the constitution of a significant part of the critical infrastructure on many fronts such as distribution infrastructure and energy production including OT systems, which are the indispensable infrastructure on which modern smart grids are built. Nowadays, the energy sector is the primary accelerator in the evolution of IIoT, besides the consumers' increasing needs for energy, energy management represents a critical aspect in the industrial field and the necessity for low-cost production of goods and services. Industrial systems are used in many other vital infrastructure areas, such as water management and transportation, in addition to the energy industry. In 2012, the General Electric company announced the Industrial Internet term, as a top head of the Industrial Internet of Things. It also introduced the set of the main constituents of the IIoT vision, such as machine-to-machine communication technology, industrial data analytics, SCADA, cybersecurity, and HMI, to cite a few. Further, their studies demonstrate the influence of the Industrial Internet to 46% on the global economy, while its impact in the energy sector reaches 100% on the production of the energy and 44% on energy consumption globally [32]. we can see the different IIoT generation presented in 1.1

1.2.3 IIoT Architecture

Principally, to understand new applications and identify the problems and challenges they face, a reference architecture framework becomes a fundamental stage. The IIoT architecture design must focus on extensibility, scalability, modularity, and

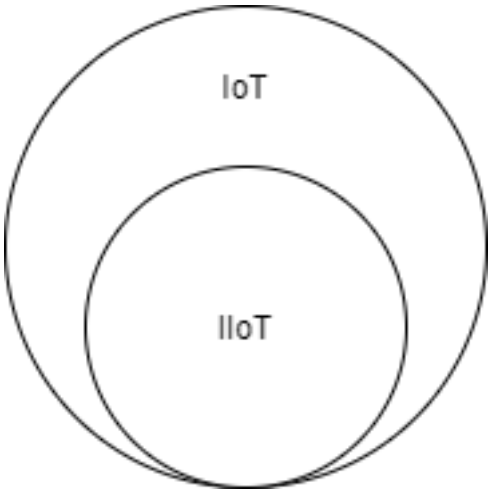


FIGURE 1.2 – Relationship between IOT and IIOT

interoperability among heterogeneous devices using different technologies. Numerous architecture frameworks came into sight in the last years in different application contexts for both IoT and IIoT. The typical approved approach is the multilayer description. The latter is organized around the services offered at each level, depending on the selected technologies, business needs, and technical requirements. For instance, the International Telecommunication Union (ITU) adopts the architecture of the Internet of Things, which consists of 5 layers : sensing, accessing, networking, middleware, and application layers. Several propositions were submitted to define the architecture framework. The majority proposed a layered architecture divided into three basic levels for IoT : perception layer (or sensing layer), network layer, and service layer (or application layer). Liu et al. proposed an IoT application infrastructure design that consists of a physical layer, transport layer, middleware layer, and applications layer. The inspiration for a four-layered architecture is derived from the perspective of offered functionalities, which include sensing, networking, service, and interface layers. However, the Reference Architectural Model Industries 4.0 (RAMI 4.0) targets the next-generation industrial manufacturing systems. It identifies a 3-D model whose axes are (i) the life-cycle & value stream, related to the product's life cycle, and (ii) the hierarchy levels, related to the different component functionalities.

Where the Hierarchy axis describes the IT representative, including the communication layer. As of late, the "Reference Architecture" white paper was released by the Industrial Internet Consortium. In specific, it focuses on diverse perspectives (business, usage, functional, and implementation views) and gives models for each one. The implementation viewpoint focused on the technologies and the framework components that are necessary for implementing the functionalities endorsed by the usage and functional perspectives. Besides the description of the IIoT system's general architecture (i.e. its structure, the distribution of components, and the topology by which they are interconnected) that it provides, it includes a description of interfaces and protocols as well. Generally speaking, the IIoT systems transfer two different kinds of information. This type of information is divided into whether the data need to be processed (data flow) or they are the results of some elaborations (control flow). Further, other architectural patterns are also specifying coherent system implementations and paving the way to innovative business models and services, usually in a multiple-tier arrangement, dictated by the very heterogeneous devices and networks. Where, the widely accepted three-tier patterns (edge, platform, and enterprise tiers) are connected by proximity, access, and service networks. The edge represents the domain in which IIoT components interact with each other. Therefore, it is composed of sensors, controllers, and actuators interconnected by independent local area networks (usually in the form of fieldbuses) to an edge gateway. The latter connects to the larger networks (access network) of the platform tier, providing global coverage. Last not least, the platform tier leverages the service network to set up links with the enterprise tier that implements domain-specific applications and provides end-user interfaces. Fig.1.3 illustrates the complexity of the IIoT hybrid architecture ; in particular, the increased latency and data aggregation of the different tiers is highlighted [33].

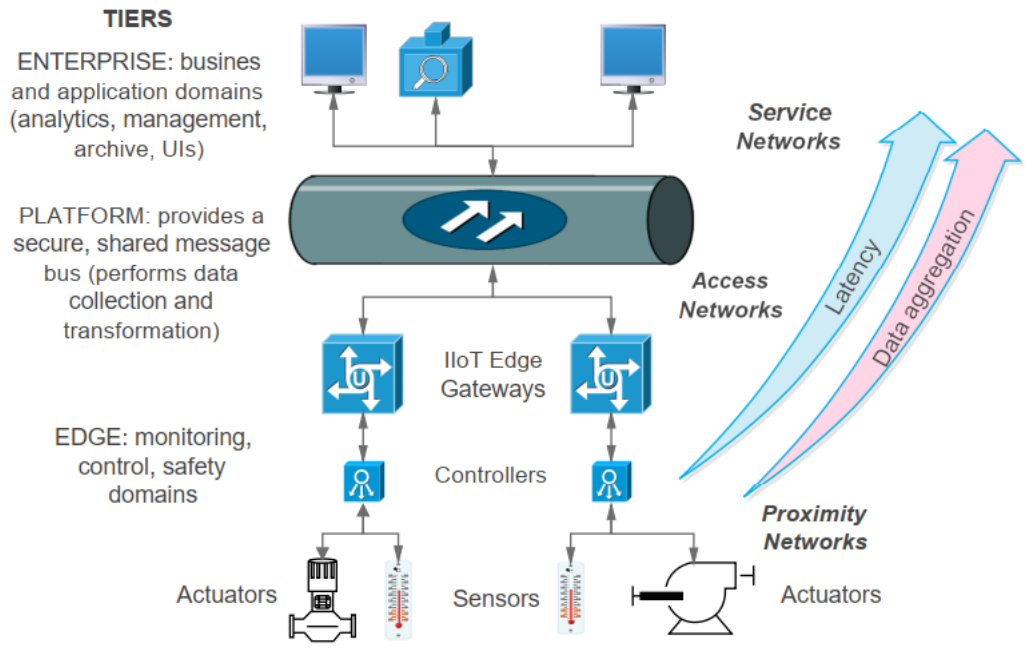


FIGURE 1.3 – The three-tier IIoT architecture.

1.2.4 Challenges in IIoT

Security :

Security is a challenge in IoT due to two characteristics : one is the devices in IoT are constrained, and the second is the scale. A device is constrained when it has limited processing power, memory, battery, and bandwidth. The constrained devices cannot protect themselves given their limited computing and battery capabilities. It also constrained the devices in terms of standard interfaces they expose for managing them. The constrained devices are an easy target for DDoS attacks and need external support[2].

Interoperability :

To address the device diversity in IoT, connected devices should follow the same language of protocols and encoding. This could be addressed by an interoperable IoT environment. Practically, it is complex in systems that have different layers with

communication protocols stacked between devices. Interoperability may also increase the economic value of the IoT market. The purchase of IoT products and services also depends on interoperability. Interoperability will be a significant consideration that enables connected devices to work in a better environment[2].

Real-time response

Some applications require real-time response, e.g., the ability to control the temperature of the boiler. And, IoT infrastructure is still evolving and is even more challenging in developing countries to be able to maintain connectivity to the cloud all the time. An IoT application should continue to function even if the connectivity to the cloud breaks[2].

Future readiness

IoT is still evolving and requires the current deployments should support future applications and new application service providers. The success of IoT deployments depends on the ability to allow innovation over the existing infrastructure. To overcome the above challenges and to enhance edge computing capability, context awareness is the key component that needs to be enabled with the use of machine learning algorithms [2].

1.2.5 IIOT vs IOT

Table 1.1 illustrates the difference between IIoT and IoT.

TABLE 1.1 – Major differences between IIoT and IoT[8].

	IIOT	IOT
01	Focuses on industrial applications such as manufacturing, power plants, oil & gas, etc.	Focuses on general applications ranging from wearables to robots & machines.
02	Uses critical equipment & devices connected over a network which will cause a life-threatening or other emergency situations on failure, therefore uses more sensitive and precise sensors.	Its implementation starts at a small scale level, so there is no need to worry about life-threatening situations.
03	Deals with large-scale networks.	Deals with small-scale networks.
04	Can be programmed remotely, i.e., offers remote on-site programming.	Offers easy off-site programming.
05	Handles data ranging from medium to high.	Handles a very high volume of data.
06	Requires robust security to protect the data.	Requires identity and privacy.
07	Needs stringent requirements.	Needs moderate requirements.
08	Has a very long life cycle.	Has a short product life cycle.
09	Has high- reliability.	Less reliable.

1.3 Botnet attacks

The wide spread of the Internet of things and its integration in various fields made it more vulnerable to penetration by various cyber-attacks, due to its simplicity and scale. It is estimated that the IoT will remain a target and attack vector for years. Table 1.2 highlights the recent attacks[5].

TABLE 1.2 – Recent attack on IoT[5].

Year	Infected devices/industry	Mechanism	Type	Severity
2018	St. Jude Medical's implantable cardiac devices	Hacker identified vulnerability present in the transmitter. Then, the transmitter connected to a device that could read the device data. In that time, the transmitter was sharing data with a physician. By gaining control of the transmitter, the hacker got control of the device data privacy	DDOS attack	Gaining control of a healthcare-related device is dangerous as it is playing with human life.
2016	Dyn internet service provider	The simplest method used by Mirai malware was to use the default credential, i.e., a list of the standard username and passwords. With this default credential, a brute force attack is made to gain access to cameras, routers, etc., Once an IoT device becomes a bot it searches for the next vulnerable device	DDOS attack using IOT Botnet	Dyn suffered from DDOS attack. Twitter, the Guardian, Netflix, Reddit, and CNN websites were down
2015	SCADA system of the power grid system of Ukraine	Hacker gained control over energy distribution system and power cut happened for three-four hours in Ukraine area	DDOS attack	Ukraine area was without power supply for three-four hours

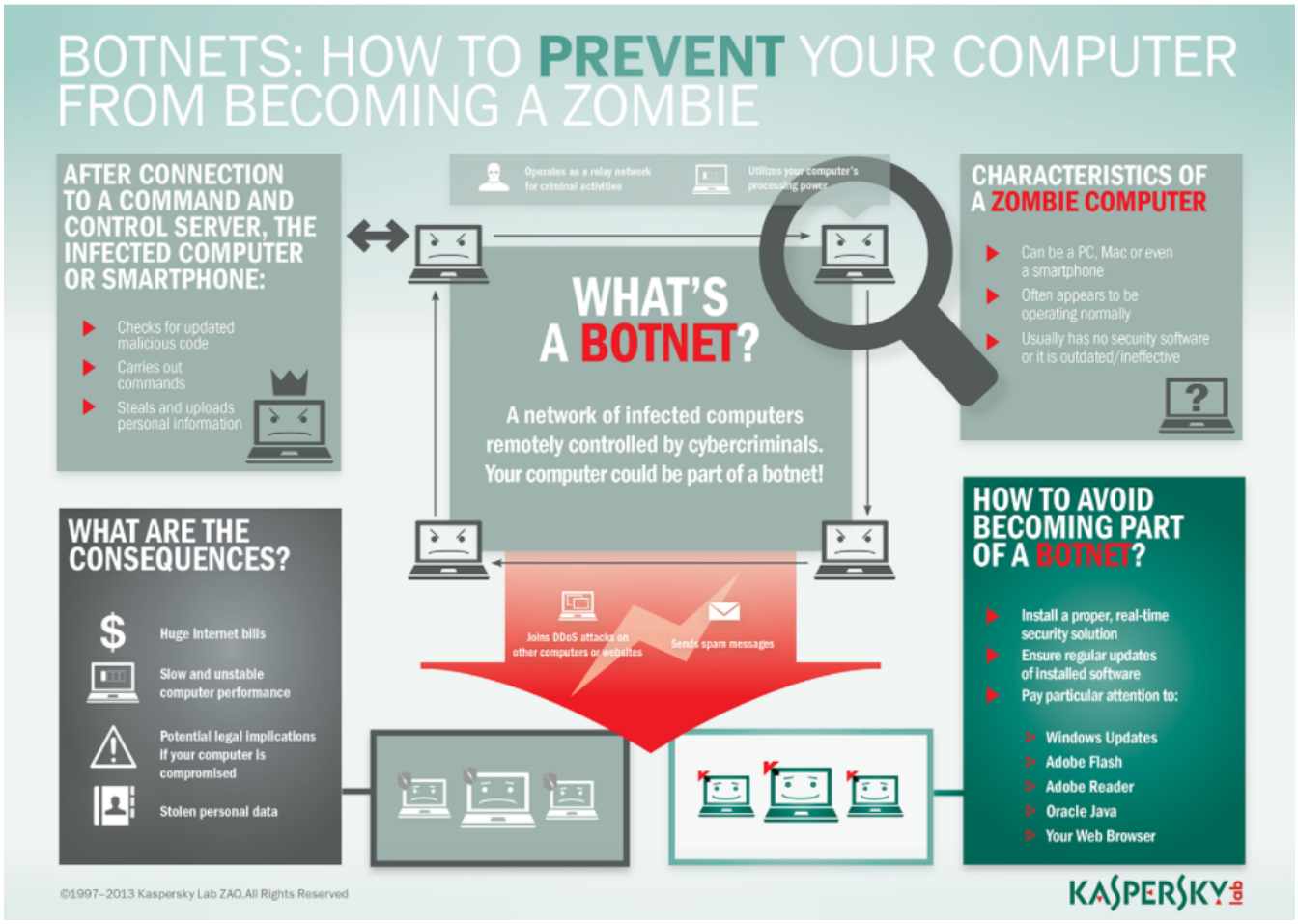


FIGURE 1.4 – description of Botnet attacks [20]

