الجـــمـــهـوريـــة الجـــزائـــريـــة الديمـقـــراطـيـــة الـشـــعـــبـيـــة
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التـعـــلـيـــم العـــالي و البـحث العـــلـــمـــي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جـــامعة أبي بـكـــر بـــلـــقـــايـد – تـــلمســـان –
Université Aboubakr Belkaïd – Tlemcen –
Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications

Spécialité : Réseaux et Télécommunications

Par : Bensenouci Dounia

Sujet

Détection des attaques DDoS à l'aide du Deep Learning

Soutenu publiquement, le 04/06/2023, devant le jury composé de :

M MOUSSAOUI Djilali	MCA	Université de Tlemcen	Président
M BOUABDALLAH Réda	MAA	Université de Tlemcen	Examinateur
M HADJILA Mourad	MCA	Université de Tlemcen	Encadreur
M BOUIDAINE El Baraa	Doctorant	Université de Tlemcen	Co-Encadreur
Mme FERHI Wafaa	Doctorante	Université de Tlemcen	Invitée

Année universitaire : 2022/2023

Remerciements

Ce mémoire est le fruit d'un long et laborieux travail. J'ai sacrifié une grande partie de mon temps pour l'élaborer. Je tiens tout d'abord à remercier, Allah tout puissant, le miséricordieux de m'avoir donné la santé, la volonté, le courage et la patience pour achever ce mémoire.

Je remercie vivement mon encadrant, Monsieur ***HADJILA Mourad****, professeur à l'Université Aboubakr Belkaïd de Tlemcen, pour ses précieux conseils qui m'ont permis d'atteindre mon objectif final. Il a toujours accepté de répondre à mes questions et de m'orienter. Il a témoigné d'une grande patience à mon égard. Allah vous bénisse !*

Je tiens à remercier, Madame ***FERHI Wafaa*** *pour l'aide et le soutien que vous m'avez apportés. Votre collaboration a facilité ma tâche, je vous souhaite donc une excellente continuation et la réussite dans votre doctorat.*

Je tiens aussi à remercier, Monsieur ***BOUIDAINE El Baraa*** *pour ses conseils, et je vous souhaite la réussite dans votre doctorat.*

Je remercie vivement, Monsieur ***MOUSSAOUI Djilali*** *maître de conférences à l'université Université Aboubakr Belkaïd d'avoir accepté de présider le jury de soutenance et Monsieur* ***BOUABDALLAH Réda*** *maître assistant à l'université Abou-Bekr Belkaid d'avoir examiné mon travail.*

Je désire aussi remercier mes professeurs du département de télécommunications de l'Université de Tlemcen en particulier Messieurs

A. BOUACHA, H .ZERROUKI, M. HADJILA, M. BOUSAHLA,

N. HASSAINE et S. BELAROUCI *pour la qualité de leur enseignement et surtout pendant la période de la Covid. Allah les bénisse !*

Dédicaces

Je dédie cet ouvrage à mes chers parents pour leur soutien, leur patience, leurs sacrifices et leurs encouragements durant mon parcours scolaire. Allah les protège.

A ma sœur et à mon frère, à qui je souhaite la réussite.

A mes grands parents paternels et maternels à qui je souhaite une longue vie avec une bonne santé.

A ma tante ***Hafida*** *qui m'a soutenue moralement.*

A toute ma famille.

A ***Rahma****,* ***Feriel*** *et* ***Fatima****, mes amis proches qui m'ont toujours aidée et accompagnée pendant ce parcours.*

A mes amis avec qui j'ai passé des moments inoubliables. Je leur souhaite la réussite.

Résumé

Une attaque par déni de service distribué (DDoS) est une attaque massive, distribuée, délibérée et coordonnée par plusieurs machines compromises pour submerger un service en ligne ou un serveur. Les attaquants tentent de s'attaquer à la disponibilité du service en envoyant des données volumineuses pour que la machine cible soit à court de ressources. Par conséquent, ce travail a proposé un réseau neuronal feedforward (modèle de Deep Learning) pour détecter et classer les attaques DDoS en utilisant quatre techniques de classification : la classification binaire, la classification multi-classe avec label-encoding, la classification multi-classe avec one-hot encoding, et la classification multi-label. Les expériences ont été menées sur le dataset CSE-CIC-IDS2018-DDoS produite à partir du dataset CSE-CIC-IDS2018. Un prétraitement de données doit être mis en place afin de supprimer les lignes dupliquées et les valeurs manquantes, convertir les données catégorielles en données numériques et standardiser les caractéristiques afin que toutes les valeurs se situent dans la même plage de valeurs. La performance du modèle proposé sur l'ensemble de données CSE-CIC-IDS2018-DDoS, qui contient des types d'attaques DDoS a été évaluée à l'aide de différentes mesures d'évaluation telles que (Accuracy, Precision, Recall, F1-score).

Mots clés : Sécurité des réseaux, attaque DDoS, CSE-CIC-IDS2018 dataset, prétraitement, apprentissage profond, classification multi-classe, classification multi-label, classification binaire.

Abstract

A Distributed Denial of Service (DDoS) attack is a massive, distributed, deliberated, and coordinated attack by multiple compromised machines to overwhelm an online service or a server. Attackers attempt to attack the availability of the service by sending voluminous dummy data to make target machine fall short of resources. Therefore, this work proposed a feedforward neural network (deep learning model) to detect and classify DDoS attacks by incorporating four classification techniques: binary classification, multiclass classification with label encoding, multiclass classification with one-hot encoding, and multi-label classification. Experiments were carried on CSE-CIC-IDS2018-DDoS dataset containing DDoS attack types which be produced from CSE-CIC-IDS2018 dataset. This dataset has undergone pre-processing such as removing duplicate observations, erasing missing values, converting categorical data to numerical data, and performing features scaling so that all values are within the same range. The performance of the proposed model on the CSE-CIC-IDS2018-DDoS dataset, which contains DDoS attack types, has been evaluated using different evaluation metrics such as (Accuracy, Precision, Recall, F1-score).

Key words: Network security, DDoS Attack, CSE-CIC-IDS2018 Dataset, pre-processing, Deep Learning, multiclass classification, multi-label classification, binary classification.

Table Des Matières

Liste Des Figures

Liste Des Tableaux

Liste Des Abréviations

ANN	Artificial Neural Network
AWS	Amazon Web Services
CIC	Canadian Institute for Cybersecurity
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSE	Communications Security Establishment
CSV	Comma-Separated Values
DDoS	Distributed Denial of Service
DL	Deep Learning
DNN	Deep Neural Network
DNS	Domain Name System
DoS	Denial of Service
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
FTP	File Transfer Protocol
GB	Giga Byte

GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
HOIC	High Orbit Ion Cannon
HTTP	Hypertext Transfer Protocol
IA	Artificial Intelligence
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
INF	Infinity
IRC	Internet Relay Chat
KNN	K-Nearest Neighbors
LOIC	Low Orbit Ion Cannon
LSTM	Long Short Term Memory
MB	Mega Bytes
ML	Machine Learning
MLP	Multi-layer Perceptron
NaN	Not a Number
NID	Network-based Intrusion Detection
NTP	Network Time Protocol
RAM	Random-Access Memory
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SGD	Stochastic gradient descent

SQL	Structured Query Language
SSH	Secure Shell
SSL	Transport Layer Security
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
TPU	Tensor Processing Unit
TTL	Time to Live
UDP	User Datagram Protocol
VANET	Vehicular Ad-Hoc Network
XSS	Cross Site Scripting

Introduction Générale

Une cyberattaque est une tentative malveillante d'exploiter les vulnérabilités d'un système informatique ou d'un réseau afin de compromettre la confidentialité, l'intégrité ou la disponibilité des données qui y sont stockées. Les attaques peuvent prendre diverses formes, telles que des attaques par déni de service distribué (DDoS), des logiciels malveillants, des attaques par phishing, etc.

Les attaques DDoS (Distributed Denial of Service) sont devenues l'une des principales menaces pour les entreprises et les organisations du monde entier. Ces attaques visent à perturber les services en ligne en saturant les serveurs ciblés avec un trafic anormal, empêchant ainsi les utilisateurs légitimes d'accéder aux ressources. Les attaques DDoS ont évolué au fil des années, devenant plus sophistiquées, plus fréquentes et plus difficiles à détecter. Les entreprises de toutes tailles sont vulnérables, et les coûts des interruptions de service peuvent être astronomiques. Des géants de la technologie comme Google et Amazon n'ont pas été épargnés par ces attaques.

En 2018, Google a subi une attaque DDoS massive, qui a perturbé ses services pendant plusieurs heures. Les pirates ont utilisé une technique appelée "Memcached Amplification", qui consiste à utiliser des serveurs Memcached mal configurés pour renvoyer des réponses très volumineuses à des requêtes simples, submergeant ainsi le serveur cible. Cette attaque a été considérée comme la plus grande attaque DDoS jamais enregistrée à l'époque.

En février 2021, Amazon a subi également une attaque DDoS importante, qui a perturbé les services de l'entreprise pendant plusieurs heures. Les pirates ont utilisé une technique appelée "Connection Flood", qui consiste à envoyer un grand nombre de demandes de connexion à un serveur en même temps, le submergeant et le rendant inaccessible. Amazon a rapidement réagi en bloquant l'adresse IP de l'attaquant, mais l'incident a néanmoins eu un impact négatif sur les services de l'entreprise.

Pour détecter ces attaques, les techniques de détection basées sur des règles ont été utilisées, mais elles ont montré leurs limites en raison de leur manque d'adaptabilité aux nouvelles attaques et de leur forte sensibilité aux faux positifs. De plus, les attaquants utilisent de plus en plus des techniques sophistiquées pour masquer leurs attaques, ce qui rend encore plus difficile la détection des attaques DDoS.

Face à ces défis, les professionnels de la cybersécurité ont cherché de nouvelles approches pour détecter et prévenir les attaques DDoS. L'une des approches les plus prometteuses est l'utilisation de l'apprentissage en profondeur (Deep Learning), une branche de l'intelligence artificielle qui permet aux ordinateurs d'apprendre à partir de données et de prendre des décisions autonomes. Les algorithmes de Deep Learning ont démontré leur capacité à détecter les attaques DDoS avec une précision élevée et à bloquer les trafics malveillants avant qu'ils ne causent des dommages importants.

Dans ce travail, nous proposons d'explorer l'utilisation d'algorithmes de Deep Learning pour la détection des attaques DDoS en utilisant des données de trafic réseau pour former et tester nos modèles. Nous évaluerons également la performance de différents algorithmes de Deep Learning pour la détection des attaques DDoS et comparerons leurs performances avec d'autres techniques de détection existantes.

Le manuscrit de ce mémoire est structuré en trois chapitres :

Dans le premier chapitre, nous aborderons, les attaques DDoS, leur architecture, leurs types, les différentes techniques utilisées pour réaliser une attaque DDoS, les problèmes de défense DDoS, et la classification des mécanismes de défense contre les attaques.

Le second chapitre est composé de deux parties. Nous présenterons dans la première le concept de Machine Learning(ML) en donnant les trois types d'apprentissage à savoir l'apprentissage supervisé, non supervisé et par renforcement, suivi par les algorithmes d'apprentissage supervisé. La seconde partie sera consacrée au Deep Learning(DL) où nous allons donner la différence entre le ML et le DL puis nous aborderons la notion de réseaux de neurones, leur structure et leurs types.

Le troisième et dernier chapitre sera consacré à l'implémentation de trois méthodes dédiées à la détection et la classification des attaques DDoS en utilisant la classification binaire, la classification multi-classe avec le "Label Encoding" et le "One-Hot Encoding" et la classification multi-label, suivie par une interprétation des résultats obtenus.

Finalement, nous couronnons ce manuscrit par une conclusion générale et des perspectives.

Chapitre 1

Les attaques par déni de service distribué (DDoS)

1.1 Introduction

La sécurité est le plus grand défi et le souci majeur dans l'environnement Internet. Les menaces les plus courantes incluent les attaques de contrôle d'accès, l'injection et l'exécution de logiciels malveillants et l'usurpation d'identité [1]. L'une des plus grandes menaces de sécurité et l'une des plus difficiles à prévoir est l'attaque par déni de service distribué. DDoS est un problème en croissance rapide dans l'environnement de réseau d'aujourd'hui. Il utilise encore d'anciennes techniques de configuration manuelle des topologies de réseau [2].

Dans ce chapitre, nous allons parler des attaques par déni de service distribué (DDoS). Tout d'abord, nous allons expliquer ce qu'est une attaque par déni de service DoS et celle de DDoS, puis présenter ses différents types, son architecture et ses principales techniques. Enfin nous allons terminer par les méthodes de défense contre les attaques DDoS.

1.2 Anomalies dans les réseaux

Les attaques dans un réseau sont également appelées anomalies dans le trafic du réseau. En général, les anomalies ou les attaques sont des événements de réseau qui s'écartent du comportement normal, attendu ou habituel, et qui sont suspects du point de vue de la sécurité. De telles anomalies dans un réseau peuvent être dues à deux types de raisons : liées à la performance et liées à la sécurité. Étant donné que de nombreuses entités individuelles sont réunies pour former un réseau informatique afin de contribuer à la fourniture de services de communication complexes, des problèmes peuvent survenir au niveau de l'une ou l'autre des entités en interaction. Une anomalie liée à la performance peut se produire en raison d'un mauvais fonctionnement des dispositifs du réseau, par exemple en raison d'une mauvaise configuration du routeur. En revanche, les anomalies liées à la sécurité sont dues à des tentatives ou à des activités malveillantes visant à perturber le fonctionnement normal du réseau [3].

1.3 Attaques par déni de service

Les attaques par déni de service (DoS) sont l'une des méthodes de cyber attaque les plus courantes dans le domaine de la sécurité des réseaux. Elles empêchent les utilisateurs légitimes d'accéder au réseau et à d'autres ressources. Les attaquants y parviennent en inondant le réseau de faux trafic (nombre excessif de paquets inutiles) qui dépassent la capacité du serveur ce qui entraîne un déni de service pour les demandes supplémentaires (voir la figure 1.1) [4]. Les victimes des attaques DoS sont souvent les organisations serveurs web bien connues telles que les banques, les entreprises commerciales et les médias [5].

Les attaques DoS peuvent durer de quelques heures à plusieurs mois et peuvent coûter du temps et de l'argent aux entreprises pendant que leurs ressources et services sont indisponibles [6].

Avec le temps, l'attaque DoS a évolué vers une attaque par déni de service distribué (DDoS) où l'attaquant compromet d'autres machines vulnérables sur l'Internet pour coordonner l'attaque à un moment donné sur la machine victime, multipliant ainsi l'effet du déni de service [7].

Figure 1.1 : Attaque DoS.

1.4 Attaques par déni de service distribuées

1.4.1 Définition

Le déni de service distribué DDoS est une attaque coordonnée à grande échelle contre la disponibilité des services d'un système victime ou des ressources d'un réseau, lancée indirectement par l'intermédiaire de nombreux ordinateurs compromis appelés «zombies» sur l'internet. En utilisant la technologie client/serveur, l'attaquant est en mesure de multiplier l'efficacité de l'attaque, de manière significative, en utilisant les ressources de nombreux zombies accomplis à leur insu, qui servent de plateformes d'attaque (voir la figure 1.2). Les zombies effectuent l'attaque proprement dite en augmentant considérablement le trafic vers une machine victime. En conséquence, la machine victime perd toutes ses ressources de calcul et de communication [8].

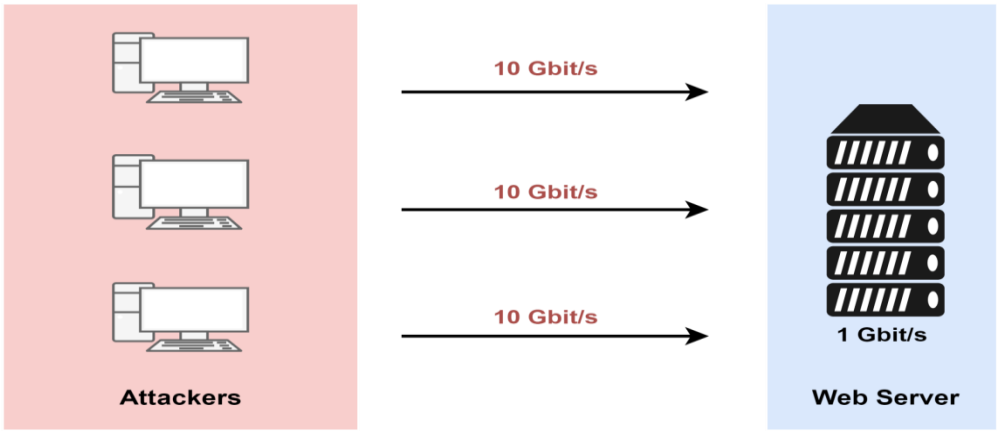

Figure 1.2 : Attaque DDoS.

1.4.2 Catégories de cibles d'attaques DDoS

Les attaques DDoS ciblent les ressources du réseau et du serveur qui sont citées ci-dessous [9].

1.4.2.1 Bande passante

Les attaques par inondation/volume consomment toute la bande passante du réseau. Elles ne permettent pas aux demandes légitimes d'atteindre le serveur en épuisant le canal.

1.4.2.2 Mémoire

Les attaques basées sur le protocole, telle que l'attaque SYN, font que la connexion TCP est ouverte en permanence, ce qui entraîne un dépassement de la mémoire tampon. Elles consomment entièrement la mémoire tampon ou la mémoire de la table de connexion TCP.

1.4.2.3 CPU

Les attaques basées sur la couche d'application visent les serveurs web. Elles rendent le service indisponible pour les utilisateurs légitimes en épuisant la puissance de traitement de l'unité centrale ou du serveur, et le serveur tombe en panne.

1.4.3 Déni de service par amplification /réflexion

1.4.3.1 DDoS par amplification

Les attaques par amplification génèrent un volume important de paquets qui sont utilisés pour submerger le site web cible sans alerter l'intermédiaire. Cela se produit lorsqu'un service vulnérable répond par une réponse volumineuse quand l'attaquant envoie sa requête, souvent appelée paquet de déclenchement. À l'aide d'outils facilement disponibles, l'attaquant peut envoyer plusieurs milliers de requêtes aux services vulnérables, provoquant ainsi des réponses considérablement plus volumineuses que la requête initiale et amplifiant de manière significative la taille et la bande passante délivrées à la cible. L'amplification peut se traduire par l'envoi de plusieurs paquets de réponse pour un seul paquet, ou par des paquets plus volumineux que l'original. L'une ou l'autre méthode entraîne une amplification [10].

1.4.3.2 DDoS par réflexion

Une autre méthode appliquée par les attaquants DDoS est la méthode de réflexion, qui consiste à utiliser des serveurs non compromis pour acheminer le trafic vers la victime et l'aider à consommer sa bande passante. Cette méthode permet à l'attaquant d'envoyer indirectement du trafic à la victime et de ne pas être détecté (voir la figure 1.3). Dans cette méthode, tous les paquets d'attaque envoyés par l'attaquant, contiennent l'adresse IP de la victime dans le champ de l'adresse source des paquets IP. Lorsque le serveur reçoit ces demandes de service, il envoie sa réponse au nœud victime, et non au nœud source réel du paquet [11].

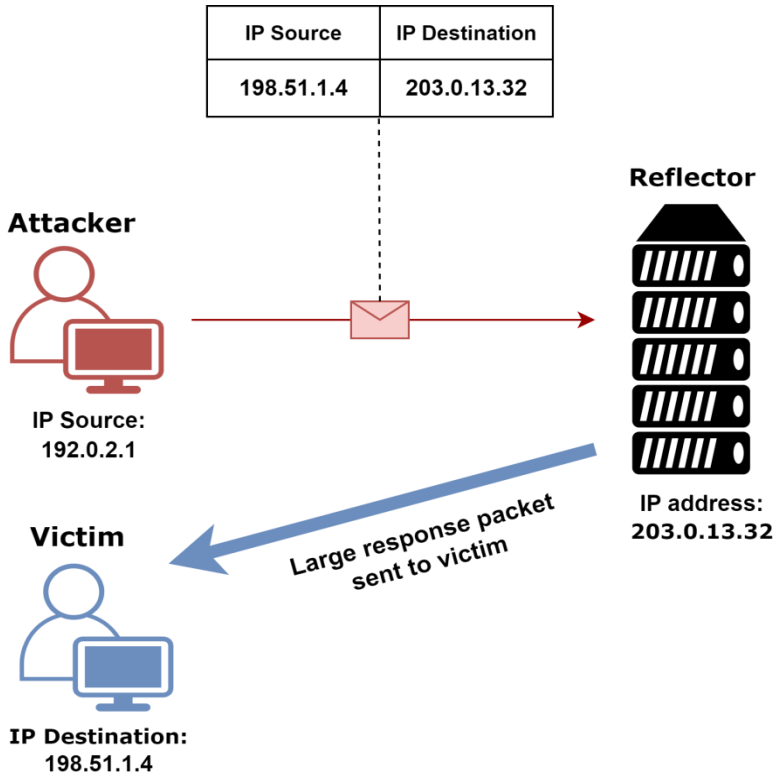

Figure 1.3 : DDoS par amplification /réflexion.

1.4.4 Classification des attaques DDoS

Les attaques DDoS sont généralement classées dans les trois catégories suivantes :

1.4.4.1 Attaques DDoS volumétriques

Les attaques volumétriques visent à consommer la bande passante du réseau et à la saturer par amplification ou par un réseau de zombies afin d'empêcher les utilisateurs d'y accéder. Celles-ci sont faciles à générer en dirigeant une quantité massive de trafic vers le serveur cible [12].

La taille des requêtes est généralement d'environ 100 Gbit/s. Cependant, les attaques récentes ont dépassé 1 Tbps [13].

Les attaques DDoS volumétriques les plus connues : l'amplification NTP, l'amplification DNS, l'attaque par inondation UDP et l'attaque par inondation TCP.

1.4.4.2 Attaques de protocole

Les attaques de protocole sont également connues sous le nom d'attaques par épuisement d'état. Ces attaques se concentrent sur les vulnérabilités des couches 3 et 4 de la pile de protocoles. Ces types d'attaques consomment des ressources telles que des serveurs, des pare-feu et des équilibreurs de charge [12].

1.4.4.3 Attaques d'application

Les attaques d'application sont des attaques ciblant les applications web d'une entreprise ou d'un individu.

Les vulnérabilités sont exploitées dans les protocoles de la couche application tels que HTTP et SSL. Le code de l'application peut également être vulnérable lorsque les pratiques de codage sécurisé ne sont pas prises en compte. Ces attaques sont les plus dangereuses car il n'est pas nécessaire de générer d'énormes quantités de trafic. Les attaques de la couche application sont très difficiles à détecter car elles sont furtives par nature et utilisent du trafic légitime [14].

1.4.5 Les étapes de l'orchestration d'une attaque DDoS

Pour lancer une attaque DDoS, un attaquant doit suivre les étapes suivantes :

1.4.5.1 Sélection des agents

L'attaquant choisit les agents qui effectueront l'attaque. La sélection des agents est basée sur l'existence de vulnérabilités dans ces machines qui peuvent être exploitées par l'attaquant afin d'y accéder. Il est important que les agents disposent de suffisamment de ressources pour pouvoir générer des flux d'attaques puissants [15].

1.4.5.2 Compromission

L'attaquant exploite les failles de sécurité et les vulnérabilités des machines de l'agent pour implémenter le code d'attaque. Il essaie également de protéger votre code contre la détection et la désactivation. Les propriétaires et les utilisateurs des systèmes d'agents ignorent souvent que leurs systèmes ont été compromis et qu'ils sont sur le point de participer à une attaque DDoS. Lors de la participation à une attaque DDoS, chaque programme agent n'utilise qu'une petite quantité de ressources (mémoire et bande passante), ce qui entraîne des changements de performances minimes pour les utilisateurs d'ordinateurs [15].

1.4.5.3 Communication

Avant que l'attaquant ne lance l'attaque, il communique avec les gestionnaires pour savoir quels agents peuvent être utilisés dans l'attaque, s'il est nécessaire de mettre à niveau les agents et quel est le meilleur moment pour planifier l'attaque. Les agents peuvent communiquer avec un ou plusieurs gestionnaires en fonction de la configuration du réseau d'attaque DDoS. Les protocoles utilisés pour la communication entre les gestionnaires et les agents sont le protocole de contrôle de transmission (TCP), le protocole de datagramme de l'utilisateur (UDP) et le protocole ICMP [15].

1.4.5.4 Attaque

À cette étape, l'attaquant commande le début de l'attaque. La victime, la durée de l'attaque et les caractéristiques spéciales de l'attaque, telles que le type, la durée, le temps de vie (TTL)

et les numéros de port, peuvent être ajustés. La variété des propriétés des paquets d'attaque peut aider l'attaquant à éviter d'être détecté [15].

1.4.6 Architecture des attaques DDoS

Comme le montre la figure 1.4, une attaque DDoS se compose de quatre éléments :

- L'attaquant.
- Les maîtres de contrôle (ou gestionnaires).
- Les agents (ou zombies)
- La victime.

L'attaquant commence par analyser des millions de machines sur l'internet pour trouver des machines vulnérables dont la sécurité peut être facilement exploitée. Ces machines sont appelées "maîtres" ou "handlers", car elles sont directement sous le contrôle de l'attaquant. Le processus de recrutement des "handlers" est entièrement automatisé et s'effectue par un balayage continu des machines distantes à la recherche d'éventuelles failles de sécurité. L'attaquant installe des codes malveillants dans ces machines infectées qui deviennent alors capables de déployer d'autres machines infectées [16]. Les machines déployées par les gestionnaires sont directement sous leur contrôle et sont connues sous le nom d'esclaves ou de zombies. L'attaquant contrôle indirectement ces machines par l'intermédiaire des gestionnaires. Sur le signal de l'attaquant, ces handlers et zombies sont utilisés pour lancer une attaque coordonnée sur la machine cible. Cette attaque rend la machine cible incapable de communiquer ou d'utiliser ses ressources. L'attaquant utilise souvent l'usurpation d'adresse IP dans les gestionnaires et les zombies pour masquer l'identité de ces machines. Cela permet à l'attaquant d'utiliser les mêmes machines pour créer une attaque DDoS [17].

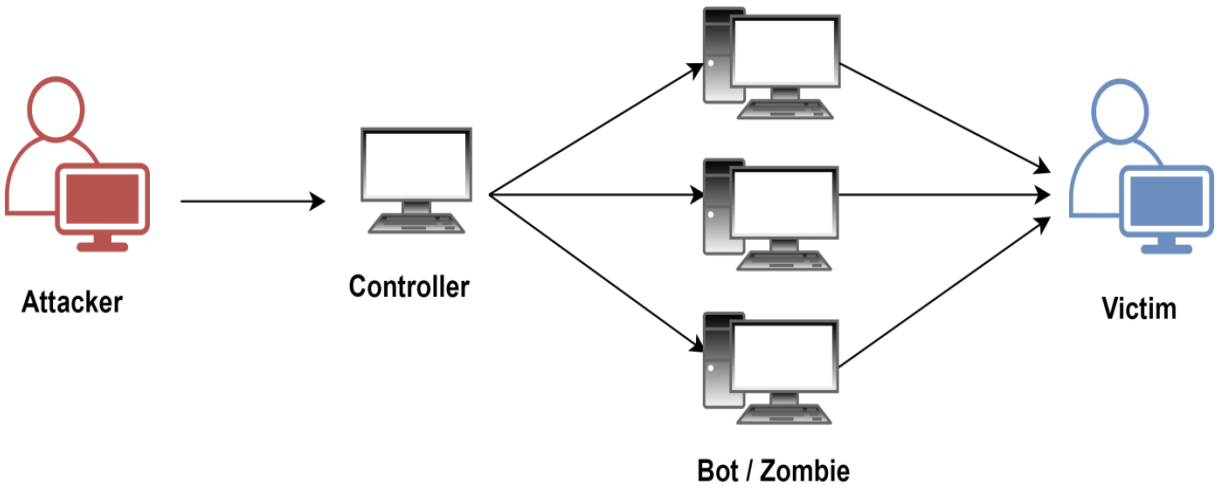

Figure 1.4 : Architecture d'une attaque DDoS.

