REPUBLIQUE DU SENEGAL

Un Peuple – Un But – Une Foi

UNIVERSITE GASTON BERGER DE SAINT-LOUIS

U.F.R DE SCIENCES APPLIQUEES ET DE TECHNOLOGIE

CENTRE DE FORMATION PROFESSIONNELLE PLURIDISCIPLINAIRE

(CFPP)

MEMOIRE DE FIN D' ETUDE POUR L' OBTENTION DU MASTER EN RESEAUX TELECOMMUNICATION

(MART)

Sujet : Classification des données du trafic réseau pour la détection d'anomalie dans l'IoT : cas d'un smart home

PRÉSENTÉ ET SOUTENU PAR:

Marième BA

Sous la direction de:

Pr. Chérif DIALLO

Année Académique: 2020 / 2021

REMERCIEMENTS

Il convient de commencer ce travail qui atteste de deux années d'étude en Master par des remerciements adressés à ceux qui m'ont encadrée, soutenu et exhortée pendant ce cycle.

Je tiens à remercier et à féliciter tous mes chers professeurs pour leur savoir, leur compétence, leur accompagnement, leur endurance… tout au long de ces deux merveilleuses années.

J'exprime particulièrement ma gratitude à mon professeur et encadreur Dr Cherif Diallo, qui a bien voulu diriger la rédaction du présent mémoire.

Je ne saurai ne pas remercier l'UFR SAT dans son ensemble et plus particulièrement le CFPP pour nous avoir offert ce cadre d'apprentissage inestimable et pour le soutien et l'accompagnement riche de savoir qu'elle a su entretenir et consolider durant ces années.

Je remercie à tous ceux qui ont participé de près ou de loin à ma réussite que ce soit amis et anciens.

J'adresse toute ma reconnaissance aux membres du jury pour avoir accepté d'évaluer mon travail de mémoire de fin d'étude.

Je remercie enfin et encore une fois mes parents et ma famille sans qui rien de tout cela ne serait possible

DÉDICACES

Je rends tout d'abord Grâce à l'Omniscient pour nous avoir ouvert les portes de l'esprit afin d'achever et de mettre à la disposition du lecteur : ce document.

A mes parents qui m'ont soutenu d'arrache-pied depuis le berceau. Je ne saurai assez être reconnaissante.

A ma famille qui ne cesse de prier pour mon humble personne. Que le digne de Louange accepte leurs prières. Amine !

A mes camarades de promotion pour leur soutien, leur réconfort, le vécu partagé, l'amitié, l'union et la sympathie dont ils ont fait preuve tout au long de notre cursus.

SIGLES ET ABRÉVIATIONS

AE : Auto-encodeurs

API : Application Programming Interface

BP : Back-Propagation

CPU : Central Processing Unit

CNN : Convolutional Neural Network

DoS : Denis of Service

DDoS : Distributed Denis of Service

DNS : Domain Name System

FAI : F FP : False Positive

FN : False Negative

FAI : Fournisseur d'Accès à Internet

GPU : Graphics Processing Unit

GRU : Gated Recurrent Unit

GAN : Génératives Adversarial Network

GCRNN : Group-Constrained Convolutional Recurrent Neural

HIDS : Host Based Intrusion Detection System

IoT : Internet of Things

IBM : International Business Machines Corporation

ISO /IEC : International Organization
for Standardization/International Electrotechnical Commission

IETF : Internet Engineering Task Force

IP : Internet Protocol

IEEE : Institute of Electrical and Electronics Engineers

IA : Intelligence Artficielle

IPTV : Internet Protocol Television

IDS : Intrusion Detection System

KNN : K-Nearest Neighbors

LDC : Liquid Crystal Display

MIT : Massachusetts Institute of Technology

MAC : Media Access Control

ML : Machine Learning

MLP : Multi Layer Perceptron

NoSQL : Not Only Structured Query Language

Nmap : Network Mapper

NIDS : Network Based Intrusion Detection System

NIPS : Network Intrusion Prevention System

ONEIROS : Open-ended Neuro-Electronic Intelligent Robot Operating System

PCA : Principal Component Analysis

QoS : Quality of Service

RFID: Radio Frequency Identification

R2L : Root to Local

RNA : Réseaux de Neurones Artificiels

RNN : Recurrent Neural Network

ReLu : Rectified Linear Unit

SVM : Support Vector Machine

TIC : Technologie de l'Information et de la Communication

TCP : Transmission Control Protocol

TP : True Positive

TN : True Negative

Tanh : Tangente hyperbolique

U2R : User to Root

UIT : Unité International des Telecommunications

WSAN : Wireless Sensor and Actuator networks

XSS : Cross-site scripting

LISTE DES FIGURES

Figure 1: Modèle architecture IoT à 3 couches (source) **6**
Figure 2: Architecture basée sur le cloud computing et le fog computing (source) **7**
Figure 3: Schéma d'un système IoT (source) **9**
Figure 4: Les differents domaines d'application IoT (source) **11**
Figure 5: Les branches de l'IA (source) **24**
Figure 6: Cycle d'un algorithme du Machine Learning **25**
Figure 7: Schéma d'hiérarchisation des modèles du Machine Learning **27**
Figure 8: Le neurone biologique (source) **29**
Figure 9: Modèle neurone formel de Warren et Walter **29**
Figure 10: Schéma illustrant la correspondance d'un neurone biologique et un neurone formel (source) **30**
Figure 11: Schéma d'un perceptron simple (source) **31**
Figure 12: Structure d'un réseau de neurones multicouche (source) **33**
Figure 13: Architecture d'un réseau de neurones convolutifs (source) **35**
Figure 14: Schéma spécifiant les trois couches du CNN **36**
Figure 15: Parcours de la filtre de convolution **37**
Figure 16: Exemple de moyen et Max Pooling avec un filtre 2 x 2 en pas de 2 (source) **37**
Figure 17: Structure d'un RNN simple (source) **39**
Figure 18: Structure d'un RNN simple (source) **40**
Figure 19: Les équations d'un réseau LSTM **41**
Figure 20: Représentation structural d'un LSTM (source) **42**
Figure 21: Les équations d'un modèle GRU **43**
Figure 22: Représentation d'un GRU (source) **43**
Figure 23: Architecture d'un auto-encodeur (source) **44**
Figure 24: Architecture d'un auto-encodeur **45**
Figure 25: Schéma représentatif d'un modèle GAN (source) **47**
Figure 26: Classification des techniques de détection d'anomalie (source) **50**
Figure 27: Téléchargement d'anaconda **54**
Figure 28: Page obtenue à l'ouverture Anaconda Navigator **55**
Figure 29: L'environnement de base anaconda **56**
Figure 30: Création de l'environnement Tensorflow **56**
Figure 31: Creation de L'environnement Tensorflow **57**
Figure 32: Ajout de l'API keras (version 2.6.0) **58**
Figure 33: Taxonomie des attaques **59**
Figure 34: Résultats d'analyse de la totalité des données **64**
Figure 35: Distribution du flux par rapport à la transmission des flag PSH **65**
Figure 36: Distribution par rappport auu nombre de champs PSH **65**
Figure 37: Distribution par rapport aux ports sources **65**
Figure 38: Distribution par rapport aux ports de destinations **65**
Figure 39: Distribution par rapport à la longueur moyenne des paquets transmis **66**
Figure 40: Distribution par rapport à la longueur moyenne des paquets **66**
Figure 41: Résultat de la corrélation entre les caractéristiques **66**
Figure 42: Résultats de la modélisation avec Decision-Tree **70**
Figure 43: Histogramme d'évaluation des caractéristiques importantes utilisées par Decision-Tree **71**
Figure 44: Résultats de la modélisation avec Random-Forest **72**
Figure 45: Résultats de la modélisation avec AdaBoost **73**
Figure 46: Résumé de la composition du modèle de réseau de neurones **74**
Figure 47: Performance du réseau de neurone par rapport aux données d'entrainement et de validation **74**

LISTE DES TABLEAUX

Tableau 1: Description des caractéristiques du trafic réseau de l'ensemble des données 63
Tableau 2: Dataset après extraction des caractéristiques 67
Tableau 3: Distribution de la sortie par rapport aux données d'entrainement et de test. 67
Tableau 4: Résultats de la modélisation avec Random-Forest 71
Tableau 5: Résultats de la modélisation avec Adaboost 72

Table des matières

REMERCIEMENTS ii
DÉDICACES iii
SIGLES ET ABRÉVIATIONS iv
LISTE DES FIGURES vi
LISTE DES TABLEAUX vii
Résumé 1
Introduction générale 2
Chapitre 1 : Généralités sur l'IoT 4
1 Définition de l' IoT 4
2 Architecture et composantes 5
2.1 Architecture 5
2.2 Composants d'une solution IoT 8
3 Caractéristiques des données dans IoT 9
4 Les applications de l'IoT 11
5 Enjeux de sécurité dans IoT 14
5.1 Les menaces dans IoT 14
5.1.1 Les menaces physiques 14
5.1.2 Les cybermenaces 14
5.2 Quelques attaques dans IoT 15
5.2.1 Dénis de service : DoS / DDoS 15
5.2.2 Mirai 15
5.2.3 R2L 16
5.2.4 U2R 16
5.2.5 Man-in-the-middle (Homme du milieu) 16
5.2.6 Attaque par injection 16
5.2.7 Attaque par mot de passe 17
5.3 Les solutions de défenses associées à un système IoT 17
5.3.1 Les solutions classiques 17
5.3.2 Les systèmes de détection d'intrusion 18
6 Conclusion 20
Chapitre 2 : Classification des données 22

1 Définition 22
2 L'Intelligence Artificielle 22
2.1 Définition de l'IA 23
2.2 Branches de l'IA 24
2.3 Machine Learning appliqué à la classification 24
2.3.1 Définition du Machine Learning 25
2.3.2 Les techniques d'apprentissage 26
2.3.3 Les algorithmes du ML 26
2.4 Les réseaux de neurones appliqués à la classification 28
2.4.1 Neurone Artificiel ou formel 28
2.4.2 Le perceptron simple 30
2.4.3 Les réseaux de neurones multicouches 32
2.5 Le Deep Learning appliqué à la classification 33
2.5.1 Définition 33
2.5.2 Les modèles du Deep Learning 34
3 Conclusion 48
Chapitre 3 : Etude ce cas, détection d'anomalies dans le trafic réseau d'un réseau d'un Smart Home 49
1 Introduction 49
2 Présentation des outils utilisés 51
2.1 Python 51
2.2 Anaconda 51
2.2.1 Jupyter Notebook 52
2.2.2 Environnement de développement 52
2.2.3 Les librairies 53
3 Installation des outils 54
4 Présentation du Dataset 58
5 Classification pour la détection d'anomalie 60
5.1 Extraction des caractéristiques pour les tests avec les algorithmes du Machine Learning 60
5.1.1 Compréhension des données 60
5.1.2 Préparation des données 67
5.2 Evaluation des Résultats 68

5.2.1 Les critères d'évaluation 68
5.2.2 Visualisation des résultats 69
6 Conclusion 75
Conclusion générale 76
REFERENCES 78

Résumé

Ce travail porte sur la detection d'anomalie de trafic dans un réseau IoT dans le cas d'un smart home. L'objectif est de mettre en place un modèle de classification basé sur l'IA pour détecter dans un réseau si un trafic est normal ou anomal. Pour atteindre ces objectifs, nous avons adopter une démarche qu'on a presenté autour de trois chapitres.

Dans le chapitre premier nous avons fait présentation sur les généralités sur l'IoT en détaillant d'abord ses éléments techniques tels que ses architectures et composants. Ensuite, de présenter les caractéristiques de données et les différents domaines d'applications. Il a été question aussi dans ce chapitre d'analyser et de discuter de la sécurité dans les réseaux IoT. Nous avons de ce fait présenté les enjeux de sécurités dans l'IoT, les menaces, les différentes attaques et les solutions proposées pour lutter contre ces différents types de menaces. Ce chapitre nous a permis de mettre le contexte de notre étude afin de mieux aborder le problème posé.

Dans le deuxième chapitre, nous avons fait une étude des différentes méthodes de classification de données qu'offre l'IA. Pour cela, nous avons d'abord donné quelques éléments introductifs sur l'IA : définitions, branches, etc. Ensuite, nous avons fait une discussion sur les modèles de classification IA par rapport à ses différents sous branches : Machine Learning, réseau de neurones, et Deep Learning. Cette discussion nous a permis de mieux comprendre les modèles les plus adaptés par rapport à notre problème et d'avoir leurs critères d'applicabilité.

Dans le dernier chapitre, nous avons fait une implémentation de modèles de classification pour la détection d'anomalie dans les réseaux de smart home. Nous avons testé et comparé trois performants modèles : Random Forest, AdaBoost et un modèle de réseau de neurone. L'AdaBoost offre le meilleur résultat avec 97% de précision dans les prédictions tout en se comportant très bien avec des données nouvelles.

En somme, nous avons mis en place un modèle IA appris à partir de données réelles qui nous permet de détecter avec une très bonne performance les trafics anormales.

Introduction générale

Depuis des années, l'évolution de l'internet a révolutionné l'interaction des êtres vivants avec son environnement permettant la communication avec ou entre des objets grâce à la présence de petits dispositifs appelés capteurs. Ces derniers pouvant être insérer sur ces objets sont capables de transformer une grandeur physique observée en données compréhensibles et utilisables. Cette évolution de l'internet plus vaste et hétérogène est connue sous le nom d'Internet des Objets (IdO), Internet of Things (IoT) en anglais. Au cours de ces derniers années, l'IoT a subit une croissance incroyable due à un nombre considérable d'objets connectés. Ceci a pour conséquence hausse considérable du flux de données dans ce type de réseau d'objets connectés par l'intermédiaire de des capteurs. Ces données contiennent des informations utiles et précieuses provenant des différents objets du réseau et applicables dans plusieurs domaines de la vie comme l'agriculture, la santé, l'industrie, les villes connectées et notamment la domotique ou le smart home qui nous intéresse particulièrement dans cette études. Cette grande quantité de données a suscité une prolifération des pirates informatiques qui ont mené beaucoup d'attaques sur les réseaux IoT au risque qu'ils deviennent des Botnet IoT (des appareils connectés et infectés par des logiciels malveillants). Ainsi, ces attaques compromettent toute la confidentialité, l'intégrité et la disponibilité des données dans l'IoT. La vulnérabilité de ces réseaux IoT s'explique en partie par les limites que présentent les méthodes classiques de sécurité. Cela remet en cause la sécurité des réseaux IoT d'où l'intérêt de faire appel à un système intelligent de protection de données et de détection de trafics malveillants. En effet, l'utilisation d'outils comme ceux l'Intelligence artificielle (IA) pourrait être une solution efficace pour extraire et détecter dans un réseau IoT la présence de trafic qui présente d'anomalies issues d'attaques ou de mauvais fonctionnement du réseau. En plus, l'IA permet d'aller encore plus loin dans la détection d'anomalie avec la puissance des modèles du Deep Learning (DL) qui sont un sous domaine d'IA issue des réseaux de neurones artificiels, qui sont eux même dérivés du Machine Learning (ML).

En partant de ces constats de l'importance de la cybersécurité dans l'IoT, l'objet de notre travail consiste à réaliser un classificateur intelligent de données pour la détection d'anomalies de trafic réseaux dans le domaine du smart home afin de diminuer la saturation du réseau et de renforcer la sécurité.

Pour atteindre nos objectifs, nous allons adopter une démarche que nous voudrions scientifique et rigoureuse. Pour ce faire, nous avons organisé notre travail selon cette démarche autour de trois chapitres. Dans le chapitre premier, nous aborderons les généralités de l'IoT afin de mieux comprendre ces concepts, son fonctionnement et analyser les risques auxquels qu'il est exposé. Puis, dans le deuxième chapitre nous ferons une étude des méthodes de classification de données qu'offre l'Intelligence Artificielle partant des modèles de l'apprentissage automatique, jusqu'à l'apprentissage profond en passant par les réseaux de neurones. Enfin, dans le troisième chapitre nous allons présenter notre apport personnel qui est une proposition de classification intelligente de trafics réseaux pour la détection d'anomalie dans un smart home. Dans cette partie nous présenterons les outils et le Dataset utilisés et le travail de détection d'anomalie par classification en adoptant les modèles du Machine Learning, des réseaux de neurones et du Deep Learning. Nous allons terminer ce chapitre par l'évaluation des performances des différents modèles adoptés en les comparant et d'en tirer le modèle le plus adapté. Nous allons terminer par une conclusion générale qui résume le travail réalisé, donne un bilan sur les apports de ce travail et ses limites et présente les perspectives qui se dégagent à partir de ce travail.

Chapitre 1 : Généralités sur l'IoT

1 Définition de l' IoT

Le terme « Internet des objets » a été inventé en 1999 par Kevin Ashton, directeur exécutif de l'Auto-IDCentre, MIT (Massachusetts Institute of Technology) qui est un laboratoire dédié à la création d'objets connectés à l'aide de l'Identification par Radiofréquence (RFID) et les réseaux de capteurs sans fil. L'IoT est devenu une actualité et une tendance marketing courantes au-delà du battage médiatique. Il est apparu à l'issue de la mécanisation et de la standardisation, appliquée à l'automatisation du traitement du document et de l'information sur support matériel puis numérique. Il s'est rapidement diffusé avec la mondialisation. Peu à peu, les objets ont été modifiés (avec des puces RFID par exemple) ou conçus pour « parler le protocole IP », devenant des « objets connectés » reliés à des serveurs capables de communiquer entre eux. L'IoT a des racines dans plusieurs technologies antérieures : système d'information omniprésente, réseaux de capteurs et l'informatique embarquée. Compte tenu du large éventail de défenseurs de la technologie IoT, aucune définition unique et claire n'a émergé. Ceci nous expose à beaucoup de définissions du concept dont nous analyserons quelques-uns afin d'en fournir une plus générale [1].

Selon l'UIT : « Au plan de la normalisation technique, on peut se représenter l'IoT comme une infrastructure mondiale pour la société de l'information, qui permet de disposer des services évolués en interconnectant des objets (physiques ou virtuels) grâce aux technologies de l'information et de la communication (TIC) interopérables existantes ou en évolutions » [2].
D'après ISO\IEC : « C'est une infrastructure des objets, personnes, systèmes et ressources d'informations avec des services intelligents pour leur permettre de traiter l'information du monde physique et du monde virtuel et réagir» [2].
Pour l'IETF : « Internet des objets (IoT) fait référence aux appareils, qui sont souvent limités dans la communication et capacités de calcul, devenant de plus en plus courantes connecté à Internet, et à divers services qui sont construits sur les capacités que ces appareils fournissent conjointement. Ceci s'attend à ce que cette évolution inaugure plus de communication de machine à machine utilisant Internet sans aucun utilisateur humain qui n'est activement impliqué » [2].

Pour l'IEEE: « L'Internet des objets envisage une configuration automatique, adaptative et un réseau complexe qui interconnecte les « objets » à l'internet grâce à l'utilisation des protocoles de communication standard. Les objets interconnectées ont des propriétés physiques ou virtuelles dans le monde numérique, détection/actionnement capacité, une fonction programmable et sont uniquement identifiables. La représentation contient des informations y compris l'identité, le statut, l'emplacement ou toutes autres informations commerciales, sociales ou privées pertinentes. Les objets offrent des services, avec ou sans intervention humaine, à travers l'exploitation de l'identification unique, la capture de données et la capacité de communication et d'actionnement » [2].

En somme, l'IoT désigne un ensemble d'infrastructures et technologies mises en place pour faire fonctionner des objets divers par le biais d'une connexion Internet. En effet, il est repéré sur deux dimensions qui sont d'ordre technique et conceptuel.

Sur le plan technique, l'IoT est avant tout un réseau d'infrastructure mondiale tout étant une extension d'internet mis en place grâce aux nouveaux outils tels que les capteurs formant les WSAN (Wireless Sensor and Actuator Network) qui permettent d'offrir des services de hautes performances aux mondes physique et virtuel.

Sur le plan conceptuel, l'accent est mis sur les différentes capacités qu'offre l'IoT tout en attribuant des identifications uniques de façon à ce que les objets physiques et virtuels peuvent se connecter et coopérer pour atteindre des objectifs notamment dans le domaine de l'intelligence des objets sans interaction humaine.

2 Architecture et composantes

2.1 Architecture

L'architecture IoT sert à illustrer comment les divers technologies sont liées les unes aux autres pour communiquer l'évolutivité, la modularité et la configuration des déploiements IoT dans différents scénarios.

Tout pareillement à la définition, il n'y a pas une architecture définitive pour l'IoT. Son architecture varie d'un système à un autre en se basant sur le type de réseau à mettre en place. Néanmoins, il existe une architecture générale à trois couches (figure 1) qui est la plus élémentaire, composée de la couche perception, de la couche réseau et de la couche application [3].