1.3.1 Botnets

A botnet is a collection of computers remotely controlled by a hacker called robots or bots that are used to perform automated tasks, usually malicious tasks. the controllers use the resources of these combined bots to carry out attacks against websites, computer networks, and Internet services, as shown in Fig 1.4. In many cases, the bots that are performing these tasks are computers that have been compromised. So, the owners of these computers may not be aware of what their computer is doing. botnets have spread widely all over the Internet without interference, due to their unobservable nature. Botnets can reach millions of computers. Bots can be programmed with methods used to infect other computers and create other bots or what can

be called "ZOMBIES". Botnets themselves can be hard to find and take down. But there are some things you can do to protect your organization. First, make sure to install antivirus software and be up-to-date on all your systems. This will help prevent infecting systems. You can also purchase heuristic-based intrusion detection systems. These systems can help to find and detect infected systems. Network monitoring can also be used. You should look for excessive network traffic between systems or excessive traffic destined for a single external system [30]. Fig 1.5 illustrates the basic stages of building a botnet into a few steps, as follows :

Prep and expose : hacker exploits a vulnerability to expose users to malware.
Infect : user devices are infected with malware that can take control of their device.
Activate : hackers mobilize infected devices to carry out attacks.
Stage 1, exposure starts with hackers finding a vulnerability in a website, application, or human behavior. The goal is to set the user up for being unknowingly exposed to a malware infection. You'll commonly see hackers exploit security issues in software or websites or deliver the malware through emails and other online messages.

In stage 2, the user gets infected with the botnet malware upon taking an action that compromises their device. Many of these methods either involve users being persuaded via social engineering to download a special Trojan virus. Other attackers may be more aggressive by using a drive-by download upon visiting an infected site. Regardless of the delivery method, cybercriminals ultimately breach the security of several users' computers.

Once the hacker is ready, stage 3 initiates by taking control of each computer. The attacker organizes all the infected machines into a network of "bots" that they can remotely manage. Often, the cybercriminal will seek to infect and control thousands, tens of thousands, or even millions of computers. The cybercriminal can then act as the boss of a large "zombie network" — i.e. a fully assembled and active botnet[20].

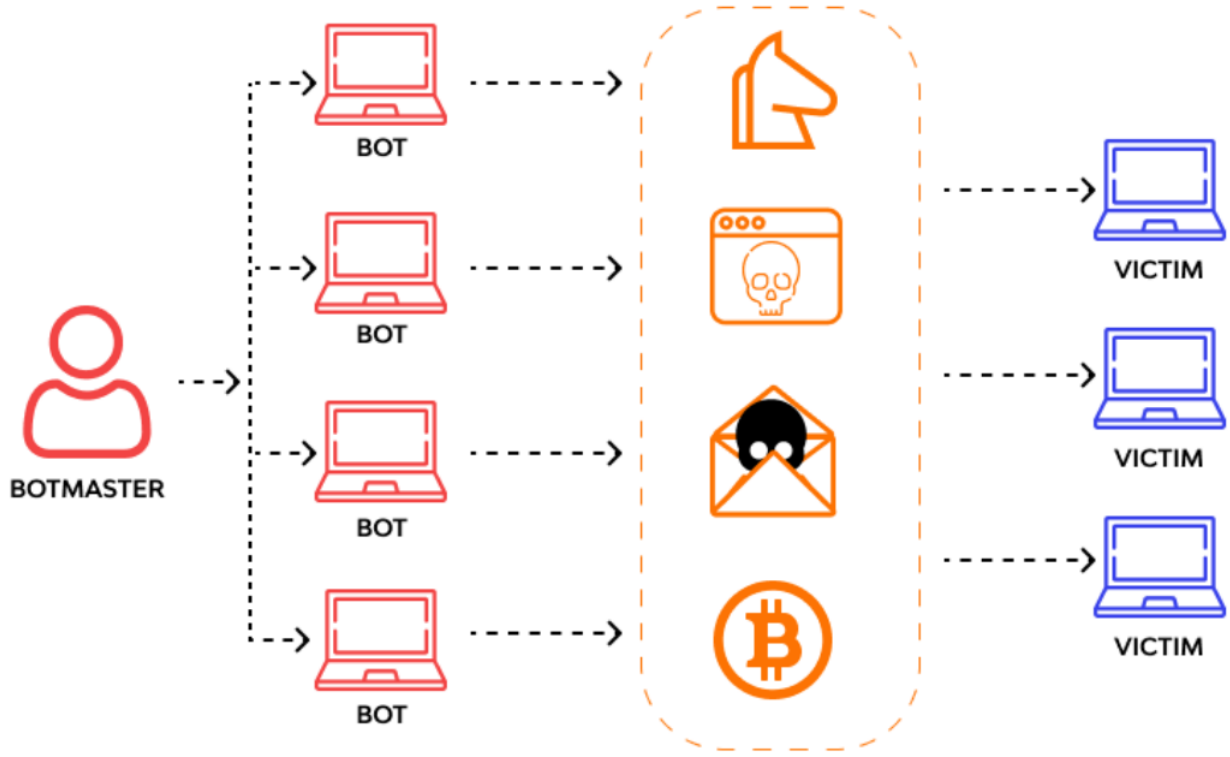


FIGURE 1.5 – Bot attacks representation[16]

How to control botnets :

Infected computers are controlled by a central server named a command-and-control server. which sends out periodic instructions to botnets in the infected computers. sometimes, there can be more than one command and control server in a botnet. This makes it even harder to stop. You may detect and shut down one command-and-control server, but the instructions to bots will be sent from another command-and-control server in the botnet. Each botnet can be led by commands either directly or indirectly in the following models[20] :

1. Centralized client-server models
2. Decentralized peer-to-peer (P2P) models

TABLE 1.3 – Types of Botnet attacks[20].

	Centralized botnet	P2P botnet
Protocol used	IRC, HTTP	P2P protocol
Pros	Easy to deploy and administer	No single point of failure
Cons	Single point of failure	Coordination of bots is difficult compared with a centralized architecture
Detection	Easy	Difficult
Life cycle	The life cycle has four phases – Initial infection – Connection with command and control – Perform botnet attack – Searching for other vulnerable devices	The life cycle has four phases – Initial infection – Connection with command and control – Perform botnet attack – Searching for other vulnerable devices

Types of Botnet attacks :

Table 1.3 shows the types of Botnet Attacks While botnets can be an attack in themselves, they are an ideal tool to execute secondary scams and cybercrimes on a massive scale. Common botnet schemes include some of the following : Distributed Denial-of-Service (DDoS) : is an attack based on overloading a server with web traffic to crash it. Zombie computers are tasked with swarming websites and other online services, resulting in them being taken down for some time.

Phishing : schemes to imitate trusted people and organizations by tricking them out of their valuable information. Typically, this involves a large-scale spam campaign meant to steal user account information like banking logins or email credentials.

Brute force attacks : run programs designed to breach web accounts by force. Dictionary attacks and credential stuffing are used to exploit weak user passwords and access their data[20].

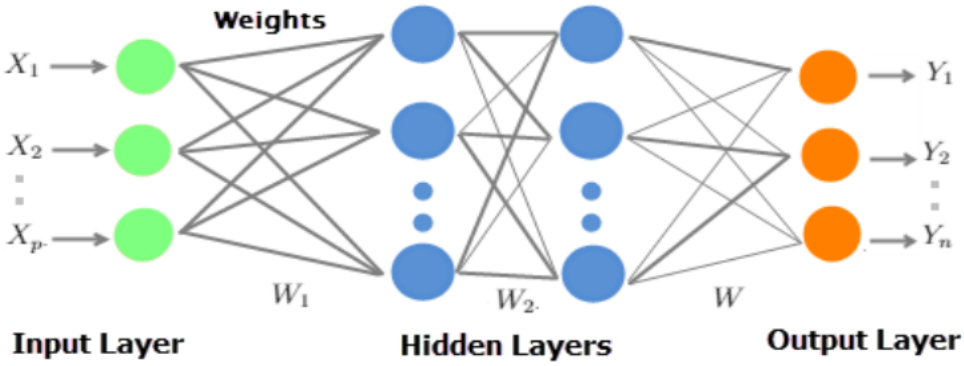


FIGURE 1.6 – The architecture of a Deep Learning model[24].

1.4 Application of deep learning in the cybersecurity

1.4.1 Deep learning :

Deep learning (DL) is a section of machine learning (ML) techniques, it has led to many remarkable successes, especially in the field of artificial intelligence(AI) compared to ML algorithms classics. Deep model architectures are relatively recent where much nonlinear information processing steps are exploited, as illustrated in Fig 1.6, in which information is processed in hierarchical layers, each receiving and interpreting information from the previous layer for learning representations of data.

1.4.2 Deep Learning for the IIOT :

IoT solutions for industries are growing significantly, especially for sensor-based data. Data released from IIoT technologies are derived from various manufacturing processes, including product lines, equipment, labor operations, and environmental conditions. Therefore, data modeling, labeling, and analysis play important roles in making industrial processes intelligent. The use of deep-learning techniques can upgrade industrial processes into highly-optimized smart facilities. For instance, in manufacturing, DL approaches can provide computing intelligence from unclear sensory

data, resulting in intelligent manufacturing. One of the advantages of DL over traditional machine-learning methods is that feature learning is performed automatically without requiring any external intervention. Fig 1.7 compares DL with traditional techniques, showing that DL outperforms other algorithms when the amount of data is large.

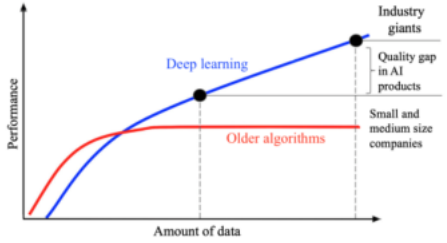


FIGURE 1.7 – Comparison of DL with traditional algorithms for the IIoT[21].

It consists of many different ways of explaining and interpreting IIoT data including descriptive analytics, predictive analytics, diagnostic analytics, and prescriptive analytics. Descriptive analytics involves capturing and analyzing a product's condition, environment, and operational parameters and summarizing the outcome. Predictive analytics employs statistical models to make future equipment improvements based on historical data. Diagnostic analytics reveal and report the major causes of failure or reduced product performance in industrial processes. Prescriptive analytics evidence the progress of activities in an industrial setup. DL, with its unconventional analytics, is transforming the industrial manufacturing facility into an augmented smart facility, having optimized operational costs and improved productivity. DL-based data-analysis techniques also develop the production process, detect failures, and predict each assessment's probable result. Fig. 1.8 shows various DL applications in IIoT. [21]

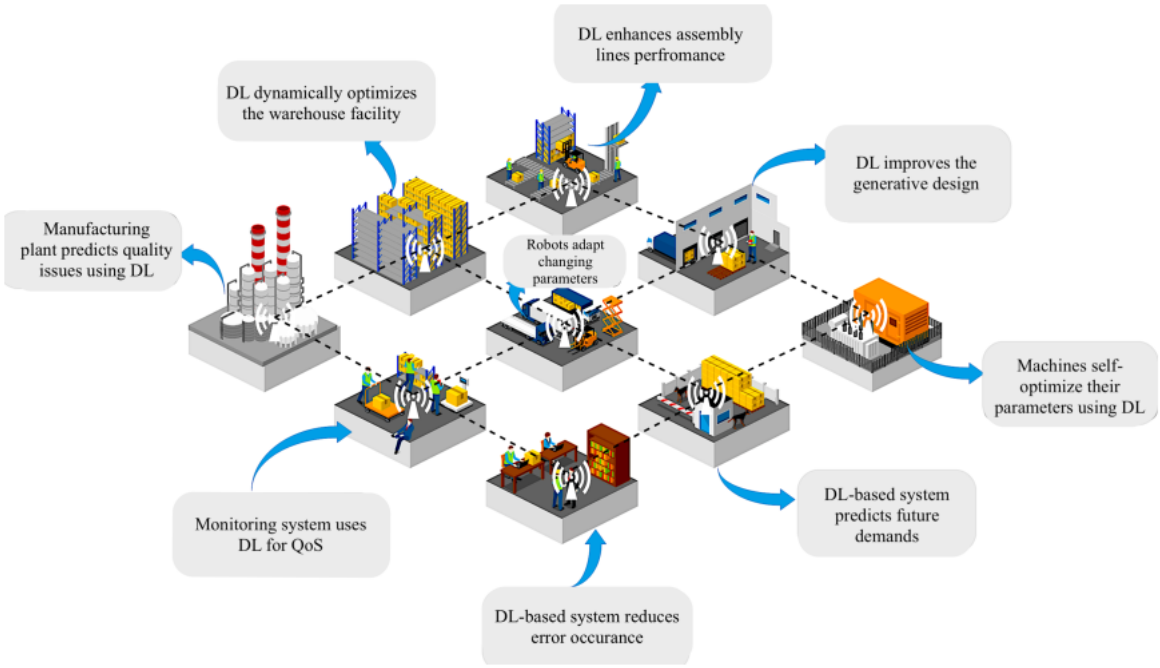


FIGURE 1.8 – Deep learning in the future Industrial Internet of Things[21].