1.4.7 Classification des attaques DDoS

Les attaques DDoS basée sur la méthodologie de communication, peuvent être classées comme suit :

1.4.7.1 DDoS basé sur l'agent et le gestionnaire

L'attaque DDoS basée sur l'agent et le gestionnaire se compose de clients, de gestionnaires et d'agents [21]. L'attaquant communique essentiellement avec le client. Ces systèmes clients sont ensuite utilisés pour communiquer avec des agents à l'aide de certains logiciels appelés "handlers" (voir la figure 1.5). En fait, les agents sont également des logiciels installés dans des systèmes compromis (bots/zombies) en exploitant les failles du système. Ces agents finissent par exécuter l'attaque, tandis que le véritable attaquant reste en sécurité [18] .

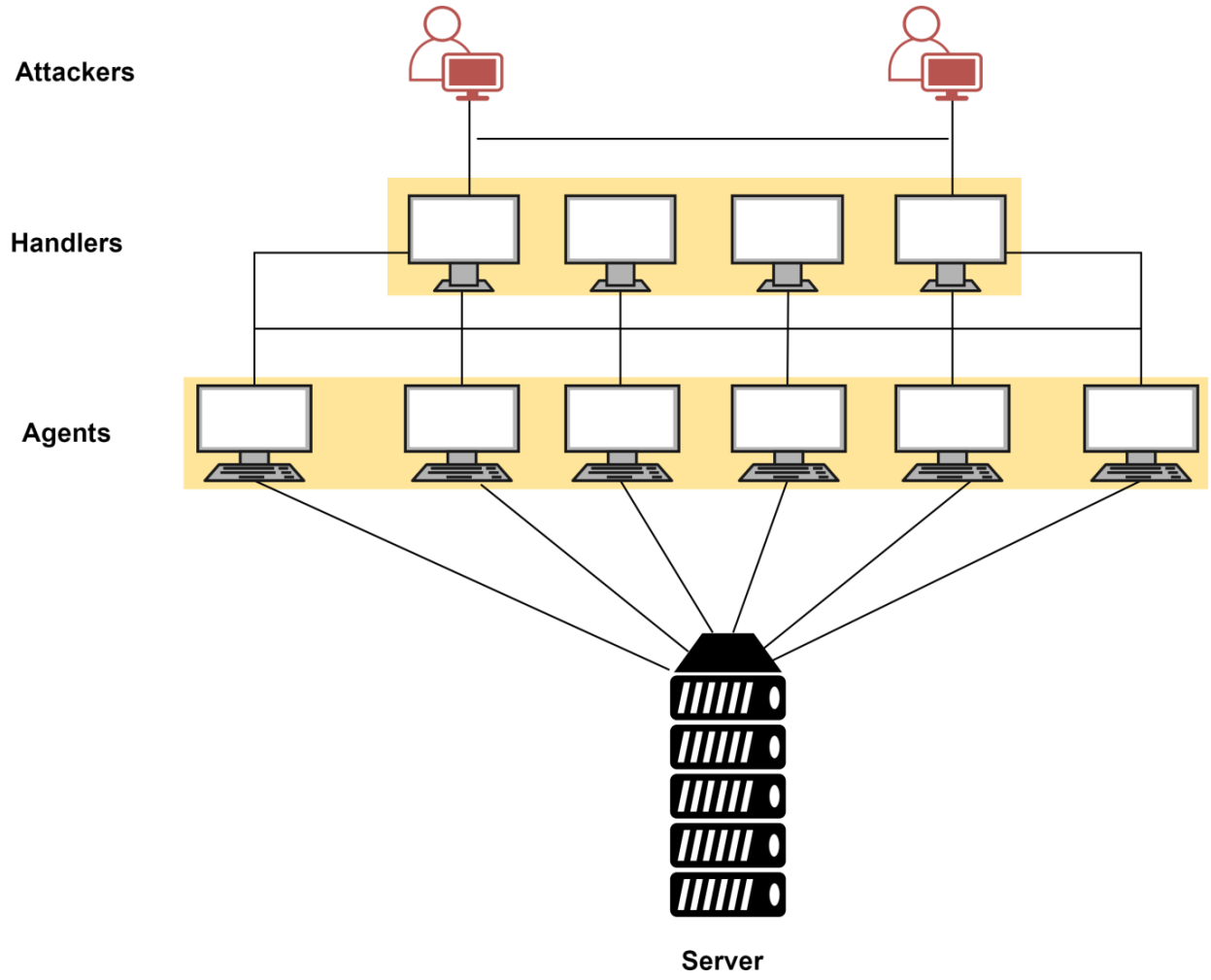

Figure 1.5 : Architecture d'une attaque DDoS basée sur Handlers.

.

1.4.7.2 DDoS basé sur l'IRC

Un système de chat en ligne multi-utilisateurs connu sous le nom de canaux IRC (Internet Relay Chat) a commencé à être utilisé pour la communication entre l'attaquant et les agents, car les réseaux de chat IRC permettent à leurs utilisateurs de créer des canaux publics, privés et secrets. Un réseau d'attaque DDoS basé sur IRC est similaire au modèle d'attaque DDoS agent-handler, sauf qu'au lieu d'utiliser un programme handler installé sur un serveur de réseau, un serveur IRC suit les adresses des agents et des handlers connectés et facilite la communication entre eux (voir la figure 1.6) [15].

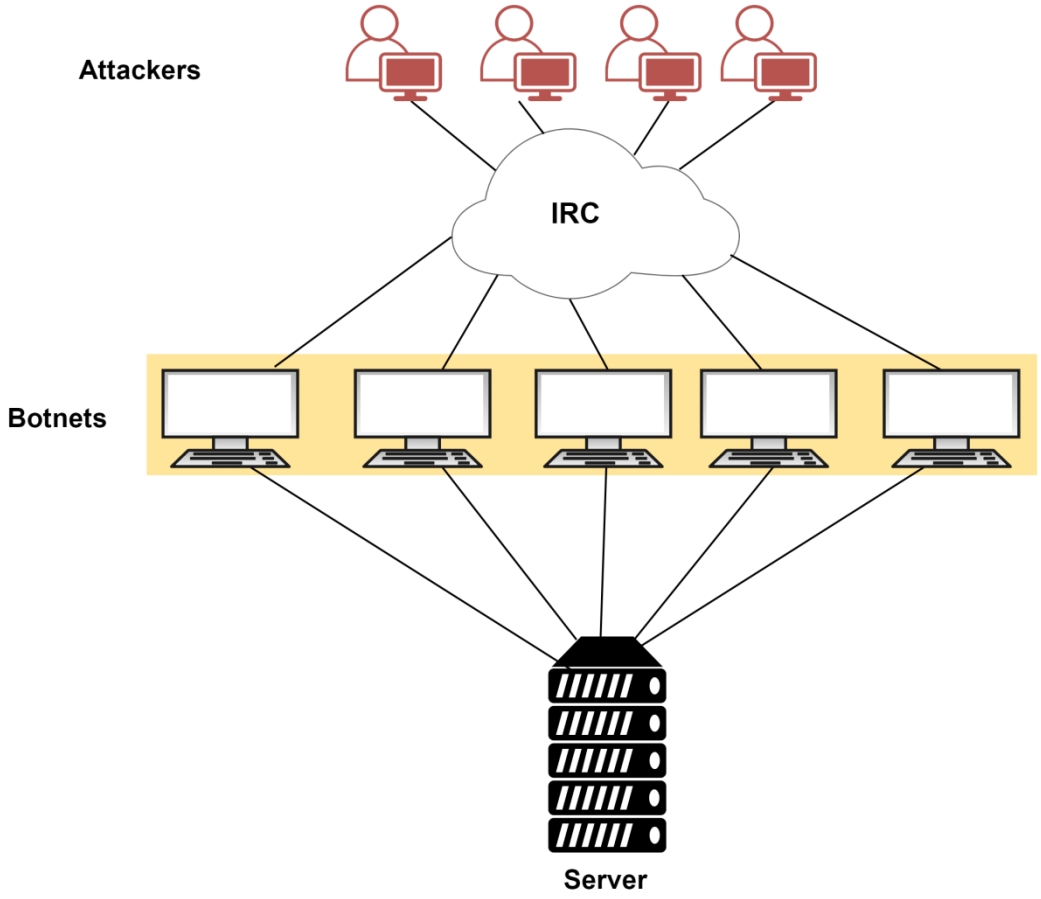

Figure 1.6 : Architecture d'une attaque DDoS basée sur IRC.

1.4.7.3 Les avantages de l'utilisation des canaux IRC

Les avantages de l'utilisation des canaux IRC par rapport aux canaux basés sur le gestionnaire d'agents sont : [18]

- En utilisant un canal IRC comme gestionnaire, les attaquants peuvent utiliser des ports IRC légitimes pour envoyer des commandes aux bots, ce qui rend le suivi des structures de commande et de contrôle DDoS beaucoup plus difficile.
- Comme les serveurs IRC ont souvent beaucoup de trafic, les attaquants peuvent facilement cacher leur présence.

- Un attaquant peut facilement partager des fichiers pour distribuer le code malveillant.
- Les attaquants peuvent simplement se connecter au serveur IRC et voir la liste de tous les bots disponibles au lieu de maintenir leur liste localement sur leur site.

1.4.7.4 Limitation

Comme il s'agit d'une approche centralisée, la principale limitation est le point central de défaillance. Ainsi, si le défenseur capture le serveur IR, l'ensemble du réseau de zombies peut être arrêté [19].

1.4.8 Techniques utilisées pour réaliser une attaque DDoS

On présente ci-dessous, quelques techniques pour réaliser une attaque DDoS.

1.4.8.1 SYN flood

L'attaque SYN Flood du protocole de contrôle de transmission (TCP) est une attaque DDoS qui exploite la conception du processus de communication TCP à trois voies entre un client, un hôte et un serveur. Tout d'abord, un client initie une nouvelle session en envoyant un paquet SYN au serveur. Le serveur répond par un paquet SYN-ACK, puis le client envoie un paquet ACK au serveur avant qu'une connexion ne soit établie [13] comme le montre la figure 1.7.

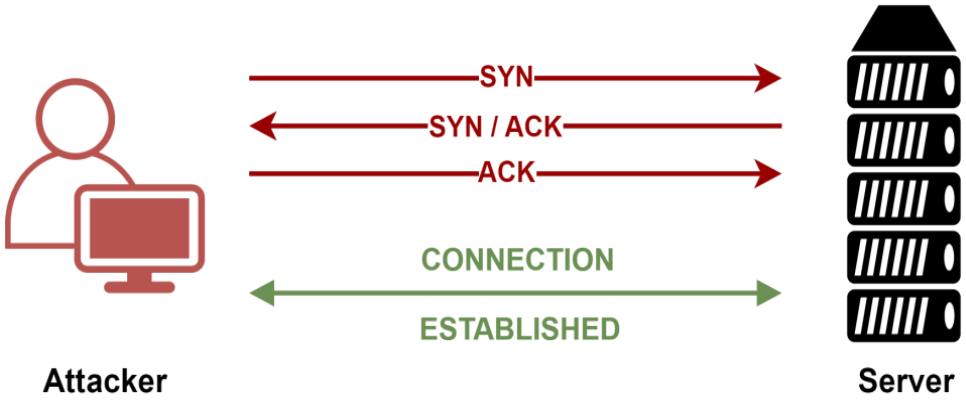

Figure 1.7 : Poignée de main à trois.

Dans une attaque par inondation SYN, l'attaquant envoie plusieurs demandes SYN à chaque port du serveur ciblé, en utilisant souvent des adresses IP usurpées. Le serveur, qui n'est pas au courant de l'attaque, reçoit de multiples demandes, apparemment légitimes, pour établir la communication. Il répond à chacune d'entre elles par un paquet SYN-ACK provenant de chaque port ouvert. Avant que la connexion ne se termine, un autre paquet SYN arrive. Cela laisse un nombre de plus en plus important de connexions semi-ouvertes [13].Ce flot de paquets TCP SYN consomme la bande passante du serveur et rend les ressources du

réseau indisponibles pour l'utilisateur légitime [20]. Enfin, le serveur peut même présenter des dysfonctionnements ou se bloquer [13] (voir la figure 1.8).

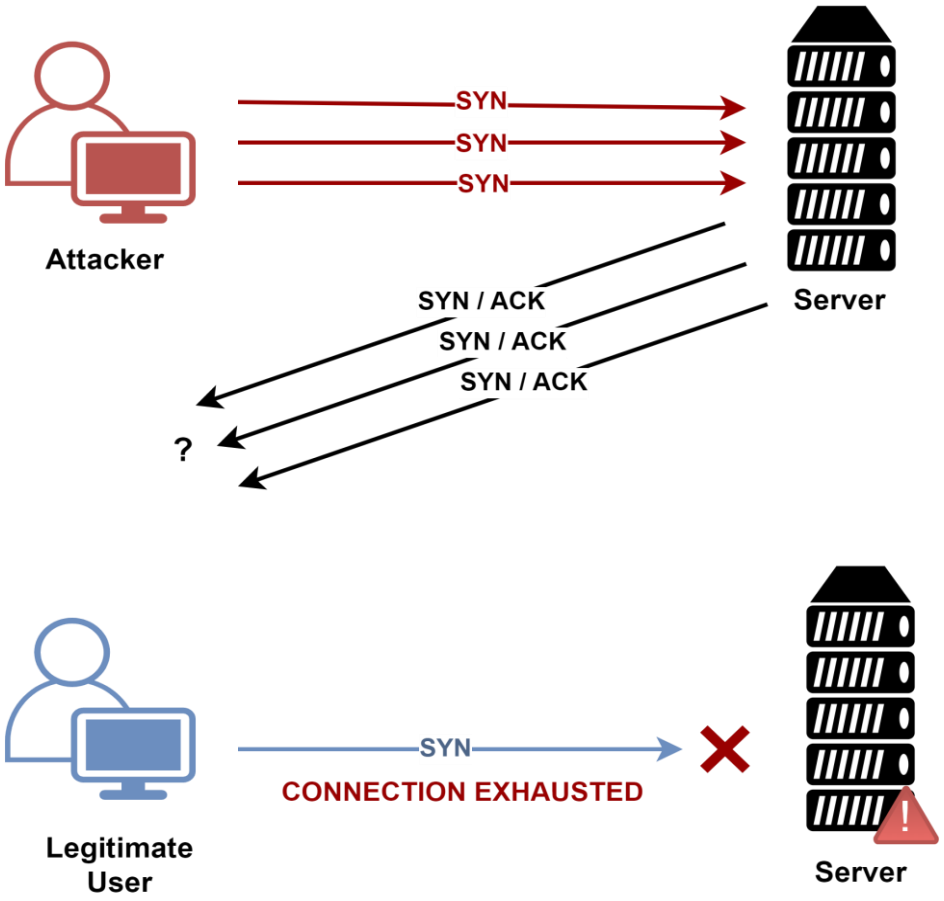

Figure 1.8 : Attaque par inondation TCP-SYN.

1.4.8.2 Réflexion DNS

Le DNS (serveur de noms de domaine) est un élément essentiel des infrastructures de réseau et est responsable de la traduction des noms de domaine en adresses IP. Les attaques DNS DDoS, en particulier celles basées sur la réflexion, constituent une menace importante pour le service de noms de domaine et la sécurité du réseau. La figure 1.9 illustre le fonctionnement des attaques par réflexion DNS. Lors d'une attaque par réflexion DNS, l'attaquant envoie un grand nombre de requêtes DNS aux serveurs DNS avec l'adresse IP source des paquets de requêtes usurpée en tant que celle de la victime. Ces requêtes DNS demandent des réponses à forte charge, telles que "Donnez-moi tous vos enregistrements DNS" ou "Donnez-moi les adresses IP de plusieurs noms de domaine". Ces réponses submergent la victime et épuisent ses ressources [21] .

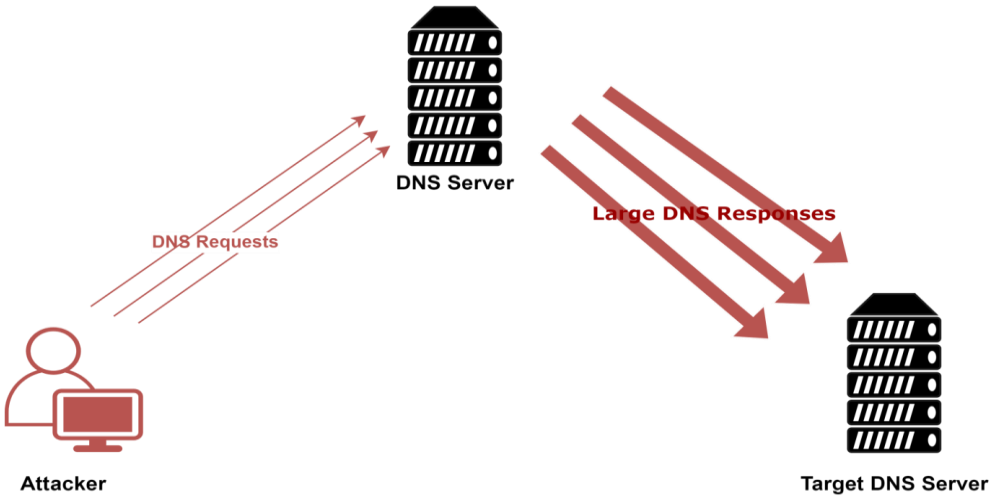

Figure 1.9 : Réflexion DNS.

1.4.8.3 Inondation UDP

Ce type d'attaque DDoS utilise le protocole UDP qui est un protocole de réseau sans connexion et sans session. Il n'est pas lié à une poignée de main à trois comme le TCP et il fonctionne avec un sur débit réduit, de sorte qu'il peut être purgé avec des ressources mineures. Dans ce cas, l'attaquant inonde les ports arbitraires de l'hôte ciblé de paquets IP contenant le datagramme UDP. Les hôtes récepteurs examinent les applications avec ces datagrammes, lorsqu'aucune application corrélée n'est à l'écoute, ils répondent par un paquet ICMP de destination inaccessible. Certaines attaques par inondation UDP peuvent prendre la forme d'attaques par amplification DNS. L'UDP amplifié et non amplifié peut émaner de groupes de botnets de différentes tailles (voir la figure 1.10). Il existe un certain nombre de logiciels accessibles dans le commerce qui permettent d'effectuer une attaque par inondation UDP (par exemple, UDP Unicorn) [22].

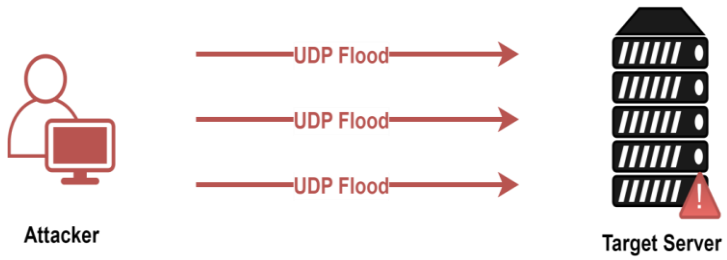

Figure 1.10 : Inondation UDP.

1.4.8.4 Inondation ICMP

Attaque par inondation ICMP (appelé aussi attaque par inondation Ping : Le protocole ICMP (Internet Control Message Protocol) est un autre protocole sans connexion utilisé pour les opérations IP, les diagnostics et les erreurs [13]. L'inondation ICMP échappe aux erreurs de configuration des appareils de réseau. L'attaquant envoie au serveur un très grand nombre de paquets ICMP avec une fausse adresse de retour, de sorte que le serveur n'a pas le temps de répondre à d'autres serveurs (voir la figure 1.11). Il n'échange pas les données entre les systèmes. La largeur de bande du réseau est réduite, ce qui fait que les paquets légitimes sont rejetés par les serveurs. Dans ce cas, l'attaquant lance un programme contrôleur (cheval de Troie) sur l'ordinateur maître, qui entraîne ensuite l'ordinateur esclave à propager l'attaque sur la victime [22].

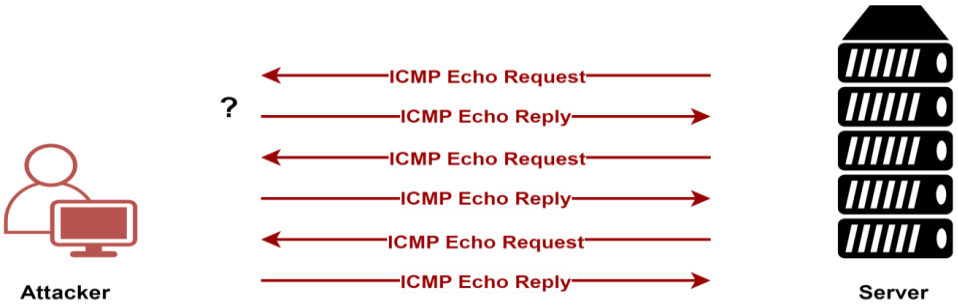

Figure 1.11 : Inondation ICMP

.

1.4.8.5 Attaque HTTP

Cette attaque nécessite une meilleure compréhension de l'activité de la victime. L'utilisation de l'attaque HTTP ne nécessite pas un trafic énorme pour affecter la victime, et elle est difficile à identifier, car le trafic de l'attaque est similaire au trafic normal [23] (voir la figure 12).

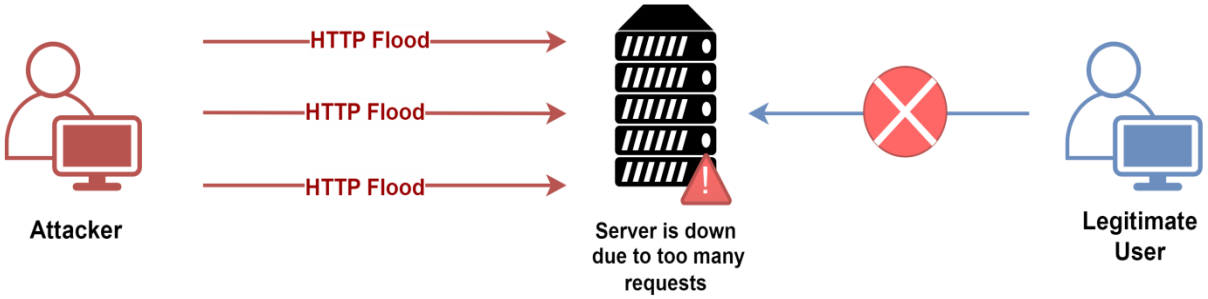

Figure 1.12 : Attaque HTTP.

1.5 Les botnets

1.5.1 Définition botnet

Un réseau de zombies ou botnet est un ensemble de machines infectées par des logiciels malveillants, appelées zombies, qui sont contrôlées par une entité malveillante appelée botmaster (voir la figure 1.13). Ce dernier contrôle les zombies à distance et leur ordonne d'effectuer des activités malveillantes par le biais de commandes. La manière dont les bots sont contrôlés dépend de l'architecture des mécanismes de commande et de contrôle du botnet, qui peuvent être basés sur IRC, HTTP, DNS ou P2P. Ces réseaux de zombies sont utilisés pour commettre des cybercrimes tels que l'envoi de spam, le lancement d'attaques par déni de service ou le vol de données personnelles telles que les comptes de messagerie ou les identifiants bancaires [24].

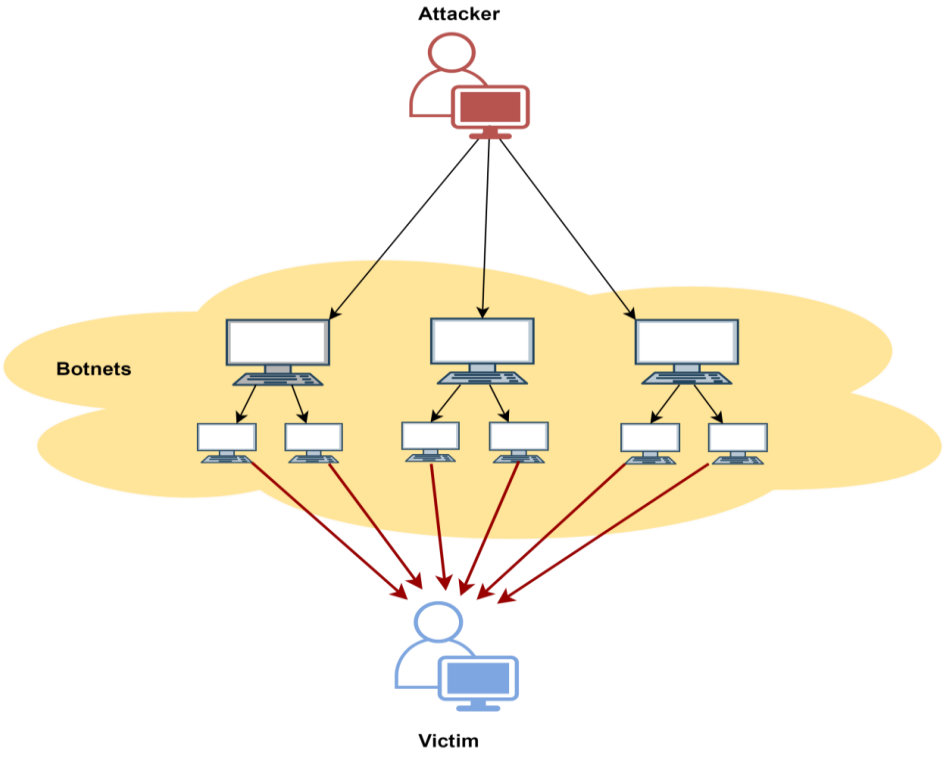

Figure 1.13 : Structure d'une attaque de botnet.

Certains botnets se composent de centaines, voire de milliers d'ordinateurs**,** ce qui en fait l'une des pires cybers menaces du moment.

1.5.2 Principales raisons préférées des attaquants

Les quatre principales raisons pour lesquelles les attaquants préfèrent cette technologie sont les suivantes : [24]

(i) L'inclusion d'un grand nombre de nœuds bots permet de générer rapidement une puissante attaque par inondation.

(ii) La difficulté à trouver l'identité de l'attaquant réel.

(iii) La possibilité de contourner les mécanismes de sécurité à l'aide de protocoles.

(iv) Difficile à détecter en temps réel en raison de la similitude avec le trafic normal.

1.5.3 Composants d'un botnet

En général, un réseau de zombies comprend quatre éléments. Il s'agit du programme du bot, de l'appareil zombie, du bot master et du serveur de commande et de contrôle (C&C). L'attaque d'un réseau de zombies commence par le maître des zombies, qui peut être un attaquant lui-même ou un programme automatisé écrit par l'attaquant. Le bot master scanne les appareils IoT cibles connectés à l'internet. Sur la base des résultats de l'analyse, le bot master exploite les appareils IoT vulnérables et y installe un programme bot. Ce programme établit une connexion avec un bot master ou un serveur C&C afin de recevoir les instructions nécessaires à l'exécution d'activités malveillantes (voir la figure 1.14).

Une brève description de chaque composant est donnée dans les sous-sections suivantes.

1.5.3.1 Programme bot

Un programme bot est un logiciel malveillant installé par un pirate sur un appareil infecté. Le programme bot réside dans l'appareil IoT victime et établit une connexion avec le serveur C&C ou le maître du bot afin de recevoir les instructions nécessaires à l'exécution d'activités malveillantes telles que l'envoi de spam, l'exécution d'attaques par inondation etc. [25].

1.5.3.2 Dispositif zombie

Un dispositif physique de la victime, sur lequel un programme de zombies est installé par l'attaquant, est appelé dispositif zombi [25].