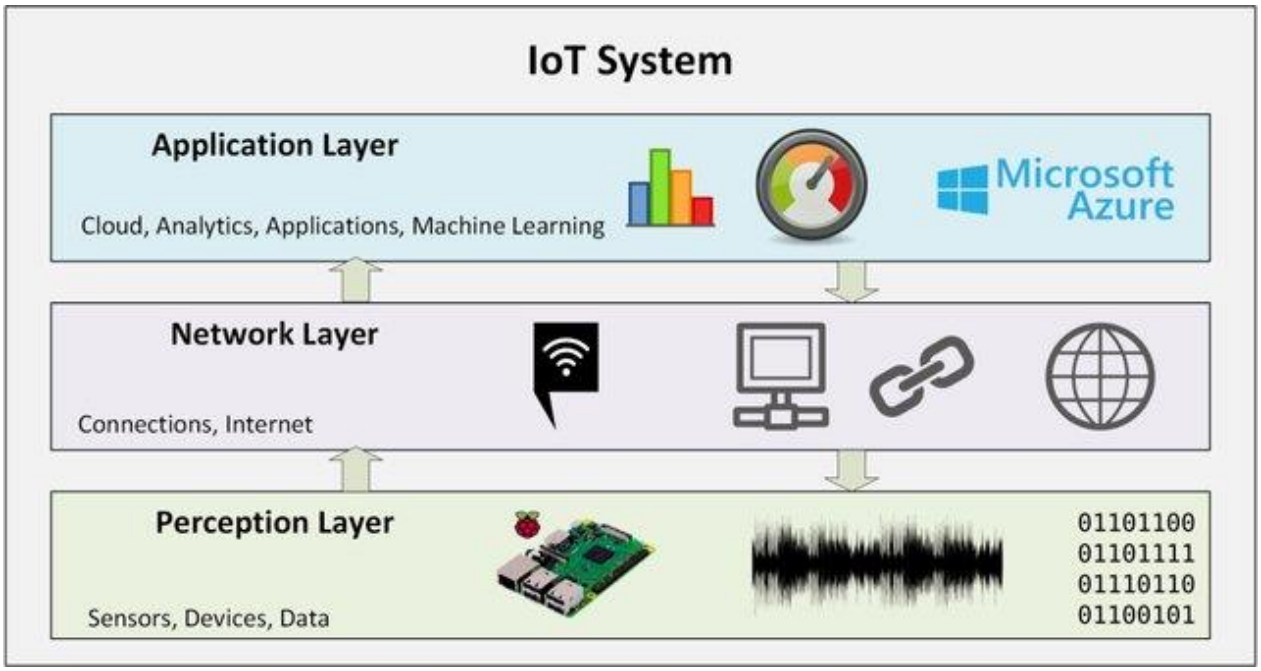


Figure 1: Modèle architecture IoT à 3 couches (source)

- **La couche perception (Perception Layer)**

Cette couche possède des capteurs et actionneurs qui détectent et recueillent des informations sur l'environnement.

- **La couche réseau (Network Layer)**

Elle est responsable de la connexion, du transport et du traitement des données issues des capteurs et actionneurs.

- **La couche application (Application Layer)**

La couche application quant à elle, elle est chargée de fournir à l'utilisateur des services spécifiques et des applications intelligentes.

Le traitement traditionnel de signal numérique et le contrôle supposent aussi des échantillons périodiques résultant des données de séries chronologiques. Cependant, les séries temporelles consomment trop d'énergie au niveau des nœuds et trop de bande passante sur le réseau. Toutes les applications ne sont pas compatibles avec les données d'acquisition apériodiques. Les contraintes de puissance et de bande passante encouragent également le calcul distribué sur les événements de capteur. Des processeurs relativement petits peuvent effectuer des traitements utiles sur de nombreux flux de données. Reconnaître des événements intéressants à l'aide du traitement

des contours réduit la quantité de bande passante réseau consommée ; il réduit également la consommation d'énergie puisque la communication sans fil nécessite de grandes quantités d'énergie. L'architecture basée sur Cloud computing (Serveurs centralisés) et le fog computing (serveurs plus proches de la périphérie) peuvent être utilisés pour effectuer un traitement supplémentaire sur ces événements extraits (figure 2).

Le fog computing, aussi appelé « informatique dans le brouillard », définit une infrastructure chargée de stocker et de traiter des données issues d'objets connectés. Concurrent direct, alternative ou solution complémentaire au cloud computing, le fog computing a comme particularité de stocker et de traiter les données via le recours à des équipements implantés à la périphérie du réseau. Il permet donc de réaliser ces deux actions en local, sans avoir à solliciter un Datacenter situé à plusieurs centaines de kilomètres ou un cloud. Dans ce domaine du stockage et du traitement des données de l'IoT, le fog computing crée une interface supplémentaire que l'on peut situer entre l'Edge Computing et le Cloud Computing [4].

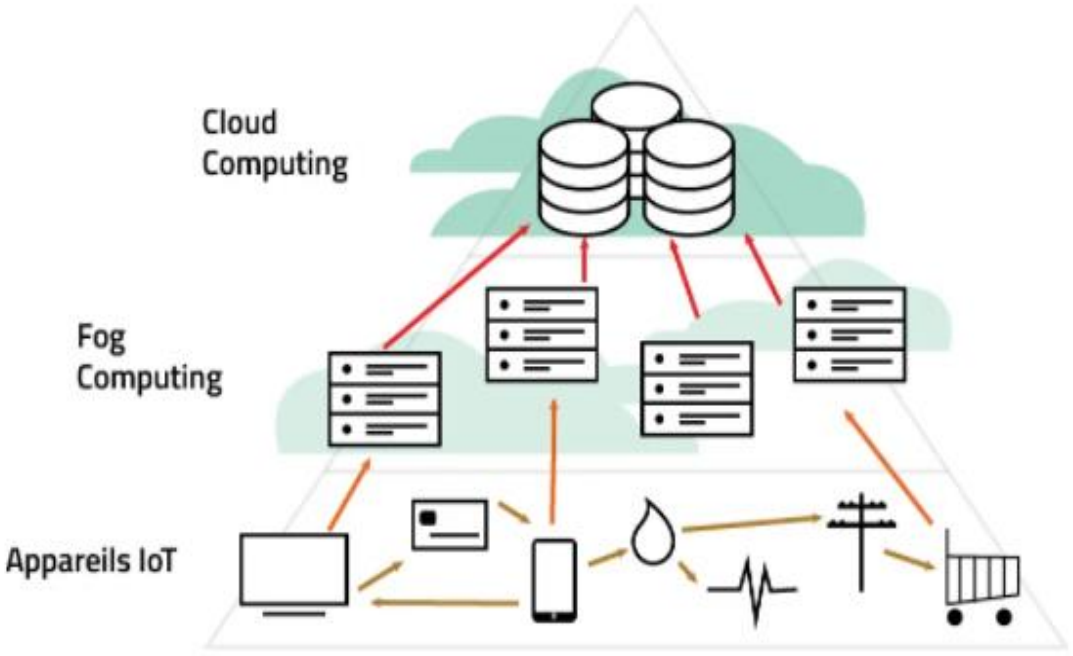


Figure 2: Architecture basée sur le cloud computing et le fog computing (source)

❖ **Appareils IoT**

Il correspond ici à la couche IoT contenant tous les objets connectés ainsi que les capteurs et actionneurs pour la collection de données, ce sont les mêmes composantes sur l'architecture simple.

❖ **Fog Computing**

C'est la couche entre IoT et le cloud, il contient le réseau local. Il a pour rôle de réduire le temps de latence, la consommation d'énergie et de rendre la gestion des données plus efficaces. Il permet aussi d'interrompre la connexion des attaques IoT tout en prévenant le FAI (Fournisseur Accès Internet) mais également il facilite la virtualisation ainsi que standardisation du réseau.

❖ **Cloud computing**

Il contient le serveur de stockage et gère une énorme quantité de données au niveau de la gestion de haut niveau.

2.2 Composants d'une solution IoT

Généralement, un réseau IoT est formé des composants suivants :

❖ **Objet (Module-capteur)**

Dans l'IoT, un « objet » peut être une personne équipée d'un pacemaker (simulateur cardiaque), un animal de ferme qui porte une puce (transpondeur), un dispositif qui mesure la température au sein d'une pièce, une voiture qui embarque des capteurs pour alerter le conducteur lorsque la pression des pneumatiques est trop faible, ou encore tout objet naturel ou fabriqué par l'être humain auquel peuvent être attribuées une adresse IP et la capacité de transférer des données sur un réseau [2].

❖ **Capteurs/Actionneurs**

Un capteur est un dispositif utilisé pour détecter un événement ou une grandeur physique, tels que luminosité, température, humidité du sol, pression, etc. et qui fournit un signal électrique correspondant. Les capteurs IoT sont généralement de petite taille, ils ont un faible coût et consomment moins d'énergie. Les signaux produits par un capteur sont traités par un microcontrôleur pour l'analyse, l'interprétation et la prise de décision.

Un actionneur une technologique complémentaire aux capteurs, convertit l'énergie électrique en mouvement ou énergie mécanique. Les actionneurs permettent de transformer

l‘énergie reçue en un phénomène physique (déplacement, dégagement de chaleur, émission de lumière …) [2].

❖ **Passerelle (Gateway)**

Une passerelle (Gateway) est une combinaison de composants matériels et logiciels utilisés pour connecter un réseau à un autre. Les gateways permettent de relier les capteurs ou les nœuds de capteurs avec le monde extérieur. Elles sont donc utilisées pour la communication de données en collectant les mesures effectuées par les nœuds de capteurs et en les transmettant à l’infrastructure Internet [2].

❖ **Cloud (Informatique en nuage)**

Le Cloud est un système centralisé qui permet de fournir et de transporter des données et divers fichiers sur Internet vers des centres de données. Les différentes données et programmes sont facilement accessibles depuis le système Cloud centralisé. Le Cloud Computing est une solution économique, car il ne nécessite pas d’infrastructure sur site pour le stockage, le traitement et l’analyse. L’évolutivité du Cloud Computing signifie qu’à mesure que votre entreprise se développe, vos capacités technologiques et analytiques le peuvent également [2].



Figure 3: Schéma d’un système IoT (source)

3 Caractéristiques des données dans IoT

Une donnée est une description élémentaire, une observation ou encore une mesure d’une réalité. Lorsqu’on parle de données dans IoT, on identifie ses caractéristiques dans les trois étapes que ce soit sont dans la collecte des données et le stockage, son traitement, et ses applications.

La collecte des données est la toute première phase d'un projet IoT. Il consiste en un flux de données brute en continu qu'il faut capter, filtrer, conserver et organiser. Cette collecte de données ne se fait pas directement avec les capteurs mais avec un fournisseur tiers par l'intermédiaire d'une plateforme comme par exemple Spot (ou GetWay d'Objenious) ou par la plateforme Azure IoT de Microsoft. Ensuite, ces données brutes sont stockées en général dans des bases de NoSQL pour éviter les pertes de données, car ces dernières ne sont pas structurées ou elles dépendent du type de capteur. Après le stockage, les données sont traitées en leur contextualisant et en leur donnant du sens afin qu'elles puissent être une information. Cette information est analysée pourvue d'identifier le passé, prédire le futur, prescrire des solutions sur les différentes situations d'IoT [5]. Les principaux caractéristiques des données IoT sont : l'hétérogénéité animée par la diversité des objets dans IoT ; le streaming à grande échelle du fait de la grande quantité de données générées ; l'horodatage des données qui est la conséquence de la corrélation de l'espace et du temps ; enfin la présence de bruit liée aux erreurs rencontrées lors de la transmission.

L'augmentation en masse de la quantité de données dans IoT, nous renvoie au mécanisme du Big data pour pouvoir gérer cette grande quantité de données. Le Big data « désigne un ensemble massif de données qu'aucun outil de gestion de données classique ne peut gérer. Le Big Data est donc un concept permettant d'accéder à des bases de données gigantesques en temps réel générés par IoT». En d'autre terme, c'est un moyen de répondre à une volumétrie de données. Le Big data fait référence à 3 caractéristiques telles que :

- Le volume très important des données,
- La capacité de gérer les variétés de données stockées,
- La vélocité qui correspond à la vitesse à laquelle les données sont générées, capturées et analysées.

Le Big Data aura un rôle important dans l'efficacité de traitement des informations et permettra ainsi aux développeurs d'IoT, une optimisation de ces outils afin d'élargir la perspective d'évolution actuelle. Il existe une interaction totale entre l'IoT et le Big Data du moment que l'IoT intervient sur notre vie avec une quantité de données qui augmente de plus en plus et que cela nécessite des systèmes capables de traiter cette quantité de données [6].

4 Les applications de l'IoT

L'IoT apparait de plus en plus comme une basique de nos vies car son application touche presque tous les secteurs de la vie, plus particulièrement les environnements intelligents dont nous allons en présenter quelques-uns conférés à la figure (4) [7].

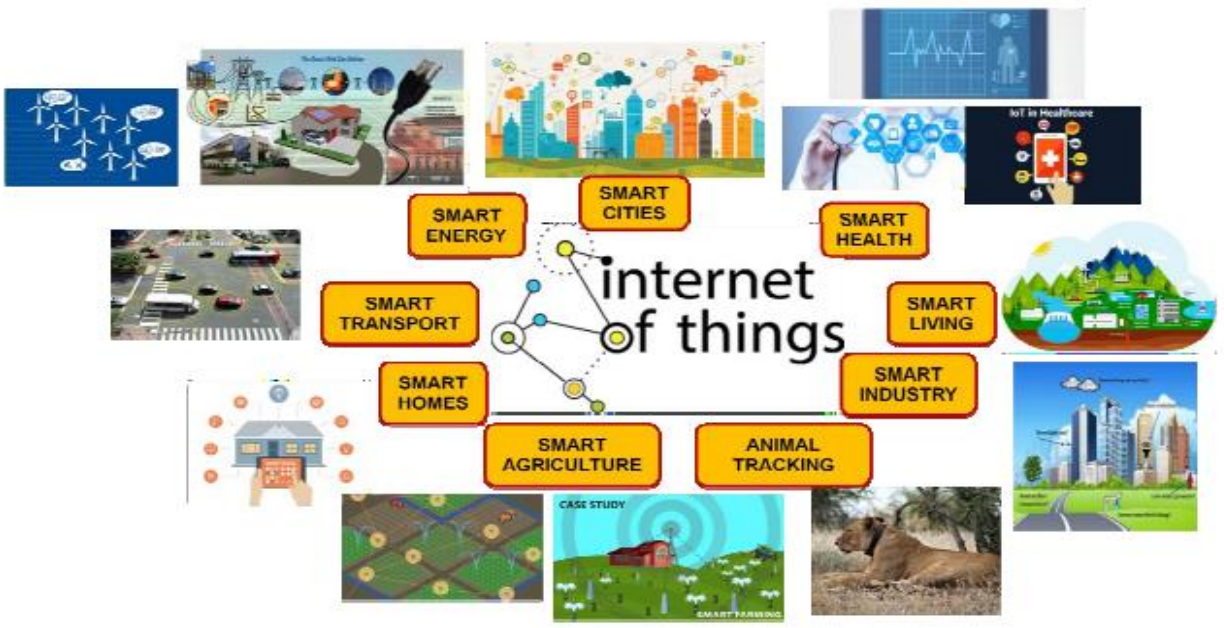


Figure 4: Les differents domaines d'application IoT (source)

❖ **Santé connecté (e-santé)**

Le paradigme IoT Cloud a été largement utilisé dans le secteur de la santé pour le diagnostic, le traitement et suivi à distance de l'état des patients. Le paradigme doit servir quatre pivots à savoir :

- le suivi : est une fonction visant à identifier le patient en mouvement,
- l'identification et l'authentification : identification visant à réduire les erreurs de diagnostic et authentification utilisés pour répondre à la sécurité d'exigences,
- collecte de données : vise généralement à réduire temps de traitement et il est lié à l'intégration de la RFID technologie avec d'autres informations sur la santé,
- la détection : utilisée pour fournir des informations en temps réel sur le patient

Les enjeux dans ce domaine sont définis comme : le contrôle, la sécurité, hétérogénéité, interopérabilité et la lecture des flux du QoS.

❖ **Vie et maison intelligente**

On y retrouve des appareils de contrôle à distance pour allumer et éteindre à distance les appareils électroménagers pour éviter les accidents et économiser de l'énergie. Dans ce contexte, beaucoup d'appareils électroménagers sont devenus intelligents tels que les réfrigérateurs avec écran LCD indiquant ce qui est à l'intérieur, les machines à laver nous permettant de surveiller le linge à distance, et les cuisinières. Avec une interface des applications d'un Smartphone nous pouvons gérer le contrôle de la température réglable à distance et la surveillance de la fonction d'auto nettoyage. Ainsi, il est possible d'afficher les conditions météorologiques extérieures telles que l'humidité, température, pression, vitesse du vent et niveaux de pluie avec capacité pour transmettre des données sur de longues distances. L'IoT offre avec une fabrication de caméras et de systèmes d'alarme à domicile grâce aux systèmes de détection d'intrusions, les gens se sentent en sécurité dans leur vie quotidienne à la maison.

❖ **Industrie connectée**

Dans le domaine de l'industrie, l'IoT permet d'avoir des prédictions sur les dysfonctionnements et l'entretien des équipements. La maintenance peut être programmée automatiquement avant une défaillance réelle des pièces en installant des capteurs à l'intérieur de l'équipement pour surveiller et envoyer des rapports. Au niveau de ces industries nous notons une communication interne entre les machines. Il constitue aussi un apport dans la surveillance de gaz toxique et d'oxygène comme par exemple dans les industries des produits chimiques et dans les mines.

❖ **Suivi des animaux**

Avec les capteurs, il est possible d'identifier l'emplacement et l'identification des animaux dans des pâturages ouverts ou dans un endroit avec de grandes écuries. Etude de la ventilation et de la qualité de l'air dans les élevages et détection de gaz nocifs dans les excréments. Il permet aussi de gérer les soins de la progéniture en contrôlant des conditions de croissance dans les élevages. Assurer leur survie et leur santé sont quelques utilisations simples de l'IoT sur les animaux.

❖ **Agriculture intelligent**

La production alimentaire agricole est la base de la survie humaine. Cependant, en raison du retard des méthodes agricoles traditionnelles et catastrophes naturelles, la productivité agricole dans

certaines zones défavorisées restent faible, ce qui entraîne un approvisionnement alimentaire insuffisant. Afin d'augmenter plus efficacement la production alimentaire et améliorer les moyens de subsistance des gens, une agriculture intelligente a été proposée et appliquée dans la pratique. L'agriculture intelligente est un système agricole qui utilise les technologies de l'information, telles que les Mégadonnées, l'IoT, l'informatique en nuage et l'informatique de périphérie. La combinaison de ces technologies rend la production plus efficace grâce à l'interconnexion, la détection et des moyens intelligents. Le système d'agriculture intelligent utilise des capteurs de diverses fonctions pour obtenir des données pertinentes dans les terres agricoles, ce qui peut améliorer la croissance des cultures en temps réel. Ces données incluent la température, teneur en dioxyde de carbone, humidité du sol, humidité de l'air, pH du sol, l'intensité lumineuse, etc…

- **Transport intelligent**

Le transport intelligent a été proposé pour remédier aux faibles niveaux de motorisation sur le guidage routier des véhicules ainsi que les problèmes de congestion du trafic urbain et l'écart entre l'offre et la demande de parkings. Ceci est réalisé en déployant un grand nombre de routes nœuds de réseau de capteurs mobiles embarqués, ainsi qu'en construisant un lot de systèmes d'application de services de trafic intelligents basés sur la technologie IoT. De cette façon, une large connaissance des éléments de trafic interconnectés est construite grâce au traitement complet d'IoT de transport intelligent pour la collecte, le traitement, l'analyse, la gestion et le service d'informations massives. Les objectifs sont d'atteindre un public plus productif, précis et humain en service d'information, et de former un environnement de transport intelligent et harmonieux afin que les systèmes de transport fonctionnent plus efficacement.

- **Energie intelligente**

L'énergie est une ressource matérielle vitale pour la survie humaine. Avec l'intensification de la population, l'urbanisation, la consommation d'énergie a également augmenté à des niveaux record. Cependant, cela a également entraîné la diminution de diverses sources d'énergie non renouvelables et renouvelables. Pour faire face à ces problèmes, ils ont formulé des stratégies de développement des énergies vertes et renouvelables. Avec le développement rapide de la technologie informatique, l'énergie intelligente a été proposée pour résoudre les problèmes de développement énergétique vert, stable, sûr et durable.

❖ **Ville intelligente**

Les villes intelligentes reposent sur des villes numériques et sont connectés à la vraie ville à travers des réseaux de capteurs omniprésents dont le calcul décisionnel, l'analyse et le stockage massif de données sont gérés par des plates-formes cloud. Elles offrent des avantages tels que la santé structurelle concernant la surveillance des vibrations et des conditions de matériaux dans les bâtiments, surveillance des passagers dans les rues ponts et monuments historiques ; l'éclairage intelligente dans les rues ; surveillance et contrôle des incendies ; détection de niveaux de déchets dans les conteneurs pour optimiser la collecte des déchets itinéraires. Les poubelles et les bacs de recyclage avec étiquettes RFID permettent le personnel d'assainissement de voir quand les ordures ont été sorties.

5 Enjeux de sécurité dans IoT

5.1 Les menaces dans IoT

Il y a eu différents défis et vulnérabilités autour de l'environnement IoT et cela est dû à la connectivité avec Internet, comme il est évident que le principal problème d'internet est sa sécurité mais aussi d'autre part ils sont liés aux emplacements des capteurs. C'est ainsi que nous repérons deux niveaux de types d'attaques que sont les menaces physiques et les cybermenaces [8].

5.1.1 Les menaces physiques

Les menaces physiques désignent l'ensemble des attaques effectuées au niveau des objets connectés ou encore au niveau des capteurs. Les contraintes de calcul et énergétiques limitées des nœuds sont un obstacle à l'application contre-mesures conventionnelles de sécurité des réseaux informatiques aux WSN. De plus, dans ces réseaux, les nœuds deviennent plus vulnérables lorsqu'ils sont placés dans des environnements non protégés comme dans les rues. Dans ces circonstances, les attaquants peuvent facilement capturer des nœuds, accéder à des informations confidentielles dans leur mémoire (par exemple, des clés cryptographiques) et reprogrammer leur comportement. Etant trop exposés, ces objets peuvent aussi subir des dommages causés par des catastrophes naturelles ou humaines [9].