1.4.3 Deep Learning methods :

Deep Neural Network (DNN) :

Deep Neural Networks (DNN) is a set of neurons organized in a sequence of multiple layers called Multilayer Perceptrons (MLP). They differ from traditional neural networks (Artificial Neural networks) by their depth and the number of layers, and nodes (neurons) that make up the network. When an ANN has two or more hidden layers, it is known as a deep neural network. They attempt to model data containing complex architectures by combining different nonlinear transformations The basic concept of perception was introduced by Rosenblatt in 1958. Perception computes a single output from multiple real-valued inputs (xi) by forming a linear combination based on its input weights (w), then placing the output through a nonlinear activation function. Mathematically, this can be written as follows :

$$y = \delta(nXn = 1Wixi + b) = \delta(WTX + b) \tag{1.1}$$

With :

• W : is the weight vector. • X : is the input vector. • b : designates the bias. • δ : represents the activation function. A typical multi-layer perception (MLP) network includes a set of source nodes forming the input layer, one or more hidden layers of compute nodes, and an output layer of nodes. The input signal propagates layer by layer on the network. The signal flow of such a network with a hidden layer is shown in (Figure 1.5). DNNs are generally used in supervised learning problems. Model training (learning) means adapting all weights and biases to their optimal values[24].

Convolutional neural networks (CNNs) :

Convolutional network or convolutional neural network or CNN is an extension of the classic feed forward network (FFN), which is used specially in the field of image processing. As shown in Figure 1.9, where all connections and hidden layers and its units are cached. CNNs outperform all other classical ML algorithms and make huge success in computer vision processing tasks (Computer Vision Tasks), they have a variety of applications [4] in image and video processing, natural language processing (NLP), recommendation systems, etc.

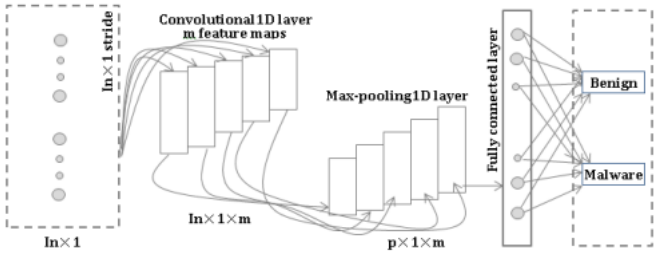


FIGURE 1.9 – Diagram of a convolutional neural network[4].

M : the number of filters Ln : the number of input characteristics P : the reduced dimension of the reduced dimensions of the features The efficiency of convolutional networks is because of the variety of layer types : convolution layers, pooling layers and fully connected layers.

Convolution layer : The purpose of convolution is to extract high-level features. It consists of a set of learner filters (or cores), each represents a certain independent functionality with the input volume. These filters consist of a layer of connection weights, they have a small receiving field (the size of the core), but when passing forward (feed forward), each filter is coevolved across the width and height of the input volume, calculating the product of the points between the inputs and the filter values producing a new feature map that better represents the information. As a result, the network learns filters that activate when it detects a type of feature that is important and specific to a certain spatial position in the input. Fig 1.10 shows a 1D convolution operation with a 1-dimensional input.

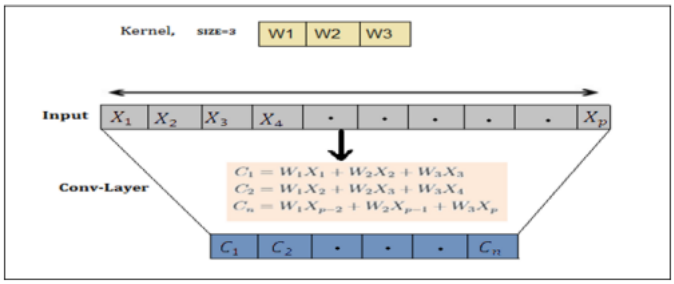


FIGURE 1.10 – convolution layer[4].

A convolutional layer shares the same convolution kernel, which greatly reduces the number of parameters needed for the convolution operation. A nonlinear activation function will be applied immediately after each convolutional layer. Deep CNNs with "Rectified Linear Units ReLU" activation function

$[F(x) = max(0; x)]$

Returns x for all values of x > 0 and returns 0 for all values of x <= 0. Train many times faster than their counterparts with "Tanh Units".

Pooling Layers : After the ReLU transformation, the pooling operation pools the activation of neurons from one layer into a single neuron from the next layer. The Pooling layer works independently on each input entity, it allows to gradually reduce the size of the representations in order to reduce the number of parameters or

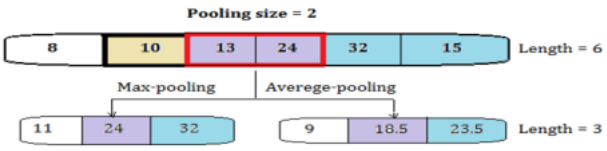


FIGURE 1.11 – pooling layer[4].

weights, which decreases the computational cost in the network, while preserving the most important information. more critical. It also helps to control overfitting. It can use two different pooling methods : Max-Pooling : uses the maximum value of each group of neurons from the previous layer. Average pooling : uses the average value of each group of neurons from the previous layer. Pooling is a form of nonlinear sub-sampling that works similarly to convolution. The Pooling kernel convolves over the input volume and divides it into a set of non-overlapping regions, and each sub-region produces a single output value which is the maximum value for Max-Pooling or the average value for Average-Pooling, Fig 1.11 depicts the Max-Pooling operation with a 1D input and a kernel of size 2 [4]. The Pooling layer has no learnable parameters. Because of this, these layers are usually not included in the total number of convolutional network layers.

Fully Connected Layers : At the end of a CNN there are one or more fully connected layers (every node in the first layer is connected to every node in the next layer). They consist in performing a classification based on the features extracted from the convolutions. The final layer contains a Softmax activation function, which generates a probability value from 0 to 1 for each of the class labels that the model attempts to predict. In some recent CNNs network architectures, the fully connected layers can be replaced by several average pooling layers. This allows these networks to significantly reduce the total number of parameters and which allows better prevention of overfitting [24].

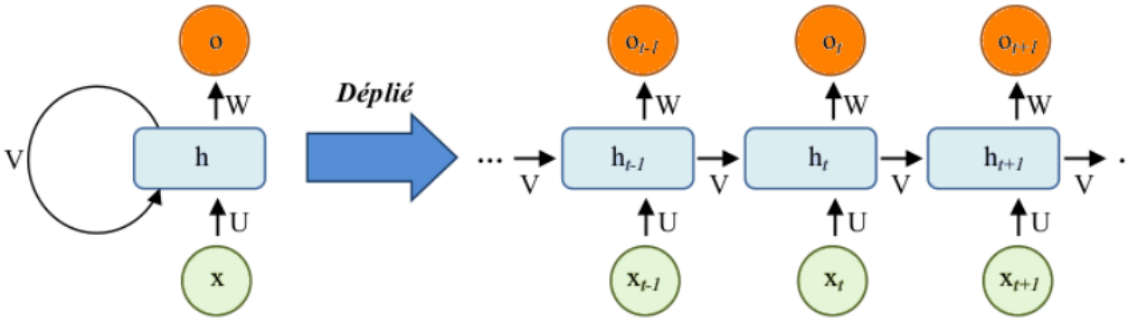


FIGURE 1.12 – The architecture of an RNN model [24]

Recurrent neural networks (RNNs) :

RNNs are Feed-Forward type networks having an internal state (or memory) that takes into account all or part of the data seen previously (already provided to the network), in addition to the data currently seen, to adapt their decision. The basic key idea of these networks is the deployment of a recurrent calculation thanks to the loops in the architecture of the network. The network output is a combination of its internal state (memory of inputs) and the last input, at the same time the internal state changes to incorporate this new data entered. This allows information to persist in memory, as shown in Fig 1.12.

Because of these properties, recurrent networks are suitable for cases where the presence of a shape is not the only discriminating information but also an order of appearance, for example. they are good candidates for tasks that process sequential data, such as textual data or data with temporal characteristics. The mathematical description of the memory transfer process is as follows :

$$ht = \delta(Uxt + Vht - 1 + bh) \tag{1.2}$$

$$Ot = \delta(Wht + by) \tag{1.3}$$

Where :

• ht : is the hidden state at time t.

• xt : is the input at the same time t.

• U; V; W : are the weight, Input-to-Hidden, Hidden-to-Hidden and Hidden-to-Output matrices respectively (known as transition matrices).

• bh : is the bias value of the hidden state.

• by : is the output bias value.

• Ot : is the output value at time t.

• δ : is a non-linearity function called activation functions. (either a logistic or tanh sigmoid function) which is a standard scaling tool for condensing very large or very small values into a logistic space, as well as making gradients workable for back-propagation.

A neural network block examines an "xt" input and outputs an "ot" value. A feed-back loop occurs at each time step, each hidden state ht contains traces not only of the previous hidden state but also of all those before "ht-1" for as long as memory can persist.[4]

Long short-term memory (LSTM) :

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture, used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). LSTM networks are well-suited to classifying, processing, and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models, and other sequence learning methods in numerous applications[24].

1.4.4 The Role of DL in IIoT Security :

The IIoT achieves high levels of accessibility by allowing commercial and industrial organizations to gather useful information from various sensors, machines, devices, instruments, and control systems. These devices may be located within specific facilities (e.g. factories) or at distant locations (e.g. in applications related to agriculture, mining, oil, and gas). To achieve this high level of connectivity on all levels of the IIoT pyramid, new entry points are introduced into the IIoT systems. However, this highly-networked architecture results in large attack surfaces and exposes whole IIoT systems to serious cybersecurity threats that must be given due consideration. In fact, the industry has already faced several serious cybersecurity incidents. For example,
•Malware called TRISIS was used to launch a cyberattack on a petrochemical plant in Saudi Arabia that resulted in the plant's shutdown. Experts believe the true aim of the attack was likely to kill people by disabling the safety system and disturbing the petrochemical process. Nobody was killed in the attack, but the site lost millions because of the abrupt shutdown.
• Internet-connected cameras were used to launch a cyberattack on the Baku-Tbilisi-Ceyhan (BTC) gas pipeline, shut down alarms, and over-pressurize the pipeline, resulting in a massive explosion.
• A human-machine interface at the Bowman Avenue Dam in New York was accessed by hackers via an internet-connected cell card with poor authentication protocols. These incidents demonstrate that security and privacy are critical concerns in any IIoT system and therefore should be active areas of research. In fact, the industry has made a coordinated effort toward identifying the threats and vulnerabilities in IIoT systems and towards devising standards and best practices to counter these. The Industrial Internet Security Framework (IISF), designed by the Industrial Internet Consortium (IIC), the security measures outlined by the European Union Agency for Cybersecurity (ENISA), and similar efforts by the National Institute of Standards and Technology (NIST) and IEEE are some promising steps towards securing the IIoT. There is a

growing trend in using deep-learning solutions to secure the IIoT. Most of this deep-learning work targets the problem of detecting intruders in different IoT and IIoT architectures[21].

1.5 Conclusion

Before getting into the the main work and the realisation of the system we had to get understand some basics and new concepts as IOT and IIOT in addition to Deep learning basics. all that had been presented in this chapter.

CHAPITRE 2

INTRUSION DETECTION SYSTEMS(IDS) BASED ON DEEP LEARNING :

2.1 Introduction :

This chapter we going to present to you the existed related works in the field of realisation of intrusion detection systems, so we have to analyze and the used methods and discuss the founded results.

2.2 Intrusion detection systems(IDS) :

The IDS is a system that will supervise the network traffic to detect any suspicious activities and known threats. to ensure its confidentiality, integrity, and availability. It may also lunch alerts to users while the detection of such activities. To handle and classify the attacks efficiently, various ML algorithms can be used. This section focuses on various techniques that are used for identifying the intrusion. IDS has a very important role in detecting attacks[31].

The operation of the intrusion detection system is divided into three phases. Monitoring is the initial phase in IDS which is based on network or host sensors. The

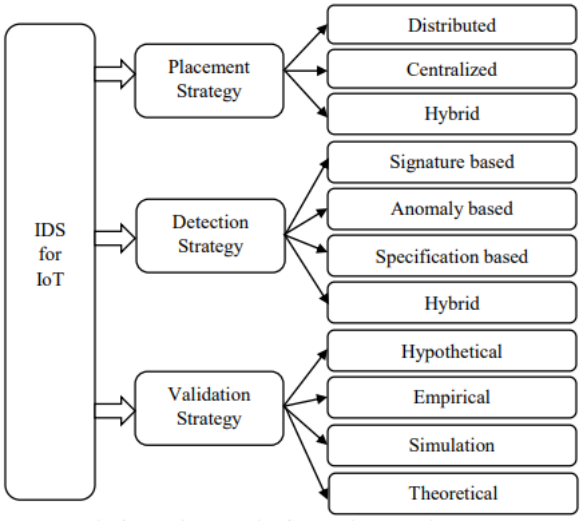


FIGURE 2.1 – Intrusion Detection Systems in IoT environment.[34]

analysis is the second phase of IDSs which performs feature extraction and pattern identification-based process. Detection is the last phase of IDSs which detects the anomaly or intrusion in a network.