1.5.3.3 Bot master

Un maître-bot ou un chef-bot est le principal contrôleur d'un réseau de zombies. Il peut s'agir d'un pirate informatique ou d'un programme automatisé contrôlé par le pirate pour organiser les attaques du réseau de zombies. Le maître des bots est l'opérateur principal d'un réseau de bots qui envoie des commandes au serveur C&C (dans une architecture client-serveur) ou à des bots spécifiques (dans une architecture peer-to-peer) [25].

1.5.3.4 Serveur C&C

Le serveur C&C est l'ordinateur central qui contrôle les dispositifs zombies en fonction des signaux de commande reçus du bot-master. Le serveur C&C n'est pas un composant obligatoire de chaque réseau de zombies. Dans le cas d'un réseau peer-to-peer, le botmaster envoie directement des signaux de contrôle aux zombies [25].

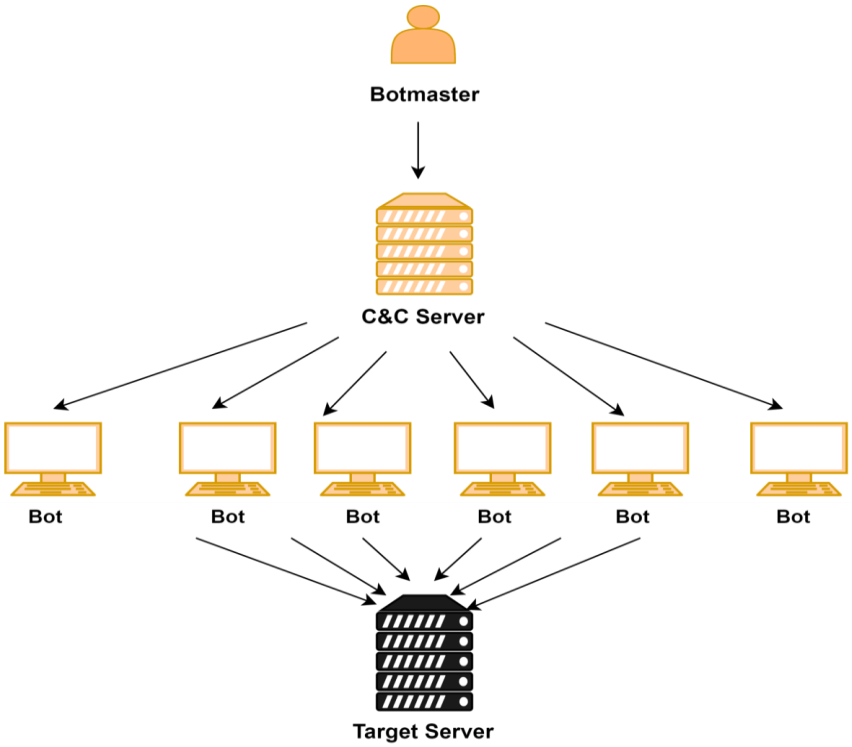

Figure 1.14 : Composants d'un botnet.

1.6 Problèmes de défense DDoS

Les attaques DDoS sont un problème difficile à résoudre car : [15]

- Il n'existe pas de caractéristiques communes des flux DDoS qui puissent être utilisées pour leur détection.
- La nature distribuée des attaques DDoS les rend extrêmement difficiles à combattre ou à retracer.
- Les outils automatisés qui rendent possible le déploiement d'une attaque DDoS peuvent être facilement téléchargés. Les attaquants peuvent également utiliser l'usurpation d'adresse IP afin de dissimuler leur véritable identité, ce qui rend la traçabilité des attaques DDoS encore plus difficile.
- Il n'existe pas de niveau de sécurité suffisant sur toutes les machines de l'internet, alors que les hôtes de l'internet présentent des failles de sécurité persistantes.

1.7 Les mécanismes de défense contre les attaques

Les mécanismes de défense contre les attaques DDoS ont été classés selon deux critères différents. La première classification classe les mécanismes de défense DDoS en fonction de l'activité déployée. La seconde classification divise les mécanismes de défense contre les attaques DDoS en fonction de l'endroit où ils sont déployés [15] .

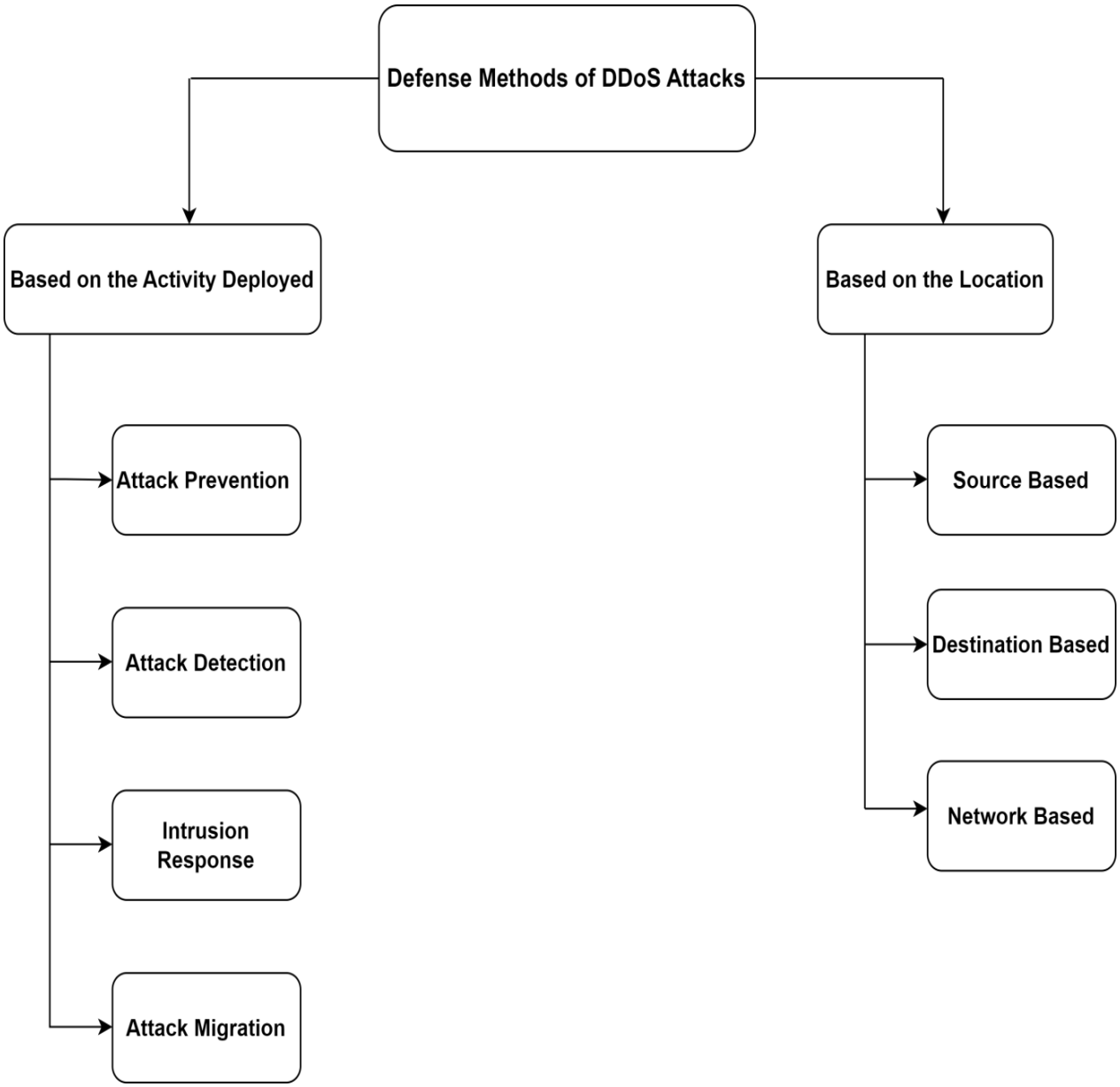

Figure 1.15 : Mécanismes de défense contre les attaques DDoS.

1.7.1 Mécanismes basés sur l'activité déployée

Les mécanismes de défense DDoS en fonction de l'activité déployée peuvent être classés comme suit :

- Prévention des intrusions.

- Détection d'intrusion.

- Tolérance et atténuation des intrusions (migration).

- Réponse aux intrusions.

1.7.1.1 Prévention des attaques DDoS

Le meilleur moment pour arrêter une attaque DDoS est celui de son lancement. En d'autres termes, la prévention des attaques est la meilleure solution de défense contre les attaques DDoS. Les mécanismes de prévention peuvent être déployés au niveau des sources d'attaque, des réseaux intermédiaires, des destinations ou d'une combinaison de ceux-ci. La plupart des mécanismes de prévention visent à corriger les failles de sécurité (protocoles non sécurisés, systèmes d'authentification faibles et systèmes informatiques vulnérables) qui peuvent être exploitées pour lancer des attaques DDoS. Plusieurs mécanismes de prévention ont été proposés dans la littérature [25].

1.7.1.2 Détection des attaques DDoS

La détection des intrusions est un domaine de recherche très actif. Grâce à la détection des intrusions, un ordinateur hôte et un réseau peuvent se prémunir contre les attaques de réseau et les attaques DDoS. Les systèmes de détection d'intrusion détectent les attaques DDoS soit en utilisant la base de données des signatures connues, soit en reconnaissant les anomalies dans les comportements du système [15].

1.7.1.3 Réponse aux intrusions

Un système de réponse DDoS, en revanche, surveille en permanence l'état du système sur la base des alertes générées par un système de détection DDoS, de sorte que les activités malveillantes ou non autorisées puissent être traitées efficacement en appliquant les contre-mesures appropriées pour empêcher les problèmes de s'aggraver et pour ramener le système à un mode sain. Un système de notification génère des alertes lorsqu'une attaque est détectée. Une alerte peut contenir des informations telles que la description de l'attaque, l'heure de l'attaque, l'adresse IP source et les comptes d'utilisateurs utilisés pour l'attaque. Un système de réponse DDoS exécute automatiquement un ensemble approprié d'actions de réponse en fonction du type d'attaque [4].

1.7.1.4 La tolérance aux intrusions et atténuation

La recherche sur la tolérance aux intrusions ou la migration admet qu'il est impossible de prévenir ou d'arrêter complètement les DDoS et se concentre sur la minimisation de l'impact de l'attaque et sur la maximisation de la qualité de ses services. La tolérance aux intrusions peut être divisée en deux catégories : la tolérance aux fautes et la tolérance à la qualité de service (QoS) [15].

1.7.2 Mécanismes basés sur l'emplacement de la défense

La deuxième classification divise les défenses DDoS en fonction de la localisation du déploiement de défense DDoS en fonction du lieu de déploiement, ce qui se traduit par les trois catégories suivantes de mécanismes de défense :

- Réseau source (source network).

- Réseau de la victime (victim network).

- Réseau intermédiaire (intermediate network).

Le tableau 1.1 représente une comparaison générale d'un système de défense contre les attaques DDoS basée sur la localisation.

Tableau 1.1 : Résumé des caractéristiques, avantages et inconvénients d'un système de défense basée sur la localisation *[19]*.

	Caractéristiques	**Inconvénients**	**Avantages**
Basé sur la source	La détection et la réponse sont déployées au niveau des hôtes sources.	Les sources sont réparties entre différents domaines ; il est donc difficile pour chacune des sources de détecter et de filtrer les flux d'attaque avec précision. Il est difficile de différencier le trafic légitime du trafic d'attaque DDoS au niveau des sources, car le volume du trafic n'est pas assez important. Faible motivation pour le déploiement, car on ne sait pas très bien qui paierait les dépenses associées à ces services.	Vise à détecter et à répondre (c'est-à-dire à filtrer) au trafic d'attaque à la source et avant qu'il ne gaspille beaucoup de ressources.
Basé sur la destination	La détection et la réponse sont déployées sur les hôtes de destination (c'est-à-dire les victimes).	Ils ne peuvent pas détecter avec précision l'attaque et y répondre avant qu'elle n'atteigne les victimes et gaspillent des ressources sur les chemins menant aux victimes.	Plus facile et moins coûteux que les autres mécanismes de détection des attaques DDoS en raison de leur accès au trafic agrégé à proximité des hôtes de destination.
En réseau	La détection et la réponse sont déployées au niveau des réseaux intermédiaires (c'est-à-dire les routeurs).	Frais généraux de stockage et de traitement élevés au niveau des routeurs. La détection des attaques est difficile en raison du manque de disponibilité d'un trafic agrégé suffisant destiné aux victimes.	Vise à détecter et à répondre (c'est-à-dire à filtrer) au trafic d'attaque sur les réseaux intermédiaires et aussi près que possible de la source.

1.8 Prévention des attaques DDoS

Pour prévenir les attaques DDoS contre les machines cibles, il existe plusieurs techniques, parmi elles [26] :

1.8.1 Le filtrage

Le filtrage de tous les paquets circulant sur le réseau est une mesure de sécurité essentielle pour détecter les attaques DDoS. En effet, cette technique permet de protéger le réseau contre les attaques provenant des réseaux voisins, tout en évitant que le réseau lui-même ne devienne un attaquant involontaire. Pour réaliser cette mesure, il est nécessaire d'installer des filtres de paquets d'entrée et de sortie sur tous les routeurs du réseau. Ces filtres permettent de détecter les adresses IP usurpées utilisées pour lancer les attaques. Cependant, la mise en place d'une approche visant à empêcher les attaques nécessite une mise en œuvre globale, ce qui peut s'avérer peu pratique.

1.8.2 Le pare-feu

Le pare-feu soit un dispositif de sécurité couramment utilisé pour autoriser ou refuser des protocoles, des ports ou des adresses IP, il ne peut pas gérer toutes les attaques, en particulier celles qui sont complexes, comme les attaques DDoS port 80. En effet, le pare-feu n'est pas en mesure de faire la distinction entre le trafic légitime et le trafic d'attaque DDoS. Il se base uniquement sur les attaques dont les signatures sont déjà connues et enregistrées dans sa base de données. Cela signifie que si une attaque présente une légère variation par rapport au modèle d'attaque original, elle pourrait passer inaperçue et ne pas être détectée. De plus, les nouvelles attaques qui n'ont pas encore de signature enregistrée dans la base de données du pare-feu ne peuvent pas être détectées [26].

1.9 Détection des attaques DDoS

Il existe plusieurs techniques de détection des attaques DDoS, parmi elles :

1.9.1 Détection basée sur les anomalies

Le système basé sur les anomalies utilise une philosophie différente. Il considère comme une anomalie toute connexion réseau qui ne respecte pas le profil normal. Une anomalie de réseau est révélée si le modèle de trafic entrant s'écarte considérablement des profils normaux [27]. Pour détecter les attaques DDoS, il faut d'abord connaître le comportement normal de notre système, puis trouver les écarts par rapport à ce comportement. Les techniques basées sur les anomalies peuvent détecter de nouvelles attaques ; cependant, elles peuvent donner lieu à un plus grand nombre de fausses alertes [26].

1.9.2 Détection basée sur les signatures

L'approche basée sur les signatures utilise une connaissance a priori des signatures d'attaques. Les signatures sont élaborées manuellement par des experts en sécurité qui analysent les attaques précédentes et sont utilisées pour correspondre au trafic entrant afin de détecter les intrusions. Les techniques basées sur les signatures ne sont efficaces que pour détecter le trafic des attaques DDoS connues, alors que les nouvelles attaques ou même les légères variations d'anciennes attaques passent inaperçues [26].

Les avantages et les faiblesses de détection DDoS basé sur la signature et l'anomalie sont donnés par le tableau 1.2.

Tableau 1.2 : Comparaison des IDS basés sur la signature et sur l'anomalie *[28]*.

	Basé sur la signature	**Basé sur les anomalies**
Avantages	- Faible taux d'alarme, faible taux de faux positifs. - Les NID basées sur la signature sont très précises. - Période de détection rapide.	- Surveille les comportements inconnus. - Détecte les attaques inconnues. - Diminue le problème des limitations.
Faiblesses	- Faible protection contre les nouvelles attaques. - Mise à jour régulière avant de sécuriser le réseau. - Aucune alarme n'est déclenchée pour le trafic autorisé.	- Produit des taux élevés de faux positifs (capture beaucoup de NID basés sur le comportement surveillent de système sur la base de leurs modèles de comportement). - Consomme beaucoup de temps pour effectuer une surveillance exhaustive en raison de la quantité de ressources utilisées.

1.9.3 Détection hybride

La détection hybride est une méthode de détection des attaques qui combine à la fois la détection basée sur les anomalies et la détection basée sur la signature pour améliorer l'efficacité de la détection des attaques tout en réduisant les taux de faux positifs.

1.9.4 Détection basée sur le Machine Learning et Le Deep Learning

La détection d'anomalies dans le trafic réseau est essentielle pour protéger les systèmes informatiques contre les attaques DDoS. Les méthodes traditionnelles basées sur des règles

peuvent manquer de précision pour détecter les comportements malveillants complexes. C'est pourquoi les méthodes de Machine Learning et de Deep Learning ont été utilisées pour améliorer la détection d'anomalies. En utilisant des réseaux de neurones profonds et d'autres techniques, les algorithmes de détection peuvent apprendre de manière autonome à partir de données historiques pour identifier les modèles de trafic inhabituels qui pourraient indiquer une attaque DDoS. Bien que cette méthode offre une précision améliorée et une réduction des fausses alertes, elle nécessite souvent une grande quantité de données et une puissance de calcul importante pour l'entraînement des modèles.

1.10 Conclusion

Dans ce chapitre, nous nous sommes concentrés sur la détection des attaques DDoS et j'ai constaté que la détection des attaques DDoS se base généralement sur deux approches différentes, une basée sur les signatures qui s'appuie sur une base de signature des attaques connues et l'autre sur les anomalies qui consistent à détecter toute action anormale de cet utilisateur.

Dans le chapitre suivant, nous allons présenter la notion d'apprentissage profond.

Chapitre 2

Deep Learning

2.1 Introduction

Le Deep Learning (DL) est un sous-ensemble du Machine Learning (ML), qui est lui-même un sous-ensemble de l'Intelligence Artificielle (IA) (voir la figure 2.1).

L'intelligence artificielle (IA) est le domaine de l'informatique consacré à la production de logiciels capables d'effectuer des calculs sophistiqués et intelligents, semblables à ceux que le cerveau humain effectue couramment. Elle comprend des méthodes, des outils et des systèmes destinés à simuler les méthodes humaines d'acquisition de connaissances logiques et inductives, le raisonnement et l'activité cérébrale pour résoudre des problèmes [29].

Dans ce chapitre, nous allons présenter les deux sous-domaines de l'intelligence artificielle.

Tout d'abord, nous allons expliquer ce qu'est le Machine Learning, son fonctionnement et ses différents types. Ensuite, nous aborderons le concept de Deep Learning. Nous commencerons par le définir, faire la différence entre le DL et le ML, avant de passer aux détails des réseaux de neurones, vu que c'est l'outil de mon travail.

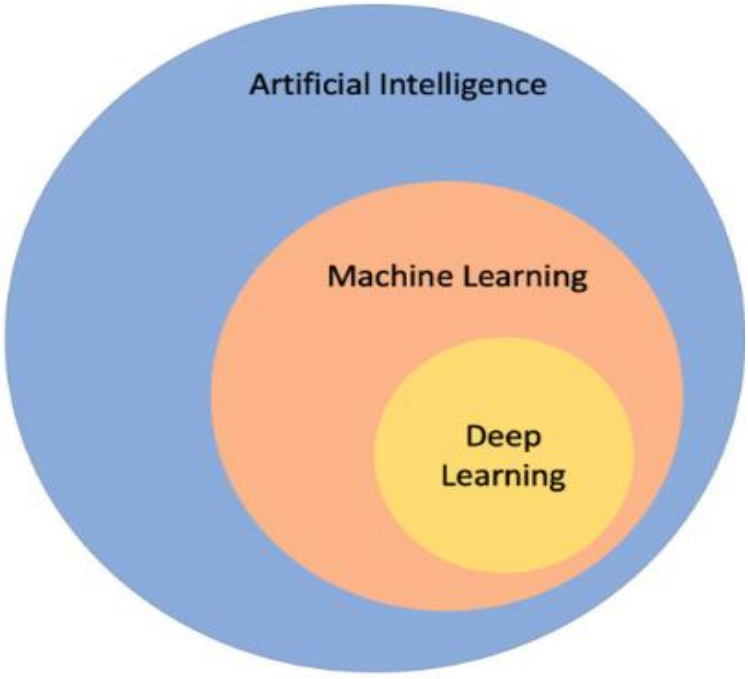

Figure 2.1 : Relation entre IA, ML et DL.

2.2 Apprentissage automatique

2.2.1 Définition

L'apprentissage automatique ou Machine Learning (ML) est l'étude scientifique des algorithmes et des modèles statistiques. Il est utilisé pour apprendre aux machines à traiter les données plus efficacement. L'objectif de l'apprentissage automatique est de laisser les machines apprendre par elles-mêmes à partir des données c'est à dire apprendre quel calcul effectuer [30], contrairement à la programmation où elle se contente d'exécuter à la lettre des règles prédéterminées.

« L'apprentissage automatique est la discipline donnant aux ordinateurs la capacité d'apprendre sans qu'ils soient explicitement programmés. » Arthur Samuel, 1959 [31].

Et voici une autre plus technique :

« Étant donné une tâche T et une mesure de performance P, on dit qu'un programme informatique apprend à partir d'une expérience E si les résultats obtenus sur T, mesurés par P, s'améliorent avec l'expérience E». Tom Mitchell, 1997 (voir la figure 2.2) [31].

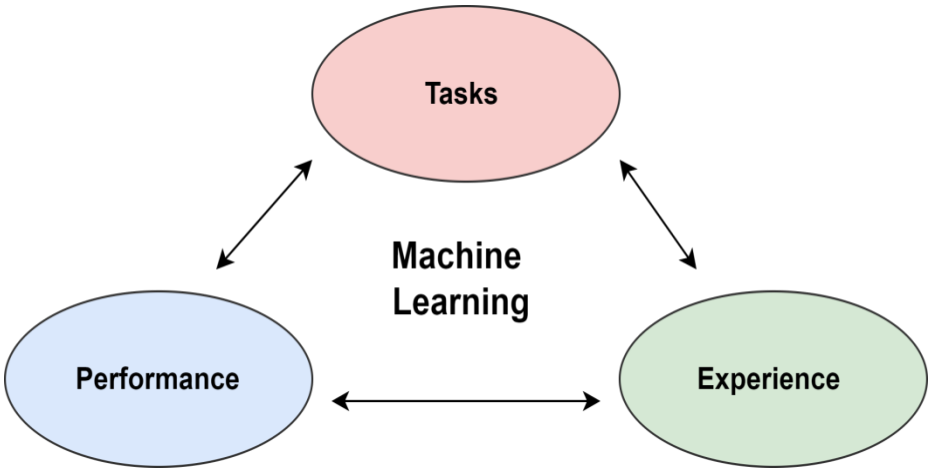

Figure 2.2 : Relation entre T, P et E.

2.2.2 Les étapes clés pour une méthode de Machine Learning

Une méthode de ML comprend principalement les quatre étapes suivantes [32]:

- Ingénierie des caractéristiques.
- Choix de l'algorithme d'apprentissage automatique approprié.
- Former et évaluer les performances du modèle.
- Utiliser le modèle formé pour classer ou prédire les données.

Ce processus est illustré par la figure 2.3.

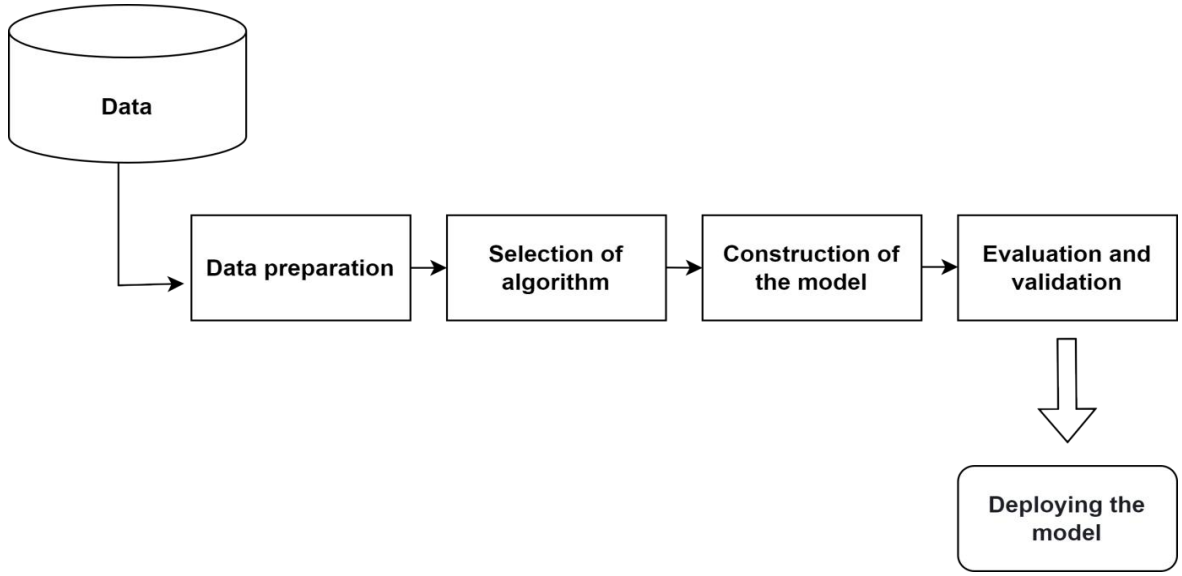

Figure 2.3 : Processus de Machine Learning.

2.3 Types d'apprentissage automatique

Pour donner à une machine la capacité d'apprendre, on utilise des méthodes d'apprentissage. Parmi ces méthodes, il existe trois catégories majeures :

2.3.1 Apprentissage supervisé

L'apprentissage supervisé est la méthode la plus courante en Machine Learning et en Deep Learning. Il constitue d'un ensemble des données (des exemples) d'apprentissage. Les données d'entraînement sont composées d'une entrée (typiquement, une instance du problème) et d'une sortie (typiquement, la réponse réputée bonne). Lors de l'apprentissage, on fournit les paires (observation, étiquette), et l'algorithme génère une fonction qui consiste à apprendre à faire correspondre des données d'entrée à des cibles (voir la figure 2.4) [33]. Avec l'apprentissage supervisé, la machine peut apprendre à faire de nombreuses applications comme : la reconnaissance optique des caractères, la reconnaissance vocale, la classification d'images, la traduction de langues [34].

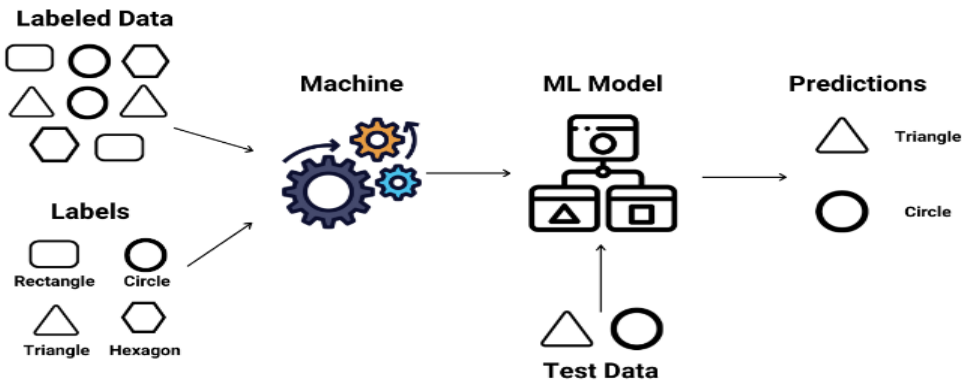

Figure 2.4 : Apprentissage supervisé.