5.1.2 Les cybermenaces

Ce sont les menaces concernant le réseau fonctionnel d'une solution IoT. Il est également courant que les attaquants profitent de la nature sans fil des communications pour intercepter les messages

ou obstruer la fréquence bandes pour empêcher la bonne réception de certains paquets. A ce niveau nous avons deux types de menaces [9]:

- **Menaces passives** : le risque latent est distinctement conduit via l'espionnage tout au long des flux de correspondance ou le système. En écoutant, l'offensive peut recueillir des données de l'instrument, l'instrument propriétaires, ou les deux.
- **Menaces actives** : L'attaque ne se résume pas à l'écoute habile de la correspondance routes mais aussi l'altération des structures IoT pour modifier leur conception, réguler leur correspondance, leur refuser toute aide, et ainsi de suite. Les attaques peuvent incorporer un regroupement de médiations, de perturbations, et modifications

5.2 Quelques attaques dans IoT

Il existe plusieurs types d'attaques dont nous allons expliquer quelques-uns les plus fréquents :

5.2.1 Dénis de service : DoS / DDoS

Une attaque DoS est une attaque où l'attaquant essaie de rendre une machine ou une ressource réseau indisponible pour un consommateur en perturbant accidentellement ou de manière non concluante des services. Une attaque DoS inonde l'appareil ciblé de demandes répétitives inutiles de requêtes essayant de surcharger les structures et bloquer certaines ou toutes les demandes légitimes d'être satisfaites.

DDoS utilise diverses machines ou ordinateurs et remplit la machine avec un trafic malveillant. L'attaque DDoS envoie un nombre illimité de requêtes au serveur en utilisant des adresses IP illégales et ces adresses sont difficiles à localiser [9].

5.2.2 Mirai

C'est un logiciel malveillant qui est utilisé pour analyser Internet et identifier un grand nombre d'appareils IoT, en particulier des caméras IP et routeurs, qui exécutent Linux et utilisent les informations d'identification par défaut pour les connexions Telnet. Il est plus souvent utilisé pour les attaques DDoS. Les attaquants ont réussi à transformer ces appareils en des Bot et les utiliser pour provoquer une attaque par déni de service sur les serveurs DNS. Ceci par conséquent abouti à couper l'accès à certains sites Web populaires [9].

5.2.3 R2L

R2L est une attaque à distance vers l'utilisateur cible d'un ou de nombreux systèmes connectés par un réseau et y installe des fichiers ou des logiciels malveillants. L'attaquant ici essaie de trouver des points vulnérables du système de sécurité de l'ordinateur pour accéder aux données personnelles. L'avantage des attaquants est, ici qu'ils peuvent voler des informations précieuses à l'utilisateur en accédant au système à distance. Le sondage est la méthode où un attaquant insère un programme ou tout autre périphérique à la jonction clé d'un réseau. Le but de la sonde est de viser pour surveiller et collecter des informations sur l'activité du réseau ainsi que sur les points faibles des périphériques réseau ou des connexions susceptibles de compromettre le système [10].

5.2.4 U2R

U2R ou un utilisateur à l'attaque racine, le pirate utilise certains privilèges en tant qu'utilisateur normal, puis plus tard, essaie d'exploiter les vulnérabilités du système pour obtenir des avantages de super utilisateur. Autrement c'est l'acte d'obtenir les privilèges de la racine tout en accédant légalement à une machine locale [10].

5.2.5 Man-in-the-middle (Homme du milieu)

Comme le nom le suggère, il s'agit d'une interférence où un attaquant infiltre la communication entre deux ou plus de parties qui ignorent l'existence de cet agresseur. L'attaquant peut soit recevoir passivement les informations échangées par les deux participants ou interférer activement et changer les données ou les informations qui sont communiquées. Pour être explicite, il joue le rôle d'intermédiaire entre les deux interlocuteurs sans que ces derniers aient des soupçons ni aucun signe d'attaque en raison d'une communication normal ou d'absence d'évènements inhabituel pendant leur échange. Ce type d'attaque pourrait être dangereux en raison de la possession du message d'origine par l'agresseur qui est dans la capacité de piéger le destinataire en lui faisant croire que le message qu'il reçoit est légitime [11].

5.2.6 Attaque par injection

Les attaques par injection font référence à une large classe de vecteurs d'attaque qui permettent à un hacker de s'octroyer une entrée dans un programme, interprété comme une commande ou requête qui modifie le déroulement de l'exécution du logiciel. Les attaques par injection sont parmi les attaques d'applications Web les plus anciennes et les plus dangereuses. Ces attaques peuvent entraîner un vol et/ou une perte de données, un déni de service, et jusqu'à la prise de contrôle complète du système. Il existe différents types d'attaques par injection tels que : code injection,

cross-site Scripting, email injection, injection SQL, etc… Ce qui rend les attaques par injection particulièrement inquiétantes, c'est leur énorme surface d'attaque (en particulier pour les SQL et XSS). Il existe d'innombrables outils disponibles gratuitement et fiables qui permettent même aux hackers inexpérimentés d'abuser de ces vulnérabilités [9].

5.2.7 Attaque par mot de passe

Les mots de passe étant le mécanisme le plus couramment utilisé pour authentifier les utilisateurs d'un système informatique, l'obtention de mots de passe est une approche d'attaque courante et efficace. Le mot de passe d'une personne peut être obtenu en fouillant le bureau physique de la personne, en surveillant la connexion du réseau pour acquérir des mots de passe non chiffrés, en accédant à une base de données de mots de passe ou simplement en le devinant. Cette dernière approche « deviner » peut s'effectuer de manière aléatoire ou systématique :

- ✓ Les attaques par **force brute** consistent à adopter une approche aléatoire c'est-à-dire essayer différents mots de passe en espérant que l'un d'entre eux fonctionnera. Une certaine logique peut être appliquée : essayer des mots de passe liés au nom de la personne, à son poste, à ses passe-temps ou à des éléments similaires ;
- ✓ Dans une **attaque par dictionnaire**, un dictionnaire des mots de passe courants est utilisé pour tenter d'accéder à l'ordinateur et au réseau d'un utilisateur. Une approche consiste à copier un fichier chiffré contenant les mots de passe, à appliquer le même chiffrement à un dictionnaire des mots de passe couramment utilisés et à comparer les résultats [11].

5.3 Les solutions de défenses associées à un système IoT

Toute la sécurité IoT se repose sur une stabilité de l'intégrité, la confidentialité, l'authenticité et la disponibilité des données du réseau.

5.3.1 Les solutions classiques

Nous disposons 3 méthodes classiques :

- ❖ **Mécanismes de tolérance aux pannes**

Pour faire face aux attaques dans les réseaux, des chercheurs avaient trouvé comme solution les méthodes de tolérance aux pannes. Ces méthodes consistent à proposer deux algorithmes pour réparer les chemins brisés dans les topologies de maillage en ne recalculant que de petites parties de la structure topologique où se trouvent les liens rompus ou bien à utiliser le mécanisme de

redondance de chemins. Ces mécanismes sont à faibles consommation d'énergie et de résilience [12].

- **Cryptographie**

La cryptographie est l'étude des algorithmes appelés cryptosystèmes permettant la protection d'informations numériques. Bien que les mécanismes précédents soient recommandables, le problème d'interopérabilité s'impose car nous retrouvons au niveau des solutions IoT beaucoup de technologies et configurations différentes. Afin d'éviter ce problème d'autres se sont focalisés sur la prévention des attaques sur la confidentialité, l'authentification, l'intégrité et la disponibilité tout en utilisant la cryptographie que ce soit en mode symétrique ou asymétrique. Pour l'intégrité, ce sont les codes d'identification (MAC) qui sont utilisés pour empêcher les modifications inaperçues sur les paquets en transit. Et pour la disponibilité les attaques qui y sont liés sont les transmissions sélectives où les nœuds capturés par les attaquants n'envoient pas la totalité des paquets et le brouillage où l'attaquant envoie des signaux de hautes puissances pour perturber le réseau ciblé qui peut même aller jusqu'à perturber les autres réseaux partageant la même bande de fréquence[12].

- **Etalement de spectre à saut de fréquence**

C'est un autre mécanisque utilisé pour la disponibilité, plus précisément pour le brouillage. Il consiste à changer constamment le canal de transmission dans la bande de fréquence du protocole. Cependant, les brouilleurs actuellement disponibles sur le marché peuvent brouiller tous les canaux utilisés par plusieurs protocoles en même temps [12].

5.3.2 Les systèmes de détection d'intrusion

Même si des solutions classiques sont proposées pour éviter ou limiter les attaques, ils ne sont pas efficaces et applicables dans tous les réseaux IoT notamment ceux à grandes échelles. Avec l'augmentation d'une forte quantité de données due aux nombreux objets connectés, les réseaux IoT sont de plus en plus exposés aux attaques. Ces attaques sont très dangereux car ils ralentissent les réseaux, provoquent l'abandon d'une grande quantité de paquets, épuisent les batteries des nœuds et peuvent compromettre plusieurs services. C'est ainsi qu'une approche très

efficace a été adoptée pour limiter ou même supprimer les attaques, les systèmes de détection d'intrusions ou IDS. On appelle IDS un mécanisme écoutant le trafic réseau de manière furtive afin de repérer des activités anormales ou suspectes et permettant ainsi d'avoir une action de prévention sur les risques d'intrusion en envoyant une alerte à l'administrateur. Nous distinguons deux types de système de détection d'intrusion [10] :

- **Network Based Intrusion Detection System**

Les NIDS ou systèmes de détection d'intrusion basés sur le réseau sont des systèmes matériels ou logiciels qui permettent d'analyser le réseau afin d'identifier les interférences du réseau ou toute menaces. Ils sont placés à des points stratégiques du réseau de sorte qu'ils puissent garder l'œil sur les périphériques ou appareils connectés du réseau. Une fois que l'intrusion est détectée, une alerte est envoyée à l'administrateur. Lorsque le NIDS a une capacité de réponse, on parle de Système de prévention des intrusions dans le réseau (NIPS). Un NIPS diffère d'un NIDS en ce que il est capable de modifier les flux de trafic réseau pour bloquer une attaque en cours par exemple.

Les NIDS utilisent deux types de solutions pour la détection d'attaques ou d'anomalies [13] :

- **Par signature :** Un NIDS peut utiliser la méthode par signature pour vérifier l'authenticité des trafics réseaux et ne détecte que des attaques spécifiques. En effet ils utilisent des motifs tels que des séquences de drapeaux TCP, d'octets, d'intrusions également appelés signatures stockés dans la base de données du NIDS. Et lorsqu'on a une séquence de trafic en entrée, cette dernière est comparée à celles stockées dans la base afin d'identifier les attaques et ils ne peuvent détecter que le attaques pour lesquelles une signature existe. Les alarmes sont déclenchées une fois qu'une séquence d'évènements qui corresponde à une des signatures est identifiée. Le principal avantage de ce type de détection est le faible taux de faux positifs (données normales considérées comme attaques), c'est à un faible taux de fausses alertes. Dans le contexte des WSN dans les villes intelligentes, la détection basée sur les signatures est utile pour identifier les attaques ciblant les réseaux avec un comportement régulier. Par conséquent, ils échouent complètement à détecter les nouvelles attaques telles que les attaques exploitant les vulnérabilités zero-day.

- **Par détection d'anomalie :** la méthode par détection d'anomalies est également utilisée par les NIDS et dépend du comportement du système. Lorsqu'un attaquant essaie de transformer un comportement normal du système en un comportement anormal, le système de détection d'anomalie le détecte. Le NIDS basé sur la détection d'anomalies apprend le profil des comportements en réseau. Lorsque le trafic réseau n'est pas conforme au profil de comportement légitime appris, une alerte est déclenchée. Alternativement, les techniques de détection d'intrusion à la recherche d'anomalies sont capables d'identifier les changements dans le système qui ne correspondent pas au comportement normal. Le principal avantage des systèmes de détection par anomalies est qu'ils peuvent détecter les nouvelles attaques ou encore appelées attaques zero-day. C'est ainsi que nous avons porté notre choix sur la détection par anomalie pour la sécurité du trafic réseau dans le cadre d'une maison intelligente.

- **Host Based Intrusion Detection System**

HIDS ou systèmes de détection d'intrusion basés sur les hôtes joue un rôle essentiel dans le réseau. Il est utilisé pour détecter les attaques ou malveillants sur les périphéries hôtes. Il intervient là où le NIDS n'a pas réussi à détecter les intrusions.

Nous notons également que la détection d'anomalie dépend de 3 critères :

- **Point :** une anomalie **ponctuelle** n'est rien plus qu'une valeur aberrante, c'est-à-dire une valeur significativement différente en taille des restes de données,
- **Contexte :** une anomalie **contextuelle** (également appelée anomalie conditionnelle) est une valeur qui est considérée anormal dans un certain contexte, alors qu'il pourrait être considéré comme normal dans un autre,
- **Collective :** l**es** anomalies **collectives** ne sont pas directement détectables par l'observation d'un nombre limité d'échantillons, car ils ne peuvent émerger que de l'observation de tendances réparties sur une période de temps suffisamment longue.

6 Conclusion

Ce chapitre introductif présente les généralités sur l'IoT en détaillant d'abord ses éléments techniques c'est à dire architectures et composants. Ensuite, de présenter les caractéristiques de

données et les différents domaines d'applications. Il a été question aussi dans ce chapitre d'analyser et de discuter de la sécurité dans les réseaux IoT. Nous avons de ce fait présenté les enjeux de sécurités dans l'IoT, les menaces, les différentes attaques et les solutions proposées pour lutter contre ces différents types de menaces. Ce chapitre nous a permis de mettre le contexte de notre étude afin de mieux aborder le problème posé.

Chapitre 2 : Classification des données

1 Définition

La classification des données est un processus d'organisation des données structurées et non structurées en catégories mutuellement exclusives selon leur niveau de sensibilité en se basant sur différentes caractéristiques telles que le type de données et son contenu. La classification permet une utilisation plus facile des données, une gestion des risques et attaques dans IoT mais aussi celle des processus de connaissances légales et de conformité et une optimisation des investissements de sécurité. Une étape fondamentale pour une sécurité de l'information est la gestion efficace des risques, ce qui est un avantage principal de la classification [19].

La classification des données IoT est faite en 4 étapes : il commence par une mise en place d'une politique de classification des données en définissant les objectifs, puis créer des flux de données basés sur les outils de classification pour identifier les données sensibles. Ensuite, identifier les catégories et les critères de classification en attribuant des étiquettes aux données et enfin définir les résultats et les utiliser pour améliorer la sécurité en connaissant les risques [19, 20]. Ainsi, la classification peut se faire à 3 niveaux : applicatif, réseau et logiciel. La classification joue un rôle important dans toutes les sciences et techniques qui font appel à la statistique multidimensionnelle. Citons tout d'abord les sciences biologiques: botanique, zoologie, écologie, etc. Ces sciences utilisent également le terme de « taxinomie » pour désigner l'art de la classification. De même les sciences de la terre et des eaux : géologie, pédologie, géographie, étude des pollutions, font grand usage de classifications.

2 L'Intelligence Artificielle

L'IA est un domaine qui a existé depuis très longtemps, mais son avènement n'a vu jour qu'à la fin des années 90 lorsque le grand champion d'échec Kasparov a perdu contre la machine deep Blue en 1997. Deep Blue est un ordinateur qui est opéré sur l'IA, produit par IBM et capable de calculer jusqu'à 2 millions de coûts par seconde. C'est ainsi que l'IA commence à être utilisé par la logistique, pour les diagnostiques médicales, pour les robots industriels, par le moteur de

recherche Google, pour la reconnaissance vocale, santé, etc… Le terme intelligence artificiel soulève des questions sur ses mots composants, à savoir le sens du mot intelligent qui assigne l'intelligence d'une machine qui se comporte comme une personne et à l'attribut artificiel qui remet en cause si le cerveau d'un être humain est quelque chose à modéliser. Cependant une très grande réflexion est penchée sur sa définition exacte et ses différentes branches [21].

2.1 Définition de l'IA

Beaucoup d'interprétations ont vu jour sur l'IA, ce qui engendre beaucoup de définitions du domaine. Pour faire face, nous allons nous référer sur quelques-uns et d'en tirer une synthèse sur la définition de l'IA.

- John McCarthy, l'un des pionniers de l'IA, a été le premier à définir le terme artificielle intelligence en 1956, comme : « intelligence artificielle » (Artificiel Intelligence ou AI, en anglais) caractérise l'élaboration de programmes informatiques capables de prendre en charge des tâches habituellement effectuées par des humains car demandant un apprentissage, une organisation de la mémoire et un raisonnement. L'objectif est de parvenir à transmettre à une machine des fonctions propres aux vivants : rationalité, raisonnement, mémoire et perception » [22]. En ce sens, nous pouvons dire que le but de l'IA est de développer des machines qui se comportent comme si elles étaient intelligentes, ainsi les robots susmentionnés peuvent être décrits aussi intelligents.
- Un de ses créateurs, Marvin Lee Minsky, définit l'intelligence artificielle de la façon suivante : « la construction de programmes informatiques qui s'adonnent à des tâches qui sont, pour l'instant, accomplies de façon plus satisfaisantes par des êtres humains car elles demandent des processus mentaux de haut niveau tels que : l'apprentissage perceptuel, l'organisation de la mémoire et le raisonnement critique » [22].

Nous pouvons tirer de ces définitions que l'IA est un domaine scientifique qui cherche à résoudre les problèmes logiques ou algorithmiques grâce à la puissance de calcul croissante et aux dispositifs imitant ou remplaçant l'être humain qu'il contient.

2.2 Branches de l'IA

L'IA est un vaste domaine qui a subi une évolution incroyable en un laps de temps très petit englobant un ensemble de concepts et de technologies qui forment ses différentes branches (voir figure 5). Il est souvent classé dans les groupes de la logique mathématique, des sciences cognitives et des sous disciplines de l'informatique.

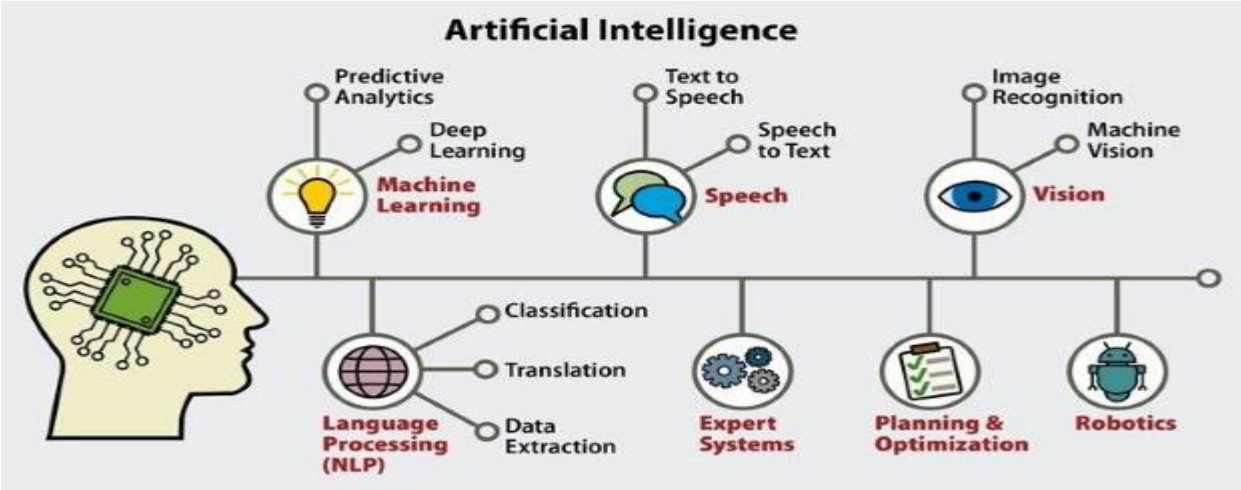


Figure 5: Les branches de l'IA (source)

Du moment que l'IA subit une évolution éclatante, grâce à l'émergence du Cloud Computing et du Big Data, soit d'une puissance de calcul peu coûteuse et de l'accessibilité à un grand nombre de données, les machines ne sont plus programmées mais elles apprennent. Ceci est à l'origine de ses branches la plus répandu, le Machine Learning.

2.3 Machine Learning appliqué à la classification

Le développement des algorithmes du ML est l'une des branches la plus étendue de l'IA. Il permet d'aborder des questions plus complexes et pour répondre à un certain type de question. La résolution avec efficacité des problèmes nécessite l'application de modèles adaptés. Ceci passe d'abord par une connaissance des différents modèles : leurs caractéristiques, les types de problème qu'ils ressoudent, leurs critères et conditions d'applicabilité etc.

Dans la suite nous allons définir le ML et examiner les différentes techniques et méthodes d'apprentissages.

2.3.1 Définition du Machine Learning

Le terme apprentissage fait référence à une technique spécifique portant sur le fait que la « machine » apprend à partir des exemples sans suivre d'instructions. "Le ML, ou apprentissage automatique, est capable de reproduire un comportement grâce à des algorithmes, eux-mêmes alimentés par un grand nombre de données. Confronté à de nombreuses situations, l'algorithme apprend quelle est la décision à adopter et créer un modèle. La machine peut automatiser les tâches en fonction des situations" [23].

En effet, le ML n'est rien d'autre une étude des algorithmes permettant l'automatisation des « machines » par l'utilisation d'expérience. En d'autre terme c'est de donner une machine la capacité d'apprendre sans la programmer de façon explicite suivant un cycle de vie de l'algorithme (voir figure 6).