Basically, conventional IDS may be categorized into three strategies as illustrated in Fig 2.1 such as placement strategy, detection strategy, and validation strategy. The detection strategy attracts more attention and most systems are developed based on this strategy [34].

— Signature-based IDS : It describes the attacks and their patterns and identifies the attacks. On detecting an attack in a network, a signature-based detection system raises an alert about suspicious activities and perform pattern matching. Based on the similarity and differences, the access or alert provided to the user detect the attacks effectively[34].

— Anomaly-based IDS : It is an initial stage intrusion detection system that collects the data and identifies the abnormalities in the system. Based on a threshold value, the normal and abnormal behaviors are identified, and an alert is

raised to the network administrator about the abnormalities. It detects the unknown attacks efficiently, but it requires large memory to process and computation costs are the limitations of anomaly-based intrusion detection systems[34].

— Specification-based IDS – Based on the specific operation, these systems continuously evaluated the system operations. The specific operation is defined by the network administrator, and it is monitoring the process and continues to validate the operation. If abnormalities are detected as per the operation, an alert is forwarded to the network administrator[34].

— Hybris IDS – A combination of anomaly and signature-based IDSs is considered a hybrid model which provides a better tradeoff between the storage and computing cost with fewer false-positive alarms. Recently, most of the systems are based on Hybrid IDS due to its effective detection and simplified operation [34].

2.3 Related works :

The application of Deep Learning has increasingly gained importance to address some emerging intrusion detection. As deep learning methods are able to directly process raw data, allowing them to learn features and perform classification at the same time, they become the mainstream approach in IDS studies (Table 2.1). the hackers are uploading their performances and the created new methods, in another side the intrusion detection systems are being developed more and more in order to be effective against this malware.

Studies	Year	Method	Datasets Used	Perfromance Metrics
Kanimozhi et al.[17]	2019	ANN, RF, k-NN, SVM, Adaboost, NB	CSE-CIC IDS2018	Accuracy, Precision, Recall, F1-Score
Kim et al.[17]	2019	CNN	CSE-CIC IDS2018	Accuracy.
Patil et al. [26]	2019	Random Forest.	CSE-CIC-IDS2017 UNSW-NB15	Accuracy, FPR.
Ferrag et al.[6]	2020	Deep disciriminative models. (DNN, RNN, CNN)Unsupervised models (RBM, DBN, DBM, DA)	CSE-CIC IDS2018 BoT-IoT	Accuracy, Time(s)
Khammassi et al. [22]	2020	C4.5, RF, NB Tree	CSE-CIC-IDS2017UNSW-NB15	Accuracy.
Zhang et al.[35]	2020	Lenet, MSCNN, HAST, MSCNN-LSTM	UNSW-NB15	Accuracy, FAR, FNR
Kasongo et al.[19]	2020	FFDNN	UNSW-NB15 AWID	Accuracy
Bouteraa et al.	2020	RF, SVM, ELM, RPART, OCSVM	ISCX2012	Accuracy, DR, FAR
Hassan et al.[15]	2020	CNN + WDLSTM	ISCX 2012 UNSW-NB15	Accuracy
Rashid et al.[27]	2020	k-NN, SVM, Naive Bayes(NB), DNN, DAE	CIDDS-001 NSL-KDD	Accuracy

TABLE 2.1 – related works for intrusion detection system based on DL methods.

Ferrag et al. [6] In their study, working with deep learning methods (recurrent neural networks (RNN), deep neural networks (DNN), restricted Boltzmann machines (RBM), deep belief networks (DBN), convoluted neural networks (CNN), deep Boltzmann machines (DBM), and deep autoencoders (DA) have applied them on CSE-CIC-IDS2018 and Bot-IoT datasets. Then, compare the classification success of deep learning and classification time of these data sets. moreover, in their study, examined intrusion detection systems based on deep learning methods, and in this sense, divided 35 attack detection data sets used in the literature into categories.

Kim et al. [23] used CNN and RNN deep learning methods on CSE-CIC-IDS 2018 dataset and compared the performance of these two methods. Kanimozhi et al.[18] used AN, RF, k-NN, SVM, ADA BOOST, and NB machine learning methods to classify the CSE-CIC-IDS 2018 data set.

Khammassi et al. [22] proposed a feature selection method based on Non-Dominated Sorting Genetic Algorithm II (NSGA II) and logistic regression. The proposed approach was tested according to the Non-Dominated Sorting Genetic Algorithm Binomial Logistic Regression (NSGA2-BLR) and Non-Dominated Sorting Genetic Algorithm Multinomial Logistic Regression (NSGA2-MLR) methods. C4.5, Random Forest (RF), and Naive Bayes (NB) methods were used to classify the best subsets obtained. In the study, the used NSL-KDD, UNSW-NB15, and CIC-IDS2017 data sets.

Ring et al. [29] conducted a comprehensive review of intrusion detection systems. In the study, the analyzed data formats of network-based intrusion detection systems. In addition, 15 features have been defined to evaluate the suitability of data sets. In addition, these features were organized into 5 groups : General Information, Quality of Data, Data Volume, Recording Environment, and Evaluation.

Kanimozhi et al. [17] classified the CSE-CIC-IDS-2018 dataset using Artificial Intelligence (AI). they achieved 99.97% success as a result of the classification. Zhang et al. [35] proposed a unified method combining Multiscale Convolutional Neural Network (MSCNN) with Long Short-Term Memory (LSTM). In the first level of the method, used MSCNN to analyze the spatial properties of the data set. In the second

level of the method, used LSTM network processes temporary features. they applied the training and testing of the model to the UNSW-NB15 dataset. The method has better accuracy, false alarm rate, and false-negative speed than models based on traditional neural networks.

Kasongo et al. [19], they conducted tests on AWID and UNSW-NB15 datasets using the Wrapper Based Feature Extraction Unit (WFEU) feature extraction method and Feed-Forward Deep Neural Network (FFDNN). Rashid et al. [27] classified the NSL-KDD and CIDDS-0001 datasets with SVM, Naive Bayes (NB), KNN, Neural networks, DNN, and DAE classification algorithms. For a reason to rise the classifier's success, before the classification algorithms, they use hybrid feature selection and sorting methods, and high accuracy rates were obtained.

Bouteraa et al. [3] Conducted a comparative study of data mining techniques for intrusion detection using the ISCX 2012 dataset. Hassan et al. [15] proposed a hybrid deep learning model in order to detect network attacks with high accuracy values. For this purpose, they used CNN and weight-dropped, long short-term memory (WDLSTM) deep learning methods together. They used UNSW-NB15 and ISCX2012 datasets to measure the performance of their proposed method.

Patil et al. [26] proposed the hypervisor level distributed network security (HLDNS) security framework of cloud computing in their study. In this method, intrusions to each server with virtual machines are monitored. Feature similarity-based Fitness Function (FSFF) and Classifier Accuracy based Fitness Function (CAFF) fitness functions are used in conjunction with the BBA algorithm for feature extraction. The performance of the proposed method has been measured using the UNSW-NB15 and CICIDS-2017 datasets.

2.4 Existed datasets :

There are some datasets that used in intrusion detection systems for the last few years presented in Table 2.2.

Dataset	Year	Attack types	Attacks
KDD Cup'99	1998	4	DoS, Probe, R2L, U2R
Kyoto 2006+	2006	2	Known Attacks, Unknown Attacks
NSL-KDD	2009	4	DoS, Probe, R2L, U2R
UNSW-NB15	2015	9	Backdoors, DoS, Exploits, Fuzzers, Generic, Port scans, Reconnaissance, Shellcode, worms
CIC-IDS2017	2017	7	Brute Force, HeartBleed, Botnet, DoS, DDoS, Web , Infiltration
CSE-CIC-IDS2018	2018	7	HeartBleed, DoS, Botnet, DDoS, Brute Force, Infiltration, Web.

TABLE 2.2 – Some existed intrusion detection datasets [1].

2.5 Conclusion

We have seen in this chapter the existing works in the intrusion detection system based on Deep learning methods as well as the reached results.

CHAPITRE 3
CONCEPTION AND REALIZATION

3.1 Introduction :

The intrusion detection systems based on Deep Learning methods becomes the new generation of intrusions detection systems. The botnet detection systems are a specific IDS, to realize that system we've been through many steps that will be presented in the following chapter.

3.2 Tools and environments

3.2.1 Google Colaboratory :

Collaborator, often shortened to "Colab", is a product of Google Research. Colab allows anyone to write and run Python code of their choice through the browser. It is an environment particularly suitable for machine learning, data analysis, and education. In more technical terms, Colab is a hosted Jupyter notebook service that requires no configuration and provides free access to computing resources, including GPUs [9].

3.2.2 TensorFlow :

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications [10].

3.2.3 Keras

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load : it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides [11].

3.2.4 Numpy :

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms basic linear algebra, basic statistical operations, random simulation and much more [12].

3.2.5 Pandas :

Pandas is an open-source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language [13].

3.3 DATABASE presentation :

In this section, we are about to explore and describe the database we're working on and the different steps that have been followed to realize it. but first, we can take a brief idea about existing datasets for cybersecurity, illustrated in Table 3.1 :

Dataset	year	Description	ML Techniques	Lacks
N-Balot	2018	Built using 9 IoT devices for legitimate traffic and two botnets (BASHLITE and Mirai) for 10 attack types	LOF, On Class SVM, and IF	- Limited IoT threat model. - IIoT traffic is not included.
Bot-IoT	2019	Built from IoT legitimate traffic as well as malicious traffic generated by botnets on IoT-specific networks.	RNN, SVM, and LSTM	- No IIoT data included. - Not suitable for IIoT security
MQTTset	2020	Built from MQTT protocol traffic and a variety of attack streams associated with IoT devices that leverage it.	NN, DT, RF, NB, MP, and GB	- Contains only MQTT traffic. - Not suitable for IIoT security.
Federated Ton_IoT	2020	Consists of three different data types, namely : IoT service telemetry, OSs logs, and network traffic	N/A	- No IDS evaluation. - Absence of IIoT - related data.
X-IIoTID	2021	Consists of device agnostic data used in the context of ML/DL based IDS for both IoT and IIoT systems.	DT, NB, SVM, KNN, LR, DNN, and GRU.	- IDS evaluations are only based on centralized learning.
WUSTL-IIOT-2021	2021	Created using legitimate and malicious data generated by various IIoT and industrial devices to mimic an actual industrial application	LR, KNN, SVM, NB, RF, DT, and ANN	- No IoT-related traffic, data, or attacks. - Not suitable for IoT security

TABLE 3.1 – Available IoT and IIoT datasets for CyberSecurity [7].

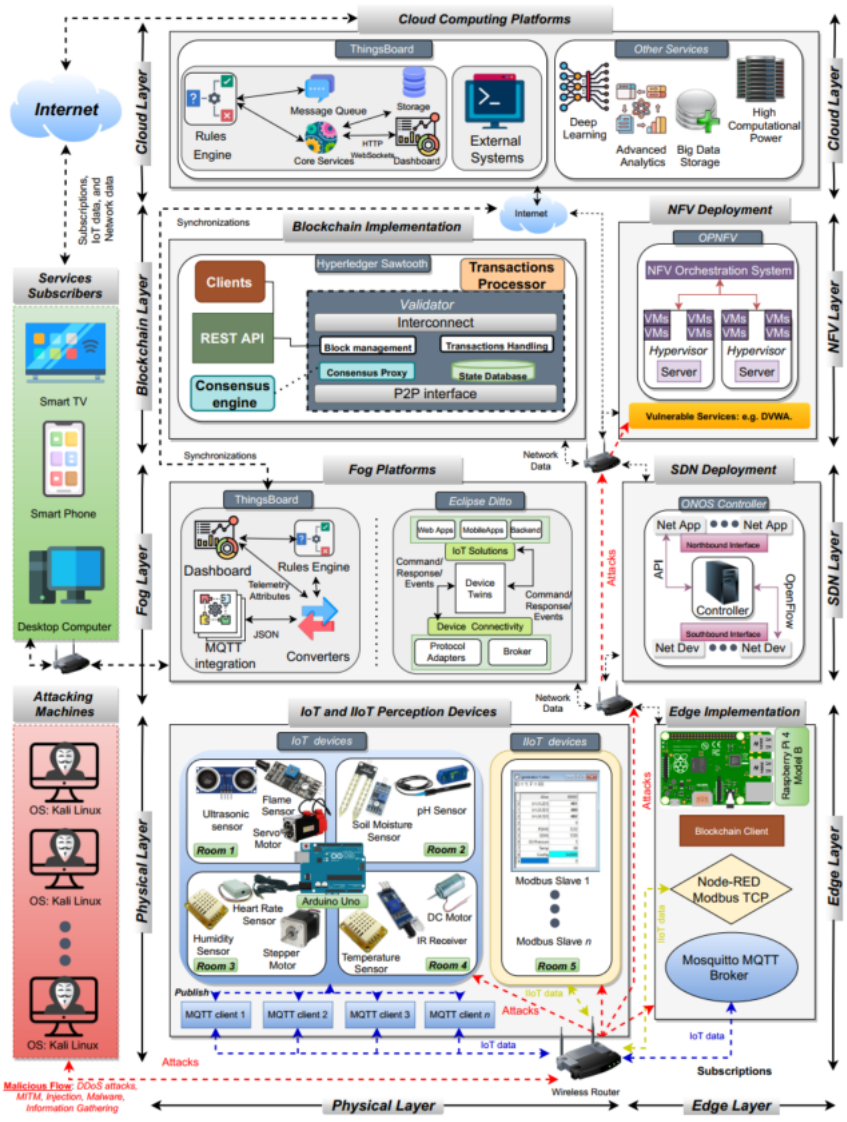


FIGURE 3.1 – The proposed testbed architecture[7].