Il existe deux principaux types d'étiquettes de sortie :

2.3.1.1 Classification

Dans ce type, on utilise des variables discrètes c'est à dire des variables qui peut prendre seulement certaines valeurs. L'algorithme doit prédire une des sorties possibles pour ce problème. L'algorithme peut prédire directement la catégorie ou produire une distribution de probabilités à chacune des catégories. La classe avec probabilité la plus haute est la classe sélectionnée [33]. Par exemple, on peut prédire si un massage spam ou pas en fonction du nombre de lien, du nombre de fautes d'orthographe, etc.).

On peut distinguer deux types de classification : la classification binaire, qui consiste à séparer les données en deux classes distinctes, et la classification multi-classe, qui permet de classifier les données en plusieurs classes (voir la figure 2.5).

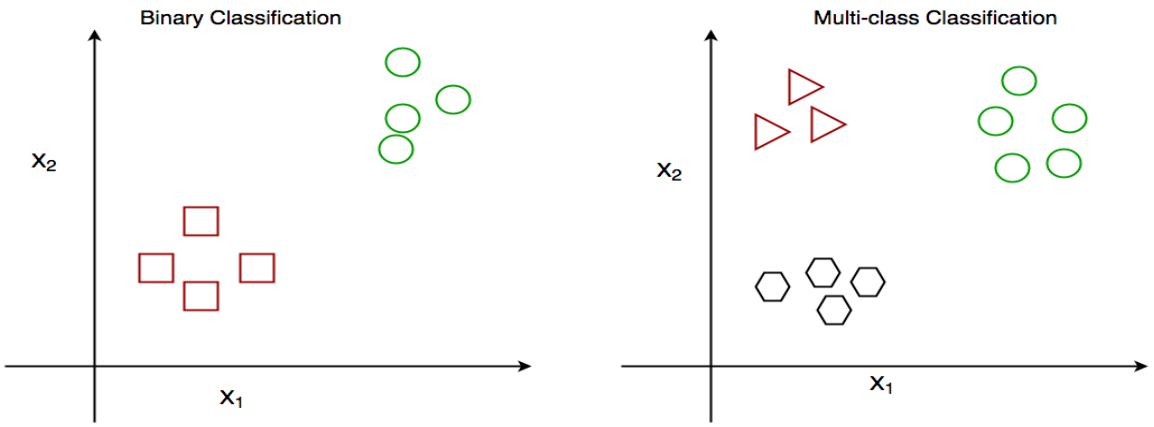

Figure 2.5 : Classification binaire et classification multi-classe.

2.3.1.2 Régression

Dans ce type, on utilise des variables continues c'est-à-dire des variables qui peuvent prendre une infinité de valeurs (voir la figure 2.6). L'algorithme fournit une valeur numérique en sortie au lieu de prédire une étiquette. Il est aussi possible de prédire plusieurs valeurs numériques en sortie [33]. Le but de la régression est d'estimer une valeur de sortie à partir des valeurs d'un ensemble des observations. Par exemple, on peut prédire le prix d'un appartement en fonction de sa surface, sa localisation, sa qualité, ...)

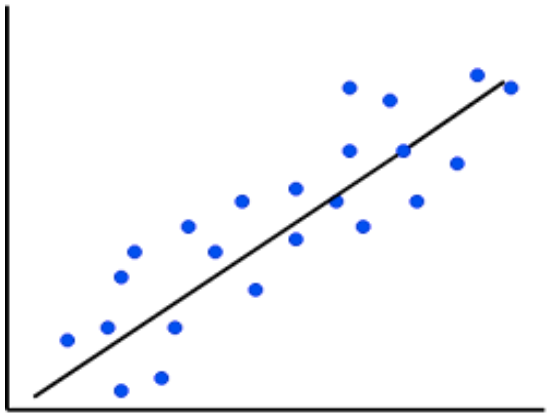

Figure 2.6 : Modèle de régression.

2.3.2 Apprentissage non supervisé

Ce type d'apprentissage doit être utilisé lorsque la catégorie de données acquises n'est pas connue à l'avance c'est-à-dire il n'y'a pas de résultat prédéterminé (on fournit à la machine un ensemble de données qui ne sont pas été étiqueté) [35]. L'objectif est de regrouper avec succès les observations disponibles dans leurs propres classes par lui- même pour des catégories d'objets basées sur des mesures de similarité entre les objets (voir la figure 2.7).

L'apprentissage non supervisé peut être classé en deux catégories d'algorithmes :

- Regroupement (Clustering).
- Association.

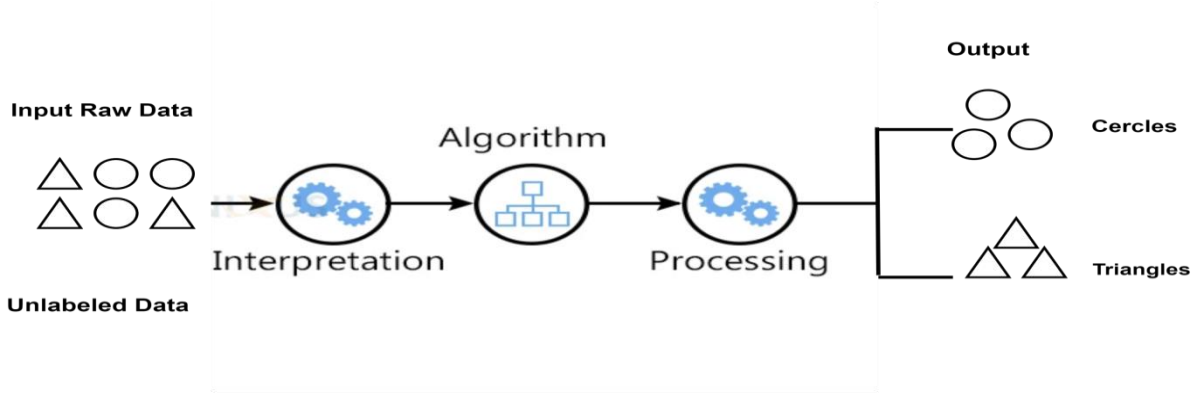

Figure 2.7 : Apprentissage non supervisé.

2.3.3 Apprentissage par renforcement

L'apprentissage par renforcement est une méthode qui utilise un processus d'essais et d'erreurs pour trouver la meilleure action à effectuer pour chaque situation qu'un robot percevra afin de maximiser la récompense [36]. Le système ajuste ses paramètres en fonction des commentaires qu'il reçoit de l'environnement (voir la figure 2.8).

On cite comme exemple un système qui simule un joueur d'échecs en utilisant les résultats des étapes précédentes pour améliorer ses performances est un système qui apprend par renforcement.

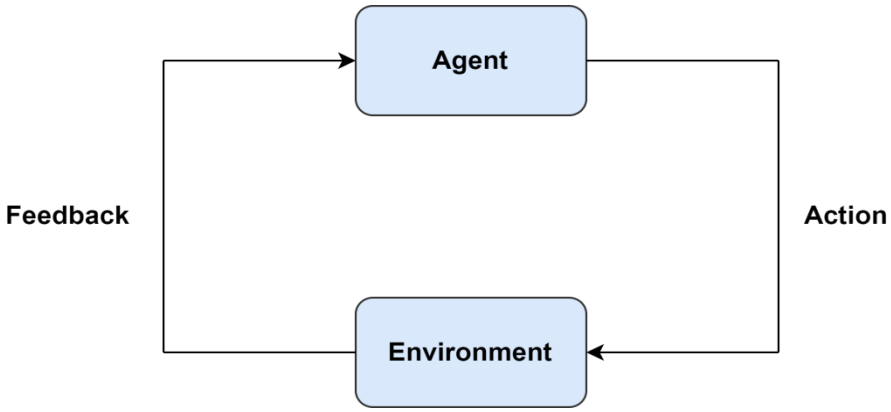

Figure 2.8 : Apprentissage par renforcement.

2.3.4 Algorithmes d'apprentissage supervisé

Les algorithmes d'apprentissage supervisé les plus importants sont :

- Régression linéaire (Linear Regression).

- Régression logistique (Logistic Regression).

- Naive Beyes gaussien (Gaussian Naive Bayes).

- Machine à vecteur de support (Support Vector Machine SVM).

- Arbres de décision (Decision Trees).

- K le plus proche (K Nearest Neighbor KNN).

- Forêt aléatoire (Random Forest).

2.4 Apprentissage en profondeur

L'apprentissage en profondeur ou Deep Learning en anglais (DL) est un sous ensemble d'apprentissage automatique qui se fait via des réseaux de neurones artificiels. Son but est de construire un réseau neuronal qui imite le cerveau humain pour l'apprentissage analytique [37]. Il s'agit d'un modèle de réseau qui contient plusieurs couches neuronales qui sont : une couche d'entrée, une couche de sortie et des couches cachées qui se trouvent entre la première et la dernière couche. Chaque couche successive reçoit les données de la couche précédente. L'apprentissage profond est adapté au traitement de données volumineux et complexes. Les informations sont transmises à chaque couche [38].

Le DL nécessite un plus grand ensemble de données par rapport au ML [39]. Comme les méthodes ML, les méthodes DL ont également les caractéristiques de l'apprentissage supervisé et de l'apprentissage non supervisé [40].

Plus le réseau est profond, plus il est capable d'apprendre des choses compliquées, mais cela rend aussi l'apprentissage plus long.

2.4.1 Histoire d'apprentissage profond

En 1943, McCulloch & Pitts [41] ont démontré une machine de Turing constituée de connexions neuronales.

En 1958, Rosenblatt [42] introduit le premier perceptron.

En 1969, les insuffisances du perceptron ont été proposées par Minsky et Papert [43] pour finir d'étudier des neurones qui fonctionnent depuis au moins une décennie.

En 1985, Geoffrey Hinton et al. [44] ont apporté l'algorithme de rétropropagation.

En 1998, YanLeCun [45] a analysé la rétropropagation pour l'analyse de documents à l'aide de CNN.

2.4.2 Différence entre le Machine Learning et le Deep Learning

L'effet des modèles de DL est nettement meilleur que les ML. La différence entre ces modèles se marque principalement dans les aspects suivants **:** [46]

Temps d'exécution : Le temps d'exécution comprend le temps de formation et le temps de test. En raison de la grande complexité des modèles profonds, le temps de formation et de test est beaucoup plus long que celui des modèles de Machine Learning.

Nombre de paramètres : Les modèles profonds ont beaucoup plus de paramètres que les modèles d'apprentissage automatique ; par conséquent, la formation et l'optimisation des modèles profonds nécessitent plus de temps.

Représentation des caractéristiques : L'entrée des modèles de ML est constituée de vecteurs de fonctionnalités, et le développement de fonctionnalités est une étape essentielle. En revanche, les modèles d'apprentissage en profondeur peuvent apprendre des représentations de caractéristiques à partir de données brutes et ne reposent pas sur l'ingénierie des caractéristiques

Capacité d'apprentissage : La structure des modèles de Deep Learning est complexe, contenant un grand nombre de paramètres (généralement des millions ou plus). Par conséquent, les modèles de DL sont plus adaptatifs que les modèles de ML. Cependant, les modèles d'apprentissage en profondeur courent également le risque de surajustement (overfitting) et nécessitent des quantités de données beaucoup plus importantes pour s'entraîner. Cependant, les modèles d'apprentissage en profondeur sont plus efficaces.

2.5 Réseaux de neurones artificiels

2.5.1 Définition d'un réseau de neurone artificiel

Les réseaux de neurones artificiels ANN (Artificial Neural Network en anglais) sont des modèles bien plus complexes que tous les autres modèles de Machine Learning, il faut souvent fournir un temps d'apprentissage plus long [46].

L'idée de création d'ANN s'inspire du fonctionnement du cerveau humain. Les ANN sont des modèles numérisés du cerveau humain, des programmes informatiques conçus pour simuler la manière dont le cerveau humain traite les informations. Ils apprennent par l'expérience avec des exemples d'apprentissage appropriés, tout comme les personnes, et non par la programmation. Les réseaux neuronaux acquièrent leurs connaissances en détectant les schémas et les relations entre les données. Le cerveau est un excellent outil de reconnaissance des formes, lorsque nous regardons un stylo, nous savons qu'il s'agit d'un stylo parce que les neurones biologiques d'une certaine zone de notre cerveau ont rencontré un modèle d'entrée similaire à plusieurs reprises et ont appris à associer ce modèle spécifique à la description de l'objet "stylo". Comme notre cerveau contient des milliards de neurones totalement interconnectés, nous pouvons apprendre et reconnaître une variété presque infinie de modèles d'entrée [29].

Un ANN se compose d'une couche d'entrée, une couche de sortie et des couches cachées. Pour créer un réseau de neurone, il suffit de connecter les nœuds (appelé aussi les neurones ou bien les unités) des couches adjacentes connectées [46]. Les neurones artificiels sont reliés par des coefficients (poids) qui composent la structure neuronale [29], (voir la Figure 2.9).

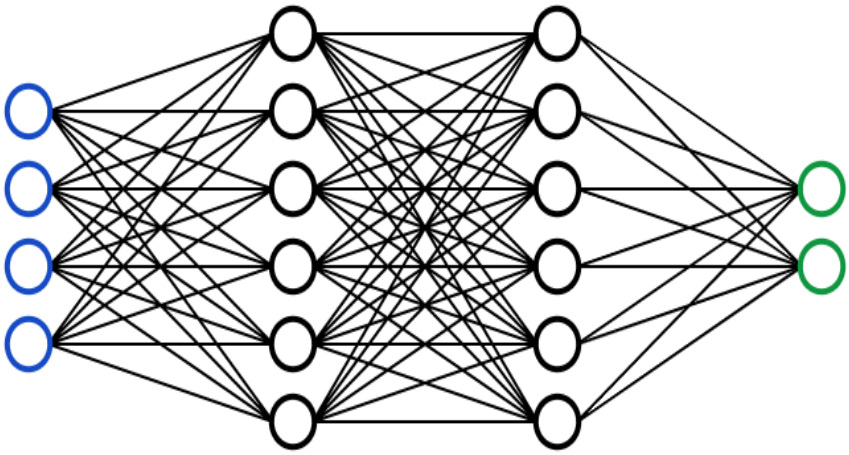

Figure 2.9 : Réseau de neurones (ANN).

Le moyen le plus simple et le plus sur d'apprendre l'intelligence artificielle est de comprendre les réseaux de neurones artificiels, et le moyen le plus simple de comprendre les réseaux de neurones artificiels est de comprendre un perceptron simple.

2.5.2 Perceptron simple

C'est le réseau de neurones le plus simple sur le plan architecture et au niveau de mise en œuvre. Il s'agit d'un modèle de classification linéaire, capable de séparer linéairement deux classes de points. La structure d'un perceptron est représentée à la figure 2.10.

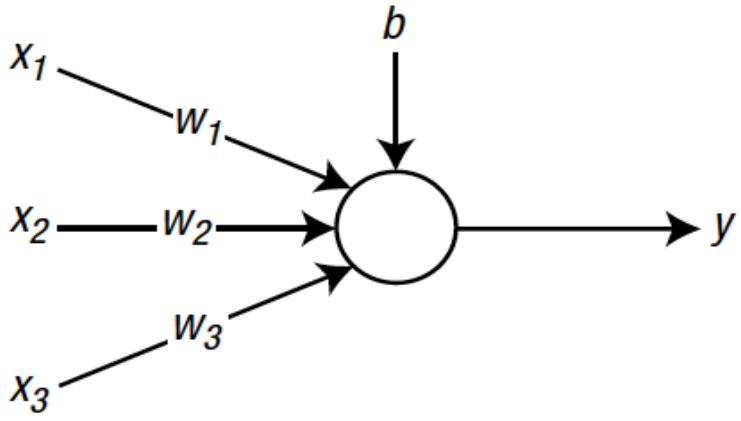

Figure 2.10 : Perceptron simple.

Le cercle indique le nœud et la flèche indique le signal.

X_1, X_2 et X_3 représentent l'entrée de neurone.

W_1, W_2 et W_3 sont les poids pour les signaux correspondants.

y : la sortie de nœud.

b : le biais qui transmet toujours la valeur 1 [47] .

Chaque neurone possède plusieurs entrées (X_1, X_2 et X_3) auxquelles sont associés des poids (W_1, W_2 et W_3).

Ensuite, il calcule la somme des produits des entrées et de leurs poids correspondants, en ajoutant un coefficient appelé biais.

$$y = W_1X_1 + W_2X_2 + W_3X_3 + b. \quad (2.1)$$

Puis, il passe à travers une fonction de transfert (fonction d'activation) pour produire le signal de sortie (voir la figure 2.11).

Grâce à la fonction d'activation le neurone décide s'il doit être "actif" ou "inactif".

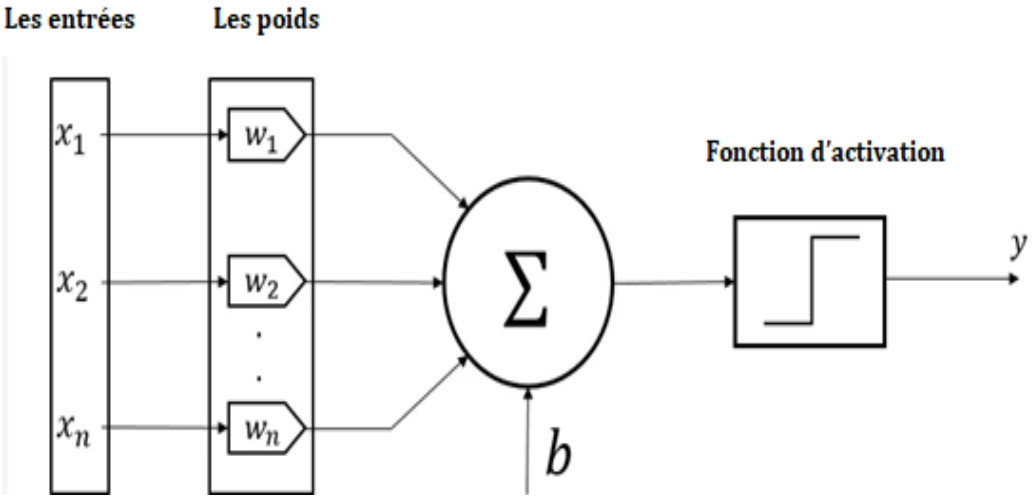

Figure 2.11 : Perceptron dans le Deep Learning.

Un perceptron simple n'est pas très utile, car sa valeur de sortie ne peut avoir qu'une des deux valeurs (0 ou 1). C'est un inconvénient majeur qui a fait stagner le domaine des réseaux de neurones. Mais ce problème a été résolu par les réseaux multicouches.

2.5.3 Fonction de coût

Lors de l'apprentissage des paramètres optimaux (poids et biais) d'un modèle, nous devons définir une fonction pour mesurer l'erreur. Cette fonction est appelée fonction de perte

et nous fournit une mesure de la différence entre les sorties prédites par le réseau et les sorties réelles de l'ensemble de données. La fonction de perte peut être définie de différentes manières, en fonction du problème et de l'objectif. Par exemple, dans le cas d'un problème de classification, une façon courante de définir la perte consiste à calculer la proportion d'entrées mal classées dans l'ensemble de données et à l'utiliser comme probabilité d'erreur du modèle. En revanche, dans le cas d'un problème de régression, la fonction de perte est généralement définie en calculant la distance entre les sorties prédites et les sorties réelles correspondantes, puis en calculant la moyenne sur tous les exemples de l'ensemble de données [48].

2.5.3.1 Binary-crossentropy

La fonction de coût binary-crossentropy est une fonction de perte pour les problèmes de classification à deux classes/binaires. En général, elle est utilisée pour calculer la perte pour les modèles où le résultat est un nombre de probabilité entre 0 et 1 [49].

2.5.3.2 Categorical-crossentropy

La fonction de coût categorical cross-entropy est utilisée pour la classification multi-classe. Elle mesure la différence entre la distribution de probabilité prédite et la distribution de probabilité réelle. Cette fonction est utilisée avec une fonction d'activation de softmax en sortie de réseau de neurones.

2.5.3.3 Sparce-crossentropy

La fonction de coût sparse categorical cross-entropy est une variante de la fonction de coût catégorical crossentropy qui est utilisée dans la classification multi-classe, mais elle est utilisée lorsque la distribution de probabilité réelle n'est pas encodée en one-hot. Elle mesure la différence entre la distribution de probabilité prédite et la distribution de probabilité réelle.

2.5.3.4 Mean_squared_error

La fonction de coût mean_squared_error est une fonction de perte pour les problèmes de régression qui calcule (résultat réel - résultat prédit)^2 pour chaque exemple de l'ensemble de données et renvoie ensuite leur moyenne [49].

2.5.3.5 Mean_absolute_error

La fonction mean_absolute_error est une fonction de perte pour les problèmes de régression qui calcule abs (real output - predicted output) pour chaque exemple dans le jeu de données, puis renvoie leur moyenne [49].

2.5.4 Régularisation

La régularisation est une technique utilisée dans l'apprentissage automatique pour améliorer un modèle qui souffre d'un sur ajustement (overfitting), ce qui signifie que cet hyper paramètre est principalement utilisé lorsqu'il est strictement nécessaire, et son objectif principal est d'augmenter la capacité de généralisation du modèle [50].

Le type de régularisation le plus courant pour les modèles d'apprentissage profond est celui qui maintient les poids du réseau à un faible niveau. Ce type de régularisation est appelé régularisation de poids et a deux variantes différentes: la régularisation L2 et la régularisation L1.

Les techniques de régularisation couramment utilisées sont les suivantes :

2.5.4.1 Régularisation L1

La régularisation ajoute une pénalité L1 au cout de la fonction d'erreur en ajoutant la somme des valeurs absolues des coefficients. Cela a pour effet de faire tendre certains coefficients à zéro.

2.5.4.2 Régularisation L2

La régularisation L2 ajoute une pénalité L2 au cout de la fonction d'erreur en ajoutant la somme des carrées des coefficients. Cela a pour effet de réduire les valeurs de tous les coefficients, mais pas nécessairement de les mettre à zéro.

2.5.4.3 Régularisation L1 et L2

La Régularisation L1 et L2 est une technique de régularisation qui combine à la fois la régularisation L1 et L2.

2.5.4.4 Dropout

La régularisation Dropout consiste à supprimer aléatoirement des nœuds d'un réseau neuronal au cours de la formation. Plus précisément, l'élimination établit une probabilité sur chaque nœud. Cette probabilité fait référence à la probabilité que le nœud soit inclus dans la formation à chaque itération de l'algorithme d'apprentissage [49] (voir la figure 2.12).

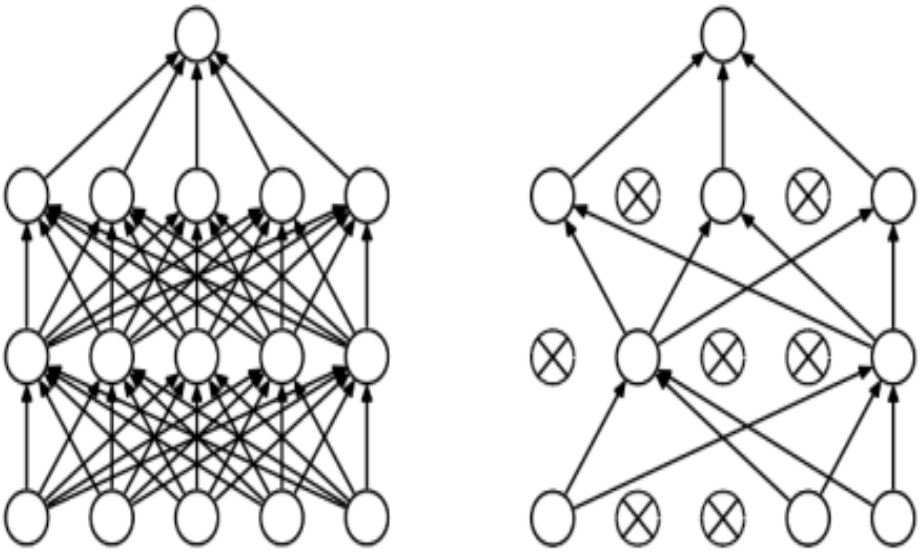

Figure 2.12 : Application de la technique de Dropout.

2.5.5 Algorithmes d'optimisation

Dans l'apprentissage profond, un optimiseur est un algorithme qui met à jour les poids d'un réseau neuronal pendant l'entrainement. L'objectif d'un optimiseur est de trouver les poids qui minimisent la fonction de coût, qui mesure la différence entre les sorties prédites du réseau et les sorties réelles.

Il existe de nombreux optimiseurs différents, Parmi les optimiseurs les plus répandus, on peut citer :

2.5.5.1 SGD

La descente de gradient stochastique (SGD), qui calcule la perte et les dérivées à chaque fois en utilisant un sous-ensemble ou un lot d'exemples de données seulement ; par conséquent, son processus d'apprentissage est plus rapide que la descente de gradient standard. La SGD utilise un seul hyperparamètre (appelé taux d'apprentissage) pour mettre à jour les paramètres [50].

2.5.5.2 Adam

L'optimisation d'Adam est une méthode stochastique de descente de gradient basée sur l'estimation adaptative des moments de premier ordre et de second ordre [48]. Ces moments sont utilisés pour mettre à jour les paramètres du modèle de manière plus efficace que SGD standard.

2.5.5.3 Adamax

Adamax est une variante d'Adam basée sur la norme infinie, est une méthode d'optimisation du premier ordre basée sur le gradient. En raison de sa capacité à ajuster le taux d'apprentissage en fonction des caractéristiques des données, elle est adaptée à l'apprentissage de processus variables dans le temps, par exemple les données vocales avec des conditions de bruit qui changent de manière dynamique [48].

2.5.6 Perceptron multicouche

Le perceptron multicouche est également connu sous le nom de MLP, qui est un modèle représentant une correspondance non linéaire entre un vecteur d'entrée et un vecteur de sortie. Les nœuds sont reliés par des poids et des signaux de sortie qui sont fonction de la somme des entrées du nœud modifiée par une simple fonction de transfert non linéaire, ou fonction d'activation. L'architecture d'un perceptron multicouche est variable [51].Un réseau MLP peut également apprendre des fonctions non linéaires. Il s'agit d'un réseau neuronal qui comporte plusieurs couches denses entièrement connectées. Pour créer un réseau neuronal, nous combinons des neurones de sorte que les sorties de certains neurones soient les entrées d'autres neurones.

Les perceptrons multicouches ont la capacité d'apprendre par le biais d'apprentissage. Ce dernier nécessite un ensemble de données d'entraînement, qui consiste en une série de vecteurs d'entrée et de sortie associés. Au cours de l'apprentissage, le perceptron multicouche se voit présenter à plusieurs reprises les données d'entraînement et les poids du réseau sont ajustés jusqu'à ce que la correspondance entrée-sortie souhaitée se produise. Les perceptrons multicouches apprennent de manière supervisée. Pendant l'apprentissage, la sortie du perceptron multicouche, pour un vecteur d'entrée donné, peut ne pas être égale à la sortie souhaitée. Un signal d'erreur est défini comme la différence entre la sortie souhaitée et la sortie réelle. L'apprentissage utilise l'ampleur de ce signal d'erreur pour déterminer dans quelle mesure les poids du réseau doivent être ajustés afin que l'erreur globale soit réduite, afin de réduire l'erreur globale du perceptron multicouche. De nombreux algorithmes peuvent être utilisés pour entraîner un perceptron multicouche. Une fois entraîné à l'aide de données d'apprentissage suffisamment représentatives, le perceptron multicouche peut se généraliser à de nouvelles données d'entrée inconnues [51].

2.5.7 Structure d'un perceptron multicouche

La figure 2.13 représente la structure d'un perceptron multicouche. Celle-ci se compose de trois couches:

Une couche d'entrée : Le nombre de nœuds dans cette couche égale au nombre de données en entrée. Les nœuds transmettent les données à la couche cachée. Aucun calcul n'est effectué à cette couche.

Une couche de sortie : Cette couche donne le résultat obtenu par le réseau.