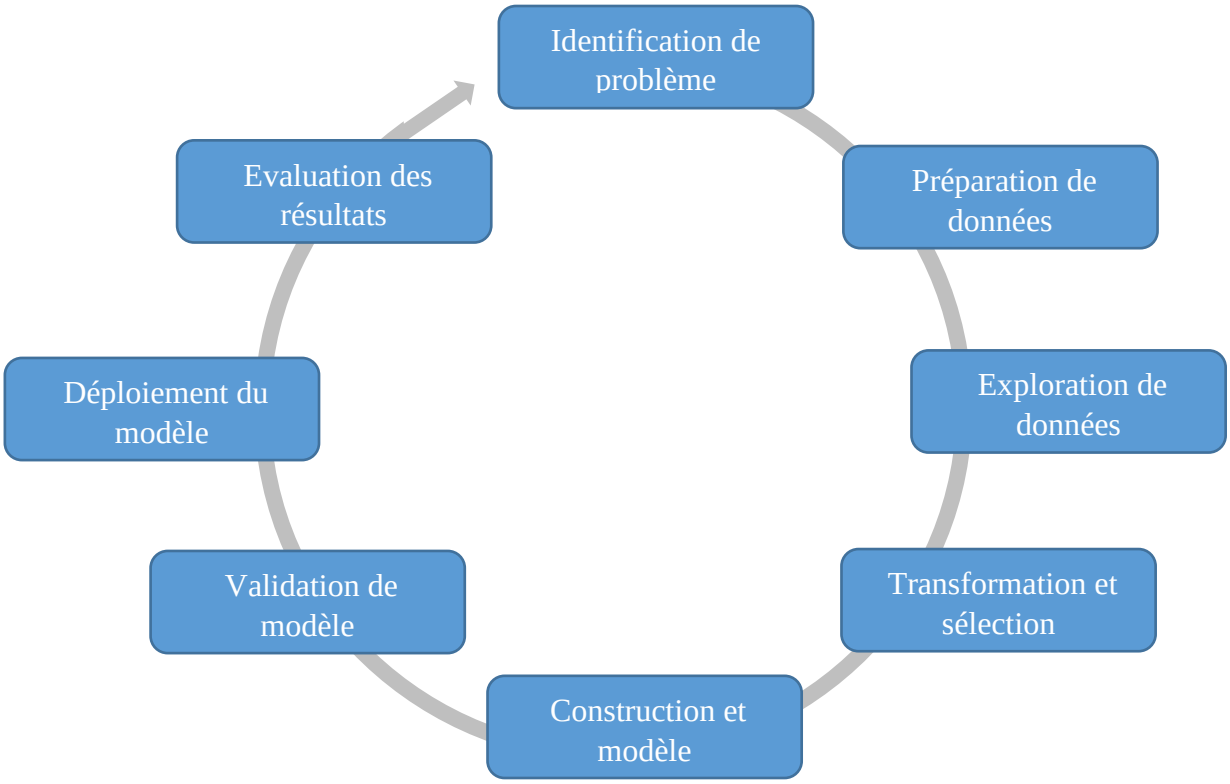


Figure 6: Cycle d'un algorithme du Machine Learning

2.3.2 Les techniques d'apprentissage

Nous distinguons 3 types d'apprentissages :

- **Apprentissage supervisé :** c'est une technique d'apprentissage automatique où on cherche à produire de manière automatique des règles à partir d'une base de données dans laquelle les objectifs sont labélisés c'est-à-dire étiquetés, indiquant la catégorie de chaque classe. Ensuite, un modèle est généré pour classer les nouveaux échantillons non étiquetés dans l'une des catégories définies. Dans ce type d'apprentissage nous avons deux phases dont la phase d'entrainement où il apprend les données labélisées afin de pouvoir identifier celles non étiquetées dans la phase test [12].
- **Apprentissage non supervisé :** il est tout le contraire de l'apprentissage supervisé. En effet ne nécessitent pas de formation de données labellisée et sont capables de diviser un ensemble de données en divers sous-ensembles sans un modèle appris au préalable. Il est fréquemment utilisé dans le clustering dont le principe est de laisser la machine apprendre à classer les données selon leur ressemblance et dans la détection d'anomalies en détectant les caractéristiques éloignés des autres échantillons. Il est également utilisé dans la réduction de dimension qui consiste à réduire la complexité d'un Dataset en projetant ses données dans un espace de plus petit dimension ayant pour but d'accélérer l'apprentissage de la machine et de lutter contre le fléau de la dimension [12].
- **Apprentissage par renforcement :** c'est une technique qui classe les problèmes d'apprentissage à partir d'expériences, ce qu'il convient à optimiser une récompense quantitative au cours du temps. Il est très précieux dans le domaine de la robotique, où les tâches à exécuter sont souvent suffisamment complexes pour défier l'encodage en tant que programmes et aucune donnée d'entrainement n'est disponible [21].

2.3.3 Les algorithmes du ML

Il existe beaucoup d'algorithmes pour le Machine Learning (voir figure 7) selon la technique d'apprentissage et l'objectif, mais nous en disposons 3 qui sont plus utilisés notamment dans la classification tels que les arbres de décisions, le Support Vector Machine et le Random Forest, ils ont tous des méthodes supervisés. Il en existe d'autres pour la classification comme k-nearest Neighbors (KNN), Bayésien algorithms, AdaBoost…

- **Les arbres de décision** (Decision Tree en anglais) sont utilisés dans la régression et la classification. C'est un modèle prédictif qui possède par structure arborescente des extraits de données d'apprentissage pour partionner les données en sous-groupe homogène du point de vue de la variable à prédire [23].
- **Les machines à vecteur de support ou SVM** sont généralement utilisées pour la classification et la régression (régression de vecteurs de support). Dans la modélisation de la classification, un SVM a un concept plus mature et plus clair sur le plan mathématique par rapport à d'autres techniques de classification. Les machines à vecteurs support peuvent également surmonter le problème de la classification et de la régression de façon linéaire ou non linéaire. Par conséquent, l'apprentissage statistique est une forme de base d'un SVM [23].
- **Random Forest :** «forêt » qu'il construit est un ensemble d'arbres de décision qu'il fusionne pour obtenir une meilleure prédiction plus précise et stable car les arbres de décision simple donnent des performances en jeu d'entrainement mais n'est pas flexible en test. Généralement formés avec la méthode d'ensachage, l'idée générale de la méthode d'ensachage est que la combinaison de modèles d'apprentissage augmente le résultat global [23].

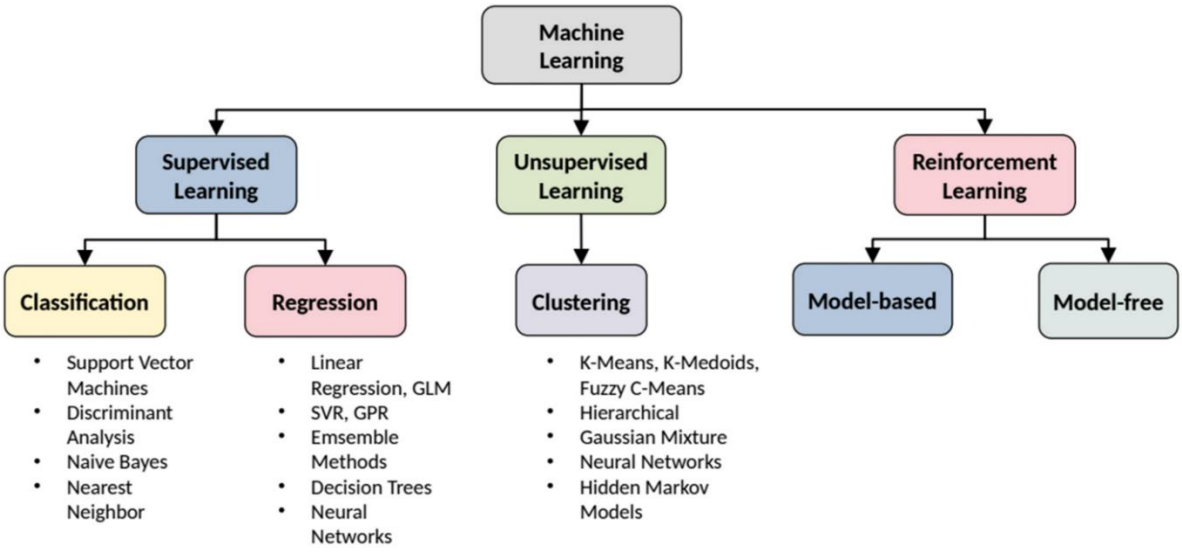


Figure 7: Schéma d'hiérarchisation des modèles du Machine Learning

2.4 Les réseaux de neurones appliqués à la classification

Pendant très longtemps, beaucoup de biologistes, psychologues et médecins ont travaillé pour comprendre comment fonctionne le cerveau. C'est plus tard que vers les années 1900 qu'ils virent que les cellules nerveuses et leurs connexions assurent le traitement et la circulation des signaux dans le cerveau et par conséquent ils sont responsables de ce fonctionnement pour la prise de confiance, des pensées et la capacité d'apprendre. C'est ainsi qu'en 1981 qu'Heinrich Waldeyer introduit le terme « neurones » comme nom pour la combinaison de ces éléments. Ainsi le neurone prend progressivement le pas sur la fibre nerveuse comme dénomination de l'objet scientifique, d'où le nom de l'intelligence artificielle [24].

2.4.1 Neurone Artificiel ou formel

Un neurone artificielle ou encore appelé neurone formel est une branche d'algorithme spéciale de l'apprentissage automatique très adaptable aux données et capable d'étudier toute fonction mathématique cachée entre les données (entrée) et le résultat (sortie). Avec la grande quantité de données que produisent l'IoT et une puissance de calcul illimitée en utilisant le cloud, ils rendent les RNA plus intéressantes pour le traitement des données. Ils sont capables de réaliser n'importe qu'elle tâche comme conduire une voiture, jouer aux échecs et de reconnaitre et classifier des images ou des données. Un RNA est inspiré d'un neurone biologique en modélisant certaines de ses propriétés. Il existe plusieurs types de neurones (Pyramide, Panier, Purkinje, etc.) avec des fonctionnements différents (sensoriel, moteur, etc.) et qui sont interconnectés entre eux et forment des réseaux, mais leur organisation structurale est en général la même [12].

Si on revient dans le domaine biologique, les neurones sont composés de dendrites, de l'axone et du corps cellulaire. Les dendrites sont les portes d'entrée du neurone, c'est à cet endroit qu'au niveau de la synapse que le neurone reçoit des signaux tels qu'excitateur ou inhibiteur à partir des autres qui le précède. Le corps cellulaire stocke les petites charges électriques provenant des signaux. Et lorsque la somme des signaux dépassent un certain seuil, le neurone s'active et produit un signal électrique, ainsi l'axone a pour rôle de faire circuler ce signal électrique jusqu'aux terminaisons pour que ce dernier puisse être encore transmis à d'autres neurones (voir figure 8) [25].

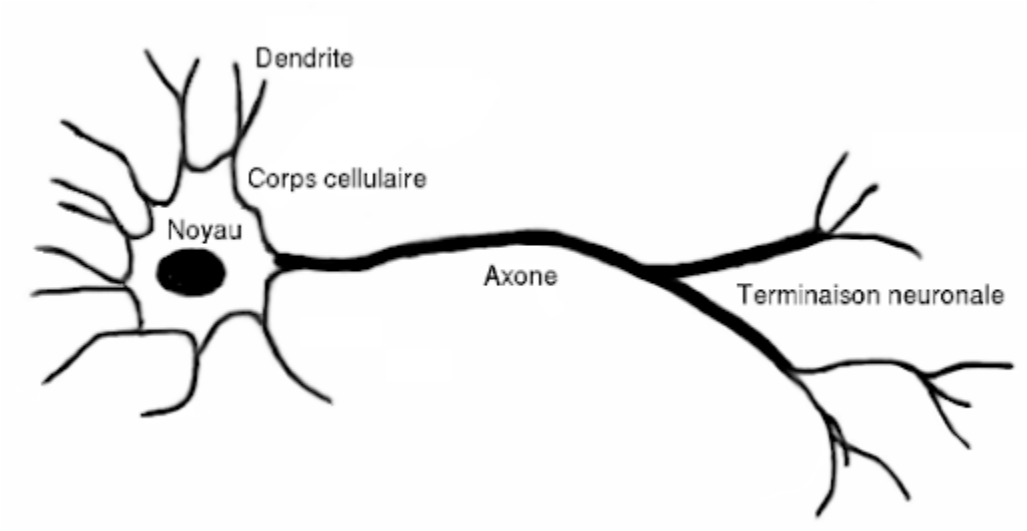


Figure 8: Le neurone biologique (source)

Le premier grand pas vers les réseaux de neurones dans l'IA a été fait en 1943 par McCulloch et Walter Pitts dans un article intitulé « Un calcul logique des idées immanentes à l'activité nerveuse ». Ils ont été les premiers à présenter un modèle mathématique binaire du neurone qu'avec des entrées logiques et sur comme base les éléments de communication du cerveau. En modélisant le neurone biologique, ils se sont basés sur le fait qu'un neurone peut être représenté par une fonction de transfert **f** qui prend en entrée des signaux $\mathbf{x_1, x_2, x_3}$, ….. , et une sortie **y (**voir figure 9) [25].

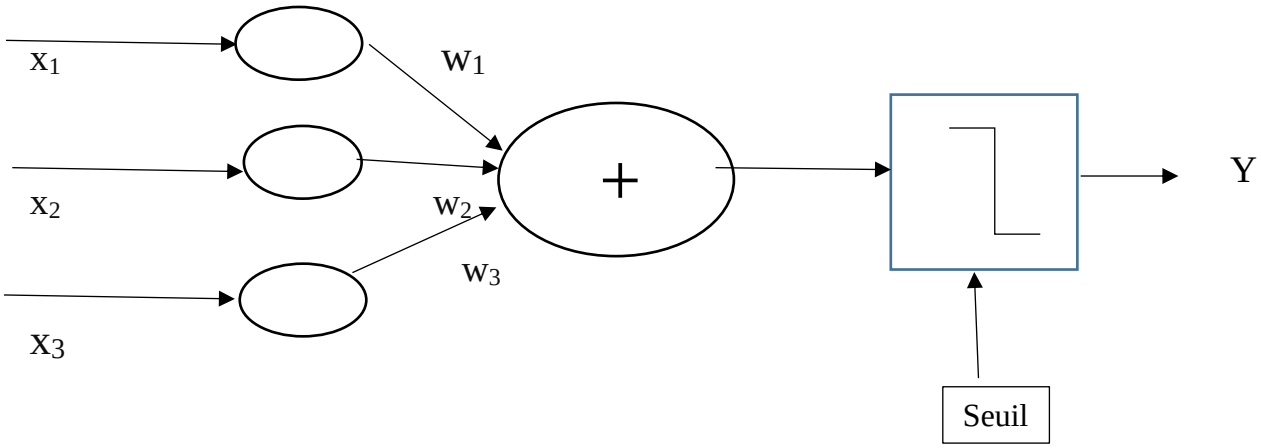


Figure 9: Modèle neurone formel de Warren et Walter

Cette fonction de sortie Y comprend deux étapes :

- **Phase d'agrégation** : correspond à la somme de toutes les entrées et on multiplie chaque entrée par un coefficient w (w_1, w_2, w_3,…) appelé poids qui correspond à l'activité synaptique soit +1 ou -1 ;

$$f = x_1 w_1 + x_2 w_2 + \ldots\ldots + x_n w_n \quad (2.1)$$

- **Phase d'activation :** c'est l'analyse du résultat obtenu précédemment, si celui dépasse un certain seuil généralement 0 alors neurone s'active et retourne une sortie y = 1 sinon le résultat reste à 0 et le neurone ne s'active pas, représentée par la fonction (2.2) suivante :

$$\begin{cases} x = 1 & si\ f \geq 0 \\ x = 0 & sinon \end{cases} \quad (2.2)$$

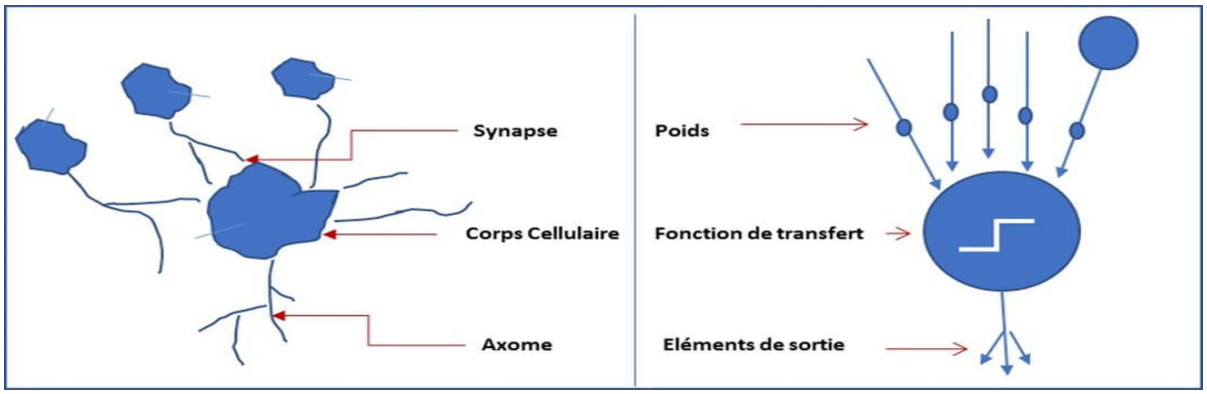


Figure 10: Schéma illustrant la correspondance d'un neurone biologique et un neurone formel (source)

2.4.2 Le perceptron simple

Walter et Warren ont aussi démontré qu'avec ce modèle, on pouvait résoudre certaines opérations logiques telles que la porte AND et la porte OR et lorsqu'on n'interconnecte les neurones en utilisant ces fonctions, il serait possible de résoudre n'importe quel problème logique booléen. Même si ce modèle constitue la base de ce que c'est les réseaux de neurones artificiels, il contient des failles comme quoi qu'il ne dispose pas d'algorithme d'apprentissage et c'est à nous

de mettre à jour les poids w pour son application dans le monde réel. C'est ainsi qu'en 1957 qu'un psychologue américain Frank Rosenblatt invente le modèle à deux couches avec une fonction d'activation par seuil appelé « perceptron » (figure 11) [26].

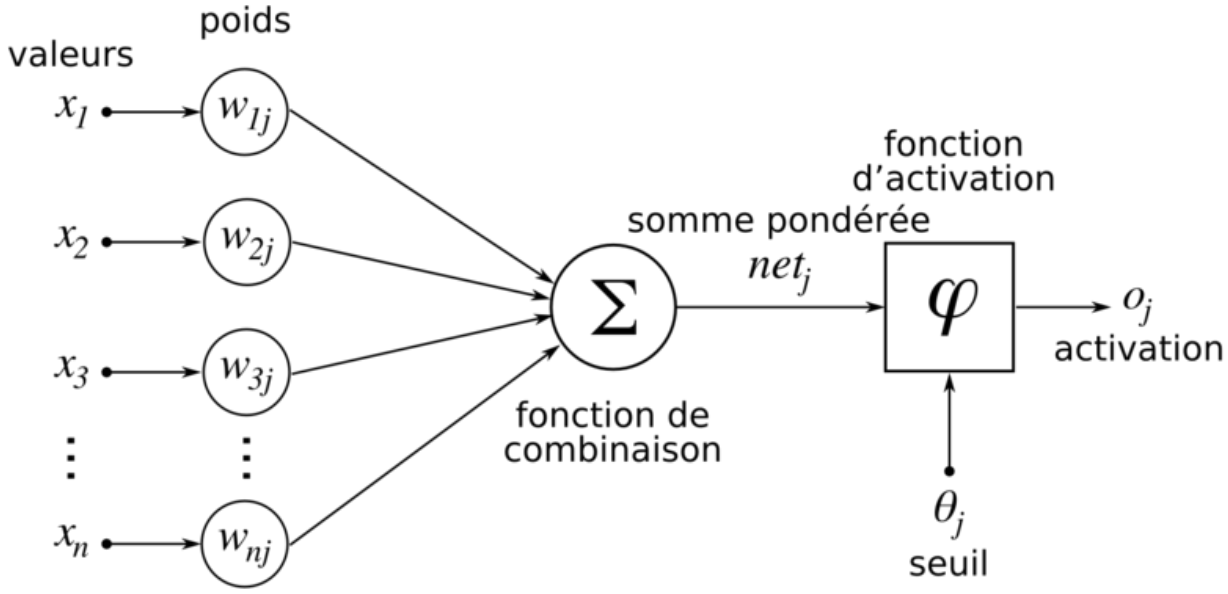


Figure 11: Schéma d'un perceptron simple (source)

Le modèle perceptron est presque identique au modèle précédent mais qui dispose d'un algorithme d'apprentissage binaire et linéaire capable de classer deux types de classes. Avec le perceptron nous avons 4 paramètres : les données d'entrées x, les poids w, la valeur obtenue y et la valeur attendue z. Le principe de l'apprentissage du perceptron est de mettre à jour de manière automatique les paramètres **w** afin d'obtenir la sortie qui nous convient. Pour développer cet algorithme Rosenblatt s'est inspiré de la théorie de Hebb qui dit que : « lorsque deux neurones biologiques sont excités conjointement, alors ils renforcent leur lien synaptique » et a utilisé la fonction d'activation Heaviside. Ainsi les valeurs d'entrées sont multipliées avec leur poids w, le neurone s'active que si les poids calculés des données d'entrées dépassent un certain seuil en général 0. Le résultat obtenu est comparé à la valeur attendue et en cas de différence, l'erreur est rétropropagée afin de mettre à jour les poids jusqu'à avoir le résultat prédit grâce à la fonction (2.3) [26] :

$$w = w_i + \alpha\,(z-y)\,x_i \qquad (2.3)$$

W= le poids corrigé

z = sortie attendue

y = sortie observée

a = le taux d'apprentissage

x_i = l'entrée du poids pour la sortie attendue

w_i = le poids actuel

2.4.3 Les réseaux de neurones multicouches

Pendant un certain moment, l'IA a subit une crise avec un problème majeur que Minsky et Papert ont soulevé en 1969 dans leur ouvrage monumental Perceptrons : « Une introduction à la géométrie computationnelle », comme quoi que les problèmes logique XOR ne sont pas linéairement séparable. Rappelez-vous que McCulloch et Pitts ont prouvé qu'un certain nombre de fonctions logiques peuvent être calculées avec un réseau de neurones. Il s'avère que les perceptrons, malgré les représentations particulières des données qu'ils présentent, ne sont que des classificateurs linéaires. La procédure d'apprentissage du perceptron est remarquable, car il est garanti de converger, mais il n'a pas ajouté de capacité en transférant des régularités non linéaires au réseau de neurones, ce qui engendre la limite du perceptron. C'est ainsi qu'n 1986 que Geoffrey Hinton, un des pères de l'IA développa le perceptron multicouche qui est une combinaison de séparateurs linéaires permettant de produire un séparateur global non-linéaire [27].