For the realization of our project, we chose the Edge-IIoTset-2022 data-set, proposed by Dr. Mohamed Amine Ferrag, which can be used by machine-learning based intrusion detection systems. Its highlevel overview is depicted in Fig 3.1. The data of this dataset is collected from a testbed consisting of sophisticated seven interconnected layers namely : cloud computing layer, NFV layer, Blockchain layer, fog layer, SDN layer, edge layer, and IoT/IIoT perception layer as shown in (figure 3.1), including more than 10 IoT devices, IIoT-based Modbus flows, 14 IoT and IIoT protocol-related attacks.

3.4 Dataset description

After introducing the existing IoT and IIOT cybersecurity datasets either the tools and the open sources used to realize the dataset we're working on now we're about to present a description of it.
The IoT data are generated from various IoT devices (more than 10 types) such as Low-cost digital sensors (for sensing temperature and humidity, Ultrasonic sensor, Water level detection sensor, pH Sensor Meter, Soil Moisture sensor, HeartRate Sensor, Flame Sensor, etc.). The following figure will present you different steps followed to get the dataset generation framework [7]. from another vision we can present you the dataset after decompose it into 3 major axes :

Normal traffic : in this part of dataset we have the normal traffics of the different sensors applied in the testbed (10 directories) stored in both extension ".pcap" and ".csv" example :in Distance file we can find 2 subfiles in the name distance.pcap and distance.csv

Attack traffic : in this side we have the traffic result of a set of attacks(will be seen later) applied on the normal traffics which called attack traffics, so we have 14 files for attacks also stored in both extensions ".pcap" and ".csv" (figure3.4).

Selected dataset for ML and DL : as we can see in figure3.5 this part of dataset is made for DL and ML application it contains treated parts of the normal and attack

traffic, which are easier to manipulate and to work on with different applications.

3.4.1 Attacks presentation

Category	Type	IoT vulnerabilities	Tools
DoS/DDoS attacks	TCP SYN Flood DDoS attack	Make the victim's IoT edge server unavailable to legitimate requests	Sending manipulated SYN packets using the tool hping3-based python scrip
	UDP flood DDoS attack	Overwhelm the processing and response capabilities of IoT devices	Sending manipulated UDP packets using the tool hping3-based python script
	HTTP flood DDoS attack	Exploits seemingly-legitimate HTTP GET or POST requests toattack IoT application	Use 200000 connections with GET requests using the slowhttptest tool
	ICMP flood DDoS attack	The IoT edge servers become inaccessible to normal traffic By flooding them with request packets (i.e., with ICMP echo-requests (pings))	Sending manipulated ICMP packets using the tool hping3-based python script
Information gathering	Port Scanning	Discover open doors or weak points in the edge-based IoT network	Discover active hosts using the Nmap and Netcat tools
	OS Fingerprinting	Analyzing IoT data packets to spot the weakness of IoT devices as well as Edge servers	An active operating system fingerprinting tool, named xprobe2
	Vulnerability scanning attack	Identifying IoT network security vulnerabilities	A web server scanner tool, named Nikto, for performing comprehensive tests against web servers
Man in the middle attacks	DNS Spoofing attack	The interception of communications between IoT devices and a DNS server	Sniffing and spoofing using Ettercap tool
	ARP Spoofing attack	Linking an attacker's MAC address with the IP address of an IoT device or Edge server	Sniffing and spoofing using Ettercap tool

Injection attacks	Cross-site Scripting (XSS) attack	Send a malicious script to an unsuspecting user, which can access sensitive information, session tokens, cookies ...etc	Detect, exploit and report XSS vulnerabilities using xsser tool in a PHP/MySQL web applications (DVMA application)
	SQL Injection	(Read/Insert/Update/Delete) sensitive data from the IoT database by the injection of a SQL query	Detecting and exploiting SQL injection flaws using sqlmap tool
	Uploading attack	Uploading files that contain malwares' command and control data	Creating php backdoor using Metasploit framework and uploading through a PHP/MySQL web application (e.g., Damn Vulnerable Web App (DVWA))
Malware attacks	Backdoor attack	Install backdoors to take control of vulnerable IoT network components	Creating python script backdoor using Metasploit framework and then transferring it using curl tool
	Password cracking attack	Identify an unknown or forgotten password to an IoT device in order to obtain unauthorized access to IoT resources	The CeWL tool is used as a ruby app for creating a list of words (password crackers) and email addresses (usernames)
	Ransomware attack	Publish or blocks access to IoT data or an IoT device system by encrypting it, until the victim pays a ransom fee to the attacker	After applying the Backdoor attack, the OpenSSL cryptography toolkit is used for creating RSA public/private keys and encrypting and decrypting victim files

TABLE 3.2 – The list of attack scenarios included in Edge-IIoTset dataset [7].

The Edge-IIoTset-2022 data-set contains fourteen different attacks against IoT and IIoT applications been identified and analyzed, and categorized into five main threats,

as illustrated in Table 3.2 :

DoS/DDoS attacks :

It make the victim's IoT edge server unavailable to legitimate requests by sending manipulated packets, which include four attacks, namely, TCP SYN Flood DDoS attack, UDP flood DDoS attack, HTTP flood DDoS attack, and ICMP flood DDoS attack.

The Information gathering :

It consists of analyzing IoT data packets to spot the weakness of IoT devices as well as Edge servers, which include three attacks, namely, Port Scanning, OS Fingerprinting, and Vulnerability scanning attack.

The man in the middle attacks :

It consists of the interception of communications between IoT devices and edge servers, which include two attacks, namely, ARP Spoofing attack and DNS Spoofing attack.

The injection attacks :

It consist of sending a malicious script to an unsuspecting user, which can access sensitive information, session tokens, cookies, ...etc.

the malware attacks :

It consist of installing backdoors to take control of vulnerable IoT network components, which include three attacks, namely, Backdoor attack, Password cracking attack, and Ransomware attack.

3.5 Concept :

An IDS is composed of two initials parts : "intrusion" and "detection system".Intrusion refers to unauthorized access to the information within a computer or network system to compromise its integrity, confidentiality, or availability. whereas a detection system is considered as a security mechanism for the detection of such malicious activity. So, IDS is a security tool that constantly supervised the host and network traffic to detect any suspicious behavior that breaks the security policy and compromises its confidentiality, integrity, and availability. The IDS will generate alerts about detected menaces to the host or network administrators [1].

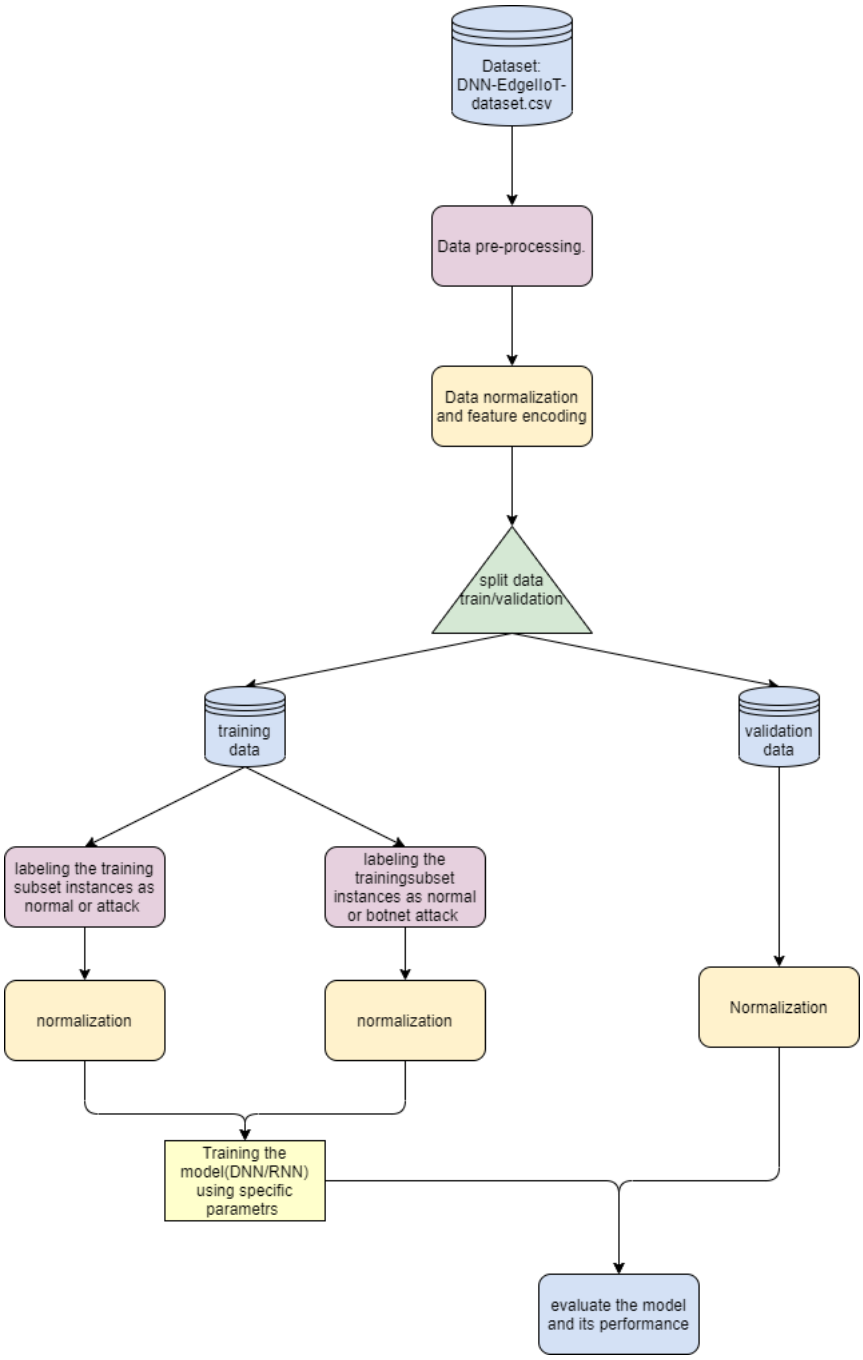


FIGURE 3.2 – Proposed methodology and implementation for botnet attack detection.

The realization of our project passed by multiple steps that have been illustrated in Fig 3.2. Starting from data pre-processing to the model application and finishing by the validation phase. Those steps are well presented in the next titles.

```
[ ]  %%capture
     !pip install stellargraph
     !pip install pretty-confusion-matrix
     !pip install numpy
     !pip install matplotlib==3.1.3
     !pip install tensorflow
     !pip install pandas
     !pip install kaggle
```

FIGURE 3.3 – Needed libraries.

3.5.1 Followed steps :

Library :

As a first step in the realization of our project we start by installing needed libraries, generally in google colab we found most of the libraries are already installed, so we just added the missing ones (such as "numpy", "kaggle", etc), as depicted in Fig. 3.3.

Uploading dataset :

Next, we need to upload the dataset "Edge-IIoTset dataset" to work on it. To upload the dataset we have first to create JSON file, which gives us the ability to upload datasets from Kaggle directly without uploading it to the computer (i.e. optimize time and memory), in addition, we upload the field "DNN-EdgeIIoT-dataset.csv" as a zip file then we unzip it to get the used file.

display dataset information :

This step allowed us to take a view about the dataset components features and instances before applying any processing (original dataset) presented in graphs to make it easier to analyze and understand, in addition we can see instances type, counts, and different information. Fig 3.4 and Table 3.3 show more information about different attacks types in the dataset. Furthermore, Fig 3.5 and Table 3.4 illustrate the attack traffic with regard to the normal traffic percentage.

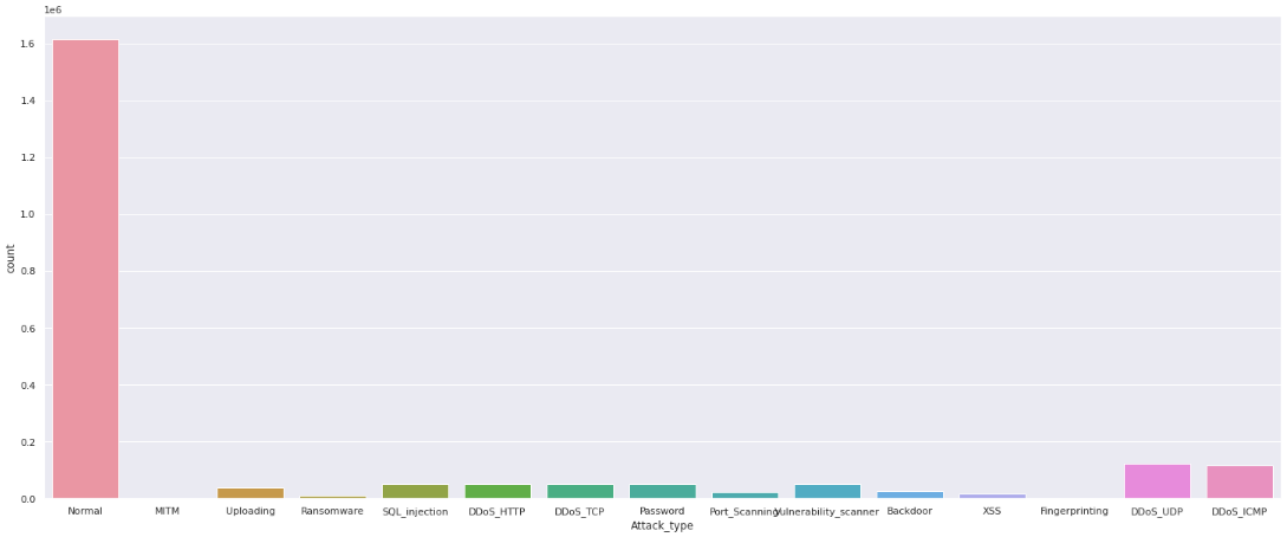


FIGURE 3.4 – Different attack types in the dataset.