Des couches cachées : Elles sont situées entre la couche d'entrée et la couche de sortie. Elles traitent l'information et la transmettent à la couche de sortie. Il existe différentes façons de configurer ces couches cachées. Parmi celles-ci, on cite :

Les réseaux neuronaux simples.

Les réseaux neuronaux convolutifs.

Les réseaux neuronaux récurrents.

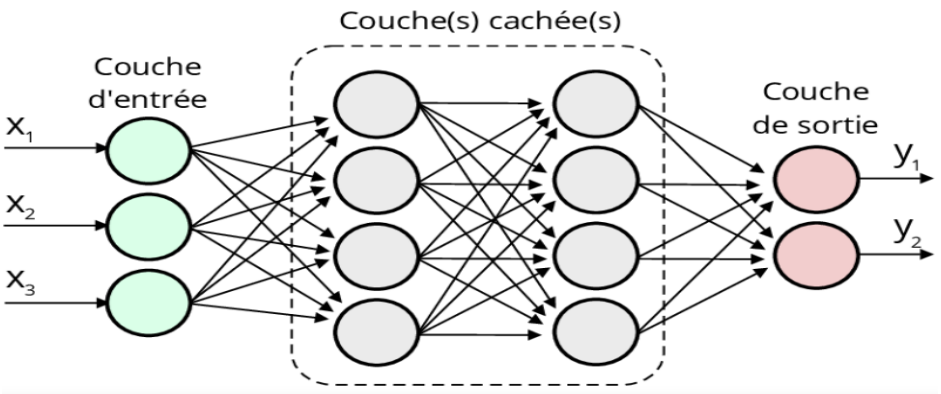

Figure 2.13 : Structure d'un perceptron multicouche.

Lors de la conception d'un modèle ANN, de nombreux facteurs doivent être pris en compte. Tout d'abord, une structure de modèle ANN appropriée doit être sélectionnée. Ensuite, la fonction d'activation doit être déterminée. Le nombre de couches et le nombre de cellules dans chaque couche doivent être choisis. En règle générale, le modèle souhaité se compose de plusieurs couches. Les modèles les plus généraux supposent une interconnexion complète entre toutes les cellules. Ces connexions peuvent être bidirectionnelles ou unidirectionnelles. Un ANN peut créer sa propre organisation ou représenter les informations qu'il reçoit au cours de l'apprentissage [52].

L'architecture de ces interconnexions est très importante dans un réseau de neurones. Le nombre de couches cachées d'un réseau MLP définit l'architecture de réseau.

2.6 Architecture des réseaux neuronaux

Il existe de nombreuses architectures des ANN, les plus connues sont :

2.6.1 Réseau de neurones Feed-Forward à une couche

Tous les nœuds d'entrée sont connectés à chacun des nœuds de sortie. Le terme "feed-forward" indique qu'il n'y a pas de retour d'information de la couche de sortie vers la couche d'entrée (voir la figure 2.14). Il s'agit donc d'un réseau non bouclé (feed-forward) à une seule couche [53].

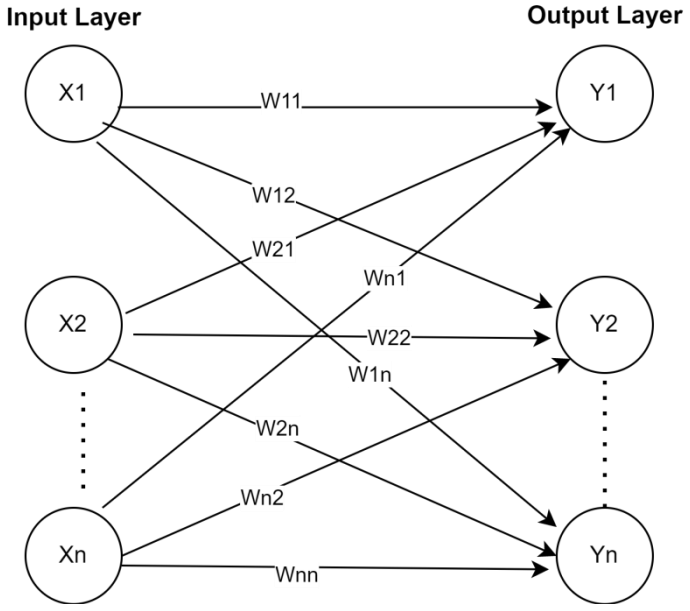

Figure 2.14 : Réseau de neurones feed-forward à une couche.

2.6.2 Réseau multicouche de type feed-forward

Dans ce réseau, les neurones de chaque couche sont connectés aux neurones de la couche suivante, et les neurones de la même couche ne sont pas connectés (voir la figure 2.15). Chaque neurone d'une couche est relié à une autre couche par un poids. Pour déterminer la configuration optimale, ANN est basé sur une série d'expériences et nécessite une certaine expérience pour y parvenir [54] [55].

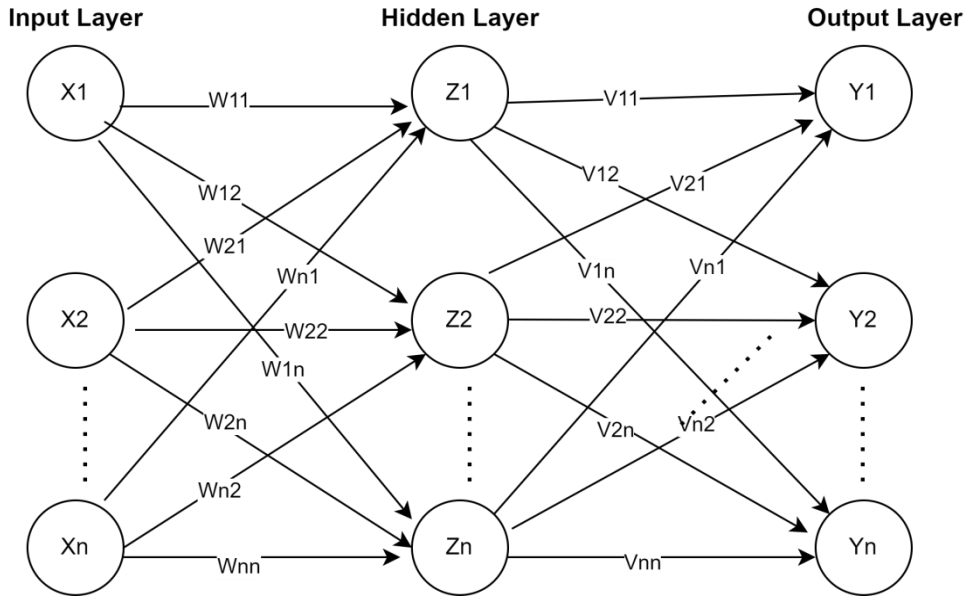

Figure 2.15 : Réseau multicouche de type feed-forward.

2.6.3 Nœud unique avec sa propre rétroaction

Dans un réseau récurrent monocouche, le réseau de rétroaction forme une boucle fermée. Dans ce modèle, un seul neurone reçoit une rétroaction sur lui-même et/ou sur d'autres neurones du réseau (voir la figure 2.16).

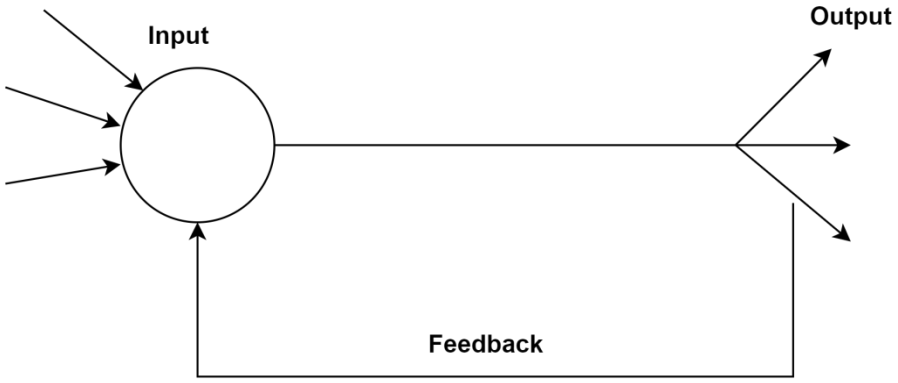

Figure 2.16 : Nœud unique avec sa propre rétroaction.

2.6.4 Réseau récurrent à une seule couche

Un réseau dans lequel la sortie d'une couche de sortie est renvoyée en tant qu'entrée vers une couche d'entrée ou une autre couche cachée est appelé un réseau de rétroaction. Un système de rétroaction à nœud unique n'a qu'une seule couche d'entrée dont la sortie est redirigée en tant que rétroaction (voir la figure 2.17).

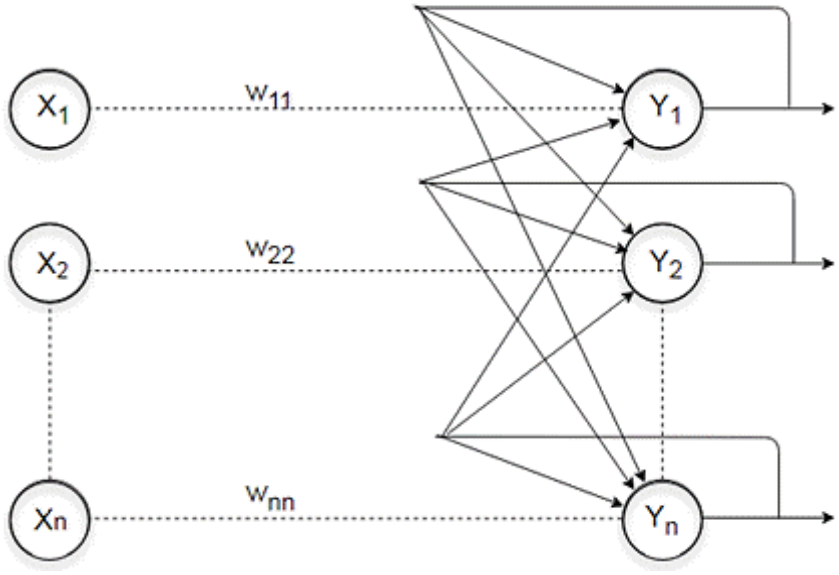

Figure 2.17 : Réseau récurrent à une seule couche.

2.6.5 Réseau récurrent multicouche

La technique de rétropropagation est un processus itératif de modification des poids de la couche de sortie à la couche d'entrée jusqu'à ce qu'aucune amélioration supplémentaire ne soit nécessaire (voir la figure 2.18). La technique de rétropropagation identifie une erreur et la propage du nœud de sortie au nœud d'entrée jusqu'à ce que l'erreur soit minimisée. Cette méthode repose essentiellement sur le principe de descendre les pentes les plus raides. L'objectif principal est de minimiser l'erreur entre les données de test réelles et la sortie du modèle [56].

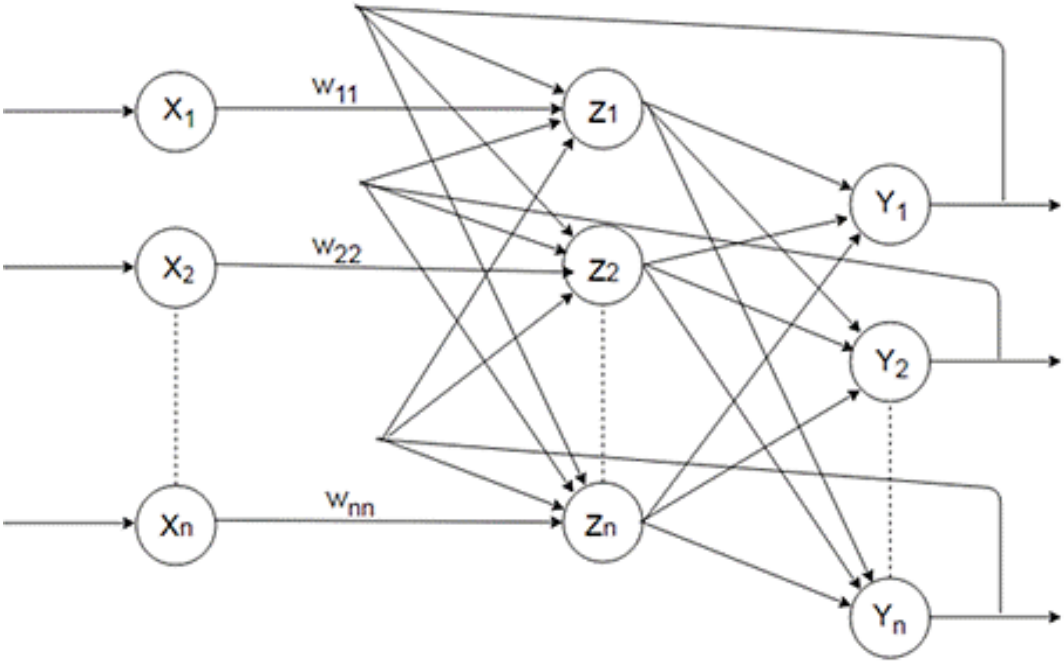

Figure 2.18 : Réseau récurrent multicouche.

2.7 Types des réseaux neuronaux

Le type d'un réseau de neurones dépend de la tâche à apprendre (problème à résoudre). Les modèles des réseaux de neurones se composent de différents réseaux profonds. Il existe de différents types, parmi lesquels :

2.7.1 Réseaux neuronaux convolutifs

Les ANN sont efficaces pour résoudre des problèmes simples, mais leur puissance de modélisation est limitée, le problème majeur lorsqu'il s'agit des applications plus complexes par exemple d'analyser de grandes images et des scènes visuelles [57]. C'est pourquoi les réseaux de neuronaux convolutifs seraient une solution idéale aux problèmes de vision par ordinateur.

Un réseau de neurone convolutif (CNN/ ConvNet) est une sous-catégorie de réseau de neurones profond. Ils sont spécialement conçus pour traiter des images (les données de pixels), analyser la parole, classifier des vidéos. Les CNN sont les modèles les plus performants pour la classification des images; ils se concentrent sur le fait que les informations d'entrée sont des images. Ils se composent généralement de trois types de couches : des couches de convolution (convolution layer), des couches de regroupement (pooling layer) et des couches entièrement connectées (Fully-connected layer) (voir la figure 2.19).

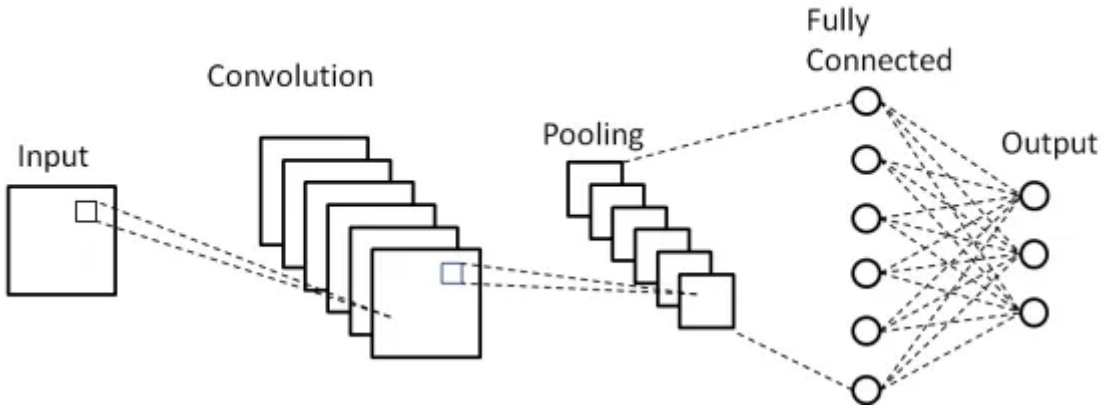

Figure 2.19 : Architecture de CNN.

2.7.1.1 Couche de convolution

La couche convolutive est un composant fondamental de l'architecture CNN qui effectue l'extraction des caractéristiques, laquelle consiste généralement en une combinaison d'opérations linéaires et non linéaires, à savoir l'opération de convolution et la fonction d'activation [58].

2.7.1.2 Couche de regroupement

Comme la couche convolutive, la couche de mise en commun balaie également un filtre sur l'image d'entrée. Mais contrairement à la couche convolutionnelle, la couche de mise en commun réduit le nombre de paramètres dans l'entrée et entraîne également une certaine perte d'informations. En revanche, cette couche réduit la complexité et améliore l'efficacité du CNN [58].

2.7.1.3 Couche entièrement connectée

La couche entièrement connectée est l'endroit où s'effectue la classification de l'image dans le CNN sur la base des caractéristiques extraites dans les couches précédentes. L'expression "entièrement connecté" signifie que toutes les entrées ou nœuds d'une couche sont connectés à chaque unité d'activation ou nœud de la couche suivante [58].

2.7.2 Réseaux neuronaux récurrents

Les réseaux neuronaux récurrents (RNN) sont des réseaux de neurones qui fonctionnent sur des données de séquence et sont utilisés beaucoup plus dans le traitement du langage naturel [59] [60]. Pour obtenir des informations contextuelles, chaque neurone du RNN reçoit non seulement l'état actuel, mais également les états précédents [46], alors que les réseaux de neurones profonds traditionnels supposent que les entrées et les sorties sont indépendantes les unes des autres (voir la figure 2.20).

Tous les réseaux de neurones récurrents ont la forme d'une chaine de modules (ou cellules) répétitifs de réseau de neurones. Dans les RNN standard, ce module répétitif aura une structure très simple, telle qu'une seule couche de Tanh.

Les RNN standard ne traitent que des séquences de longueur limitée. Pour éviter le problème de la dépendance à long terme, on utilise des cellules plus complexes telles que : LSTM (Long ShortTrem Memory ou mémoire à long terme et à court terme) a été proposé par Hochreiter et Schmidhuber en 1997 [61], GRU (Gated Recurrent Unit) a été proposé par Chung et al en 2014 [62] et le bi-RNN [63].

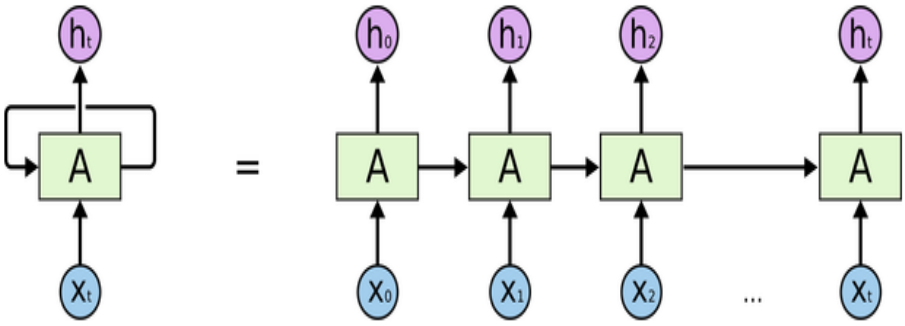

Figure 2.20 : Architecture de RNN.

2.8 Fonctions d'activation

La fonction d'activation c'est la partie essentielle de la conception d'un réseau de neurones, spécialement utilisée dans les réseaux neuronaux artificiels pour transformer un signal d'entrée en un signal de sortie qui, à son tour, sert d'entrée à la couche suivante. Elle va prendre une décision si le neurone est actif ou non.

Il existe plusieurs types de fonction d'activation, les fonctions les plus utilisées sont :

2.8.1 La fonction Sigmoid

C'est une fonction non-linéaire, généralement utilisée dans la couche de sortie d'une classification binaire, où le résultat est soit 0, soit 1. Comme la valeur de la fonction sigmoïde se situe uniquement entre 0 et 1, il est facile de prédire que le résultat sera 1 si la valeur est supérieure à 0,5 et 0 dans le cas contraire [64] (voir la figure 2.21).

$$y(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

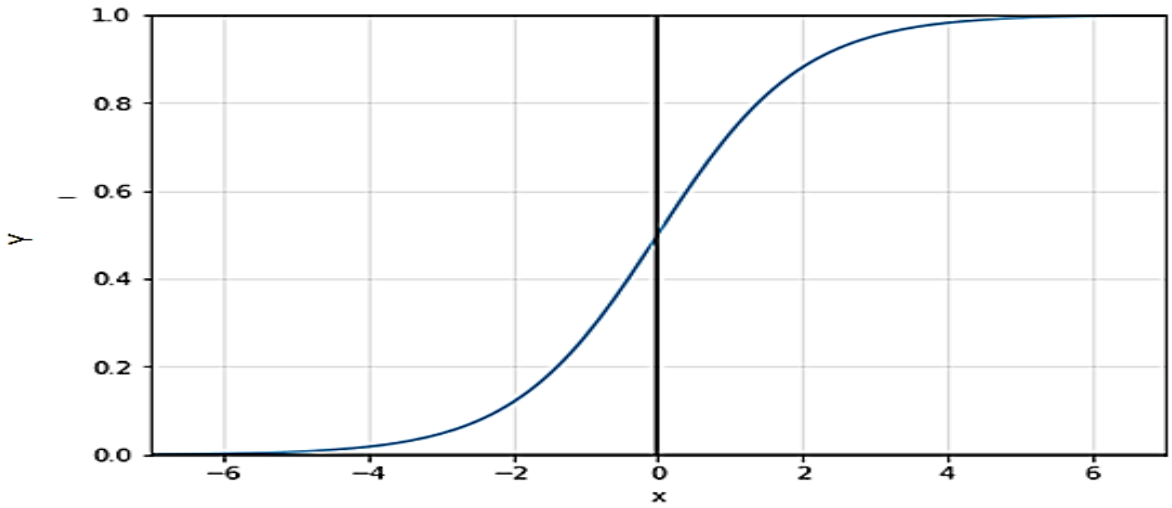

Figure 2.21 : Fonction Sigmoid.

2.8.2 La fonction Tanh

Il s'agit de la fonction tangente hyperbolique. Cette fonction est similaire à la fonction sigmoïde, mais elle est symétrique par rapport à l'origine [65]. Habituellement utilisé dans les couches cachées d'un réseau neuronal, ses valeurs sont comprises entre -1 et 1. La moyenne de la couche cachée est donc égale à 0 ou très proche, ce qui permet de centrer les données en rapprochant la moyenne de 0. Cela rend l'apprentissage de la couche suivante beaucoup plus facile [64] (voir la figure 2.22).

$$y\,(x) = \frac{e^{x} - e^{-x}}{e^{x} - e^{-x}} \tag{2.3}$$

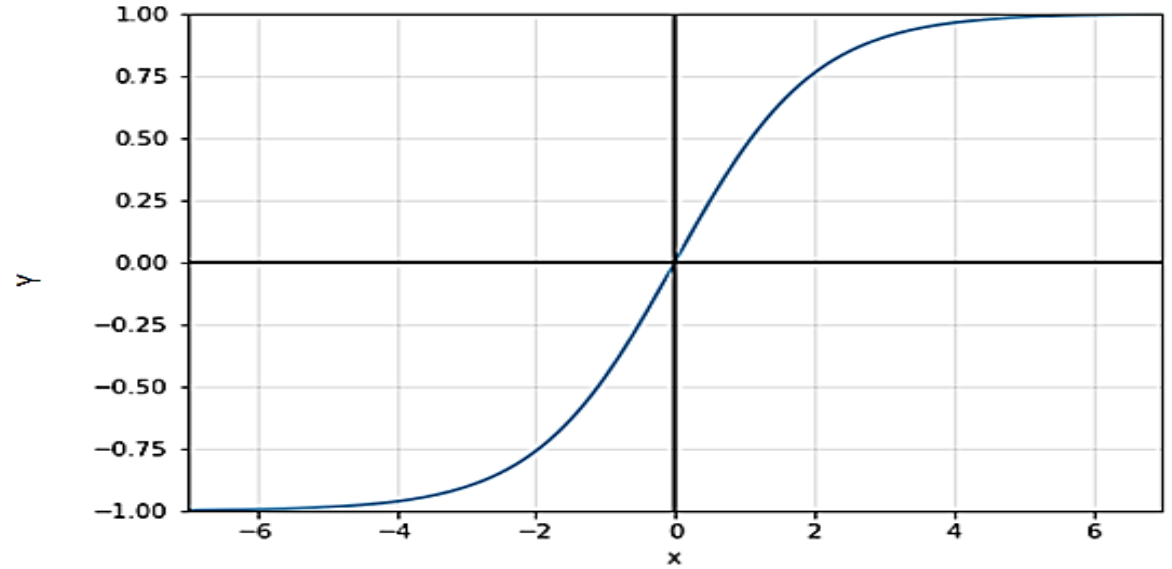

Figure 2.22 : Fonction Tanh.

2.8.3 La fonction Relu

ReLU est l'abréviation de Rectified Liner Unit (unité linéaire rectifiée).C'est une fonction d'activation non linéaire largement utilisée dans les réseaux neuronaux (voir la figure 2.23).

L'avantage de la fonction ReLU est que tous les neurones ne sont pas activés en même temps. Cela implique qu'un neurone ne sera désactivé que lorsque la sortie de la transformation linéaire est égale à zéro [65], aussi elle est moins coûteux que tanh et sigmoïde car il implique des opérations mathématiques plus simples. RELU apprend beaucoup plus rapidement que les fonctions sigmoïde et Tanh [64].

$$y(x) = \text{Max}(0, x) = \{0 \text{ si } x<0 \quad 1 \text{ sinon}\} \tag{2.4}$$

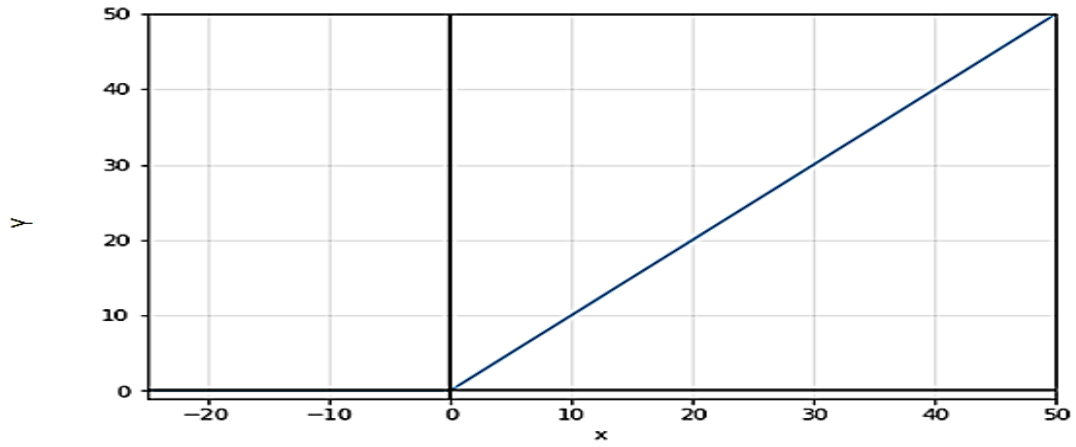

Figure 2.23 : Fonction ReLU.

2.8.4 La fonction Softmax

C'est une fonction non-linéaire. La fonction Softmax est une association de plusieurs fonctions sigmoïdes (voir la figure 2.24). La différence entre les deux c'est que la fonction Softmax traite des problèmes de classification multi-classes contrairement à la fonction sigmoïde qui est utilisée pour la classification binaire [65].

$$y(x)_j = \frac{e^{xj}}{\sum_{k=1}^{K} e^{xk}} \tag{2.5}$$

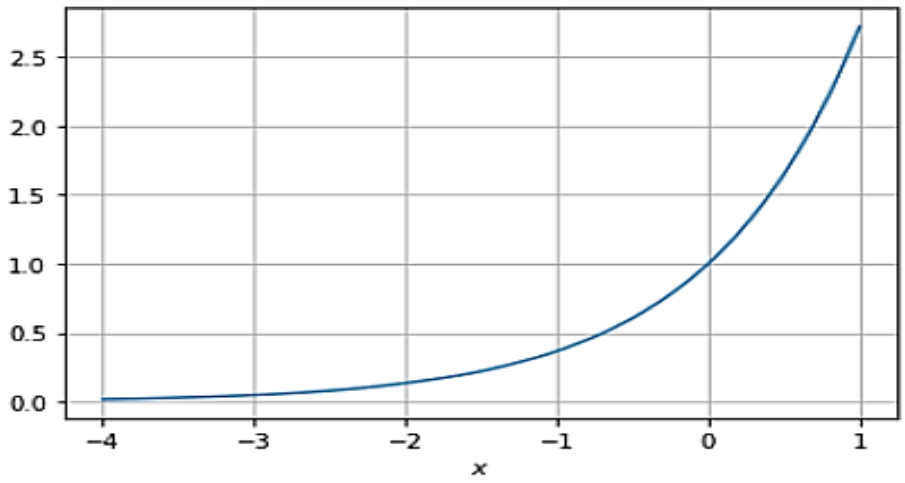

Figure 2.24 : Fonction Softmax.