Le perceptron multicouche est le premier véritable réseau de neurones artificiels. Un modèle multicouche est composée de minimum 3 couches (voir figure 12) que sont la couche d'entrée qui ne contient aucun neurone, la ou les couches cachés constitués de neurones et afin la couche de sortie qui est également formée de neurones. Les neurones sont reliés entre eux par des connexions pondérées. Ce sont les poids de ces connexions qui gouvernent le fonctionnement du réseau et "programment" une application de l'espace des entrées vers l'espace des sorties à l'aide d'une transformation non linéaire. Le choix du nombre de couches et de nombre de neurones est arbitraire ; plus on en met, plus nous obtenons des résultats complexes et intéressants.

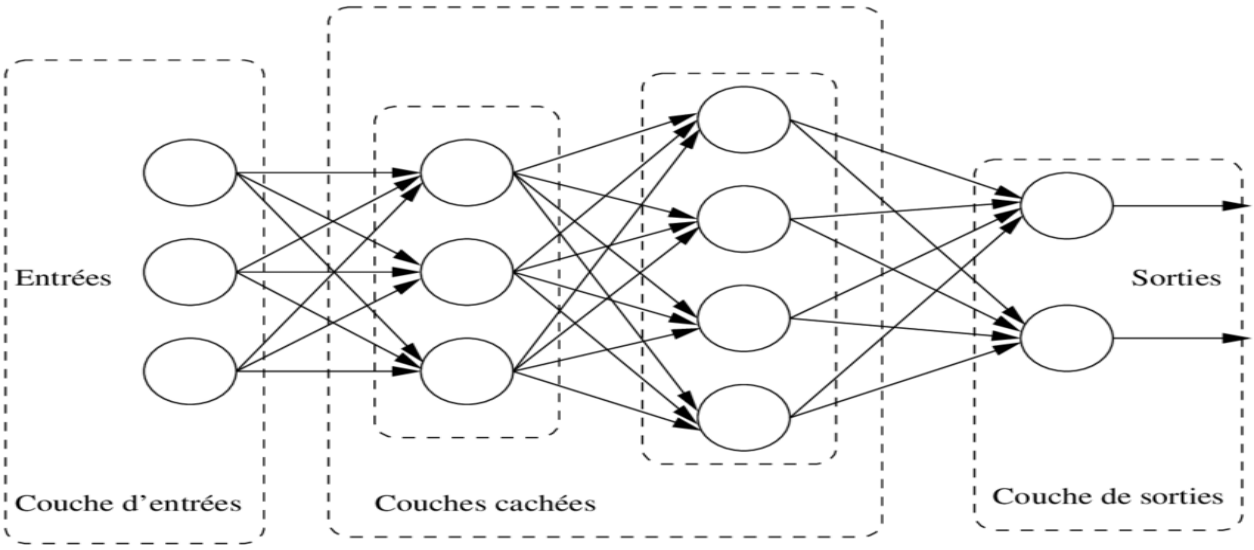


Figure 12: Structure d'un réseau de neurones multicouche (source)

Le principe des MLP est basé sur l'approche du BackPropagation (BP) qui consiste à déterminer comment la sortie du réseau varie en fonction des paramètres présents dans chaque couche tels que le poids w et le biais b de telle sorte à minimiser les erreurs. Il se réalise en calculant une chaine de gradient tout indiquant comment la sortie varie de la dernière couche et puis comment la dernière couche varie en fonction de l'avant dernière et ainsi de suite. Pour développer un MLP, il y a 4 étapes à répéter en boucle [26]:

- **Forward Propagation :** circulation des données de la première couche jusqu'à la dernière couche pour obtenir une sortie y,
- **Calculer la fonction coût :** calculer la vraisemblance du résultat obtenu et celui prédit,
- **Back Propagation :** mesurer comment cette fonction coût varie en fonction des couches en partant de la sortie à l'entrée,
- **Corriger chaque paramètre** grâce à l'algorithme de la Descente de Gradient.

2.5 Le Deep Learning appliqué à la classification

2.5.1 Définition

Les MLP ont connu des limites du fait qu'ils requièrent une grande puissance de calcul et un accès à une grande quantité de données. Ainsi le suivi de certains chercheurs sur les réseaux de

neurones comme Yann LeCun a engendré une étude sur ce qu'on appelle le Deep Learning ou l'apprentissage profond, qui est devenue aujourd'hui la branche phare de l'intelligence artificielle. Il a été élaboré vers la fin des années 80 mais il a connu une explosion qu'en 2012 lors de la compétition annuelle de reconnaissance d'image (ImageNet) où les meilleurs algorithmes s'affrontent. Le Deep Learning se base sur les réseaux de neurones artificiels imitant le cerveau humain. Cette structure est disposée en plusieurs couches, interconnectées entre elles. À chaque information intégrée, les connexions entre neurones s'étendent et se modifient. C'est pour cela qu'un système avec un IA à apprentissage profond a la capacité d'apprendre de nouvelles choses en autonomie. Il améliore également de lui-même ses prévisions et ses prises de décision, sans aucune intervention humaine. Il a donc pour particularité d'apprendre de ses propres erreurs. La révolution du Deep Learning est liée à la puissance des ordinateurs qui ne cesse d'augmenter grâce à l'apparition des GPU et qui permet la création et l'entraînement de réseaux neuronaux avec des dizaines de couches cachées. L'essor du deep Learning vient également de la quantité de données qui ne cesse de s'accumuler [28].

2.5.2 Les modèles du Deep Learning

Les modèles du Deep Learning sont construits selon les types d'apprentissages que ce soit supervisé ou non supervisé. Nous allons étudier quelques modèles tels que les CNN, les RNN, les autoencoders, les GAN.

a. Les réseaux de neurones convolutifs (CNN)

Les CNNs sont les premières variantes des MLP développés dans les années 90 par le célèbre Yann LeCun. Le principe de base pour les CNN est le traitement et la reconnaissance d'image qui s'effectue pour des raisons d'améliorer la qualité de l'image, détecter les formes et les classifier. Les limites du MLP sur le traitement d'image telle que la perte de la sémantique de l'image lors de sa transformation et le fléau de sa dimension ont suscités cette nouvelle méthode du deep Learning. La conception des réseaux de neurones convolutifs suit la découverte de mécanismes visuels dans les organismes vivants. Depuis 1968, des travaux ont montré chez l'animal que le cortex visuel contient des arrangements complexes de cellules, responsables de la détection de la lumière dans les sous-régions du champ visuel qui se chevauchent, appelés champs réceptifs. En effet Leur fonctionnement est inspiré par les processus biologiques [29].

❖ **Architecture des CNN**

L'architecture d'un CNN est divisée en deux parties (figure 13):

- La première partie correspondant aux modules de convolution pour l'extraction des caractéristiques en appliquant des opérations de filtre par convolution. La première couche filtre l'image avec plusieurs noyaux de convolution, et renvoie des « features maps », qui sont ensuite normalisées avec une fonction d'activation en général ReLu. Ce procédé peut être réitéré plusieurs fois: on filtre les features maps obtenues avec de nouveaux noyaux, ce qui nous donne de nouvelles features maps à normaliser et redimensionner, et qu'on peut filtrer à nouveau, et ainsi de suite. Finalement, les valeurs des dernières « features maps » sont concaténées dans un vecteur. Ce vecteur définit la sortie du premier bloc, et l'entrée du second [29].
- La deuxième partie est la classification contenant une connexion de neurones artificiels. Les valeurs du vecteur en entrée obtenues dans le bloc précédent, sont transformées (avec plusieurs combinaisons linéaires et fonctions d'activation) pour renvoyer un nouveau vecteur en sortie. Ce dernier vecteur contient autant d'éléments qu'il y a de classes : l'élément i représente la probabilité que l'image appartienne à une classe. Chaque élément est donc compris entre 0 et 1, et la somme de tous vaut 1. Ces probabilités sont calculées par la dernière couche de cette partie (et donc du réseau de neurones), qui utilise une fonction logistique (classification binaire) ou une fonction softmax (classification multi-classe) comme fonction d'activation [29].

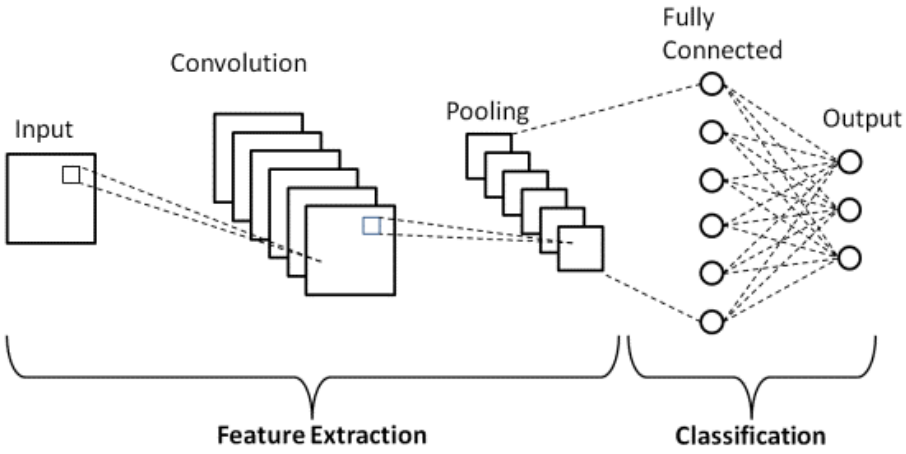


Figure 13: Architecture d'un réseau de neurones convolutifs (source)

❖ **Les couches de l'architecture d'un CNN**

La figure (14) illutre les différentes couches de l'architecture d'un CNN qui sont au nombre de 3 : la couche convolution, la conche de pooling ou down-subsampling et la couche fully-connected ou classificaion .

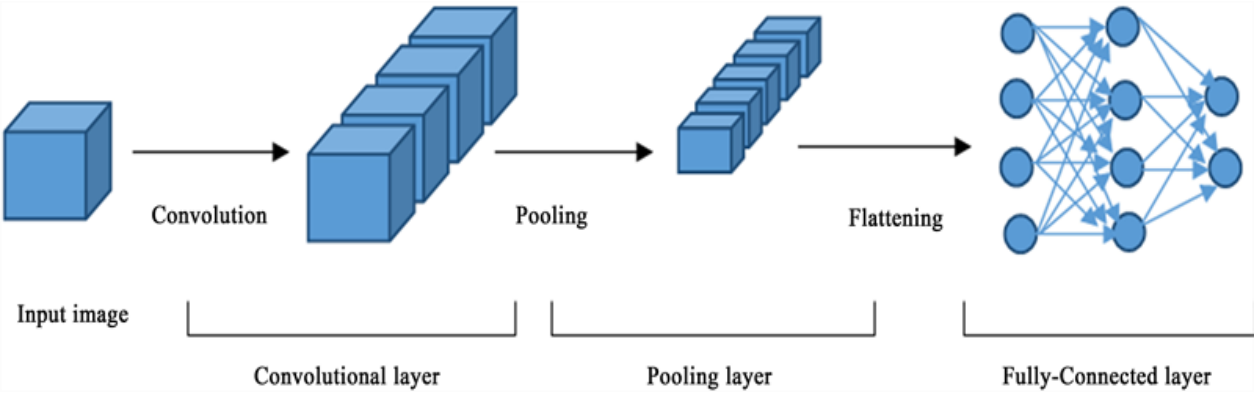


Figure 14: Schéma spécifiant les trois couches du CNN

- La couche convolution : c'est la couche de construction de base des CNN. Cette couche est là où les images sont traduites en des cartes de caractéristiques de données par l'application d'un noyau de convolution ou encore appelé kernel en anglais. Ainsi, la couche de convolution est juste une convolution d'image de la couche précédente, où les poids spécifient le filtre de convolution. Ils sont initialisés puis mis à jour par rétropropagation du gradient. Les trois hyperparamètres qui permettent de dimensionner le volume de la couche de convolution sont : la Profondeur qui détermine nombre de noyaux de convolution, le Pas qui contrôle le chevauchement des champs récepteurs et le Padding pour mettre des zéros à la frontière du volume d'entrée [30]. La figure (15) montre le parcours du filtre sur une image :

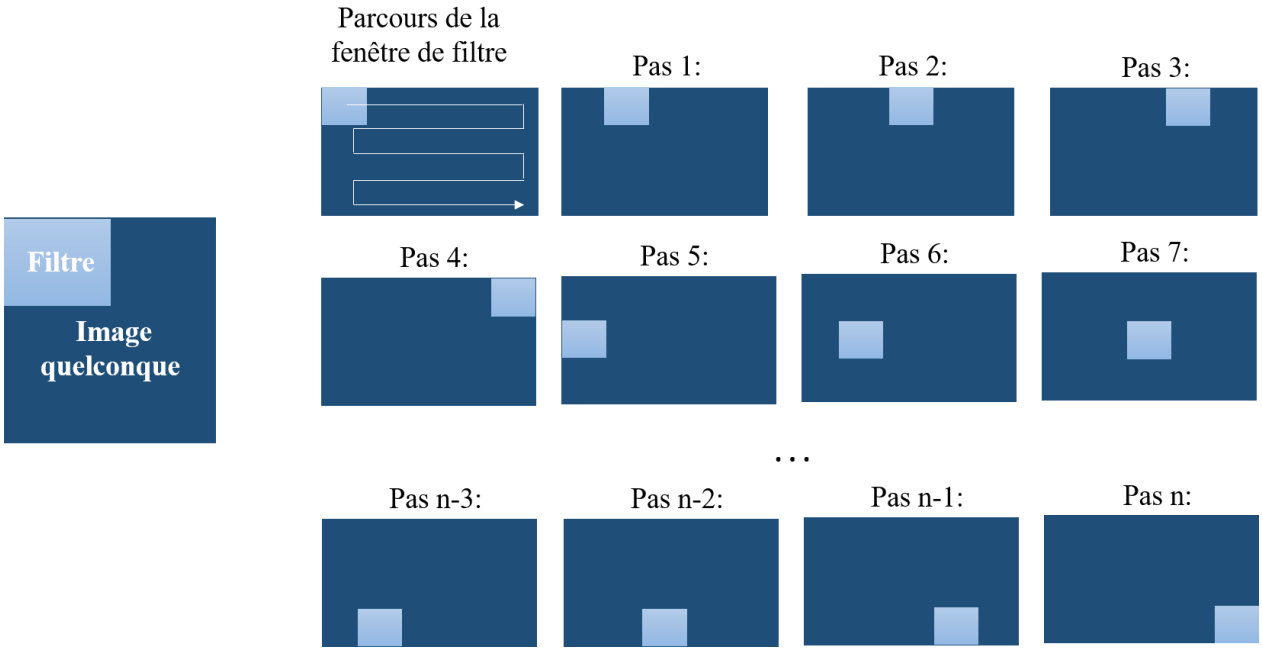


Figure 15: Parcours de la filtre de convolution

- La mise en commun (pooling), ou sous-échantillonnage, est effectuée sur la sortie convolutive. Pendant la mise en commun, un filtre se déplace sur la sortie convolutive pour prendre soit la moyenne appelée average pooling qui résume les éxcitations de neurones voisins par l'excitation moyenne. Soit la valeur maximale appelée max pooling qui résume les éxcitations de neurones voisins par l'excitation maximale. L'objectif de la couche de mise en commun est de réduire progressivement la taille spatiale de la matrice afin de diminuer le nombre de paramètres et contrôler l'ajustement (voir figure 16) [30].

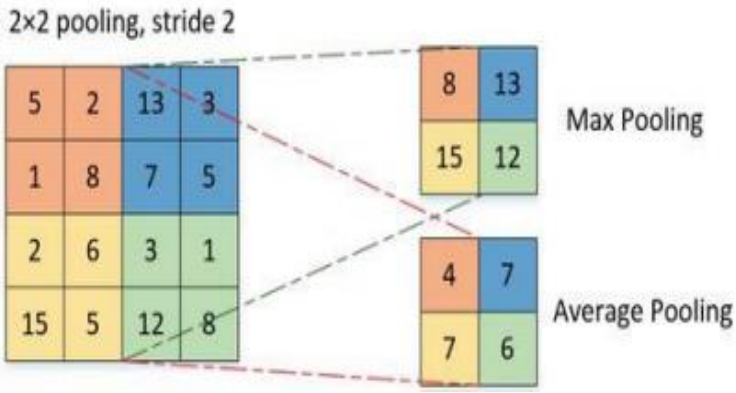


Figure 16: Exemple de moyen et Max Pooling avec un filtre 2 x 2 en pas de 2 (source)

- La couche full-connected : Elle est toujours la dernière couche d'un réseau de neurones. L'objectif fondamental d'une couche entièrement connectée est de prendre les résultats des processus de convolution et de pooling et de les utiliser pour classer l'image dans une étiquette. Cette couche reçoit un vecteur en entrée et produit un nouveau vecteur en sortie. Pour cela, elle applique une combinaison linéaire puis éventuellement une fonction d'activation aux valeurs reçues en entrée, dont la plus utilisée est le softmax car elle permet d'obtenir des valeurs de probabilité entre 0 et 1. La normalisation par lots est utilisée pour améliorer la vitesse d'entraînement et réduire le surajustement. Chaque valeur du tableau en entrée "vote" en faveur d'une classe. Les votes n'ont pas tous la même importance : la couche leur accorde des poids qui dépendent de l'élément du tableau et de la classe. Ce traitement revient à multiplier le vecteur en entrée par la matrice contenant les poids. Le fait que chaque valeur en entrée soit connectée avec toutes les valeurs en sortie explique le terme fully-connected. Le réseau de neurones convolutifs apprend les valeurs des poids de la même manière qu'il apprend les filtres de la couche de convolution lors de phase d'entraînement, par rétropropagation du gradient. La couche fully-connected détermine le lien entre la position des features (caractéristiques) dans l'image et une classe. En effet, le tableau en entrée étant le résultat de la couche précédente, il correspond à une carte d'activation pour une caractéristique donnée : les valeurs élevées indiquent la localisation (plus ou moins précise selon le pooling) de cette caractéristique dans l'image. Si la localisation d'une caractéristique à un certain endroit de l'image est caractéristique d'une certaine classe, alors on accorde un poids important à la valeur correspondante dans le tableau.

❖ **Applications des CNNs**

Ils ont des applications dans la reconnaissance d'image et vidéo , les systèmes de recommandation , classification d'images , segmentation d'images , analyse d'images médicales, le traitement du langage naturel , interfaces cerveau-ordinateur et financières séries chronologiques .

b. Les réseaux de neurones récurrents (RNNs)

Jusqu'à présent nous avons connu que les réseaux de neurones feedforward que sont les MLP et les CNN. Mais ces approches d'aprentissage présentent des limites du fait qu'ils ne disposent que des vecteurs d'entrée et de sorties fixes, des nombres fixes d'étapes de calcul c'est-à-dire un nombre de couches fixes et ils sont incapables de modéliser des données de séries temporelles. C'est ainsi que dans les années 80 que les réseaux de neurones récurrents ont été inventés pour relever ces limites et surtout sur le problème de la modélisation des données séquentielles. Un RNN est similaire au réseau de neurones feedforward mais la différencence en est qu'un RNN est constitué d'une interconnexion de neurones qui inteagissent de façon non-linéaire tout en formant un cycle, car les sorties (de la couche de sortie et/ou de la couche cachée) calculées à l'instant **t** sont réinjectées en entrée du réseau et/ou en entrée de la couche cachée. En effet on peut en théorie conserver dans le réseau en mémoire de ce qui s'y est passé depuis le début car ils utilisent leur mémoire interne pour chaque séquence de données et les sorties précédentes sont utilisées comme entrées supplémentaires, ils sont parfaitement adaptés au traitement de données séquentielles**.** La structure d'un RNN est représentée par la figure (17) et une fonction de sortie par (2.4).

Nous distinguons deux types de structure RNN développés par Jordan et Elman. Jordan utilise un vecteur d'état qui contient les copies des activations de la couche de sortie du réseau à l'instant précédent. Quant à Elman il est basé sur le même modèle que Jordan, mais il utilise les copies des activations de la couche cachée à l'instant précédent pour le vecteur d'état [31].