N	Attack_Type	Count	%
0	Backdoor	24862	1.120313
1	DDoS_HTTP	49911	2.249053
2	DDoS_ICMP	116436	5.246753
3	DDoS_TCP	50062	2.255857
4	DDoS_UDP	121568	5.478008
5	Fingerprinting	1001	0.045106
6	MITM	1214	0.054704
7	Normal	1615643	72.802914
8	Password	50153	2.259958
9	Port_Scanning	22564	1.016762
10	Ransomware	10925	0.492294
11	SQL_injection	51203	2.307272
12	Uploading	37634	1.695836
13	Vulnerability_scanner	50110	2.258020
14	XSS	15915	0.717150
Total		2219201	100.00

TABLE 3.3 – Detailed Dataset compenents

Attack_label	Count	%
Normal	1615643.00	72.80
Attack	603558.00	27.20
Total	2219201.00	100.00

TABLE 3.4 – Dataset compenents

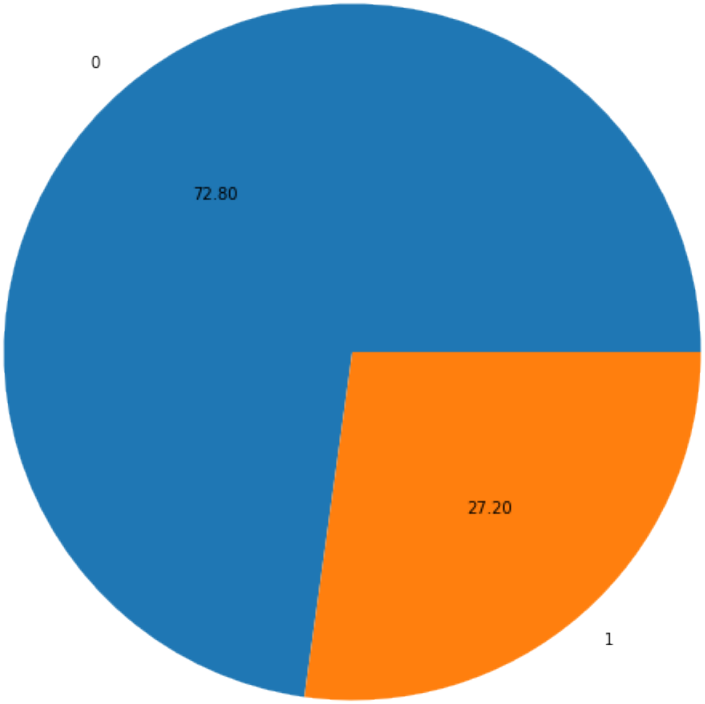


FIGURE 3.5 – Illustration of normal traffic percentage (0) vs attack traffic (1).

Data pre-processing :

For preparing the dataset to apply the processing on it, we have to make a copy first to use it in the different treatments.

First, in the application of the deep learning methods we must have enough features that satisfy the classification (depends on quantity and quality of DATA) and reach the best results, but in the dataset we may find some useless features that may influence the results in the worst way possible, so as an urgent step to make is to drop those features like :
"frame.time", "ip.src_host", "ip.dst_host", "arp.src.proto_ipv4","arp.dst.proto_ipv4", "http.file_data", "http.request.full_uri", "icmp.transmit_timestamp", "http.request.uri.query", "tcp.options", "tcp.payload", "tcp.srcport", "tcp.dstport", "udp.port", "mqtt.msg". In another way we can say that the more we have good quality data, the more we have better and more specific results, so we need to keep only traffic features that's all it matters. In the other part we have also the duplicated rows as well as the features that their value is ="NULL" or "NUN", all that are useless information that have to be dropped to ensure efficiency of the application. Table 3.5 and Table 3.6 illustrate the different features with their types and choice of features, respectively. Further, Table 3.7, Fig 3.6 and Fig 3.7 represent the botnet attacks type and their percentage in the dataset, respectively.

Further, as I work only on botnet attacks I have to drop the other rows that present the other attacks type like :"MITM", "XSS", "SQL_injection"..., to make manipulating data much easier.

Attack_type	count	%
DDoS_HTTP	48544	2.938303
DDoS_ICMP	67939	4.112256
DDoS_TCP	50062	3.030186
DDoS_UDP	121567	7.358287
Normal	1363998	82.560967
Total	1652110	100.000000

TABLE 3.7 – Botnet Attacks working on them.

For the attack_Label column, which represents the class of each instance, it has

Features	Type	Features	Type
frame.time	object	http.request.full_uri	object
tcp.payload	object	mqtt.hdrflags	float64
ip.src_host	object	http.request.version	object
tcp.seq	float64	mqtt.len	float64
ip.dst_host	object	http.response	float64
tcp.srcport	float64	mqtt.msg_decoded_as	float64
arp.dst.proto_ipv4	object	http.tls_port	float64
udp.port	float64	mqtt.msg	object
arp.opcode	float64	tcp.ack	float64
udp.stream	float64	mqtt.msgtype	float64
arp.hw.size	float64	tcp.ack_raw	float64
udp.time_delta	float64	mqtt.proto_len	float64
arp.src.proto_ipv4	object	tcp.checksum	float64
dns.qry.name	float64	mqtt.protoname	object
icmp.checksum	float64	tcp.connection.fin	float64
dns.qry.name.len	object	mqtt.topic	object
icmp.seq_le	float64	tcp.connection.rst	float64
dns.qry.qu	float64	mqtt.topic_len	float64
icmp.transmit_timestamp	float64	tcp.connection.syn	float64
dns.qry.type	float64	mqtt.ver	float64
icmp.unused	float64	tcp.connection.synack	float64
dns.retransmission	float64	mbtcp.len	float64
http.file_data	object	tcp.dstport	float64
dns.retransmit_request	float64	mbtcp.trans_id	float64
http.content_length	float64	tcp.flags	float64
dns.retransmit_request_in	float64	mbtcp.unit_id	float64
http.request.uri.query	object	tcp.flags.ack	float64
mqtt.conack.flags	object	Attack_label	int64
http.request.method	object	tcp.len	float64
mqtt.conflag.cleansess	float64	Attack_type	object
http.referer	object	tcp.options	object
mqtt.conflags	float64	\	\

TABLE 3.5 – dataset features and their type.

Operation	Drop	Labling	Nothing
1	frame.time	http.request.method	Others
2	ip.src_host	http.referer	
3	ip.dst_host	http.request.version	
4	arp.src.proto_ipv4	dns.qry.name.len	
5	arp.dst.proto_ipv4	mqtt.conack.flags	
6	http.file_data	mqtt.protoname	
7	http.request.full_uri	mqtt.topic	
8	icmp.transmit_timestamp		
9	http.request.uri.query		
10	tcp.options		
11	tcp.payload		
12	tcp.srcport		
13	tcp.dstport		
14	udp.port		
15	mqtt.msg		

TABLE 3.6 – Choice of features.

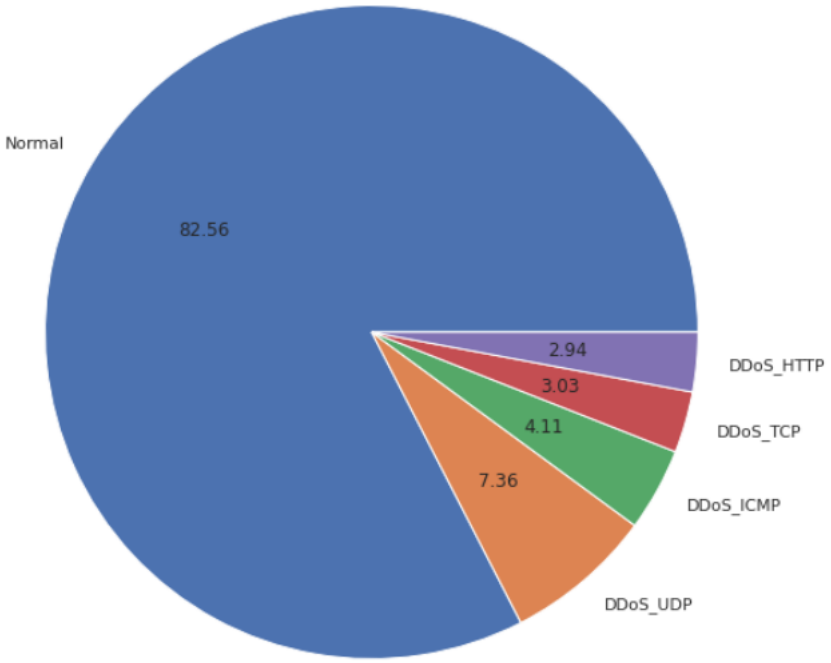


FIGURE 3.6 – Normal traffic percentage vs different attacks.

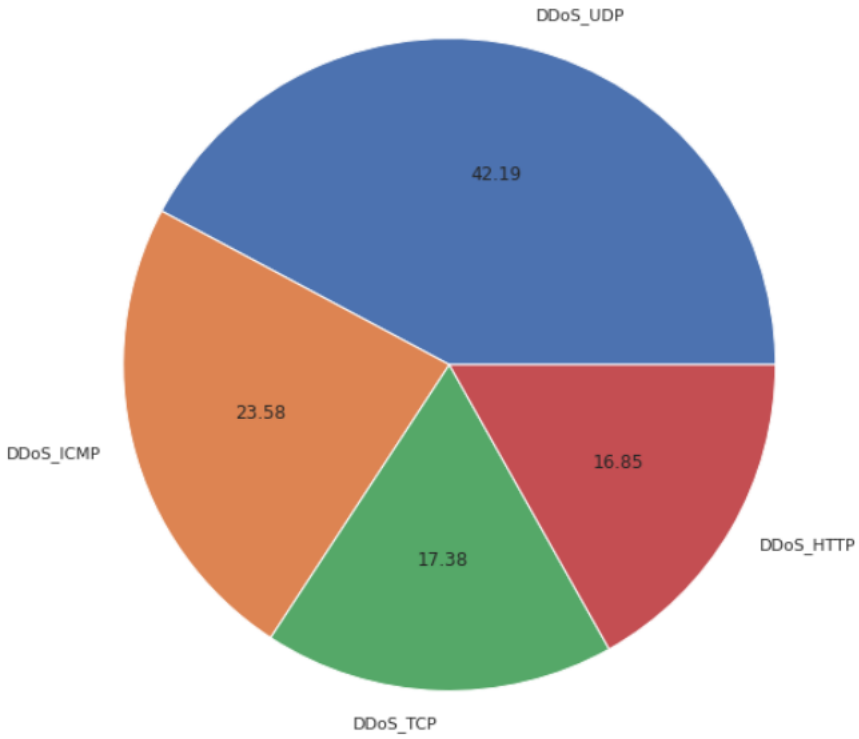


FIGURE 3.7 – Percentage of Botnet Attacks

been encoded with a popular technique called "One-Hot-Encoding". The coding will convert rows containing categories to their own column with a value 1 means true (this instance is of this class) or 0 means false (this instance is not of this class)

The last step in this procedure is normalization(also called standardization), the purpose of this step is to make the input data values between[-1,1], it influences the model building by reducing the learning rate, model training converges quickly, besides to other effects.

After we get a well-treated data, well presented then it's time to identify test and training data, so we divided the data into two parts(0.3 for the test and the rest for the train).

3.6 Realisation of a Botnets Detection System :

In the realization of this system, we tried to use 2 different models of deep learning, which are : deep neural network (DNN), and recurrent neural network (RNN). those models have been applicated on the "DNN-EdgeIIoT-dataset.csv" with 2 different experiments :

Binary classification : For this type of classification we have two classes to treat either in the DNN model or the RNN model (Table 3.8) after splitting the treated data into 70% for training the rest for the test, so the data have been presented in two classes class for normal traffic and the other one for the threats (Botnets Attack).
The training data is for model training and validation, while the test data is for evaluation. this phase is about evaluating the efficiency of the model in Botnets detection.

Multi-classifications(5 classes) : In this experiment we have 5 classes one for normal trafic and 4 for different Botnet attack classes (Table 3.9) this experiment is about testing the reliability of the detection of the different Botnet attacks In this case, the system efficiency depends on the detection rate and the total classification accuracy.

Class	NB of training instances	NB of test instances
Normal	954798,6	409199,4
Attaque	201678,4	86433,6
Total	1156477	495633

TABLE 3.8 – Binary classification subsets.

classes	NB of training instances	NB of test instances
Normal	954798,6	409199,4
DDoS-HTTP	33980,8	14563,2
DDoS-ICMP	47557,3	20381,7
DDoS-TCP	35043,4	15018,6
DDoS-UDP	85096,9	36470,1
Total	1156477	495633

TABLE 3.9 – Multi-classification subsets.