2.9 Techniques d'évaluation des performances du modèle

Pour l'évaluation d'un modèle de classification, on définit les paramètres suivants

2.9.1 Matrice de confusion

Une matrice de confusion contient des informations sur les classifications réelles et attendues réalisées par un système de classification. Les performances de ces systèmes sont souvent évaluées à l'aide de données matricielles. Sur la diagonale principale, on trouve donc les valeurs bien classées, hors diagonale les éléments mal classés, (voir le tableau 2.1) [66].

Tableau 2.1 : Matrice de confusion.

		Classe Prédites	
		Décision Positive	Décision Négative
Classes Réelles	Etiquette Positive	True Positive **(TP)**	False Négative **(FN)**
	Etiquette Négative	False positive **(FP)**	True Négative **(TN)**

Chaque cellule de la matrice de confusion représente un facteur d'évaluation.

- **True Positive(TP):** Il s'agit du nombre de résultats initialement positif et prédits comme étant positifs.

- **True Negative(TN):** Il s'agit du nombre de résultats initialement négatifs et prédits comme étant négatifs.

- **False Negative(FN):** Il s'agit du nombre de résultats initialement négatifs et prédits comme étant positifs.

- **False Positive(FP):** Il s'agit du nombre de résultats initialement positifs et prédits comme étant négatifs.

2.9.2 Accuracy

Accuracy est une mesure de la précision des prédictions positives d'un modèle. Elle est définie comme le rapport entre les vraies prédictions et le nombre total de prédictions faites par le modèle.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

2.9.3 Précision

La précision est utilisée pour mesurer la performance du modèle. Il s'agit du rapport des vrais positifs et le total des vrais positifs et des faux positifs.

$$Précision = \frac{TP}{TP + FP} \tag{2.7}$$

2.9.4 Spécificité

La spécificité est également connue sous le nom de taux de vrais négatifs (TNR). Elle s'agit du nombre de prédictions négatives divisé par le nombre total de négatifs réels.

$$Spécificité = \frac{TN}{TN + FP} \tag{2.8}$$

2.9.5 Rappel (Recall)

Le rappel ou la sensibilité est également connue sous le nom de taux de vrais positifs (TPR). Le rappel mesure l'efficacité d'un modèle de classification à identifier toutes les instances pertinentes d'un ensemble de données. Il s'agit du rapport entre le nombre d'instances réellement positives (TP) et la somme des instances réellement positives et faussement négatives (FN).

$$TPR = \frac{TN}{TN+FP} \tag{2.9}$$

2.9.6 Taux de faux positifs

Le taux de faux positifs FPR est calculé comme le rapport entre le nombre d'échantillons faussement positifs et le nombre total d'échantillons réellement négatifs.

$$FPR = \frac{FP}{FP + TN} \tag{2.10}$$

2.9.7 Taux de faux négatifs

Le taux de faux négatifs FNR est calculé comme le rapport entre le nombre d'échantillons faussement négatifs et le nombre total d'échantillons réellement positifs.

$$FNR = \frac{FN}{FN + TP} \tag{2.11}$$

2.9.8 Score F1

Le score F1 (F-measure) est utilisé pour évaluer la performance globale d'un modèle de classification. Il s'agit de la moyenne harmonique de la précision et du rappel.

$$F1 - score = 2 \times \frac{\Pr ecision \times \mathrm{Re} call}{\Pr ecision + \mathrm{Re} call} \tag{2.12}$$

2.9.9 Caractéristique de performance

La caractéristique de performance, plus fréquemment désignée sous le terme courbe ROC (de l'anglais Receiver Operating Characteristic) est utilisée pour visualiser les performances des données catégorielles multidimensionnelles. Elle est considérée comme l'une des mesures d'évaluation les plus importantes pour évaluer la précision des modèles de classification. Elle est également connue sous le nom d'AUROC (Area Under Receiver Operating Characteristics). Modélisons le réseau neuronal et faisons des prédictions (voir la figure 2.25) [67].

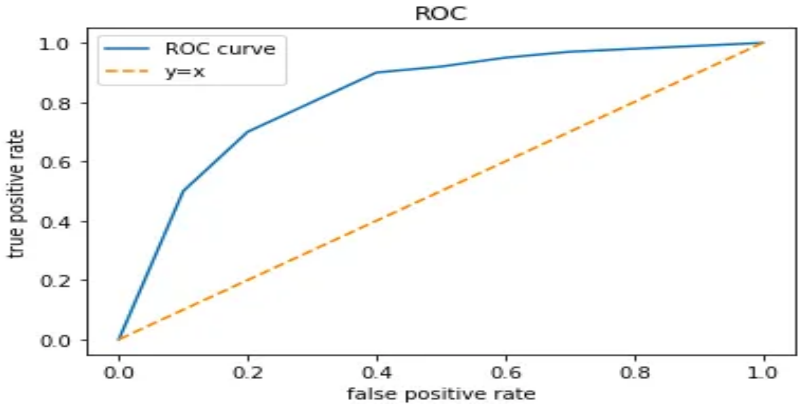

Figure 2.25 : Courbe ROC.

2.10 Conclusion

Ce chapitre me permettra de mettre en pratique ce que j'ai appris dans la partie théorique.

Le chapitre suivant couvrira la conception et la mise en œuvre du projet, en commençant par créer un modèle capable de classer les attaques DDoS à l'aide de perceptrons multicouches puis discuter les différents résultats obtenus.

Chapitre 3

Implémentation, résultat et discussion

3.1 Introduction

L'objectif de ce chapitre est de présenter la réalisation de modèles de réseaux neuronaux profonds (DNN) pour la détection des attaques DDoS en utilisant différentes approches et techniques d'apprentissage en profondeur. Nous avons également évalué la performance de ces modèles en termes d'accuracy, de précision, de rappel et de f1-score. En tirant parti des techniques d'apprentissage profond, ce travail vise à améliorer la robustesse de l'infrastructure du réseau contre les activités malveillantes. Pour atteindre cet objectif, nous avons utilisé le dataset CSE-CIC-IDS2018, qui est un ensemble de données complet et largement utilisé dans le domaine de la sécurité des réseaux.

3.2 Outils de développement

Les outils de développement sont utilisés pour pouvoir réaliser la classification à l'aide du Deep Learning pour la détection des attaques DDoS. Il est important d'utiliser un ensemble d'outils informatiques adéquats et performants pour obtenir les meilleurs résultats. Les outils que nous avons utilisés lors de notre recherche sont les suivants :

3.2.1 Hardware

Lors de la réalisation de notre projet, nous avons utilisé un ordinateur personnel pour effectuer nos recherches. Le tableau 3.1 représente les différentes caractéristiques de la machine utilisée durant le développement de ce travail.

Tableau 3.1 : Caractéristiques de l'ordinateur utilisé.

Modèle de processeur	Intel® Core™ i5-4310U
Fréquence CPU	2.00GHz 2.60 GHz
RAM	8 GB
OS	Microsoft Windows 10 -64 bits.

3.2.2 Software

3.2.2.1 Langage utilisé

Python

Le langage Python est un langage de programmation open source multiplateformes et orienté objet. Grâce à des bibliothèques spécialisées, Python s'utilise pour de nombreuses situations comme le développement logiciel, l'analyse de données, ou la gestion d'infrastructures. Python permet de créer des programmes de manière simple et rapide [68].

3.2.2.2 Bibliothèques utilisées

TensorFlow

Tensorflow est une plateforme open-source pour l'apprentissage automatique et une bibliothèque mathématique symbolique utilisée pour les applications d'apprentissage automatique [69].

Keras

Il s'agit d'une bibliothèque de réseaux neuronaux open source qui s'exécute au-dessus de Tensorflow. Elle est conçue pour être rapide et facile à utiliser pour l'utilisateur. C'est une bibliothèque utile pour construire n'importe quel algorithme d'apprentissage profond de n'importe quel choix [69].

Pandas

Pandas est une bibliothèque open source conçue pour Python. Elle a été créée pour s'attaquer à différentes tâches liées à la manipulation et à l'analyse des données [50].

NumPy

NumPy est une bibliothèque Python open source utilisée pour manipuler de grands tableaux multidimensionnels. Elle a également été créée avec un grand nombre de fonctions mathématiques permettant d'opérer sur de tels tableaux [50].

Scikit-Learn

Scikit-Learn permet d'intégrer facilement et rapidement des méthodes d'apprentissage automatique dans le code Python. La bibliothèque Scikit-Learn comprend une large gamme de méthodes de classification, de régression, de réduction de la dimensionnalité, de prétraitement des données [70].

Matplotlib

Matplotlib est une bibliothèque conçue pour tracer et visualiser des graphiques via le langage de programmation Python [71].

3.3 Plateforme et environnement de développement

Google Colab

Google Colab ou Colaboratory est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur [72].

Anaconda

Anaconda est une plateforme libre et gratuite, largement utilisée pour apprendre et utiliser les langages de programmation Python pour les domaines du calcul scientifique, de la science des données et de l'apprentissage automatique. Elle simplifie la gestion des paquets et offre de nombreuses bibliothèques/packages [73].

Jupyter

Jupyter est une application web qui permet de créer et de partager des documents contenant du code, des équations et des visualisations [73].

Kaggle

La plateforme kaggle, spécifiquement destinée aux experts en données et aux amateurs de l'apprentissage automatique, est un espace communautaire en ligne qui favorise la collaboration entre utilisateurs. Elle offre également la possibilité de trouver et de publier des jeux de données.

Notre choix s'est porté sur Kaggle en tant que plateforme pour nous aider dans cette étude. En effet, Kaggle est une plateforme qui offre plusieurs avantages. Tout d'abord, elle propose un accès facile à des ressources de calcul haute performance telles que des CPU, GPU et TPU pour des tâches de traitement intensif. De plus, Kaggle permet également l'importation directe de données à partir de sources en ligne sans avoir à les télécharger sur votre ordinateur local, ce qui peut être très pratique pour travailler avec de grands ensembles de données (big data). En outre, la plateforme propose des outils pour la visualisation et l'analyse des données, ainsi que des fonctionnalités pour la collaboration et le partage de projets.

3.4 Dataset

Le jeu de données choisi pour cette étude est le dataset CSE-CIC-IDS2018. Ce jeu de données a été créé dans le cadre d'un projet de collaboration entre le Centre de la Sécurité des Télécommunications (CSE) et l'institut canadien de cybersécurité (CIC), qui utilise des profils pour générer des ensembles de données de cybersécurité de manière systématique. Il fournit une description détaillée des intrusions, ainsi que des modèles de distribution abstraits pour les applications, les protocoles et les entités de réseau de niveau inférieur [74]. L'ensemble de données est disponible en téléchargement sur Amazon Web Services (AWS) [75].

3.4.1 Description du dataset CSE-CIC-IDS2018

Ce dataset contient dix fichiers CSV représentant dix jours du flux de réseau capturé, avec plus de 16,2 millions d'échantillons. En outre, plus de 80 caractéristiques ont été extraites par l'outil CICFlowMeter (voir le tableau 3.2).

Tableau 3.2 : Caractéristiques du dataset CSE-CIC-IDS2018.

Nom de dataset	**CSE-CIC-IDS2018**
Type de dataset	Multi-classe
Nombre total des lignes (Nombre d'instances)	16232943
Total des colonnes (Nombre de features)	84
Nombre de classes distinctes	14

Ce dataset comprend six principaux types d'attaques d'intrusion : déni de service distribué (DDoS), déni de service (DoS), Botnet, Brute Force, infiltration, and web attacks, comme le montre le tableau 3.3.

Tableau 3.3 : Description des fichiers contenant le dataset CSE-CIC-IDS2018.

Jour	Nom de fichier	Type d'attaque	Nombre	Taille
1	Wednesday-14-02-2018	Benign FTP-BruteForce SSH-Bruteforce	667626 193360 187589	358.22 MB
2	Thursday-15-02-2018	Benign DoS attacks-GoldenEye DoS attacks-Slowloris	996077 41508 10990	375.95 MB
3	Friday-16-02-2018	Benign DoS attacks-Hulk DoS attacks-SlowHTTPTest	446772 461912 139890	333.72 MB
4	Tuesday-20-02-2018	Benign DDoS attacks-LOIC-HTTP	7372557 576191	4.05 GB
5	Wednesday-21-02-2018	Benign DDOS attack-HOIC DDOS attack-LOIC-UDP	360833 686012 1730	328.89 MB
6	Thursday-22-02-2018	Benign Brute Force-Web Brute Force-XSS SQL Injection	1048213 249 79 34	382.64 MB
7	Friday-23-02-2018	Benign Brute Force-Web Brute Force-XSS SQL Injection	1048009 362 151 53	382.84 MB
8	Wednesday-28-02-2018	Benign Infilteration	544200 68871	209.25 MB
9	Thursday-01-03-2018	Benign Infilteration	238037 93063	107.84 MB
10	Friday-02-03-2018	Benign Botnet	762384 286191	352.37 MB

3.4.1.1 Types des colonnes dans le dataset CSE-CIC-IDS2018

Le tableau 3.4 liste les différents « Features » ainsi que leurs types appartenant au dataset CSE-CIC-IDS2018.

Tableau 3.4 : Types des Features dans le dataset CSE-CIC-IDS2018.

Features	**Type**	**Features**	**Type**
Flow ID	Object	Fwd Pkts/s	float64
Src IP	Object	Bwd Pkts/s	float64
Src Port	float64	Pkt Len Min	float64
Dst IP	Object	Pkt Len Max	float64
Dst Port	int64	Pkt Len Mean	float64
Protocol	int64	Pkt Len Std	float64
Timestamp	Object	Pkt Len Var	float64
Flow Duration	int64	FIN Flag Cnt	Int 64
Tot Fwd Pkts	int64	SYN Flag Cnt	int64
Tot Bwd Pkts	int64	RST Flag Cnt	int64
TotLen Fwd Pkts	float64	PSH Flag Cnt	int64
TotLen Bwd Pkts	float64	ACK Flag Cnt	int64
Fwd Pkt Len Max	float64	URG Flag Cnt	int64
Fwd Pkt Len Min	float64	CWE Flag Count	int64
Fwd Pkt Len Mean	float64	ECE Flag Cnt	int64
Fwd Pkt Len Std	float64	Down/Up Ratio	float64
Bwd Pkt Len Max	float64	Pkt Size Avg	float64
Bwd Pkt Len Min	float64	Fwd Seg Size Avg	float64
Bwd Pkt Len Mean	float64	Bwd Seg Size Avg	float64
Bwd Pkt Len Std	float64	Fwd Byts/b Avg	int64
Flow Byts/s	float64	Fwd Pkts/b Avg	int64
Flow Pkts/s	float64	Fwd Blk Rate Avg	int64
Flow IAT Mean	float64	Bwd Byts/b Avg	int64
Flow IAT Std	float64	Bwd Pkts/b Avg	int64
Flow IAT Max	float64	Bwd Blk Rate Avg	int64
Flow IAT Min	float64	Subflow Fwd Pkts	int64
Fwd IAT Tot	float64	Subflow Fwd Byts	int64
Fwd IAT Mean	float64	Subflow Bwd Pkts	int64
Fwd IAT Std	float64	Subflow Bwd Byts	int64
Fwd IAT Max	float64	Init Fwd Win Byts	int64
Fwd IAT Min	float64	Init Bwd Win Byts	int64
Bwd IAT Tot	float64	Fwd Act Data Pkts	int64
Bwd IAT Mean	float64	Fwd Seg Size Min	int64
Bwd IAT Std	float64	Active Mean	float64
Bwd IAT Max	float64	Active Std	float64
Bwd IAT Min	float64	Active Max	float64
Fwd PSH Flags	int64	Active Min	float64
Bwd PSH Flags	int64	Idle Mean	float64
Fwd URG Flags	int64	Idle Std	float64
Bwd URG Flags	int64	Idle Max	float64
Fwd Header Len	int64	Idle Min	float64
Bwd Header Len	int64	Label	object

CSE-CIC-IDS2018 est un dataset composé de dix fichiers dont deux parmi eux contiennent les attaques DDoS (les fichiers quatre et cinq). Le premier fichier a une taille de 4Go et contient deux types de trafic : le flux bénigne (trafic normal) et une attaque DDoS. Le trafic normal (bénigne) représente 93% du fichier alors que les 7% restant constitue l'attaque DDoS-LOIC-HTTP. Le second fichier à son tour est composé du trafic normal avec un taux de 34% ainsi que de deux sortes d'attaques DDoS, il s'agit des attaques DDOS-HOIC et DDOS-LOIC-UDP avec un pourcentage respectivement de 65% et 1% (voir tableau 3.5).

Cependant, nous n'allons pas utiliser la totalité du trafic Benign (7372557 instances) du quatrième fichier. Nous avons supprimé 60% du flux normal (Benign) afin d'éviter un énorme déséquilibre entre le trafic d'attaque et le trafic normal, ce qui rendrait plus difficile la convergence du modèle dans la phase d'apprentissage. Les caractéristiques du dataset combiné et l'occurrence détaillée par classe sont présentées dans le tableau 3.6.

Après, nous avons fusionné ces deux fichiers en un seul fichier CSV appelé "data.csv" qui contient 4573789 instances. Le dataset ainsi obtenu est nommé CSE-CIC-IDS2018-DDoS. Les caractéristiques du dataset combiné sont présentées dans le tableau 3.7.

Tableau 3.5 : Description des fichiers contenant le dataset CSE-CIC-IDS2018-DDoS.

Jour	Nom de fichier	Features	Type d'attaque	Nombre	Taille
4	Thursday-20-02-2018	84	Benign DDoS attacks-LOIC-HTTP	7372557 576191	4.05 GB
5	Wednesday-21-02-2018	80	Benign DDOS attack-HOIC DDOS attack-LOIC-UDP	360833 686012 1730	328.89 MB

Tableau 3.6 : Description des fichiers contenant le dataset CSE-CIC-IDS2018-DDoS.

Jour	Nom de fichier	Features	Type d'attaque	Nombre	Taille
4	Thursday-20-02-2018	84	Benign DDoS attacks-LOIC-HTTP	2949023 576191	1.87 GB
5	Wednesday-21-02-2018	80	Benign DDOS attack-HOIC DDOS attack-LOIC-UDP	360833 686012 1730	328.89 GB

Tableau 3.7 : Caractéristiques de dataset CSE-CIC-IDS2018-DDoS.

Nom de dataset	CSE-CIC-IDS2018-DDoS
Type de dataset	Multi-classe
Total des lignes (Nombre total d'instances)	4573789
Total des colonnes (Nombre de Features)	84
Target	Label
Nombre de classes distinctes	4
classe-Label	Benign DDoS attacks-LOIC-HTTP DDOS attack-HOIC DDOS attack-LOIC- UDP

3.5 Implémentation

La figure 3.1 représente une classification schématique des attaques DDoS utilisant un modèle de Deep Learning.

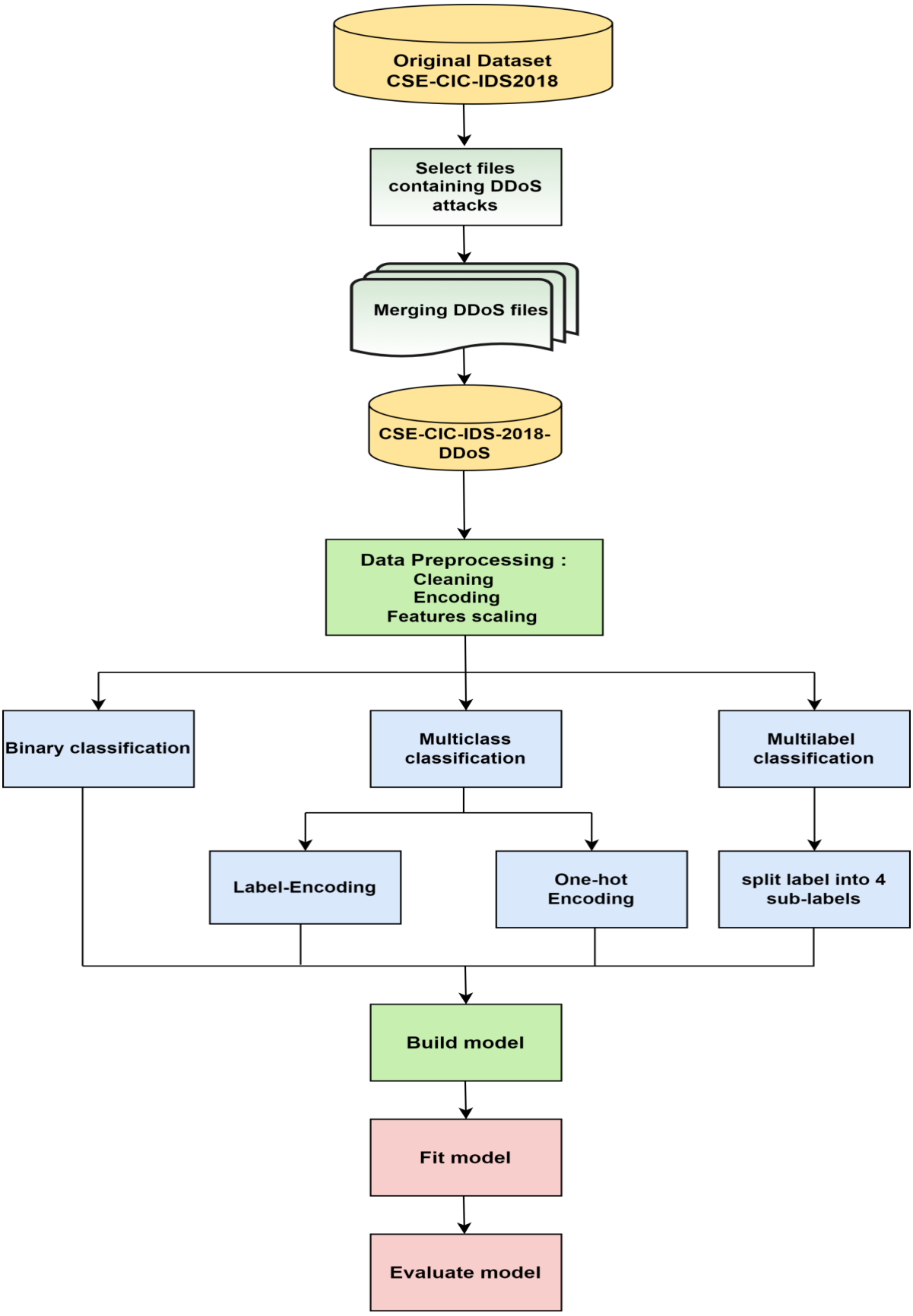

Figure 3.1 : Diagramme de détection des attaques DDoS.

3.6 Prétraitement de dataset

Lorsqu'on travaille avec un dataset, il est essentiel de s'assurer de la qualité et l'organisation ainsi que la clarté des données avant de les utiliser pour entraîner un modèle de Deep Learning. Les réseaux de neurones profonds nécessitent généralement des quantités massives de données pour leur apprentissage, ce qui peut entraîne des problèmes : des données inutiles, redondantes, erronées ou manquantes NaN (Not a Number), infini (INF) et d'autres problèmes.

Le prétraitement des données est une étape cruciale pour garantir que les données d'entrée du modèle de Deep Learning sont propres, cohérentes et pertinentes pour la tâche de classification ou de prédiction que l'on souhaite réaliser. Il permet d'améliorer la qualité des données et de les rendre plus cohérentes et de les adapter aux besoins du modèle que l'on souhaite construire. Le prétraitement des données peut également contribuer à réduire les temps d'apprentissage et de test du modèle. En veillant à ce que nos données soient propres et prêtes à être utilisées pour entraîner notre modèle, nous nous assurons que les performances de celui-ci ne sont pas compromises par des données manquantes, erronées ou redondantes. Le prétraitement des données est donc une étape importante pour garantir la fiabilité et l'efficacité de notre modèle de Deep Learning.

Les étapes de prétraitement des données pour le Deep Learning peuvent inclure diverses techniques telles que la normalisation des données, la suppression des données infinis (INF), la gestion des données manquantes (NaN), la suppression des lignes dupliquées, la suppression des données inutiles et le codage des caractéristiques et d'autres techniques.

3.6.1 Nettoyage des données

Le nettoyage des données permet d'avoir un dataset propre et fiable pour effectuer des analyses et des modélisations plus précises. Après analyse du dataset, nous avons trouvé un pourcentage de 1.1% des valeurs manquantes (voir la figure 3.2). Nous avons procédé au nettoyage de l'ensemble des données en supprimant les fonctionnalités inutiles ("Timestamp", "Flow ID", "Src IP", "Src Port" et "Dst IP") car leurs propriétés sont liées aux informations de contact et ne représentent pas des caractéristiques importantes des attaques. Puis, nous avons supprimé les valeurs de l'infini, du NAN et les lignes dupliquées. Le tableau 3.8 représente les différentes fonctions utilisées pour le nettoyage des données.

Tableau 3.8 : Fonctions utilisées pour le nettoyage des données.

La fonction	Description
drop()	permet de supprimer les lignes ou les colonnes d'un dataset.
replace ([np.inf , -np.inf] , np.nan)	permet de remplacer les valeurs infinies inf ou -inf par des valeurs manquantes NaN.
isna().sum()	Permet de compter le nombre de NaN dans chaque colonne d'un dataset.
dropna (axis=0)	permet de supprimer les lignes contenant des valeurs manquantes.
duplicated ()	permet de détecter les lignes dupliquées.
drop_duplicates ()	permet de supprimer les lignes dupliquées.

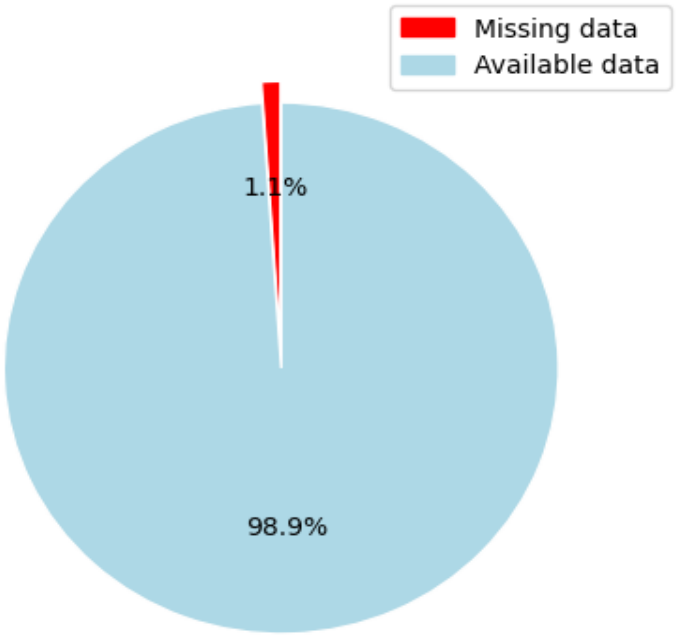

Figure 3.2 : Pourcentage des valeurs NaN dans le dataset.