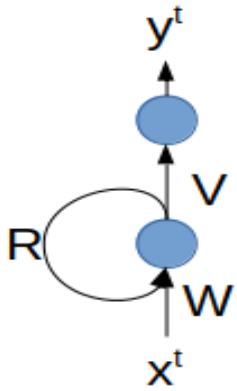


Figure 17: Structure d'un RNN simple (source)

Fonction de sortie d'un RNN : $y(t) = \sigma\,(Wx \,.\, X(t) + Wy \,.\, y\,(t\text{-}1) + b)$ *(2.4)*

Y(t) : sortie du réseau

ϭ : fonction d'activation

Wx : poids à l'instant t

Xt : l'entrée à l'instant t

Wy : deuxième pois succèssif

Y(t-1) : sortie de l'ittération précédende

b : le biais

Qu'elle soit détaillée ou simplifiée, la représentation d'un réseau récurrent n'est pas aisée, car il est difficile de faire apparaître la dimension temporelle sur le schéma. C'est notamment le cas pour les connexions récurrentes, qui utilisent l'information du temps précédent. Pour solutionner ce problème, on utilise souvent une représentation du réseau "déplié dans le temps", afin de faire apparaître explicitement celui-ci. La figure (18) suivante montre un exemple de réseau RNN déplié :

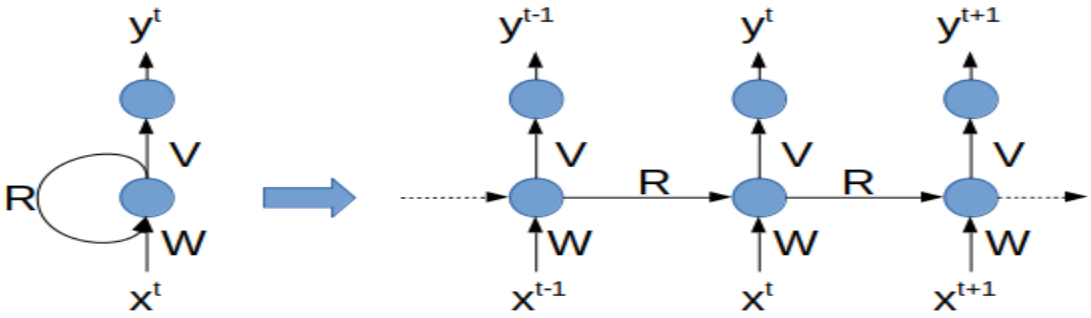


Figure 18: Structure d'un RNN simple (source)

Cette version dépliée dans le temps fait désormais apparaître clairement les variables d'entrée au cours du temps : x^{t-1}, x^t, x^{t+1}, etc. (idem pour la sortie), et l'impact des sorties précédentes sur la sortie courante du réseau.

Malgré que les RNNs sont en principe simples et puissants mais ils sont difficiles à entrainer correctement en pratique. En effet ils n'ont pas la capacité de se souvenir de beaucoup de séquences

au fil du temps dû au problème de l'explosion et à la disparition du Grandient. Pour remédier à ce problème deux autres modèles récurrents sont créés dont les LSTM et les GRU.

❖ Les modèles à Long-Court Terme (LSTM)

Vue la limite de la quantité de séquences que peut se souvenir les RNNs simples, un modèle a été proposé par Hochereiter et Schmidhuber en 1997 pour régler les problèmes de la descente de Grandient et éviter les problèmes de dépendance à long terme. Les LSTMs sont formés d'un ensemble de sous-réseaux connectés de manière récurrente appelé bloque de mémoire. Cette bloque est composée d'une cellule de mémoire pour le stockage des états et d'un ensemble de portes tels que la porte d'entrée pour contrôler ce qu'il faut apprendre, la porte d'oubli pour contrôler ce qu'il faut oublier et enfin une porte de sortie qui contrôle la quantité de contenu à modifier [32].

Cet ensemble de bloques est représenté par les équations suivantes de la figure (19) :

$$f_t = \sigma\,(W_f \,.\, x_t + U_f \,.h_{t-1} + b_f) \qquad (2.5)$$

$$i_t = \sigma\,(W_i x_i + U_i h_{t-1} + b_i) \qquad (2.6)$$

$$\tilde{c}_t = tanh\,(W_c xt + U_c h_{t-1} + b_c \qquad (2.7)$$

$$c_t = ft * c_{t-1} + i_t * \tilde{c}_t \qquad (2.8)$$

$$ot = \sigma\,(W_0 xt + U_o h_{t-1} + b_0) \qquad (2.9)$$

$$h_t = o_t * tanh\,(c_t) \qquad (2.10)$$

Figure 19: Les équations d'un réseau LSTM

Où **σ** est la fonction sigmoïde, $\mathbf{f_t}$ est le vecteur d'activation de porte d'oubli (Forget gate), $\mathbf{i_t}$ vecteur d'activation de porte d'entrée, $\mathbf{o_t}$ vecteur d'activation de porte de sortie, $\mathbf{x_t}$ vecteur d'activation d'entrée de cellule, $\mathbf{c_t}$ état de la cellule, $\mathbf{h_t}$ vecteur de sortie de l'unité LSTM, tous les **w** et **u** sont des poids, **b** est un vecteur de biais et le symbole * pour le produit Hadamard. Poids w, u et les biais **b** doivent être appris durant le processus d'entrainement [32].

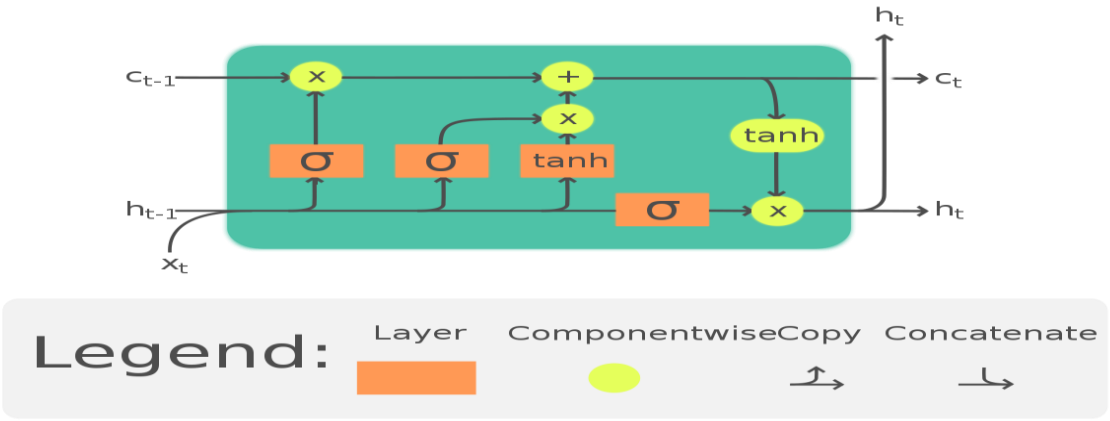


Figure 20: Représentation structural d'un LSTM (source)

La procédure d'un modèle LSTM peut être résumée en 5 étapes :

1. Nous avons une nouvelle valeur xt et une nouvelle valeur du nœud précédent ht-1 qui arrive.
2. Ces valeurs sont combinées ensemble et passent par la fonction d'activation sigmoïde, où il est décidé si la vanne d'oubli doit être ouverte, fermée ou ouverte dans une certaine mesure.
3. Les mêmes valeurs, ou en fait des vecteurs de valeurs, passent en parallèle par une autre opération de couche « tanh », où il est décidé quelle valeur nous allons passer au pipeline de mémoire, et aussi l'opération de couche sigmoïde, où il est décidé, si cette valeur va être transmise au pipeline de mémoire et dans quelle mesure.
4. Ensuite, nous avons une mémoire qui circule dans le pipeline supérieur. Si nous avons la vanne d'oubli ouverte et la vanne de mémoire fermée, la mémoire ne changera pas. Sinon, si nous avons la vanne d'oubli fermée et la vanne de mémoire ouverte, la mémoire sera complètement mise à jour.
5. Enfin, nous avons xt et ht-1 combinés pour décider quelle partie du pipeline de mémoire va devenir la sortie de ce module.

❖ **Les Unités recurrentes fermées (GRU)**

Un modèle GRU est une autre solution pour les problèmes des réseaux de neurones récurrents simples. Il a été introduit en 2014 par Cho et Al pour résoudre aussi les problèmes de disparition du Grandient mais avec une structure simple en utilisant moins de paramètres que les LSTM. Les GRU disposent d'une porte de mise à jour en combinant la porte d'entrée et d'oubli des LSTM et une porte d'initialisation qui correspond à la sortie [32]. Les équations de base sont les suivantes, regroupées dans la figure (21) et sa structure illustrée par la figure (21)

$$Z_t = \sigma\,(W_z.x_t + U_z.h_{t-1} + b_z \qquad (2.11)$$

$$r_t = \sigma\,(W_r x_t + U_r h_{t-1} + b_r \qquad (2.12)$$

$$\tilde{h}_t = tanh\,(W_h x_t + U_h\,(r_t * h_{t-1}) + b_h \qquad (2.13)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \qquad (2.14)$$

Figure 21: Les équations d'un modèle GRU

Où **σ** est la fonction sigmoïde, $\mathbf{z_t}$ est le vecteur d'activation de la porte de mise à jour, $\mathbf{r_t}$ le vecteur d'activation de la porte de réinitialisation, $\tilde{h}_t$ le vecteur candidat et $\mathbf{h_t}$ est le vecteur output de l'unité GRU. **w** et **u** sont des poids, **b** est le vecteur biais (poids et biais sont à entrainer durant le processus d'apprentissage) et le symbole ***** pour le produit de Hadamard [32].

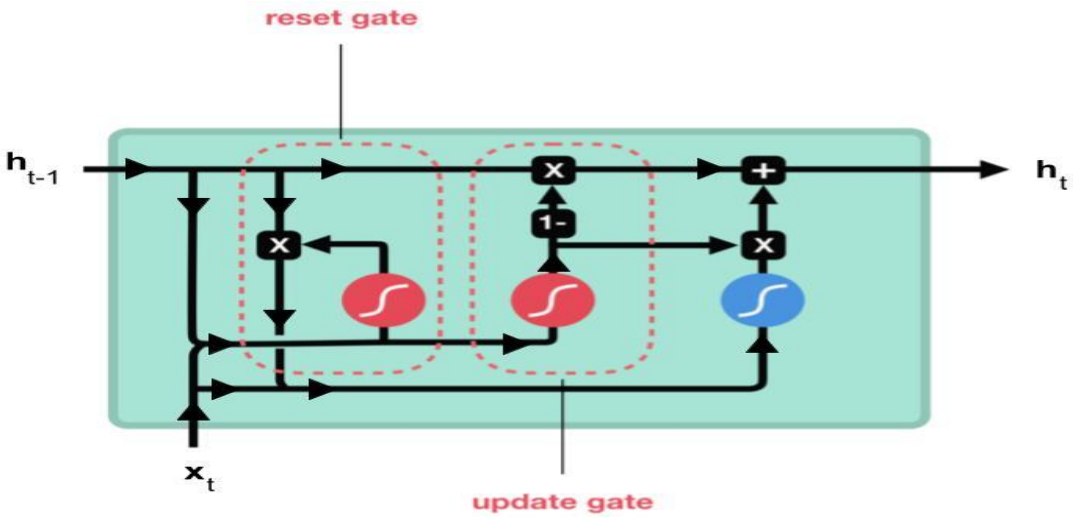


Figure 22: Représentation d'un GRU (source)

❖ **Applications des RNN**

Les réseaux de neurones récurrents sont adaptés pour des données d'entrée de taille variable. Ils conviennent en particulier pour l'analyse de séries temporelles. Ils sont utilisés en reconnaissance automatique de la parole ou de l'écriture manuscrite, plus en général en reconnaissance de formes , ou encore en traduction automatique. Les RNN sont appliqués au suivi et à la surveillance: systèmes de prévision du trafic basés sur les données à l'aide de Graph RNN convolutifs (GCRNN).

c. Les auto-encodeurs (AE)

Les modèles du Deep Learning que nous avons étudié utilisent des données étiquettées et sont tous basés sur l'apprentissage supervisé. Ainsi la question posée est comment faire lorsqu'on dispose d'une très grande quantité de données non étiquées ? Pour répondre à cette question nous disposons un modèle sous le nom d'auto-encodeur utilisé pour l'apprentisage non supervisé. Un auto-encodeur est une variante du réseau de neurones classiques avec une particularité dans le sens où sa couche d'entrée et sa couche de sortie comportent exactement le même nombre de neurones. Les autoencodeurs sont des réseaux de neurones optimisés pour convertir l'entrée en une version d'elle-même via une couche cachée agissant comme un code appris. Ils sont spécifiquement conçus pour déduire les caractéristiques utiles des données en apprenant à ignorer le bruit et en se concentrant sur les modèles généralisables pertinents [33].

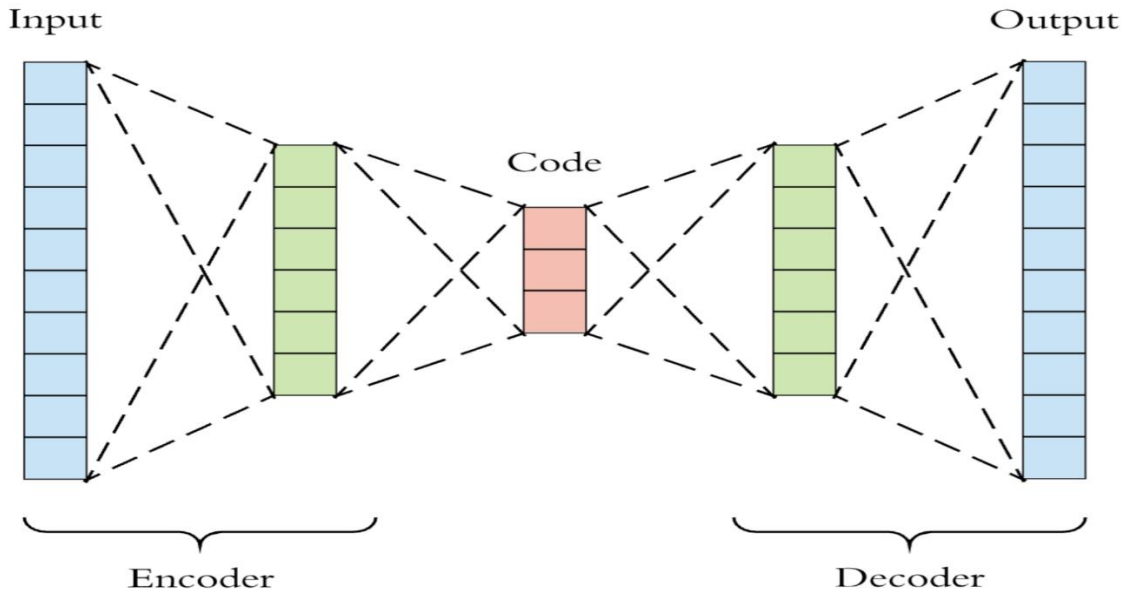


Figure 23: Architecture d'un auto-encodeur (source)

Cette technique d'auto-encodeur est divisée en deux parties (voir figure 23) : l'encodeur et le décodeur . L'encodage correspond à la phase dans laquelle le modèle apprend à réduire les dimensions d'entrée et à compresser les données d'entrée dans une représentation codée. Et dans la phase de décodage le modèle apprend à reconstruire les données à partir de la représentation codée pour être aussi proche que possible de l'entrée d'origine. Entre ces deux parties , il dispose d'une structure interne appelée code ou Boottleneck ou encore espace latent implementé dans les couches cachées, ce qui signifie que le nombre de couches internes est inférieure au nombre de couches de l'entrée et de la sortie. Ce code permet une compréssion des données ou représentation de dimension réduite des données. Les auto-encodeurs peuvent être représentés avec l'utilisation des fonctions mathématiques. La segmantation des deux parties est représentée par les fonctions suivantes (figure 24) [34] :

$$Ø : X \rightarrow F \qquad (2.15)$$

$$\Psi : F \rightarrow X \qquad (2.16)$$

$$Ø , \psi = arg\ min\ Ø,\psi\ ||\ X - (\psi\ o\ Ø)\ X\ ||^2 \qquad (2.17)$$

Figure 24: Architecture d'un auto-encodeur

Ψ: fonction de la partie encodeur

ϕ : fonction de la parie décodeur

X : données d'entrées

F : l'espace latent

Φ, ψ : transition de l'encodeur au décodeur

Ainsi pour chaque partie une équation est définie :

- Pour le réseau d'encodage :

$$Z = \sigma\ (Wx + b) \qquad (2.18)$$

- Pour le réseau de décodeur

$$x' = \sigma'\ (\ W'z + b') \qquad (2.19)$$

z est la dimension latente, **w** le poids , **b** le biais , **Ϭ** la fonction d'activation, **x'** données décompressées avec un poids, une fonction d'activation et un biais différents du réseau d'encodage. Toutefois nous avons aussi une fonction d'erreur appelée LOSS qui représente la différence entre la fonction d'entrée et la fonction de sortie.

Fonction LOSS :

$$L\,(x,x') = ||x\text{-}x'||2 = ||x - \sigma'\,(W'\,(\sigma(Wx + b\,)) + b'\,)||^2 \qquad (2.20)$$

En effet les auto-encoderus sont liés à l'analyse des composants principales dont l'approche est la réduction de dimensionnalité linéaire qui fonctionne en trouvant les principaux composants ou bases qui conduisent à la moindre erreur de reconstruction sur l'ensemble de données. Les auto-encodeurs sont une approche non linéaires, essayant à nouveau de découvrir le code qui minimise l'erreur de reconstruction des données. En fait, si le décodeur est linéaire et que la fonction de perte est l'erreur quadratique moyenne, un auto-encodeur déduit le même sous-espace que le PCA. En d'autres termes, dans ces circonstances, l'autoencodeur découvrira le sous-espace principal des données d'apprentissage, un peu comme le PCA mais ils utilient en générale des fonctions d'activation non- liénaires comme ReLu ou Sigmoïd [35].

❖ **Application des auto-encoders**

Les auto-encodeurs sont beaucoup utilisés dans la cybersécurité notament dans la détection d'anomalies sur les données. Cependant, si vous avez des données d'entrée corrélées, la méthode d'encodeur automatique fonctionnera très bien car l'opération d'encodage repose sur les fonctionnalités corrélées pour compresser les données.

d. Les réseaux accusatoires génératifs (GAN)

La promesse de l'apprentissage profond est de découvrir des modèles riches et hiérarchiques qui représentent la probabilité distributions sur les types de données rencontrées dans les applications d'intelligence artificielle, telles que des images, des formes d'onde audio contenant de la parole et des symboles dans des corpus en langage naturel. Les GAN sont une classe d'algorithmes d'apprentissage non supervisé devenues un des prometteurs de l'intelligence artificielle introduits par Goodfellow et al. en 2014.

Les réseaux d'accusatoires sont conçus pour résoudre des jeux d'imitation des données, qu'il s'agisse de texte, d'images, de sons ou de vidéo basés sur la concurrence entre deux réseaux. Les GAN sont composés de deux réseaux qui se complètent pour analyser, capturer et copier les variations au sein d'un ensemble de données. Les deux modèles de réseau qu'ils entrainent sont à savoir le générateur et le discriminateur. Le générateur apprend la distribution des données et est entrainé à augmenter la probabilité que le réseau de discrimination fasse des erreurs en faisant passer du bruit aléatoire. Quant au discriminateur, il estime la probabilité que l'échantillon qu'il a obtenu provienne des données d'apprentissage mais pas du générateur, donc il sert à faire la classification entre les vrais et faux échantillons. Nous notons que les deux réseaux des GAN sont tous des perceptrons multicouches et ils utilisent la fonction d'erreur LOSS et le mécanisme de la descente de Grandient pour la BackPropagation [36].

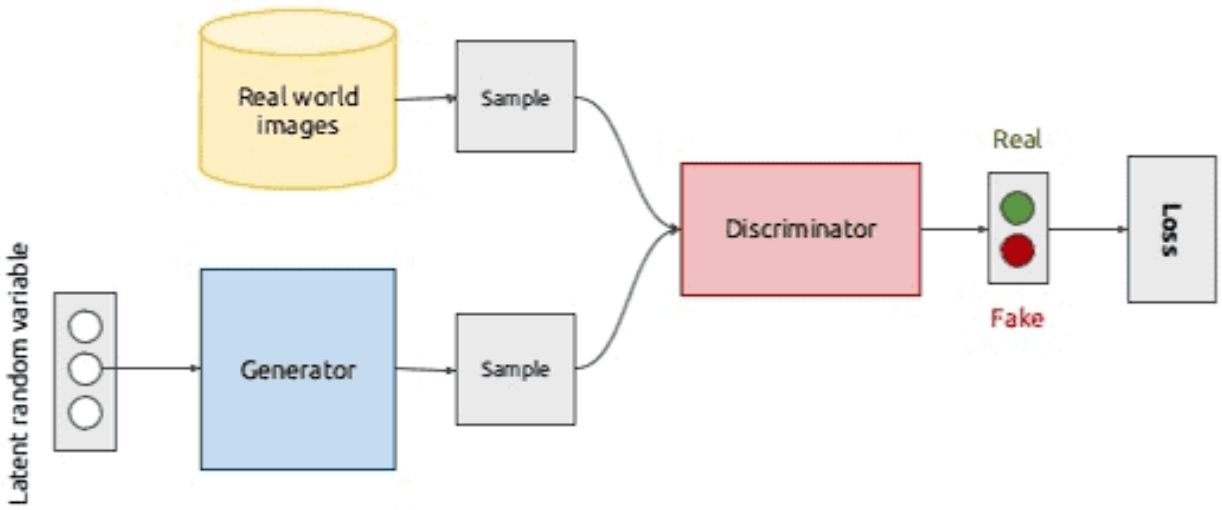


Figure 25: Schéma représentatif d'un modèle GAN (source)

Le jeu entre les deux réseaux, générateur et discriminateur, peut être considéré comme un jeu entre le minimum et le maximum modélisé par la fonction V (D,G) suivante [36]:

$$\min_G \max_D V(D, G) = E_{x \sim pdata(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (2.21)$$

Pour apprendre la distribution P_g du générateur sur les données x, nous définissons un a priori sur variables de bruit d'entrée P_z (z), puis représentent un mappage à l'espace de données

comme G (z; θ g), où G est une fonction dérivable représentée par un perceptron multicouche de paramètres θg. Nous définissons également un deuxième perceptron multicouche D(x; θ d) qui produit un seul scalaire. D(x) représente la probabilité que x provenait des données plutôt que P_g .Nous entraînons D pour maximiser la probabilité d'attribuer l'étiquette correcte aux exemples d'apprentissage et aux échantillons de G. Nous entraînons simultanément G pour minimiser *log (1 – D (G(z)))*.