3.6.1 Models Structure :

The models we chose to work with from all types of Deep Learning methods are : the Deep Neural Network (DNN), and Recurrent Neural Network (RNN), the main structure of each one of them is completely different from the other, in addition to the modifications approved by trying different combinations and parameters. However that doesn't mean that there are no similarities, these proposed model structures have multiple common points :

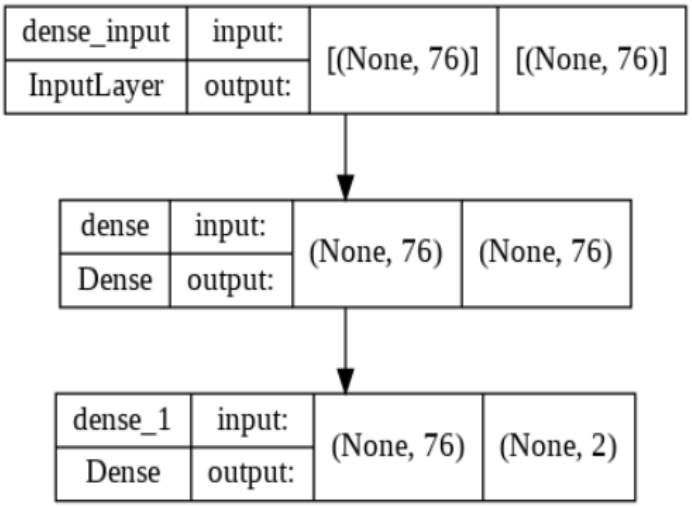
The input layers have the same dimensions (number of neurons) as the number of features (Features) in the input vector for each model.

The activation function used was "Sigmoid Activation Function" , this function have been selected over other functions that been tested and experienced.

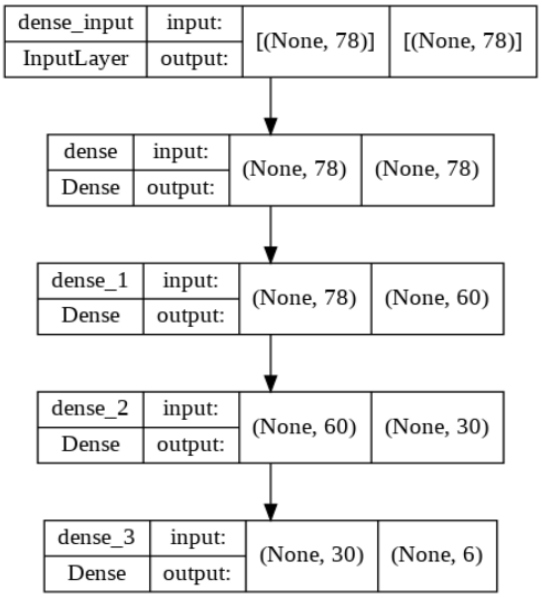
The activation function used for DNN multiclassification was " Softmax function" The output layers have the same dimensions as the number of classes, for the multi-class classification we have chose the "Softmax" activation function, gives a probability (whose sum is 1) as an output neurons (each at time), the greatest probability as an

output is then the responsible one to make the decision that its associated class is the predicted class.

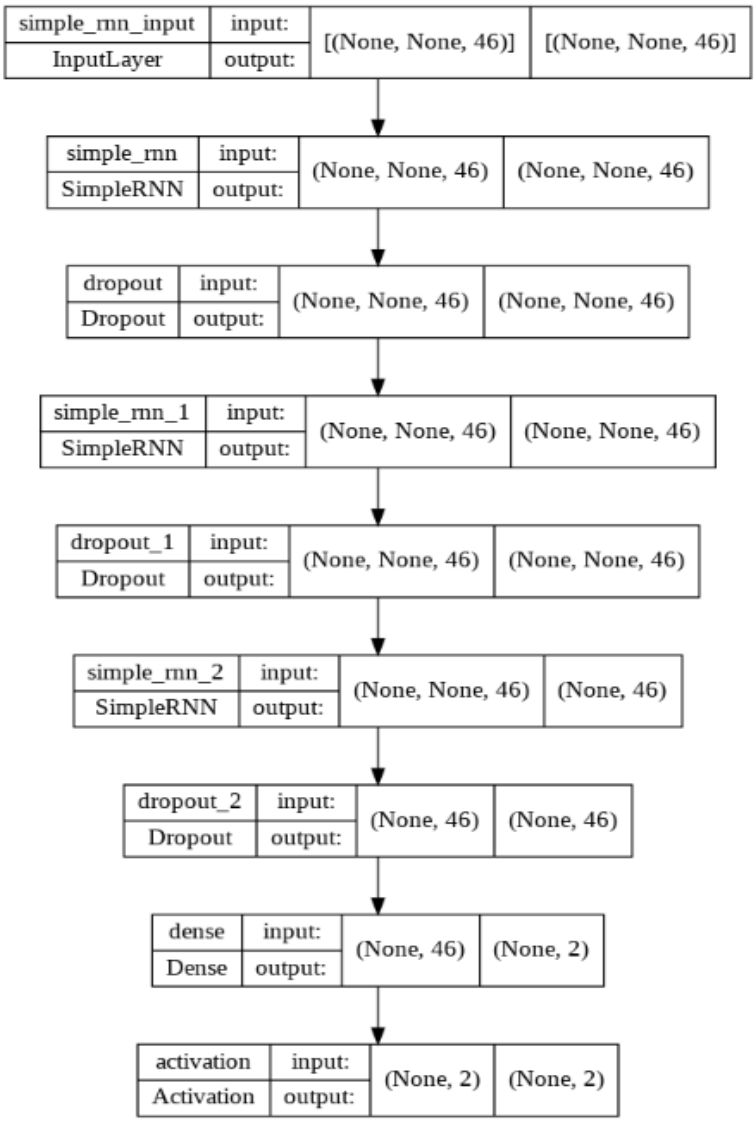
The "categorical_crossentropy" function was selected as a loss function in multi-class classification, however the “binary-cross-entropy” for binary (normal/attack) classification. and for the optimization we chose "Adam" for the reason that it gives us more efficiency in our case.



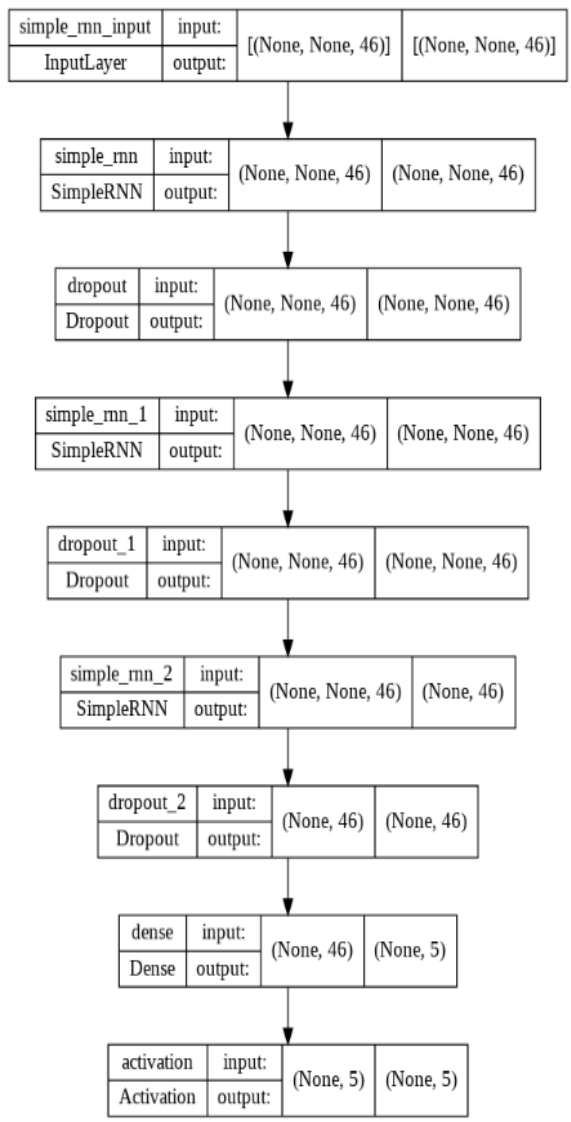
(A) the DNN model used for binary classification



(B) the DNN model used for 5-Classes classification



(C) the RNN model used for binary classification



(D) the RNN model used for 5-Classes classification

FIGURE 3.8 – The architecture of the models proposed for the "deep learning" classification

3.7 Botnets detection system based on Deep Neural Network (DNN) model

The main structure for the DNN model used has many hidden layers, with input and output layers (Fig. 3.8)
Those layers have a very important role in the classification with defined parameters that we tried to define the right values so we can have the best results. the DNN model can extract automatically complex characteristics and information from raw data. This is for the purpose of determining the underlying statistical properties of normal packets and packets of different attacks. More hidden layers lead to a more complex model, the results can be better, but they can lead to over-learning. After setting the activation functions, the number of samples per batch, and the optimization function. the model converges faster than the CNN model. This model was trained over 10 epochs.

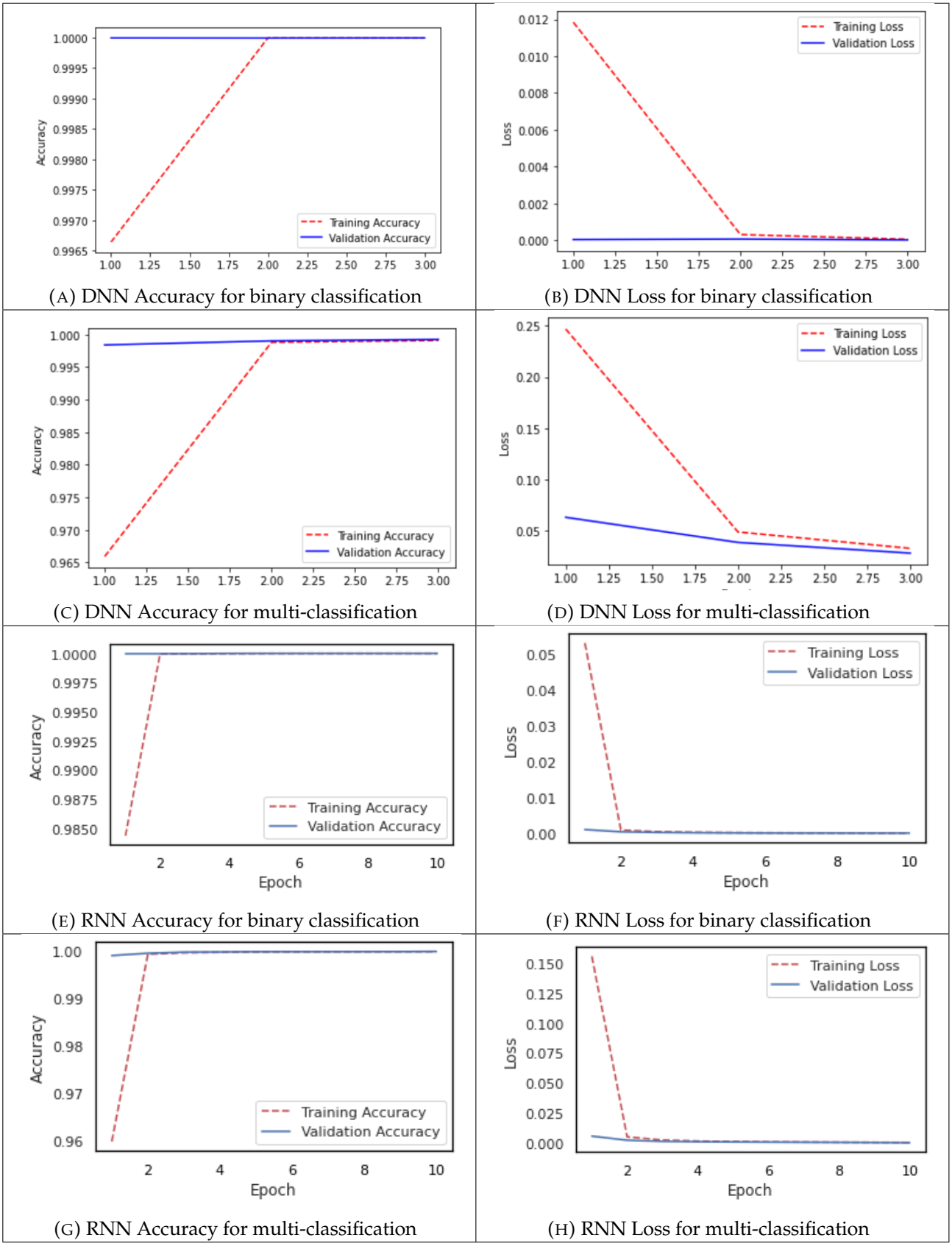
3.8 Botnets detection system based on Recurrent Neural Network (RNN) model

the same thing for the RNN model it is constructed from multiple layers, with input and output layers (Figure3.8)
the RNN model has also a combination of parameters that make the difference in the classification so we tried to choose the right ones that fit with the data we're working on

3.9 Results

We have realized 2 models of deep learning, Deep Neural Network (DNN), and Recurrent Neural Network (RNN). Those models have been realized and tested on

the treated and normalized data from DNN-EdgeIIoT-dataset. We have experimented with many tests in order to get the right and fitted parameters for everyone (each model). Those parameters play the main role in the training phase because they may influence the performance and the efficiency of the model in this step, so we are not able to make any changes to those parameters before the training ends. They include the variables that determine the structure of the network (Nbr of neurons, Nbr of layers, activation function, etc.), the batch of samples (Batch Size) and the number of iterations, etc. once we reach the results we want (minimum error rate and the maximum accuracy), then we will be able to apply the model to the test data. The results of the models are illustrated in Fig. 3.9 with loss graphs and accuracy graphs, those results become after training the model on 70% of the DNN-EdgeIIoT-dataset, and evaluating its performance on the 30% rest (test phase), both of model were trained in 10 epochs. As we can see in the same figure(Figure3.9)for both models, the accuracy for the two phases (training and validation) constantly increases from the start to the end (devolve into 1). while the loss graphs decrease and reach a minimum value that tends toward 0. which means that after every single epoch the model improves its training (learns better).