3.6.2 Encodage des données

Dans certains cas, les variables catégorielles peuvent être difficiles à utiliser dans des analyses statistiques ou des algorithmes de Machine Learning. Pour cette raison, il faut transformer les variables catégorielles en une forme numérique appropriée. Pour cela, nous avons utilisé la méthode str. replace () de Pandas. Cette méthode nous a permis de remplacer les différentes valeurs catégorielles par des entiers. Nous avons ensuite utilisé la méthode astype() pour convertir le type de données en entiers 64 bits. Cette étape était nécessaire pour que la colonne puisse être utilisée comme variable d'entrée dans certains algorithmes de Machine Learning. En utilisant ces deux méthodes, nous avons pu transformer efficacement la colonne "Label" en une variable numérique appropriée pour notre analyse.

3.6.3 Normalisation et standardisation

3.6.3.1 Normalisation

La normalisation permet de mettre sur une même échelle toutes les variables quantitative afin qu'elles se situent dans un intervalle compris entre 0 et 1 ce qui facilite beaucoup l'apprentissage de la machine.

$$\mathrm{Zi} = \frac{\mathrm{Xi} - \min(\mathrm{X})}{\max(X) - \min(X)} \tag{3.1}$$

3.6.3.2 Standardisation

La standardisation est une méthode de mise à l'échelle qui transforme les données en une distribution gaussienne avec une moyenne égale à 0 et un écart type égal à 1 [50].

$$\mathrm{Zi} = \frac{\mathrm{Xi} - \mathrm{mean}(\mathrm{X})}{std(X)} \tag{3.2}$$

Afin d'améliorer les performances et la fiabilité de notre modèle DNN et de l'aider à converger rapidement, nous avons appliqué la méthode de standardisation aux données d'apprentissage et de test avec la fonction StandardScaler du module sklearn.preprocessing de la bibliothèque Python Scikit-learn. Cette méthode est une étape importante dans le prétraitement des données car elle permet de rendre les données comparables ce qui facilite la comparaison entre elles.

3.7 Création de données d'apprentissage et de test

En Deep Learning, il ne faut jamais évaluer la performance de modèle sur les mêmes données qui ont servi à son entrainement.

Lorsque l'on entraîne un modèle d'apprentissage profond, il est important de créer deux ensembles de données : un ensemble de données d'apprentissage et un ensemble de données de test. L'ensemble de données d'apprentissage est utilisé pour entraîner le modèle, tandis que

l'ensemble de données de test est utilisé pour évaluer les performances du modèle. En général, on met 80% des données dans la partie apprentissage et 20% pour la partie test (voir la figure 3.3). Pour faire tout ça dans python, on utilise la fonction train_test_split() qui appartient au module model_selection.

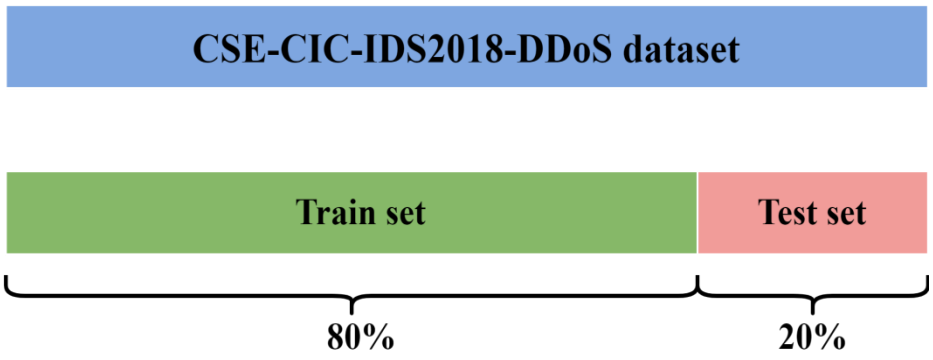

Figure 3.3 : Division du dataset CSE-CIC-IDS2018-DDoS en deux parties.

Pour l'entraînement du modèle, nous avons utilisé trois approches différentes :

3.7.1 Classification "multi-classe"

Dans cette approche, nous avons utilisé deux méthodes pour faire l'encodage de la colonne "Label" qui constitue notre target.

3.7.1.1 Label Encoding

Le "Label Encoding" est une méthode simple et rapide qui consiste à transformer des variables catégorielles en variables numériques. Cette méthode remplace chaque catégorie par un entier unique. L'objectif de cette approche est de faciliter l'entraînement du modèle.

Dans ce cas, nous allons appliquer cette méthode à la variable cible du dataset appelée "Label" qui contient quatre différentes classes : Benign, DDoS attacks-LOIC-HTTP, DDOS attack-HOIC et DDOS attack-LOIC-UDP.

Avec la méthode de "Label Encoding", les catégories Benign, DDoS attacks-LOIC-HTTP, DDOS attack-HOIC et DDOS attack-LOIC-UDP seront remplacées respectivement par les entiers 0, 1, 2 et 3 (voir la figure 3. 4).

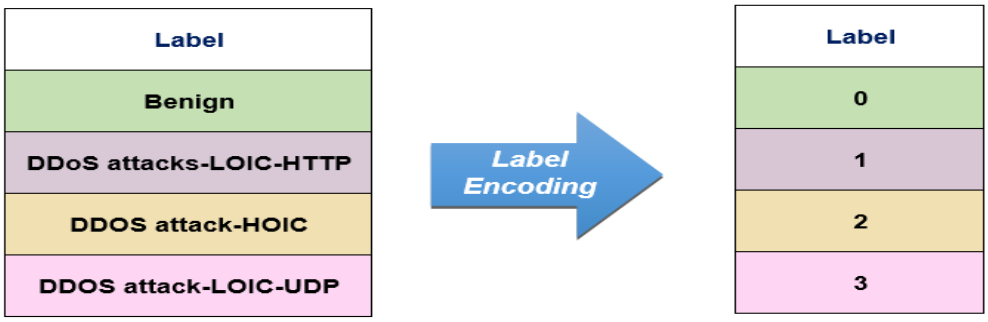

Figure 3.4 : Label Encoding.

3.7.1.2 One-Hot Encoding

Dans cette approche, nous allons utiliser une autre méthode qui s'appelle One-Hot Encoding. Cette méthode consiste à transformer chaque classe de la variable cible "Label" en un vecteur binaire unique. Chaque vecteur aura une longueur égale au nombre de classes différentes dans la variable cible, et aura une seule valeur 1 qui correspond à la classe de la variable cible pour cette observation et la valeur 0 pour les autres. Par exemple, si l'observation appartient à la classe Benign, le vecteur binaire correspondant serait [1, 0, 0, 0] (voir la figure 3.5).

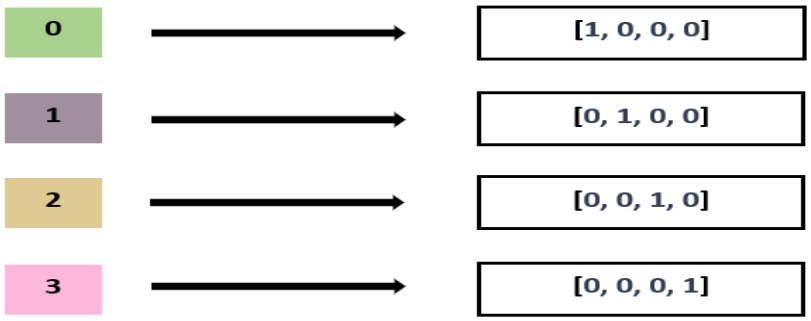

Figure 3.5 : One-hot Encoding.

3.7.2 Classification binaire

Dans cette approche, nous avons regroupé les trois types des attaques DDoS (DDoS attacks-LOIC-HTTP, DDOS attack-HOIC et DDOS attack-LOIC- UDP) en un seul type nommé "DDoS", plus le flux "Benign" (voir Tableau 3.9).

L'objectif de cette approche est de simplifier la variable cible en réduisant le nombre de classes, tout en conservant une distinction importante entre les observations DDoS et Benign.

Tableau 3.9 : Conversion de dataset en deux étiquettes : "Benign" et "DDoS".

Classes étiquettes	Nouvelles étiquettes	Nombre d'instances
DDoS attacks-LOIC-HTTP DDOS attack-HOIC DDOS attack-LOIC-UDP	DDoS	1263933
Benign	Benign	3309856

La colonne Label qui représente la classe de chaque instance, a été encodée. L'encodage convertit chaque valeur de chaîne dans la colonne Label en "0" ou "1". Le "0" représente le Benign et le "1" représente le DDoS. La figure 3.6 montre les types d'attaques et la répartition des flux de données dans le dataset.

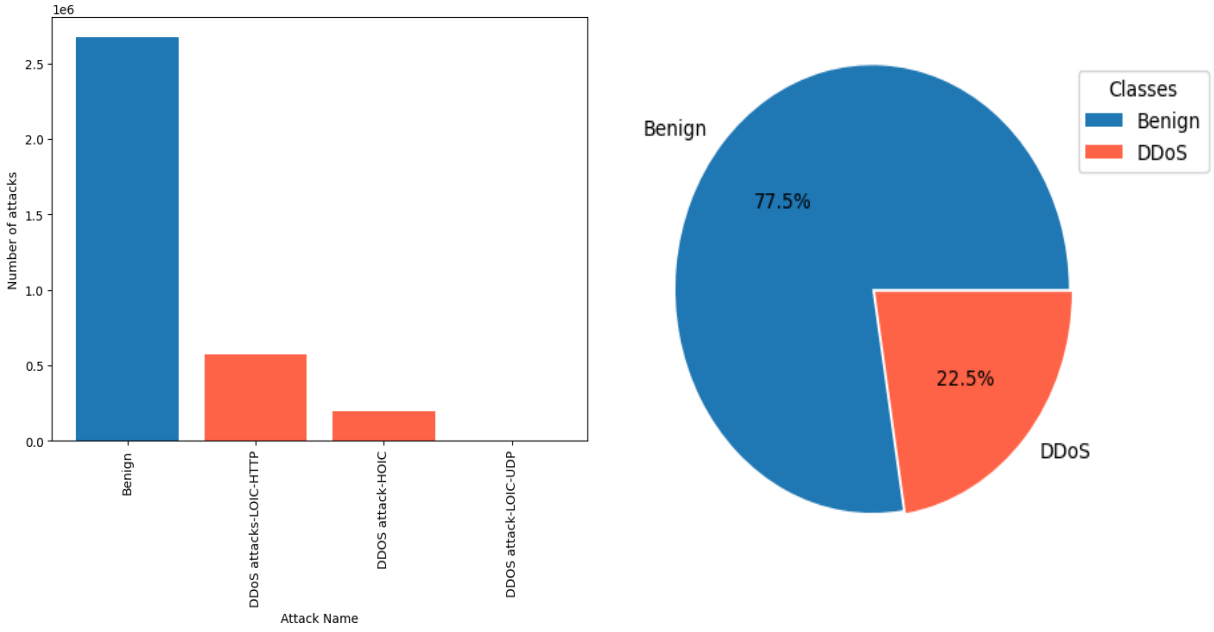

Figure 3.6 : Types d'attaques et la répartition des flux de données dans le dataset.

3.7.3 Classification "multi-label"

Dans cette approche, on va utiliser une autre technique d'encodage qui s'appelle multi-label encoding. Cette méthode consiste à créer une nouvelle colonne pour chaque catégorie de

la variable cible "Label", et à attribuer une valeur de 1 à la colonne correspondant à la catégorie de la variable, et 0 à toutes les autres colonnes. Dans ce cas, on aura quatre variables cibles différentes : Benign, DDoS-LOIC-HTTP, DDOS attack-HOIC et DDOS-HOIC-LOIC-UDP (voir la figure 3.7).

L'objectif de cette approche est de permettre aux modèles de distinguer les différentes classes de la variable cible de manière plus fine.

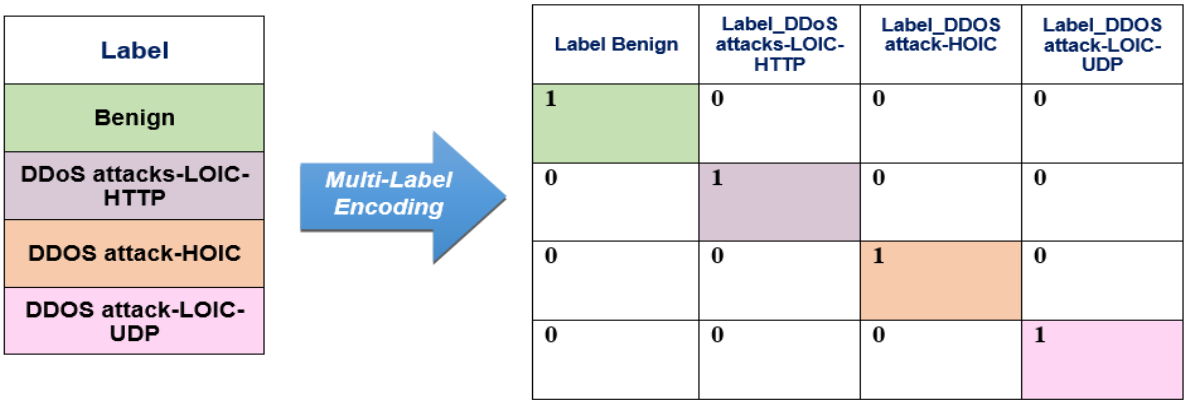

Figure 3.7 : Multi-label encoding.

Dans cette section, nous avons présenté les trois différentes approches utilisés pour préparer la variable "Label" de dataset CSE-CIC-IDS2018-DDoS. Maintenant, nous allons passer au modèle.

3.7.4 Architecture du modèle

Nous avons construit un modèle de réseau de neurones multicouches (Multilayer Perceptron) avec quatre couches cachées de type Dense. Les couches cachées utilisent la fonction d'activation ReLU, tandis que la couche de sortie utilise la fonction d'activation sigmoid pour la classification binaire/classification multi-label et softmax pour la classification multi-classe. Le nombre de neurones dans chaque couche est de 128 pour la première couche cachée, 64 pour la deuxième couche cachée, 32 pour la troisième et la quatrième couche cachée (voir la figure 3.8). La couche de sortie dépend de l'approche utilisée.

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 dense (Dense)                (None, 128)               10112

 dense_1 (Dense)              (None, 64)                8256

 dense_2 (Dense)              (None, 32)                2080

 dense_3 (Dense)              (None, 32)                1056

 dense_4 (Dense)              (None, 4)                 132

=================================================================
Total params: 21,636
Trainable params: 21,636
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.8 : Modèle DNN utilisé.

Pour éviter le surapprentissage (overfitting), le modèle a été régularisé en utilisant une pénalisation L2, avec un coefficient de régularisation de 0.0001 appliqué sur le nœud.

Le modèle est compilé avec l'optimiseur Adamax. Le choix de la fonction de perte dépend de l'approche utilisée, soit categorical_crossentropy pour la classification multi-classe, soit binary_crossentropy pour la classification binaire et la classification multi-label. L'évaluation du modèle se fait à l'aide de la métrique "accuracy". Le modèle est entraîné avec un batch size de 128. Les données sont divisées en 90% pour l'apprentissage et 10% pour la validation. Enfin, le modèle est évalué sur l'ensemble de test. Le tableau 3.10 récapitule les hyperparamètres utilisés dans chaque approche.

Tableau 3.10 : Hyperparamètre expérimental du modèle DNN proposé pour chaque approche.

Approche	**Neurones par couche cachée**	**Fonction de perte**	**Métrique d'évaluation**	**Nombre de sorties**
Multi-classe (Label Encoding)	128,64, 32,32	Sparse-categorical-crossentropy	Softmax	Nombre de classes
Multi-classe (One-hot Encoding)	128,64, 32,32	categorical-crossentropy	Softmax	Nombre de classes
Classification binaire	128,64, 32,32	Binary-crossentropy	Sigmoid	1
Multi-Label Classification	128,64, 32,32	Binary-crossentropy	Sigmoid	Nombre de classes

3.8 Résultats et Discussion

Nous avons implémenté un modèle d'apprentissage en profondeur DNN avec quatre différentes approches. Cette section présente les résultats des tests réalisés pour la configuration des modèles. Plusieurs tests ont été effectués pour obtenir les bons hyperparamètres pour chaque modèle. Ces paramètres comprennent, par exemple, le nombre de couches, le nombre de nœuds dans chaque couche, le nombre d'itérations (epochs), type d'optimiseur et la fonction d'activation pour les couches cachées. Nous avons ensuite entraîné chaque modèle sur l'ensemble d'entraînement tout en le validant sur l'ensemble de validation. Nous avons répété ce processus avec diverses valeurs de paramètres jusqu'à ce que nous ayons trouvé les meilleurs résultats. Nous avons enfin confirmé les performances de notre modèle en le testant sur l'ensemble de test. Le modèle qui présente la plus grande "accuracy" et le plus faible taux d'erreur (loss) est considéré comme le meilleur modèle.

3.8.1 Classification "multi-classe"

3.8.1.1 Label Encoding

La figure 3.9 montre le "Loss" et l'"Accuracy" du modèle en fonction des itérations au cours du processus d'apprentissage.

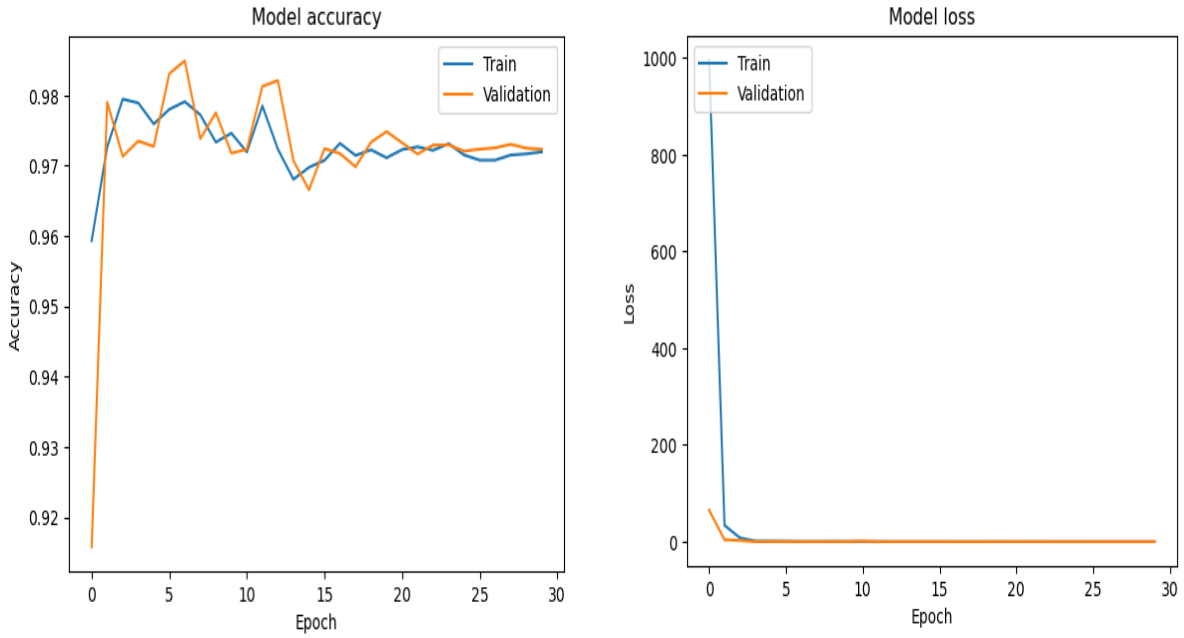

Figure 3.9 : Accuracy et Loss (4 classes).

Cette approche a produit des résultats satisfaisants avec une accuracy de 97.18% pour les données d'entrainement et de 97.24% pour les données de validation.

Le modèle entrainé avec l'approche "Label Encoding" a un "Loss" plus élevé de 9.04% pour les données d'entrainement et de 8.88% pour les données de validation.

3.8.1.2 One-Hot Encoding

La figure 3.10 montre le "Loss" et l'"Accuracy" du modèle en fonction des itérations au cours du processus d'apprentissage.

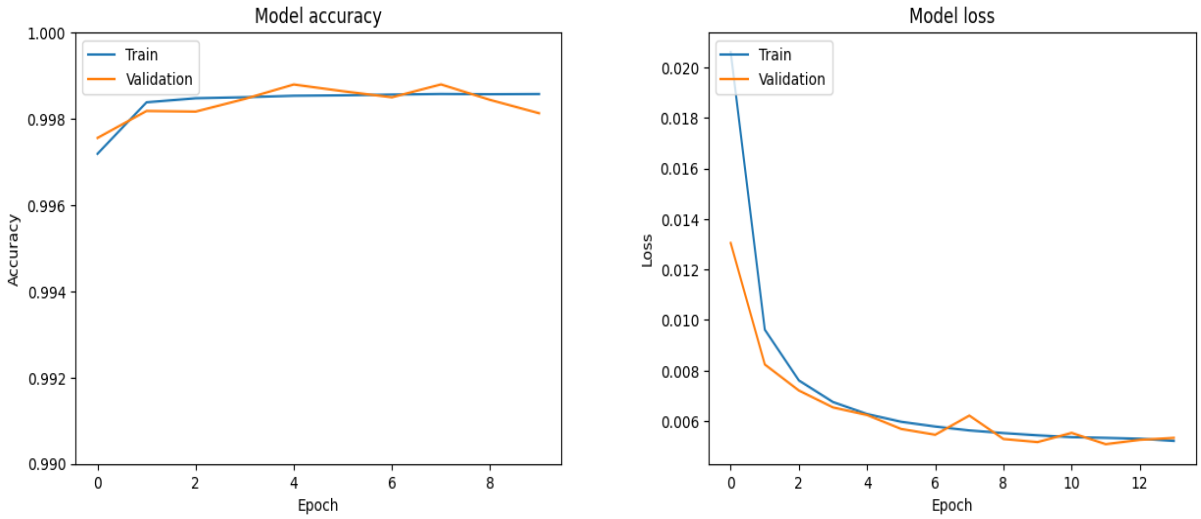

Figure 3.10 : Accuracy et Loss (4 classes).

Cette approche a produit des résultats très satisfaisants avec une "Accuracy" de 99.84% pour les données d'entrainement et de 99.85% pour les données de validation.

Les résultats de cette approche montrent également une fiable valeur de "Loss" pour les données de test et de validation, avec une perte de 0.53% pour les données d'entrainement et de 0.52% pour les données de validation.

Le tableau 3.11 affiche les métriques "Accuracy", "Precision", "Recall" et "F1-score" pour les 4 classes Benign, DDoS attacks-LOIC-HTTP, DDOS attacks-HOIC et DDOS attacks-LOIC-UDP dans le cas d'une classification multi-classe avec le "One Hot Encoding". La matrice de confusion correspondante est donnée par la figure 3.11.

Tableau 3.11 : Rapport de classification one-hot encoding (multi-classe).

Label	Accuracy	Precision	Recall	F1-score
Benign	99.90%	100%	100%	100%
DDoS attacks-LOIC-HTTP	99.50%	99%	100%	100%
DDOS attack-HOIC	99.99%	100%	100%	100%
DDOS attack-LOIC- UDP	98.28%	74%	98%	85%

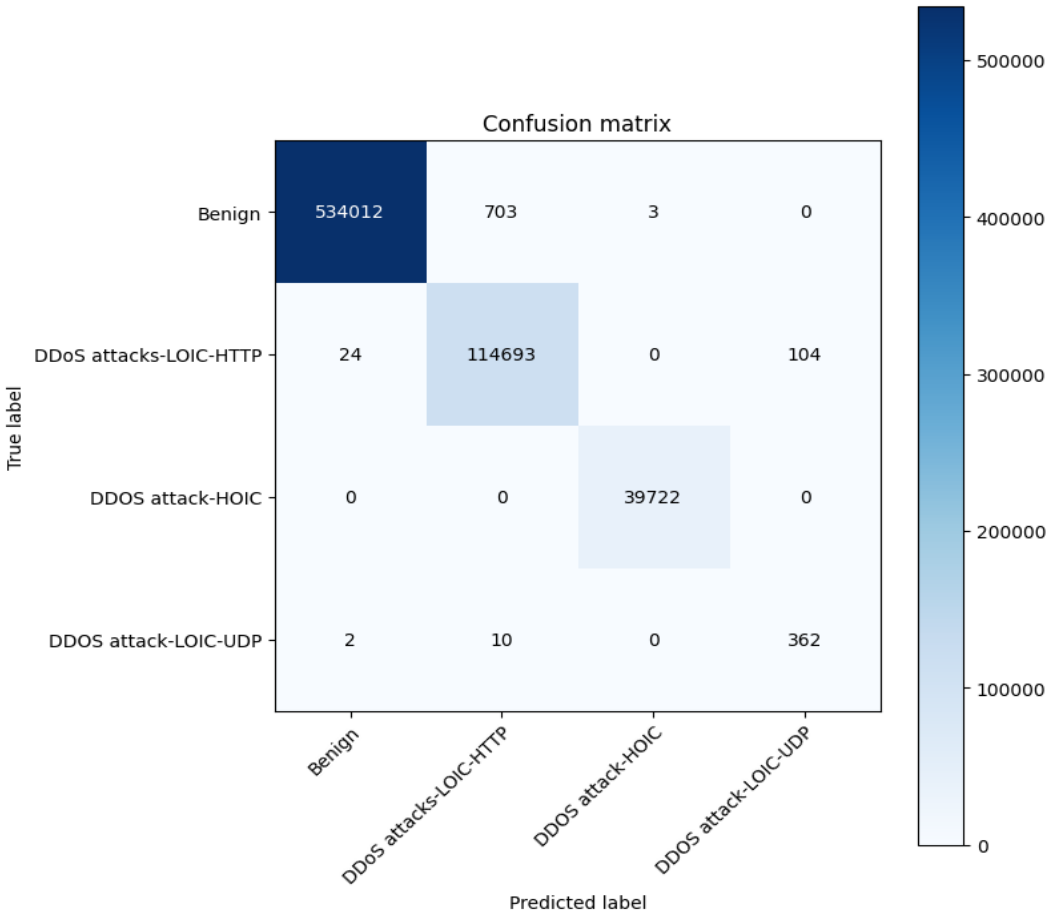

Figure 3.11 : Matrice de confusion pour la classification multi-classe (one-hot encoding).

3.8.2 Classification binaire

La figure 3.12 montre le "Loss" et l'"Accuracy" du modèle en fonction des itérations au cours du processus d'apprentissage.

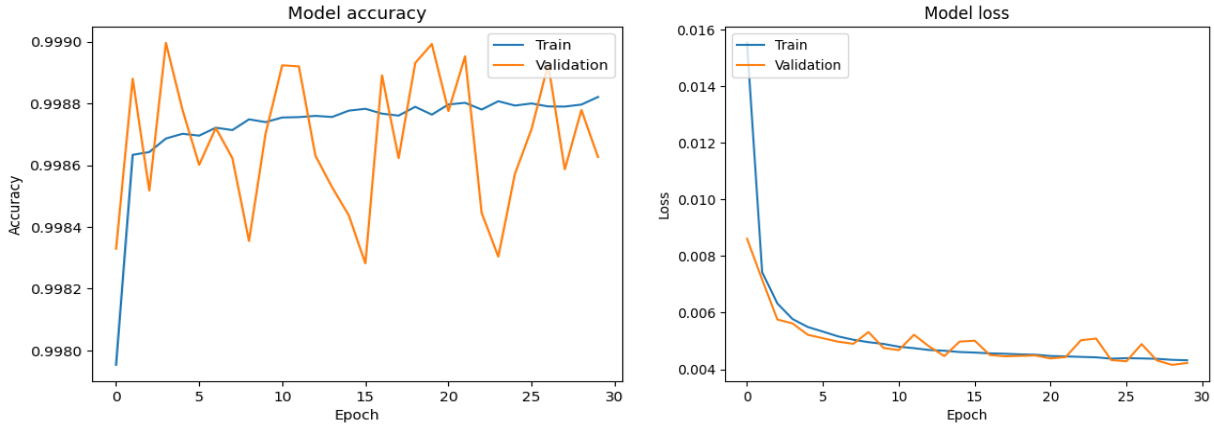

Figure 3.12 : Accuracy et Loss (2 classes).