- **Applications des GAN**

Les GAN peuvent être utilisés pour le traitement d'images en générant des modèles 3D à l'aide d'images 2D d'objets sous plusieurs perspectives et peuvent aussi créer des images réalistes à partir de descriptions textuelles d'objets simples. Les GAN peuvent avoir une application potentielle dans la sécurité de l'IoT car ils peuvent apprendre différents scénarios d'attaque pour générer des échantillons similaires à une attaque zero-day et fournissent des algorithmes avec un ensemble d'échantillons au-delà des attaques existantes.

3 Conclusion

Dans ce chapitre, nous avons fait une étude des différentes méthodes de classification de données qu'offre l'IA. Pour cela, nous avons d'abord donné quelques éléments introductifs sur l'IA : définitions, branches, etc. Ensuite, nous avons fait une discussion sur les modèles de classification IA par rapport à ses différents sous branches : Machine Learning, réseau de neurones, et Deep Learning. Cette discussion nous a permis de mieux comprendre les modèles les plus adaptés par rapport à notre problème et d'avoir leurs critères d'applicabilité.

Chapitre 3 : Etude ce cas, détection d'anomalies dans le trafic réseau d'un réseau d'un Smart Home

1 Introduction

Une maison intelligente est née de la combinaison de l'internet des objets (IoT) et de la domotique, dont la maison intelligente est considérée comme son successeur. En effet, la domotique est le mariage de l'informatique et des télécommunications au service de la maison et de l'environnement. Une maison intelligente n'est rien d'autre que l'automatisation des tâches de cette maison dont les différents éléments sont pilotables à distance grâce à l'avènement des nouvelles technologies telles que les smartphones, ordinateurs portables et les objets connectés. Ces fonctions d'automatisations sont beaucoup axées autour de la sécurité pour déclencher des alarmes en cas de danger, de la communication pour déclencher des activités à des heures fixes et de la gestion de l'énergie notamment avec l'électricité. Partant de ces fonctions d'automatisation, elle offre des avantages sécuritaires, nous permettant de mieux gérer notre temps ainsi que de limiter les dépenses énergétiques. Les données manipulées par les smart homes sont très sensibles et très critiques ce qui pose le problème de la sécurisation de ces derniers. Il devient donc impératif de mettre en place des dispositifs de sécurité pour lutter contre d'éventuelles attaques sur les réseaux de smart home. Cela passe par une surveillance du trafic réseau de smart homes pour détecter d'éventuels disfonctionnement du réseau.

La détection d'anomalie devient une solution adaptive pour résoudre les problèmes de cybersécurité d'une maison intelligente. En effet, cette détection d'anomalie revient à distinguer tous les avènements ou observations suspects et rares appelés données aberrantes qui relèvent des différences avec la majorité des autres données. Cependant, il existe plusieurs types de données avec des caractéristiques différentes, ce qui implique l'existence de différentes méthodes de détection d'anomalie trouvant chacune son efficacité dans un domaine particulier suivant un objectif défini. La plupart de ces méthodes utilisent des seuils de décision en s'appuyant des techniques basées sur la classification, le clustering, la régression, les plus proches voisins et les

outils statistiques. Chaque méthode est évaluée en mesurant ses performances de critères de précision de la détection, de sa rapidité et de son passage à l'échelle par rapport au débit du flux.

Les techniques de détections d'anomalies sont classées suivant 4 catégories [18] :

- **Les techniques statistiques :** elles peuvent être paramétriques ou non-paramétriques dont les modèles construis sont basés sur un intervalle de confiance à partir des données existantes et les données qui ne correspondent pas à ce modèle sont considérées comme anormales.
- **Les techniques basées sur la proximité :** il regroupe la technique basée sur les voisins les plus proches et le clustering. Ils utilisent tous deux l'approche de la distance et celle de la densité. En effet, le clustering classe les données selon leur ressemblance.
- **Les techniques basées sur le Deep Learning :** représentent des algorithmes d'apprentissage en profondeur supervisé et non supervisé pour la détection d'anomalie basée sur la classification
- **Autres techniques :** les autres techniques regroupent les modèles basés sur l'apprentissage automatique et les réseaux de neurones.

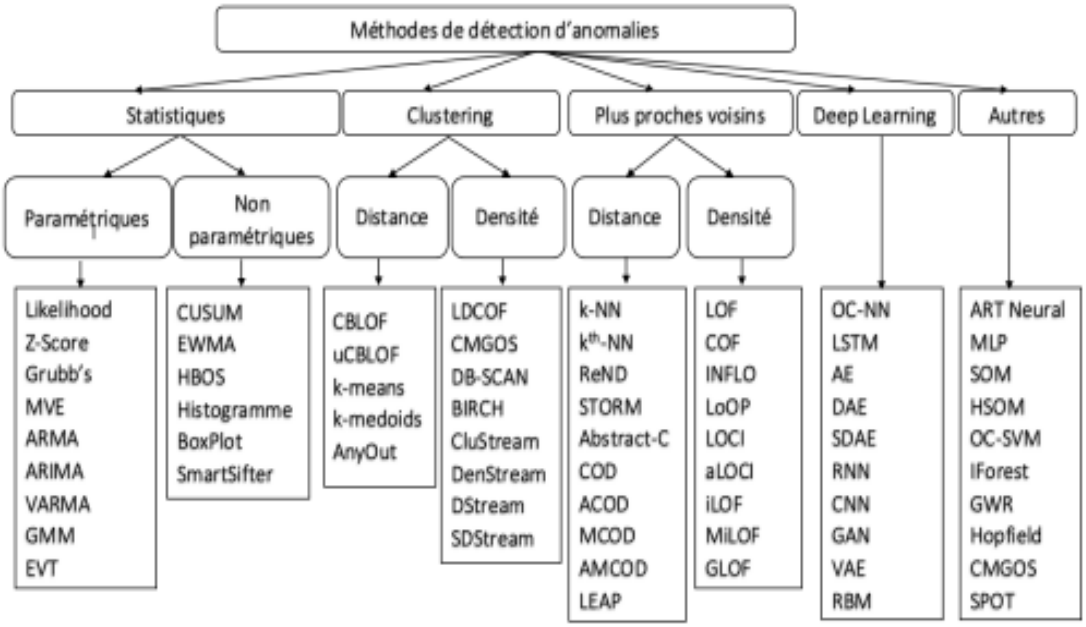


Figure 26: Classification des techniques de détection d'anomalie (source)

La détection d'anomalie est applicable dans tout système qui exploite des données. Cependant, chaque méthode est utilisable dans des domaines bien définis. Elle est applicable dans les domaines de détection d'intrusion, de détection de fraude, de détection de défaillance d'un système, dans la santé, dans la détection des évènements d'un réseau, dans le traitement d'images et de textes, etc.

Pour mettre en place un système de détection d'anomalie pour notre smart home nous allons utiliser les données de trafic réseau de ce dernier afin de concevoir des méthodes IA qui permettent de prédire ou de détecter un trafic malicieux. Pour ce faire nous avons utilisé un ensemble outils pour le traitement et l'analyse des données du réseau et fait une implémentation de plusieurs modèles IA. La suite présente l'ensemble de ces outils et l'implémentation.

2 Présentation des outils utilisés

2.1 Python

Le choix de langage de programmation approprié à notre travail est porté sur Python. C'est un langage de programmation orienté objet créé en 1989 par Guido Van Rossum avec des caractéristiques qui marquent ses performances telles que la portabilité, le dynamisme, l'extensibilité et la modularité. Son utilisation est fréquente dans le développement des modèles d'apprentissage automatique et en profondeur grâce aux nombreuses bibliothèques qu'il détient, ce qui rend plus facile leur mise en œuvre.

2.2 Anaconda

Anaconda est une distribution libre et open source utilisée pour faire des programmes en Python et en langage R pour la science des données et l'apprentissage automatique concernant l'analyse prédictive, le traitement des données et des calculs scientifiques. Il dispose de 250 paquets populaires en science de données et d'un navigateur appelé navigateur anaconda qui est une interface graphique permettant aux utilisateurs d'accéder facilement aux applications ou outils tels que Jupyter Notebook, Visual Studio Code, Studio, Spyder, Jupyter Lab., QtConsol, etc. et de gérer des environnements, des packages des canaux conda sans utiliser la ligne de commande.

2.2.1 Jupyter Notebook

Pour la partie d'apprentissage initiale du travail, c'est l'interface de programmation interactive Jupyter Notebook qui est utilisée, offrant la possibilité de rassembler du texte, des images, des formules mathématiques (en format latex) et du code exécutable. Il est initialement développé pour les langages de programmation Julia, Python et R, d'où le nom Jupyter. Les cellules sont les éléments de base de Jupiter notebook, elles peuvent contenir du code informatique ou du texte au format markdown. Jupyter notebook permet des tests efficaces puisque nous pouvons exécuter les cellules individuellement, et ainsi nous n'avons pas à exécuter l'ensemble du programme à chaque fois qu'une modification est apportée. Il est bénéfique lors de l'exécution l'apprentissage automatique puisque nous n'avons pas à exécuter le prétraitement à chaque fois. Également notebook fournit des sorties visuelles afin que nous puissions voir directement des graphiques ou des tableaux. C'est une interface full web qui s'ouvre directement dans un navigateur existant sur votre machine permettant d'exécuter du code Python et faciliter sa mise en forme et sa compréhension. Il est livré avec comme kernel ou noyau IPython, néanmoins il existe actuellement plusieurs autres noyaux comme Python 3 utilisé dans notre cas.

2.2.2 Environnement de développement

❖ Tensorflow

Tensorflow est un Framework open source publiée par Google Brain (maintenant Google AI) en 2015 pour permettre aux développeurs de créer, de former et de travailler plus facilement avec des modèles et des données d'apprentissage en profondeur pour faire différentes types de prédictions grâce à un écosystème complet et flexible d'outils, de bibliothèques et de ressources communautaires qu'il possède. Initialement c'était un Framework développé pour les calculs numériques mais il n'a cessé de gagner en popularité pour être l'un des plus utilisés dans le développement de l'apprentissage automatique, des réseaux de neurones et du Deep Learning. L'un de ses principales particularités est que les opérations courantes sur les réseaux de neurones sont faits via des tables de données multidimensionnelles appelés Tenseurs (Tensor), d'où le nom Tensorflow. Ces tenseurs incluent également des scalaires, des vecteurs ou des matrices de n dimensions représentant tous les types de données.

❖ Keras

keras est une API d'apprentissage en profondeur de haut niveau écrite en python capable de fonctionner sur TensorFlow ou Theano pour les réseaux de neurones. Il prend en charge plusieurs calculs de réseaux de neurones et facilite leur mise en œuvre. Il a été développé dans le cadre de l'effort de recherche du projet ONEIROS, et son principal auteur et mainteneur est François Chollet, un ingénieur Google. Il fonctionne sans problème sur le CPU et le GPU. Keras prend en charge presque tous les modèles de réseau de neurones et étant de nature modulaire, il est incroyablement expressif, flexible et apte à la recherche innovante. Keras est une API open-source et enveloppe des bibliothèques de calcul numérique efficaces Theano et tensorflow et il permet de définir et d'entraîner un réseau de neurones en quelques lignes de code. Il minimise les actions de l'utilisateur requises pour les cas d'utilisation courants et fournit un retour clair en cas d'erreur de l'utilisateur, cela rend Keras facile à apprendre et à utiliser.

2.2.3 Les librairies

Dans Anaconda nous disposons plusieurs librairies qui facilitent le traitement de données et l'implémentation de modèle IA. Dans notre cas nous utilisons les librairies suivantes :

- ❖ NumPy et Pandas lors du prétraitement de l'ensemble de données. NumPy permet de manipuler des tableaux multidimensionnels et peut effectuer des opérations mathématiques rapides sur eux. Pandas nous fournit la structure de trame de données, nous permettant de convertir facilement vers et depuis CSV des dossiers. NumPy et Pandas fonctionnent tous deux efficacement sur des données volumineuses.
- ❖ Pour la phase d'apprentissage du projet, nous avons la bibliothèque Scikit-learn, qui est une bibliothèque open source qui fonctionne avec NumPy et Pandas. Il propose une large gamme d'outils et est simple à utiliser, offrant tout le nécessaire pour l'apprentissage dans ce projet. Il a été aussi utilisé pour la standardisation des données d'entrainement et de test avec la fonction StandardScaler.
- ❖ Egalement la bibliothèque matplolib est utilisée pour tracer des graphiques et visualiser les données avec ses sous-bibliothèques pyplot qui crée une interface proche du logiciel commercial Matlab qui contient des fonctions très similaires à celui-ci et seaborn pour la visualisation de données, spécialisée dans l'analyse statistique.

3 Installation des outils

Nous avons commencé par télécharger le Framework anaconda à partir de ce lien : https://www.anaconda.com/products/individual, et on obtient la page suivante :



Figure 27: Téléchargement d'anaconda

Une fois que cette page est affichée, nous cliquons sur Download. Ici le choix est directement centré pour le système d'exploitation Windows, donc pour le télécharger sur MacOs il faut cliquer sur l'image de MacOs et il va directement lancer le programme de téléchargement pour MacOs. Pour notre cas nous avons téléchargé pour Windows de 64-Bit qui vient avec la version 3.9 de python. Dès que le téléchargement est fini, nous avons lancé le fichier exécutable et l'application peut être ouverte sous le nom d'Anaconda Navigator à partir du bouton démarrer, s'il n'y est pas il faut le chercher à partir de la barre des recherches.

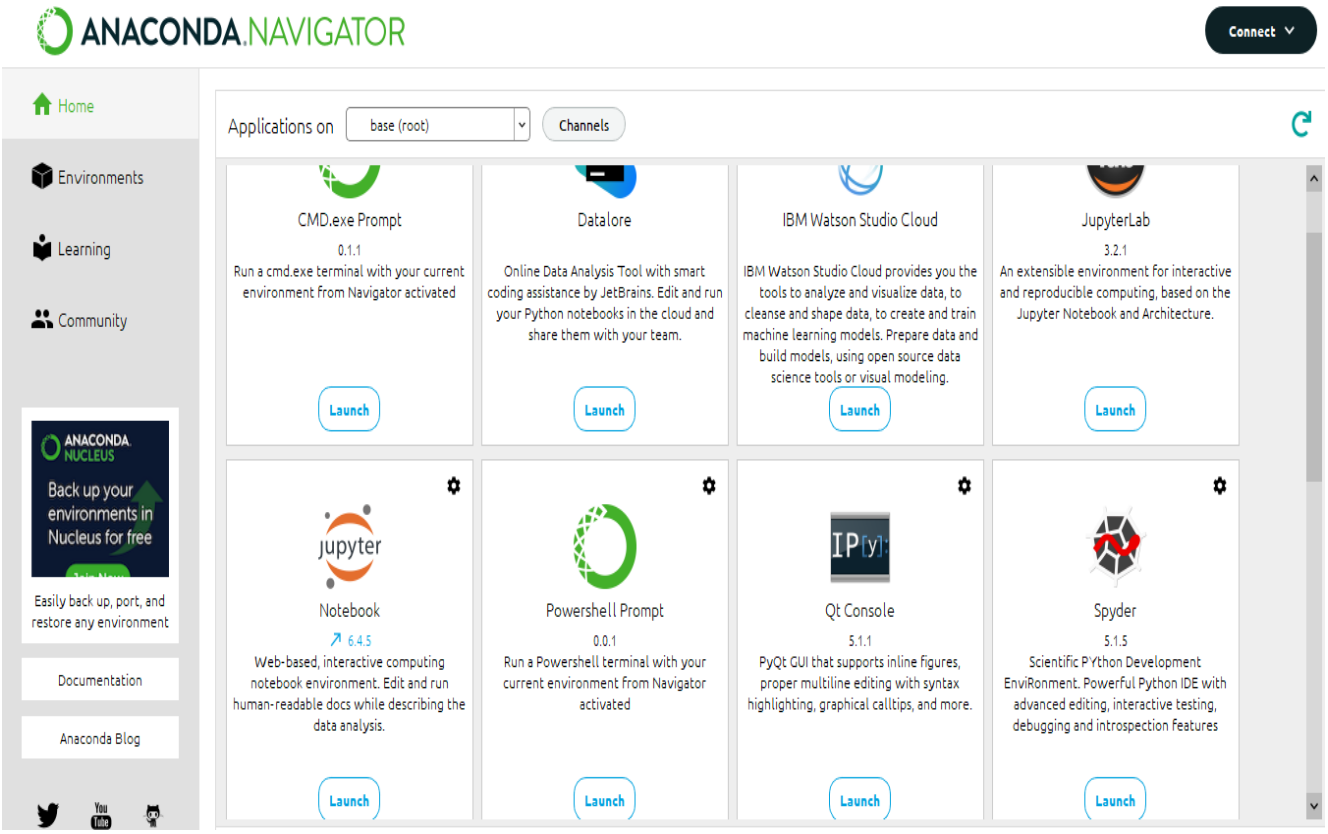


Figure 28: Page obtenue à l'ouverture Anaconda Navigator

Comme nous l'avions dit anaconda vient avec de nombreux outils dont Jupiter Notebook que nous avons choisis pour notre travail. Pour les outils déjà installés il faut simplement cliquer sur « Launch » pour pouvoir y accéder sinon « Install » pour les autres qui ne les ont pas encore.

A présent nous pouvons accéder à Jupiter Notebook en cliquant sur « Launch » qui va s'ouvrir à partir d'un navigateur web. Ainsi, des notebooks peuvent être créés et y développés des programmes en python mais encore du deep Learning. Afin de développer des modèles Machine Learning et de deep Learning, nous allons crées un environnement sous le nom de Tensorflow dans notre cas à partir d'Anaconda tout en vérifiant la version de python est celle installée (3.9.7 pour notre cas) et y ajouté le Framework Tensorflow et son API keras.

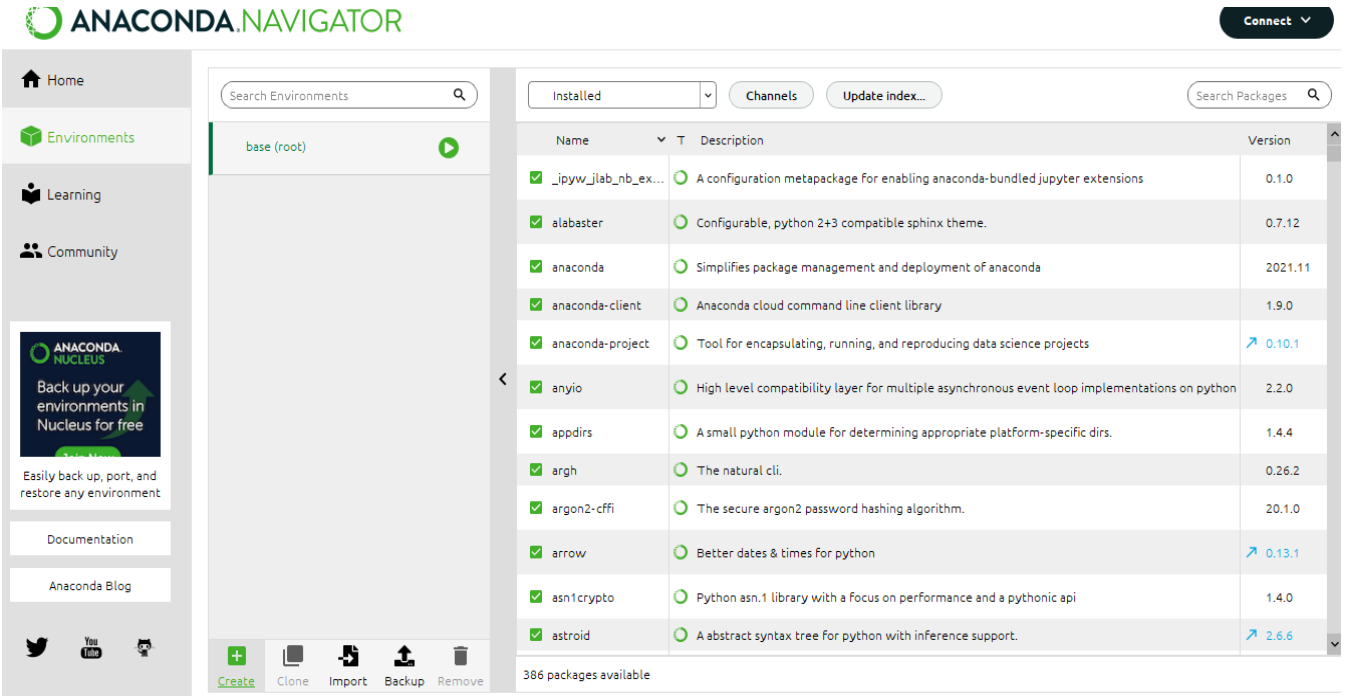


Figure 29: L'environnement de base anaconda

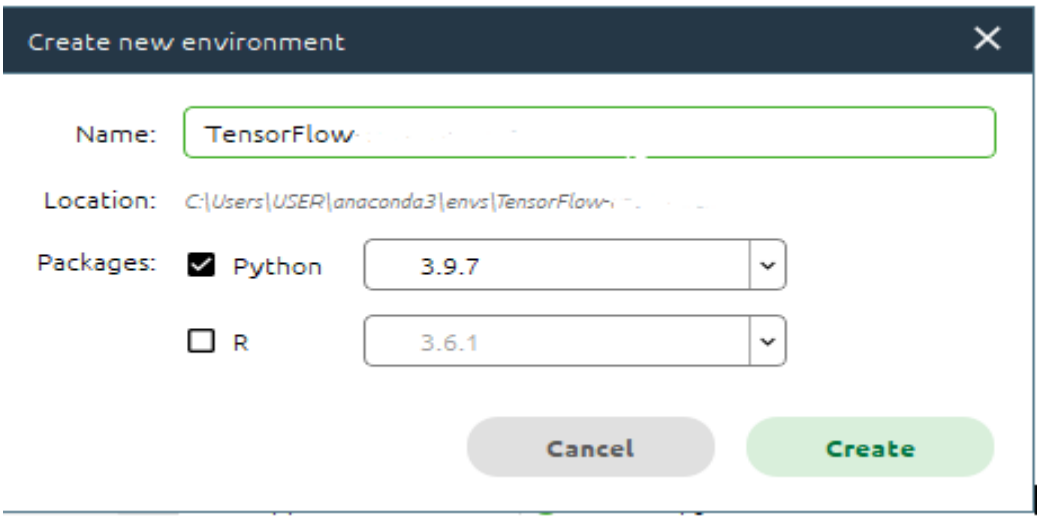


Figure 30: Création de l'environnement Tensorflow

Pour l'ajout du framework Tensorflow, on entre d'abord dans l'environnement Tensorflow créé sur l'interface Anaconda Navigator et on regarde ceux qui ne sont pas installées dans la cage « Not Installed » puis on recherche Tensorflow, le cocher, cliquer sur Apply et attendre que l'installation finisse. C'est la même procédure pour l'ajout de Keras.