(A) DNN Accuracy for binary classification

(B) DNN Loss for binary classification

(C) DNN Accuracy for multi-classification

(D) DNN Loss for multi-classification

(E) RNN Accuracy for binary classification

(F) RNN Loss for binary classification

(G) RNN Accuracy for multi-classification

(H) RNN Loss for multi-classification

FIGURE 3.9 – Accuracy and loss curves of the proposed models with respect to the training and validation epochs

3.10 Evaluation metrics :

The efficiency and performance of the Botnets detection system may be measured by some metrics like accuracy, precision, recall, FScore ... etc. Some of those metrics are discussed below : TP : this value represents that the data instances are correctly predicted as an Attack by the classifier. TN : this value represents that the data instances are correctly classified as Normal instances. FP : this value represents that the data instances were wrongly classified as an Attack. FN : This value represents that the data instances were wrongly predicted as Normal instances.

— Precision (pr) and it is given by :

$$Pr = \frac{TP}{TP + FP}$$

— Recall (Rc) : Or we can call it the Detection Rate :

$$Rc = \frac{TP}{TP + FN}$$

— F1-score ($F1$) : It also called as F-Measure, and it is given by :

$$F1 = \frac{2 * (Acc * Rc)}{(Acc + Rc)}$$

— Confusion Matrix : As depicted in Fig. 3.10 and Fig. 3.11, is a specific array layout allowing to visualize the performance of an ML algorithm for a classification problem, it is known as the error matrix.

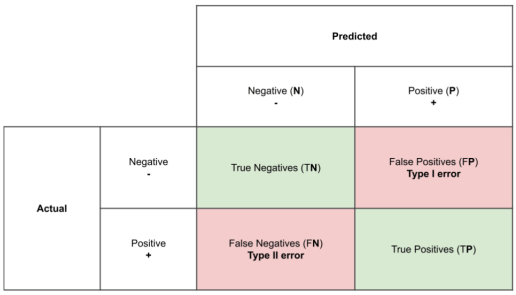
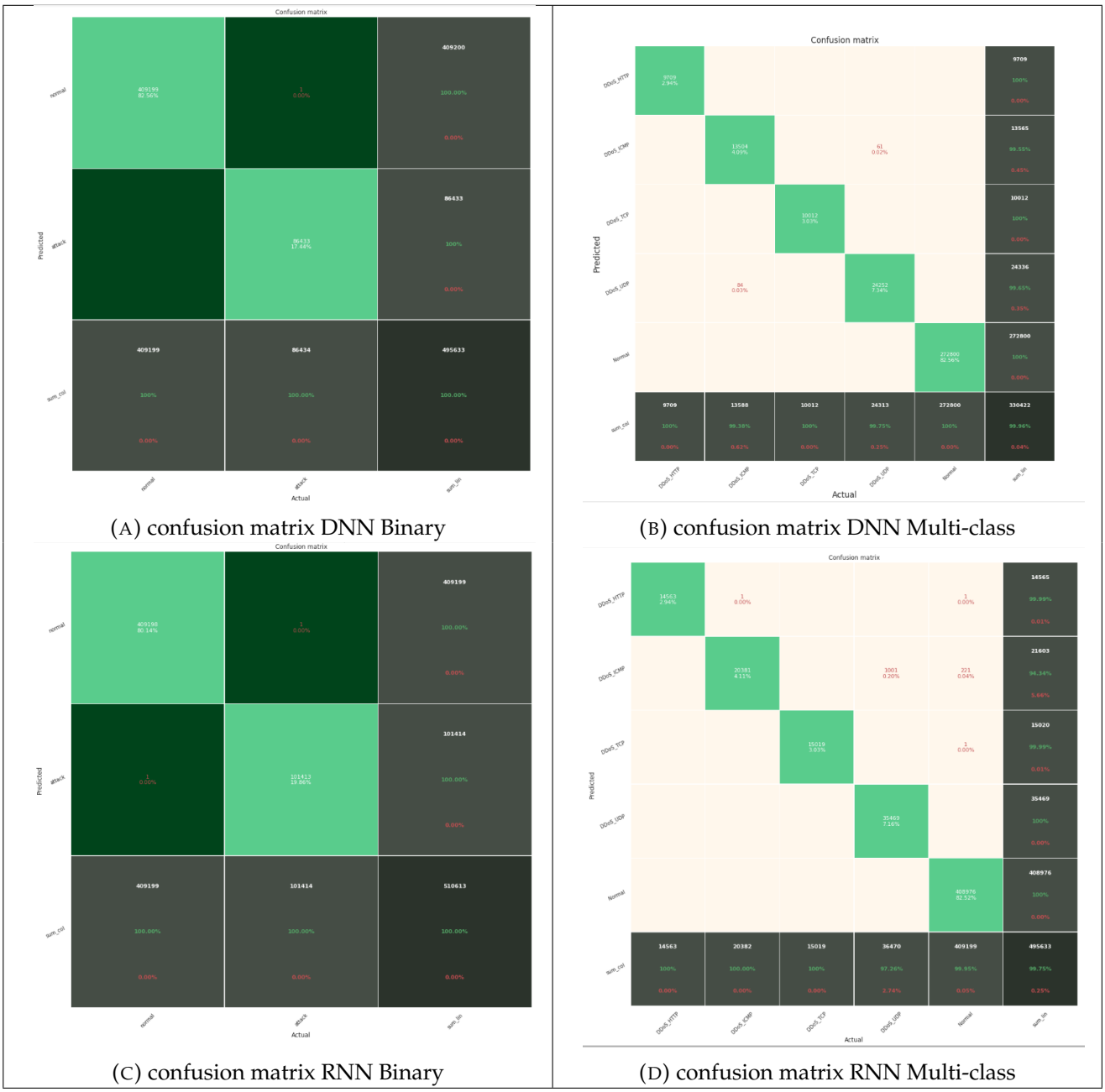


FIGURE 3.10 – Confusion Matrix [25]



(A) confusion matrix DNN Binary

(B) confusion matrix DNN Multi-class

(C) confusion matrix RNN Binary

(D) confusion matrix RNN Multi-class

FIGURE 3.11 – confusion matrix detailed

3.11 Conclusion

We have presented in this chapter the different tools as well as the different stages that been followed in the Botnet detection system that we realized.

GENERAL CONCLUSION

Cyber attacks have become stronger and developed day by day, these attacks present a real danger to companies and systems, for this the today's developers of cyber-security systems aim to find new techniques for detecting intrusions based on new methods. So they choosed the Deep Learning for that. After the research we have done and the reading of the existing works we tried to realize a botnet attacks detection system for the purpose of protecting systems from this type of attacks, during the realization of this project we have at the new Dataset called "Edge-IIoTset dataset" created by Dr.MOHAMED AMINE FERRAG, after that we've treated the data in a way to make it easier to use and reach good results, we applied a number of preprocessing methods. The Deep Learning model that we chose were the Deep Neuron Network (DNN) and Recurrent Neuron Network (RNN) with 2 type of classification : Binary classification and Multi-classification. the realization of the two models wasn't easy the deficulty was in the parameters choice, because the parameters (activation function,...) are the main characters to define the effectiveness of the model.
After all that, after treating the data and realizing the model and define the wright parameters we reached a good results that have been discussed in precedent chapters.

BIBLIOGRAPHY

[1] Zeeshan AHMAD et al. « Network intrusion detection system : A systematic study of machine learning and deep learning approaches ». In : *Transactions on Emerging Telecommunications Technologies* 32.1 (2021), e4150.

[2] P AMBIKA. « Machine learning and deep learning algorithms on the Industrial Internet of Things (IIoT) ». In : *Advances in computers* 117.1 (2020), p. 321-338.

[3] Imad BOUTERAA, Makhlouf DERDOUR et Ahmed AHMIM. « Intrusion Detection using Data Mining : A contemporary comparative study ». In : *2018 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS)*. IEEE. 2018, p. 1-8.

[4] ABDERRAOUF CHERIFI. « Un Framework de sécurité pour détecter les malwares Android ». In : (2021).

[5] Smita DANGE et Madhumita CHATTERJEE. « IoT Botnet : the largest threat to the IoT network ». In : *Data Communication and Networks*. Springer, 2020, p. 137-157.

[6] Mohamed Amine FERRAG et al. « Deep learning for cyber security intrusion detection : Approaches, datasets, and comparative study ». In : *Journal of Information Security and Applications* 50 (2020), p. 102419.

[7] Mohamed Amine FERRAG et al. « Edge-IIoTset : A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning ». In : (2022).

[8] GEEKS. https ://www.geeksforgeeks.org/difference-between-iiot-and-iot/.

[9] GOOGLE. `https:research.google.com/colaboratory/faq.html?hl=fr`.

[10] GOOGLE. `https://www.tensorflow.org/`.

[11] GOOGLE. `https://keras.io/`.

[12] GOOGLE. https ://numpy.org/doc/stable/.

[13] GOOGLE. https ://pandas.pydata.org/docs/.

[14] Djallel HAMOUDA et al. « Intrusion Detection Systems for Industrial Internet of Things : A Survey ». In : *2021 International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS)*. IEEE. 2021, p. 1-8.

[15] Mohammad Mehedi HASSAN et al. « A hybrid deep learning model for efficient intrusion detection in big data environment ». In : *Information Sciences* 513 (2020), p. 386-396.

[16] IONOS. https ://www.wallarm.com/what/what-is-a-botnet.

[17] V KANIMOZHI et T Prem JACOB. « Artificial intelligence based network intrusion detection with hyper-parameter optimization tuning on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing ». In : *2019 international conference on communication and signal processing (ICCSP)*. IEEE. 2019, p. 0033-0036.

[18] V KANIMOZHI et T Prem JACOB. « Calibration of various optimized machine learning classifiers in network intrusion detection system on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing ». In : *International Journal of Engineering Applied Sciences and Technology* 4.06 (2019), p. 209-213.

[19] Sydney Mambwe KASONGO et Yanxia SUN. « A deep learning method with wrapper based feature extraction for wireless intrusion detection system ». In : *Computers & Security* 92 (2020), p. 101752.

[20] KASPERSKY. https://usa.kaspersky.com/resource-center/threats/botnet-attacks.

[21] Ruhul Amin KHALIL et al. « Deep learning in the industrial internet of things : Potentials, challenges, and emerging applications ». In : *IEEE Internet of Things Journal* 8.14 (2021), p. 11016-11040.

[22] Chaouki KHAMMASSI et Saoussen KRICHEN. « A NSGA2-LR wrapper approach for feature selection in network intrusion detection ». In : *Computer Networks* 172 (2020), p. 107183.

[23] Hyun Young KIM et al. « Development and Evaluation of Self-Management Program for Patients with Coronary Artery Disease ». In : *Journal of Multimedia Information System* 6.4 (2019), p. 317-322.

[24] Liran LERMAN, Olivier MARKOWITCH et Gianluca BONTEMPI. *Les systèmes de détection d'intrusion basés sur du machine learning*. 2008.

[25] NBSHARE. https://www.nbshare.io/notebook/626706996/Learn-And-Code-Confusion-Matrix-With-Python/.

[26] Rajendra PATIL, Harsha DUDEJA et Chirag MODI. « Designing an efficient security framework for detecting intrusions in virtual network of cloud computing ». In : *Computers & Security* 85 (2019), p. 402-422.

[27] Azam RASHID, Muhammad Jawaid SIDDIQUE et Shahid Munir AHMED. « Machine and deep learning based comparative analysis using hybrid approaches for intrusion detection system ». In : *2020 3rd International Conference on Advancements in Computational Sciences (ICACS)*. IEEE. 2020, p. 1-9.

[28] RESEARCHGATE. https ://www.researchgate.net/figure/From-Industry-10-to-Industry-50$_{f}ig1_{3}$36816740.

[29] Markus RING et al. « A survey of network-based intrusion detection data sets ». In : *Computers & Security* 86 (2019), p. 147-167.

[30] Derrick ROUNTREE. *Security for Microsoft Windows system administrators : introduction to key information security concepts*. Elsevier, 2011.

[31] T SARANYA et al. « Performance analysis of machine learning algorithms in intrusion detection system : A review ». In : *Procedia Computer Science* 171 (2020), p. 1251-1260.

[32] Dimitrios SERPANOS et Marilyn WOLF. « Industrial internet of things ». In : *Internet-of-Things (IoT) Systems*. Springer, 2018, p. 37-54.

[33] Emiliano SISINNI et al. « Industrial internet of things : Challenges, opportunities, and directions ». In : *IEEE transactions on industrial informatics* 14.11 (2018), p. 4724-4734.

[34] S SMYS, Abul BASAR, Haoxiang WANG et al. « Hybrid intrusion detection system for internet of things (IoT) ». In : *Journal of ISMAC* 2.04 (2020), p. 190-199.

[35] Jianwu ZHANG et al. « Model of the intrusion detection system based on the integration of spatial-temporal features ». In : *Computers & Security* 89 (2020), p. 101681.