Cette approche a également produit des résultats très satisfaisants avec une précision de 99.87% pour les données d'entrainement et de 99.86% pour les données de validation.

Cette méthode a également produit des résultats satisfaisants pour la perte, avec une perte de 0.39% pour les données d'entrainement et de 0.42% pour les données de validation.

Le tableau 3.12 affiche les métriques "Accuracy", "Precision", "Recall" et "F1-score" pour les 2 classes Benign, DDoS dans le cas d'une classification binaire. La matrice de confusion correspondante est donnée par la figure 3.13.

Tableau 3.12 : Rapport de classification binaire.

Label	Accuracy	Precision	Recall	F1-score
Benign	100%	100%	100%	100%
DDoS	100%	100%	100%	100%

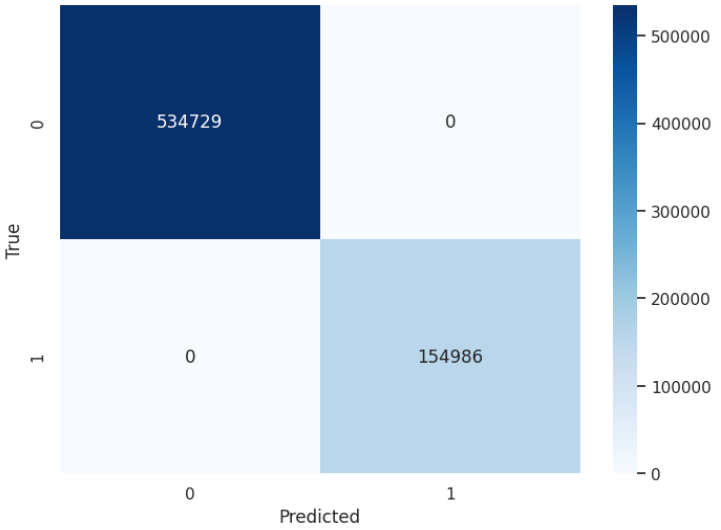

Figure 3.13 : Matrice de confusion pour la classification binaire.

3.8.3 Classification "multi-label"

La figure 3.14 montre le "Loss" et l'"Accuracy" lors de l'ajustement du modèle.

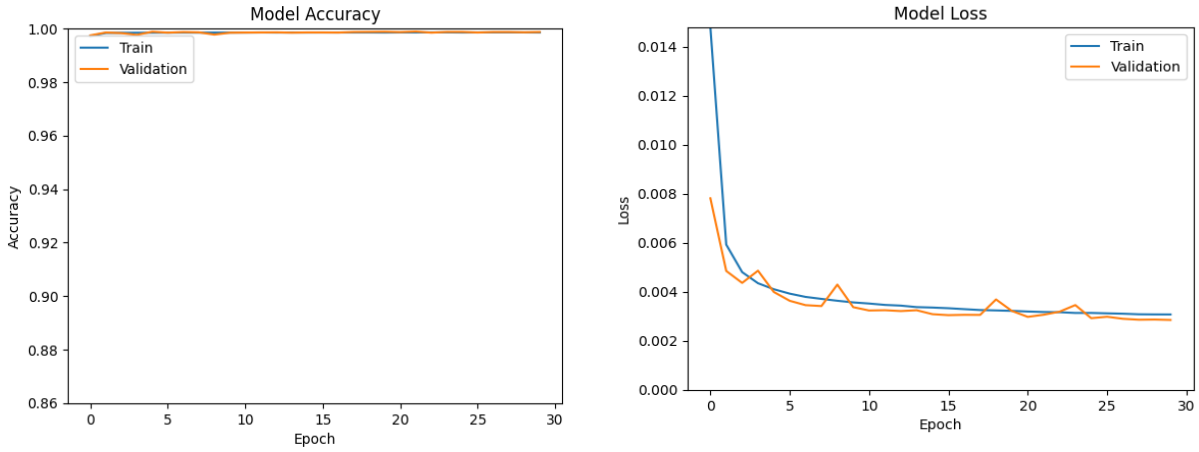

Figure 3.14 : Accuracy et Loss.

Cette approche a également produit des résultats très satisfaisants avec une précision de 99.86% pour les données d'entrainement et de 99.87% pour les données de validation. Cette méthode a transformé les variables catégorielles en un encodage multiple, ce qui permet de mieux représenter les relations complexes entre les différentes catégories.

Pour l'approche multi-label encoding, les valeurs de loss de 0.29% pour les données d'entrainement et 0.28% pour les données de validation peuvent être considérées comme

bonnes car elles sont assez faibles, ce qui indique que le modèle a bien appris à classer les données et qu'il n'y a pas de sur-apprentissage (overfitting).

Le tableau 3.13 affiche les métriques "Accuracy", "Precision", "Recall" et "F1-score" dans le cas d'une classification multi-label. La matrice de confusion correspondante est donnée par la figure 3.15.

Tableau 3.13 : Rapport de classification multi-label.

Label	Accuracy	Precision	Recall	F1-score
Benign	99.89%	100%	100%	100%
DDoS attacks-LOIC-HTTP	99.70%	99%	100%	100%
DDOS attack-HOIC	99.99%	100%	100%	100%
DDOS attack-LOIC- UDP	97.00%	76%	97%	86%

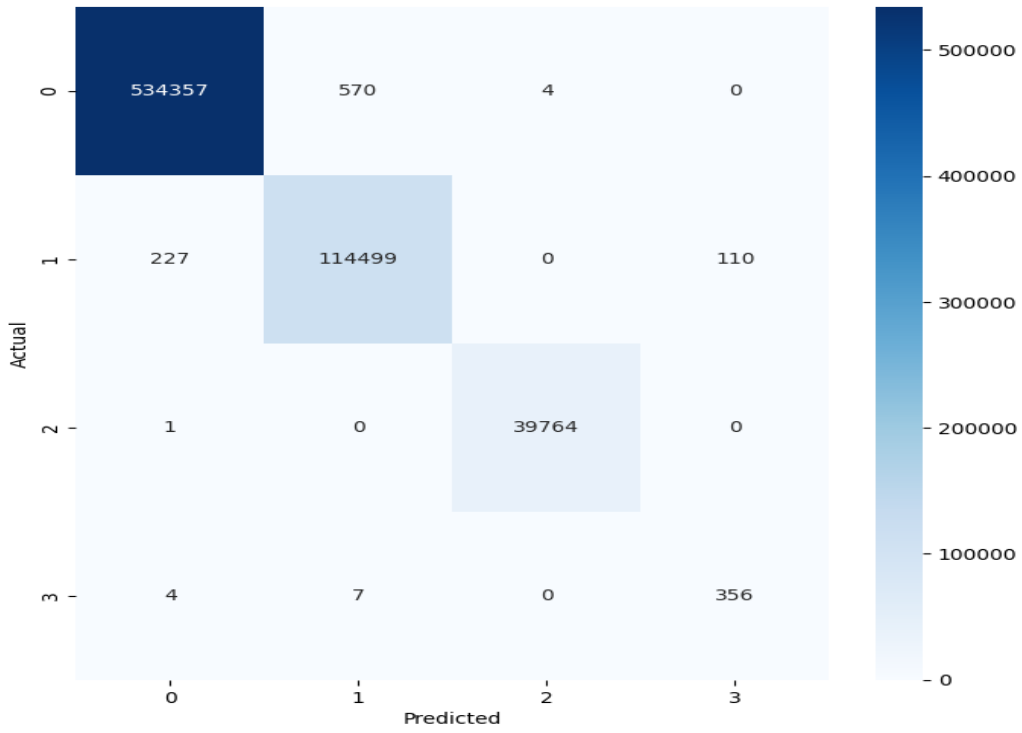

Figure 3.15 : Matrice de confusion pour la classification multi-label.

La figure 3.16 représente les matrices de confusion par chaque classe (Benign, DDoS attacks-LOIC-HTTP, DDOS attack-HOIC, DDOS attack-LOIC- UDP) dans le cas de classification multi-label.

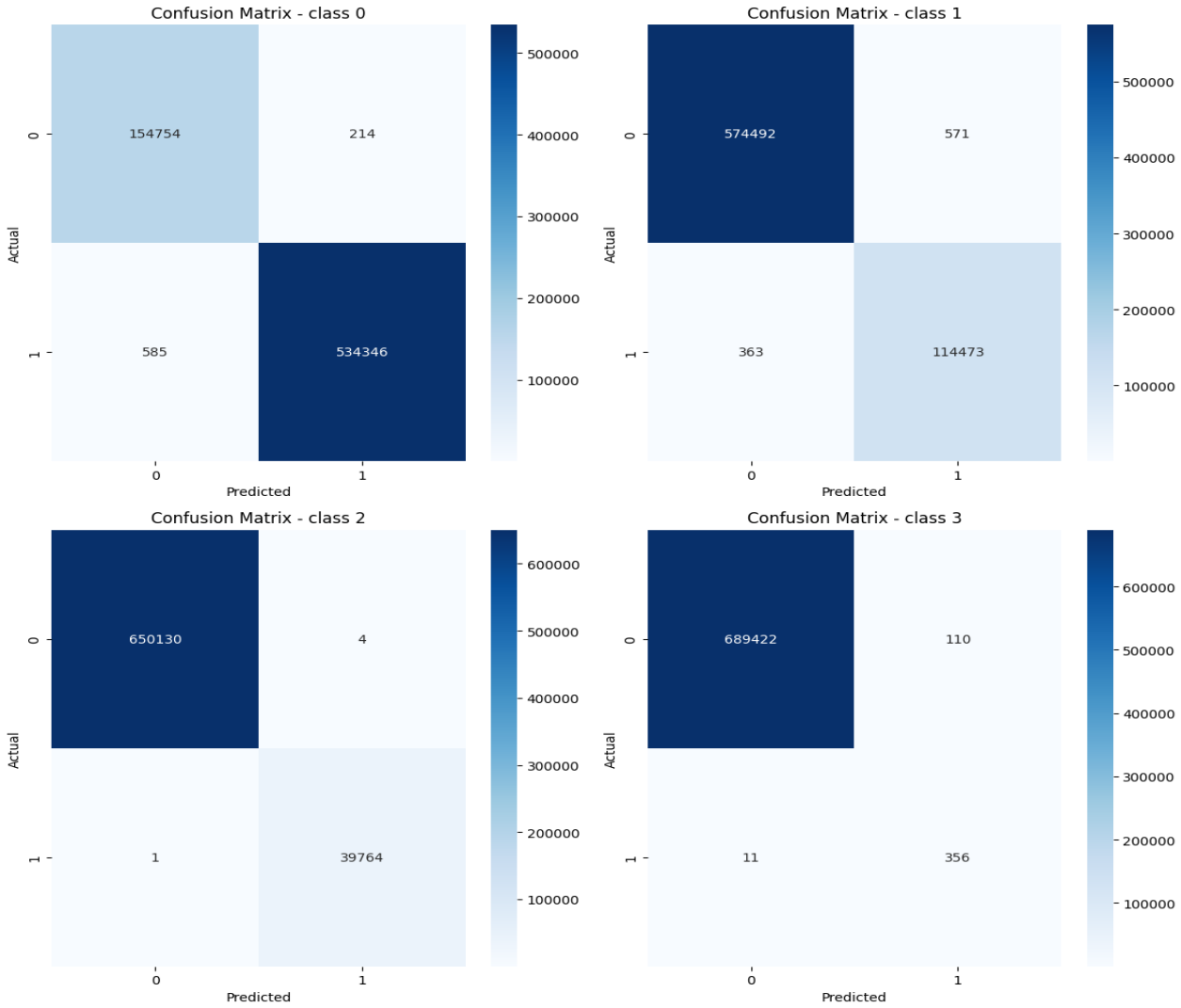

Figure 3.16 : Matrice de confusion pour chaque classe.

3.8.4 Comparaison des résultats

En comparant les résultats des différentes approches, on peut constater que les approches multi-classe, one-hot encoding et classification binaire ont produit des résultats très similaires et très performants en termes d'accuracy et de loss, tandis que l'approche label encoding a montré des performances légèrement moins bonnes en termes d'accuracy et de loss.

L'approche multi-label encoding a produit des résultats impressionnants avec une accuracy parfaite et une très faible valeur de loss. Cependant, il est important de noter que ces résultats peuvent varier en fonction du dataset et des objectifs de l'analyse, et qu'il est donc important de choisir la méthode la plus adaptée.

On peut aussi remarquer que les valeurs de Loss sont très faibles pour toutes les approches, ce qui indique que les modèles ont réussi à minimiser l'erreur de prédiction. Cependant, on peut constater que l'approche de label encoding a produit des valeurs de Loss plus élevées que les autres approches, ce qui confirme que cette méthode de prétraitement n'est pas la plus adaptée pour ce dataset. En revanche, les autres approches ont produit des valeurs de loss très similaires et très faibles, ce qui montre qu'elles sont efficaces pour cette tâche de classification.

En somme, le tableau 3.14 présente une comparaison entre les approches précitées en termes des métriques "Accuracy", "Precision", "Recall" et "F1-score".

Tableau 3.14 : Résultats des approches proposées.

Approches	Accuracy	Precision	Recall	F1-score
Classification binaire	100%	100%	100%	100%
Multi-classe (One-hot Encoding)	99.84%	93%	99%	96%
Classification multi-label	99.86%	94%	99%	96%

Ces résultats indiquent que les performances de classification varient selon l'encodage utilisé et le type de modèle. Dans la classification binaire, les performances sont excellentes avec une précision, un rappel et un score F1 de 100%. En revanche, avec l'encodage "multi-label encoding" et "one-hot encoding", les performances de la précision et du score F1 sont légèrement inférieures, mais le rappel est toujours assez élevé. Les résultats semblent indiquer que le modèle est capable de prédire les classes avec une bonne précision et un bon rappel.

3.9 Conclusion

L'objectif de ce dernier chapitre était de proposer une solution qui détecte les attaques par déni de service distribué (DDoS). Dans ce chapitre, nous avons examiné les différentes

approches pour détecter les attaques DDoS en utilisant diverses techniques d'encodage et de classification. Nous avons analysé et discuté les résultats de chaque méthode. Les approches proposées donnent des résultats très satisfaisants.

Les résultats montrent que les modèles de réseaux neuronaux profonds ont une grande capacité à détecter les attaques DDoS, en particulier lorsqu'ils sont entraînés sur des données réalistes telles que celles fournies par le dataset CSE-CIC-IDS2018-DDoS.

Conclusion Générale

Les attaques DDoS sont devenues une menace majeure pour les entreprises et les organisations (de toutes tailles) du monde entier. Ces attaques peuvent perturber ou même mettre hors service des systèmes en ligne, causant d'importants dommages financiers.

Les techniques de détection traditionnelles ont montré leurs limites en raison de leur manque d'adaptabilité et de leur forte sensibilité aux faux positifs.

A cause de l'augmentation du nombre d'attaques DDoS, des méthodes de sécurité ont été développées et la gestion des accès est devenue très importante. Pour répondre à la problématique posée dans l'introduction générale, nous avons présenté dans ce mémoire l'avantage d'utiliser le Deep Learning assurant la protection des ressources du réseau contre les attaques DDoS.

Le Deep Learning est une technique prometteuse pour la détection des attaques DDoS. Les modèles de détection intelligente peuvent apprendre à identifier des schémas dans le trafic réseau qui indiquent des attaques DDoS, même lorsque ces schémas sont difficiles à détecter à l'aide de méthodes traditionnelles. Cela fait du Deep Learning un outil puissant pour les organisations qui cherchent à se protéger des attaques DDoS.

Les algorithmes de Deep Learning offrent une approche prometteuse pour détecter et prévenir les attaques DDoS, grâce à leur capacité à apprendre à partir de données et à prendre des décisions autonomes.

Le travail présenté dans ce mémoire porte sur la conception et le développement d'un modèle utilisant des techniques de Deep Learning pour la détection et la classification d'un large éventail d'attaques DDoS.

Dans le cadre de ce travail, nous avons choisi le dataset CSE-CIC-IDS2018. Ce dataset est composé de dix fichiers dont deux parmi eux contiennent les attaques DDoS. Ces deux fichiers ont été fusionnés pour entraîner et évaluer notre modèle. Sachant que la performance de notre modèle dépend entièrement de la qualité de ces données, nous avons appliqué un prétraitement des données telles que l'élimination des valeurs NAN, des valeurs infini, des lignes dupliquées et les colonnes inutiles qui pourraient affecter négativement la performance du modèle.

Par ailleurs, nous avons constaté que la structure et la qualité des données utilisées pour l'entraînement des modèles jouaient un rôle crucial dans les performances obtenues. Nous avons également étudié les effets de différents hyper paramètres sur les performances des modèles d'apprentissage profond, comme le nombre de couches cachées, le nombre de neurones dans chaque couche cachée, les fonctions d'activation, l'optimiseur, le type et la valeur de régularisation, le nombre d'itération, le taux d'apprentissage, etc. qui ont un impact significatif sur la performance des modèles.

Dans notre étude, nous avons adopté différentes approches pour détecter les attaques DDoS. Tout d'abord, nous avons utilisé une classification binaire qui permet de distinguer entre le trafic normal et le trafic malveillant. Ensuite, nous avons mis en place une classification multi-classe qui permet d'identifier plusieurs types d'attaques DDoS spécifiques. Enfin, nous avons utilisé une classification multi-label capable de détecter simultanément plusieurs attaques DDoS pouvant se produire en même temps. Ces différentes approches nous ont permis d'évaluer la performance de chaque modèle pour la détection des attaques DDoS.

Les résultats ont montré que la classification binaire a atteint la plus grande Accuracy, suivie par la classification multi-label et ensuite la classification multi-classe.

En perspectives, nous envisageons la détection des attaques DDoS dans les réseaux véhiculaires car ces attaques sont plus graves dans le réseau ad hoc véhiculaire (VANET) en raison de la diffusion de cette attaque qui se propage sur une large zone du réseau.

Bibliographie

[1] N. Kumar et S. Sharma, «Study of Intrusion Detection System for DDoS Attacks in CloudComputing,» 2013.

[2] V. K. Yadav, M. Trivedi et B. Mehtre, «DDA: An approach to handle DDoS (ping flood) attack,» 2015.

[3] D. K. Bhattacharyya et J. K. Kalita, DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance, 2016.

[4] k. Kim, M. Aminanto et H. Tanuwidjaja, Network intrusion detection using deep learning: a feature learning approach, 2018.

[5] L. Liang, K. Zheng et Q. Sheng, «A denial of service attack method for an iot system,» 2016.

[6] «Denial-of-Service (DDoS) Attack: Examples and Common Targets,» [En ligne]. Available: https://www.investopedia.com. [Accès le 30-3-2023].

[7] J. Mirkovic et P. Reiher, «A taxonomy of DDoS attack and DDoS defense mechanisms,» May 2004.

[8] M. Suresh et R. Anitha, «Evaluating Machine Learning Algorithms for Detecting DDoS Attacks,» 2011.

[9] R. Swami, M. Dave et V. Ranga, «Software-defined Networking-based DDoS Defense Mechanisms,» 2019.

[10] «What is a Reflection/Amplification DDoS Attack,» [En ligne]. Available: https://www.csoonline.com/article/3629476/what-is-a-reflection-amplification-ddos-attack.html. [Accès le 02-04-2023].

[11] M. Masdari et M. Jalali, «A survey and taxonomy of DoS attacks in cloud computing,» 2016.

[12] «Denial of Service DDoS attack,» [En ligne].Available:https://www.geeksforgeeks.org/denial-of-service-ddos-attack.[Accès le 04-04-2023].

[13] R. Tandon, «A survey of distributed denial of service attacks and defenses,» 2020.

[14] A. Praseed et P. Thilagam, «DDoS attacks at the application layer: Challenges and research perspectives for safeguarding web applications,» 2018.

[15] C. Douligeris et A. Mitrokotsa, «DDoS attacks and defense mechanisms: classification and state-of-the-art,» 2004.

[16] Y. Xiao et M. Lee, «MIMO multiuser detection for CDMA systems,» February 2006.

[17] D. Nagamalai, E. Renault et M. Dhanuskodi, «Advances in Parallel, Distributed Computing,» 2011.

[18] Rajkumar et M. Nene, «A Survey on Latest DoS Attacks: Classification and Defense Mechanisms,» 2013.

[19] S. Zargar, J. Joshi et D. Tipper, «A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks,» 2013.

[20] D. Kshirsagar, . S. Sawant, A. Rathod et S. Wathore, «CPU load analysis & minimization for TCP SYN flood detection,» 2016.

[21] Z. He, . T. Zhang et R. Lee, «Machine learning based DDoS attack detection from source side in cloud,» 2017.

[22] . P. Kamboj, M. Trivedi, V. Yadav et V. Singh, «Detection techniques of DDoS attacks: A survey,» 2017.

[23] A. Soliman, C. Salama et H. Mohamed, «Detecting DNS Reflection Amplification DDoS Attack Originating from the Cloud,» 2018.

[24] N. Hoque , D. Bhattacharyya et J. Kalita, «Botnet in DDoS Attacks: Trends and Challenges,» 2015.

[25] X. Geng et A. Whinston, «Defeating Distributed Denial of Service attacks,» pp. 36-42, 2002.

[26] A. Srivastava, B. Gupta, A. Tyagi et A. Sharma, «A Recent Survey on DDoS Attacks and Defense Mechanisms,» 2011.

[27] B. Gupta, R. Joshi et M. Misra, «Defending against Distributed Denial of Service Attacks: Issues and Challenges,» 24 11 2009.

[28] M. Shurman, R. Khrais et A. Yateem, «DoS and DDoS Attack Detection Using Deep Learning and IDS,» 2020.

[29] S. Agatonovic-Kustrin et R. Beresford, «Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research,» 2000.

[30] B. Mahesh, «Machine Learning Algorithms - A Review,» 2020.

[31] A. Géron, Machine Learning avec Scikit-Learn, 2017.

[32] C. Ebert et . P. Louridas, Machine learning, 2016, p. 110–115.

[33] V. Martineau, «Utilisation de automatique en remplacement des simulateurs de débitage de billots de bois,» 2022.

[34] F. Chollet, Deep Learning with Python FRANÇOIS CHOLLET, 2017.

[35] A. Fischer, «Deux méthodes d'apprentissage non supervisé :synthése sur la méthode des centres mobiles et présentation des courbes principales,» 2014.

[36] D. Filliat, «Robotique mobile,» 2011.

[37] Y. LeCun, Y. Bengio et G. Hinton, «Deep Learning,» May 2015.

[38] L. Zhang, S. Wang et B. Liu , «Deep Learning for Sentiment Analysis: A Survey,» 2017.

[39] «Introduction à l'apprentissage en profondeur,». [En ligne]. Available: https://www.geeksforgeeks.org/introduction-deep-learning/.[Accès le 16-04-2023]

[40] L. Deng et D. Yu, «Deep Learning: Methods and applications,» Jun 2014.

[41] W. McCulloch et W. Pitts, «A logical calculus of the ideas immanent in nervous activity,» 1943.

[42] F. Rosenblatt, «The perceptron: a probabilistic model for information storage and organization in the brain,» 1958.

[43] M. Minsky et A. SEYMOUR, Papert Perceptrons: An Introduction to Computational Geometry, 1969.

[44] D. Ackley, G. Hinton et T. Sejnowski, «A learning algorithm for Boltzmann machines,» 1985.

1[45] Y. LeCun, L. Bottou, Y. Bengio et P. Haffner, Haffner Gradient-based learning applied to document recognition Proc, 1998.

[46] H. Liu et B. Lang, «Machine learning and deep learning methods for intrusion detection systems: A survey,» 2019.

[47] C.-A. Azencott , Introduction au Machine Learning, 2019.

[48] «Keras API reference,» [En ligne]. Available: https://keras.io/api/.[Accès le 10-05-2023]

[49] M. Moocarme, M. Abdolahnejad et B. Ritesh , The Deep Learning With Keras Workshop, 2020.

[50] H. Saleh, The Machine Learning Workshop_ Get ready to develop your own high-performance machine learning algorithms with scikit-learn, 2020.

[51] M. Gardner et S. Dorling, «Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences,» 1998.

[52] D. Hanbay, I. Turkoglu et Y. Demir, «Prediction of wastewater treatment plant performance based on wavelet packet decomposition and neural networks,» 2008.

[53] Un guide complet du réseau neuronal artificiel dans l'apprentissage automatique. [En ligne]. Available: https://fre.myservername.com/complete-guide-artificial-neural-network-machine-learning. [Accès le 22-03-2023]

[54] A. Sadrmomtazi, Sobhani, J. Sobhani et M. Mirgozar, «Modelling compressive strength of EPS lightweight concrete using regression, neural network and ANFIS,» 2013.

[55] C. Bilim, C. Atis, H. Tanyildizi et O. Karahan, «Predicting the compressive strength of ground granulated blast furnace slag concrete using artificial neural network,» 2009.

[56] H. Yaprak, A. Karacı et I. Demir, «Prediction of the effect of varying cure conditions and w/c ratio on the compressive strength of concrete using artificial neural networks,» 2013.

[57] H. Yoo , «Deep convolution neural networks in computer vision: a review,» 2015.

[58] Convolutional neural network (CNN). [En ligne]. Available: https://www.techtarget.com/searchenterpriserai/definition/convolutional-neural-network. [Accès le 22 03 2023].

[59] A. Graves, A. Mohamed et G. Hinton, «Speech recognition with deep recurrent neural networks,» 26–31 May 2013.

[60] I. Sutskever, O. Vinyals et Q. Le, «Sequence to sequence learning with neural networks,» 2014.

[61] S. Hochreiter et . J. Schmidhuber, «Long short-term memory,» 1997.

[62] J. Chung, C. Gulcehre, K. Cho et Y. Bengio , «Empirical evaluation of gated recurrent neural networks on sequence modeling,» 2014.

[63] M. Schuster et K. Paliwal, «Bidirectional recurrent neural networks,» 1997.

[64] S. Tiwari, «Activation functions in Neural Networks,». [En ligne]. Available: https://www.geeksforgeeks.org/activation-functions-neural-networks/. [Accès le 24-03-2023].

[65] S. Sharma et A. Athaiya, «Activation functions in neural networks,» 2017.

[66] A. Santra et C. Josephine Christy, «Genetic algorithm and confusion matrix for document clustering,» 2012.

[67] V. Kanimozhi et T. Jacob, «Artificial Intelligence based Network Intrusion Detection with Hyper-Parameter Optimization Tuning on the Realistic Cyber Dataset CSE-CIC-IDS2018 using Cloud Computing,» 2019.

[68] Python. [En ligne]. Available: https://www.futura-sciences.com/tech/definitions/informatique-python-19349/. [Accès le 23-04-2023].

[69] «Difference between TensorFlow and Keras,» [En ligne]. Available: https://www.geeksforgeeks.org/difference-between-tensorflow-and-keras/?ref=gcse.[Accès le 23-04-2023].

[70] O. Kramer, Machine Learning for Evolution Strategies, 2016.

[71] «Matplotlib,». [En ligne]. Available: https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501867-matplotlib/. [Accès le 23-04-2023].

[72] «google colab,». [En ligne]. Available: https://ledatascientist.com/google-colab-le-guide-ultime/. [Accès le 23-04-2023].

[73] «Anaconda,». [En ligne]. Available: https://blog.hubspot.com/website/anaconda-python. [Accès le 23-04-2023].

[74] «A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018),». [En ligne]. Available: https://registry.opendata.aws/cse-cic-ids2018. [Accès le 05-05-2023].

[75] «CSE-CIC-IDS2018 on AWS,». [En ligne]. Available: https://www.unb.ca/cic/datasets/ids-2018.html. [Accès le 05-05-2023].