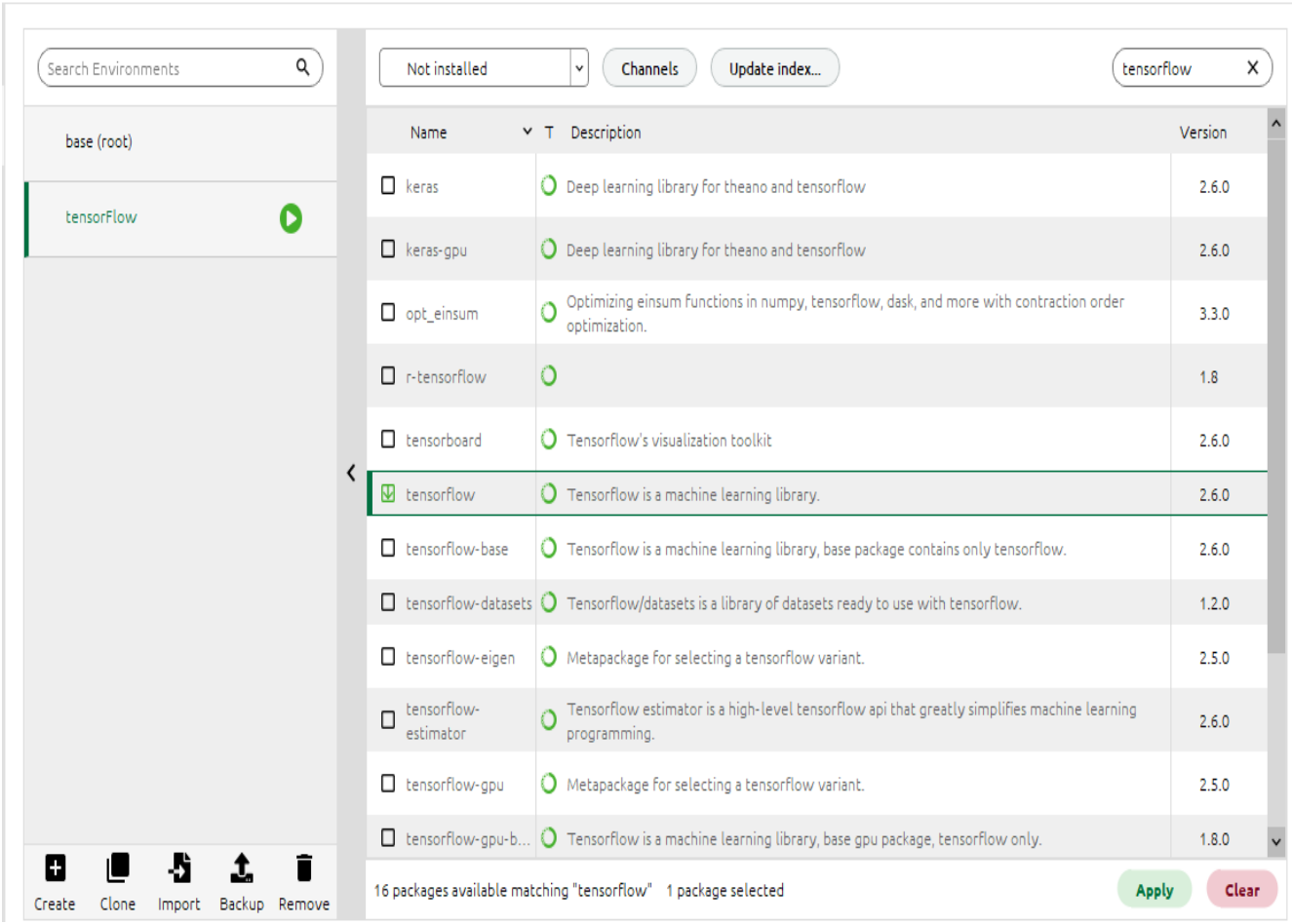


Figure 31: Creation de L'environnement Tensorflow

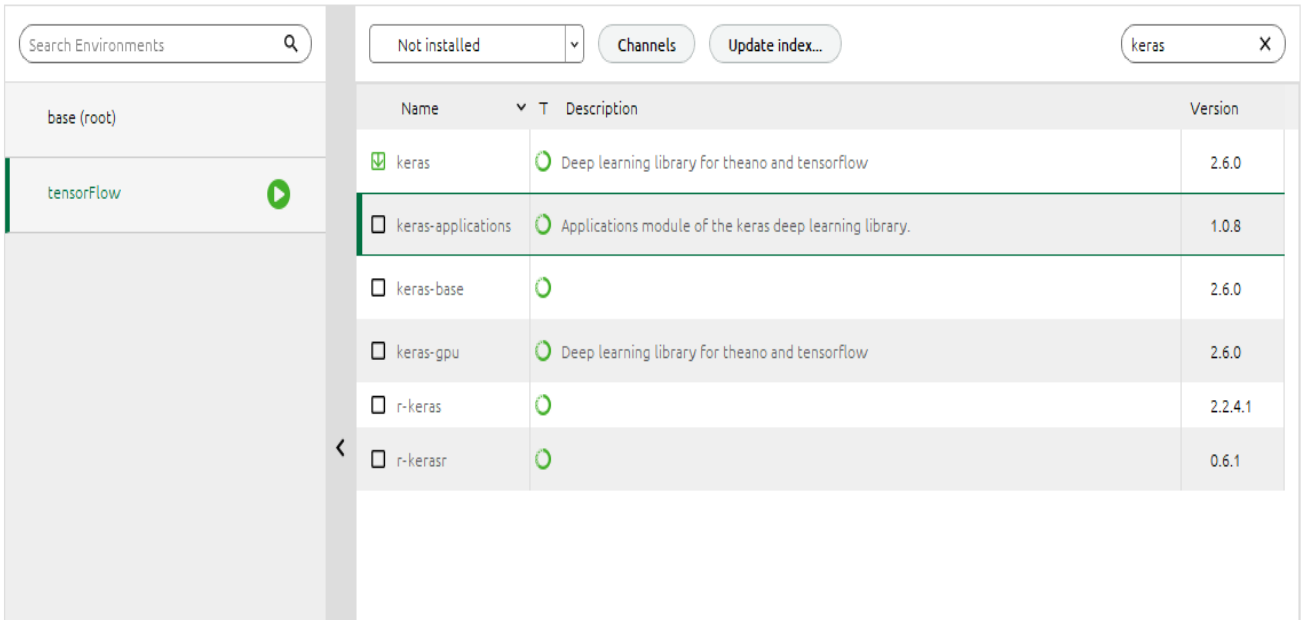


Figure 32: Ajout de l'API keras (version 2.6.0)

Pour l'installation des autres bibliothèques utilisées pour le traitement de nos données, on suit la même procédure. Enfin nous pouvons commencer à développer des modèles du d'apprentissage et évaluer leur performance.

4 Présentation du Dataset

Pour l'étude de cas nous avons utilisé un ensemble de données publics appelés IoT Network Intrusion Dataset développé par Huy Kang Kim, dont les principaux auteurs sont Hyunjae Kang, Dong Hyun Ahn, Gyung Min Lee, Jeong Do Yoo, Kyung Hose Garer et Huy KangKim. La dernière mise à jour du Dataset a eu lieu le Vendredi 27 Septembre 2019 à 04 :57. Les données recueillies proviennent d'un système de maison intelligente sur une durée de 3 mois 21jours allant du 20 Mai 2019 au 10 Septembre 2019, composée de deux appareils domestiques intelligents tels que SKT NUGU (NU 100) et la caméra Wi-Fi EZVIZ (C2C Mini O Plus 1080P).

SKT NUGU (NU 100) est un haut-parleur intelligent développé en Corée, qui dispose d'une borne de sortie de ligne et qui peut émettre du son lorsqu'elle est connectée à un périphérique externe. Il fonctionne également sur une batterie intégrée plutôt que sur une batterie filaire. Il offre 30 fonctionnalités comme par exemple l'appréciation de la musique, la maison intelligente, la

gestion des horaires, l'alerte météo, le commerce, l'IPTV, les informations sur le trafic et la livraison des commandes ainsi que certains services financiers et d'informations sur les films.

La caméra Wi-Fi EZVIZ (C2C Mini O Plus 1080P) est un dispositif qui permet de surveiller facilement toutes les pièces d'une maison en fournissant une sécurité discrète et efficace. Cette caméra est munie d'un détecteur de mouvement qui va percevoir tout mouvement anormal. Une fois connectée à un réseau wifi, nous pouvons assurer la sécurité de la maison en regardant la vidéo HD à tout moment depuis un smartphone, une tablette ou un ordinateur et y recevoir des messages d'alerte via l'application dédiée.

Ces deux dispositifs IoT sont des dispositifs victimes utilisés et sont câblés à un routeur Wi-Fi de la maison intelligente. Des ordinateurs portables, des tablettes et des smartphones sont également reliés au routeur de la maison intelligente. Ces appareils ont été utilisés comme dispositifs d'attaque dans le banc d'essai. L'ensemble de données se compose de 42 fichiers de paquets réseau bruts (pcap) à différents moments. Les fichiers de paquets sont capturés en utilisant le mode moniteur de l'adaptateur réseau sans fil. Les en-têtes sans fil sont supprimés par Aircracking. Toutes les attaques, à l'exception de la catégorie Mirai Botnet, sont des paquets capturés lors de la simulation d'attaques à l'aide d'outils tels que Nmap. Dans le cas de la catégorie Mirai Botnet, les paquets d'attaque ont été générés sur un ordinateur portable, puis manipulés pour le faire apparaître comme s'ils provenaient de l'appareil IoT. Nous avons quatre catégories d'attaques, qui sont elles-mêmes divisées en huit sous-catégories, la figure (33) montre la taxonomie des attaques :

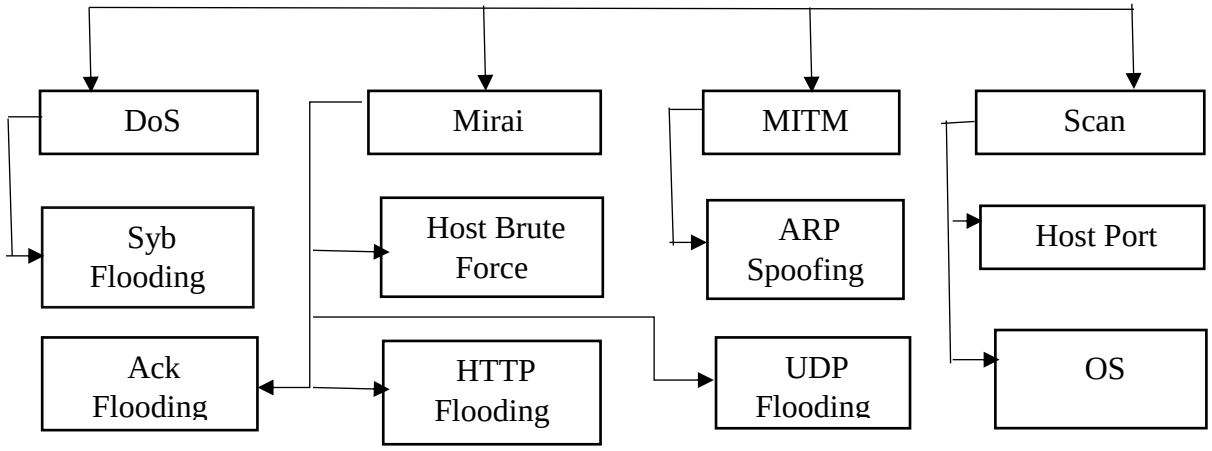


Figure 33: Taxonomie des attaques

5 Classification pour la détection d'anomalie

Le développement d'un model performant nécessite de suivre un processus qui part de la collecte et du traitement des données jusqu'au développement du ou des modèles. En effet, ce processus commence par une analyse et une description statistiques des données afin d'avoir une compréhension de celle-ci aussi fine que possible. Cette compréhension est nécessaire puisqu'elle oriente le choix du modèle à appliquer. Ensuite, cette phase de compréhension des données est suivie d'une phase de prétraitement qui prépare les données à l'application des modèles pour en tirer le meilleur résultat. Après cela, un ensemble de modèles adaptés au problème étudié est choisi et implémenté. En fin, ces modèles sont entrainés avec des données d'entrainement et tester avec des données de test et le celui qui offre le résultat de classification est choisi comme modèle pour répondre à notre problème.

5.1 Extraction des caractéristiques pour les tests avec les algorithmes du Machine Learning

5.1.1 Compréhension des données

Pour comprendre les données de trafic réseau dont nous disposons nous allons faire une analyse aussi bien sur la forme que sur le fond c'est à dire faire un ensemble de manipulation et de visualisation avec des outils statistiques et ainsi faire une description et une analyse statistique un peu plus approfondie.

- **Analyse de forme**

Nous avons commencé par repérer la « variable target » qui représente notre sortie, spécifiée ici sous le nom de « **label** », en d'autres termes c'est une variable dont les valeurs doivent être modélisées et prédites par les autres variables de notre Dataset. L'ensemble des données est réparti sur 625783 lignes et dans 86 colonnes qui représentent les caractéristiques du trafic réseaux (voir tableau 1). Chaque caractéristique est représentée par une variable en fonction du numéro de flux. L'analyse de la nature des données montre que notre Dataset contient deux catégories de variables dont 45 variables qualitatives et 41 quantitatives. En plus, ces variables sont soit des valeurs réelles, des entiers ou de type objets. Pour terminer avec l'analyse de forme nous avons cherché à savoir s'il existe des données manquantes et les résultats de la figure (33) nous ont montré qu'il n'existe aucune donnée manquante.

Caractéristiques	**Description**
Flow_ID	identifiant du flux
Src_IP	adresse IP source
Src_Port	numéro Port source
Dst_IP	adresse IP destination
Dst_Port	Numéro Port destination
Protocol	protocole utilisé
Timestamp	Horodatage
Flow_Duration	durée du flux
Tot_Fwd_Pkts	nombre total de paquets en transmission directe
Tot_Bwd_Pkts	nombre total de paquets transmis en sens inverse
TotLen_Fwd_Pkts	la longueur totale des paquets transmission directe
TotLen_Bwd_Pkts	la longueur totale des paquets transmis en sens inverse
Fwd_Pkt_Len_Max	la longueur maximale des paquets transmission directe
Fwd_Pkt_Len_Min	la longueur minimale des paquets transmission directe
Fwd_Pkt_Len_Mean	la longueur moyenne des paquets transmission directe
Fwd_Pkt_Len_Std	ecart type de la longueur des paquets transmission directe
Bwd_Pkt_Len_Max	la longueur maximale des paquets transmis en sens inverse
Bwd_Pkt_Len_Min	la longueur minimale des paquets transmis en sens inverse
Bwd_Pkt_Len_Mean	la longueur moyenne des paquets transmis en sens inverse
Bwd_Pkt_Len_Std	ecart type de la longueur des paquets transmis en sens inverse
Flow_Byts/s	le flux en octects par seconde
Flow_Pkts/s	le flux en paquets par seconde
Flow_IAT_Mean	temps moyenne entre les arrivées des paquets
Flow_IAT_Std	ecart type du temps d'inter-arrivée des paquets
Flow_IAT_Max	temps maximum entre les arrivées des paquets
Flow_IAT_Min	temps minimum entre les arrivées des paquets
Fwd_IAT_Tot	temps total entre les arrivées des paquets (sens avant)
Fwd_IAT_Mean	temps moyen entre les arrivées de paquets (sens avant)
Fwd_IAT_Std	Écart type du temps d'inter-arrivée des paquets (sens avant)
Fwd_IAT_Max	temps mamixmun entre les arrivées de paquets (sens avant)
Fwd_IAT_Min	temps minimum entre les arrivées de paquets l(sens avant)
Bwd_IAT_Tot	Temps total entre les arrivées de paquets (sens inverse)

Bwd_IAT_Mean	Temps moyen entre les arrivées de paquets (sens inverse)
Bwd_IAT_Std	ecart type temps d'inter-arrivée de paquets (sens inverse)
Bwd_IAT_Max	Temps maximun entre les arrivées de paquets (sens inverse)
Bwd_IAT_Min	Temps minimum entre les arrivées de paquets (sens inverse)
Fwd_PSH_Flags	Nombre d'indicateurs PSH (push) (sens avant)
Bwd_PSH_Flags	Nombre d'indicateurs PSH (push) (sens inverse)
Fwd_URG_Flags	Nombre d'indicateurs URG (urgent) (sens avant)
Bwd_URG_Flags	Nombre d'indicateurs URG (urgent) (sens inverse)
Fwd_Header_Len	longueur de l'entête (sens avant)
Bwd_Header_Len	longueur de l'entête (sens inverse)
Fwd_Pkts/s	nombre de paquets transmis par seconde
Bwd_Pkts/s	nombre de paquets transmis par seconde (sens inverse)
Pkt_Len_Min	longueur minimum d'un paquet
Pkt_Len_Max	longueur maximum d'un paquet
Pkt_Len_Mean	longueur moyenne d'un paquet
Pkt_Len_Std	ecart type de la longueur d'un paquet
Pkt_Len_Var	variation de la longueur des paquets
FIN_Flag_Cnt	nombre de champs de FIN de la connecxion
SYN_Flag_Cnt	nombre de champs pour la synchronisation
RST_Flag_Cnt	nombre de champs pour la réinitialisation
PSH_Flag_Cnt	nombre de champs push
ACK_Flag_Cnt	nombre de champs d'acquisition
URG_Flag_Cnt	nombre de champs d'urgence
CWE_Flag_Count	nombre de champs
ECE_Flag_Cnt	nombre de champs
Down/Up_Ratio	Taux de paquets livré dans les deux sens
Pkt_Size_Avg	la taille moyenne d'un paquet
Fwd_Seg_Size_Avg	la taille moyenne d'un segment de paquet (sens direct)
Bwd_Seg_Size_Avg	la taille moyenne d'un segment de paquet (sens inverse)
Fwd_Byts/b_Avg	la moyenne d'octets transmis (sens directe)
Fwd_Pkts/b_Avg	la moyenne des paquets transmis (sens directe)
Fwd_Blk_Rate_Avg	le taux moyen de blocage en transmisson en sens directe
Bwd_Byts/b_Avg	la moyenne d'octets transmis (sens inverse)

Bwd_Pkts/b_Avg	la moyenne des paquets transmis (sens inverse)
Bwd_Blk_Rate_Avg	le taux moyen de blocage en transmisson en sens inverse
Subflow_Fwd_Pkts	le nombre de paquets transmis en sens directe
Subflow_Fwd_Byts	le nombre d'octects transmis en sens directe
Subflow_Bwd_Pkts	le nombre de paquets transmis en sens inverse
Subflow_Bwd_Byts	le nombre d'octects transmis en sens inverse
Init_Fwd_Win_Byts	Nombre d'octets envoyés dans la fenêtre initiale (directe)
Init_Bwd_Win_Byts	Nombre d'octets envoyés dans la fenêtre initiale (sens inerse)
Fwd_Act_Data_Pkts	les paquets données actives qui sont transmis (dircte)
Fwd_Seg_Size_Min	la taille minimum d'un segment de paquer (sens directe)
Active_Mean	Durée moyenne d'activité d'un flux avant de devenir inactif
Active_Std	ecart typedu temps pendant lequel un flux était actif avant de devenir inactif
Active_Max	Durée maximale d'activité d'un flux avant de devenir inactif
Active_Min	Durée minimale d'activité d'un flux avant de devenir inactif
Idle_Mean	Durée moyenne d'activité d'un flux avant de devenir actif
Idle_Std	Écart type du temps pendant lequel un flux était inactif avant de devenir actif
Idle_Max	Durée maximale d'activité d'un flux avant de devenir actif
Idle_Min	Durée minimale d'activité d'un flux avant de devenir actif
Label	Label
Cat	Catégorie
Sub-Cat	sous-catégorie

Tableau 1: Description des caractéristiques du trafic réseau de l'ensemble des données

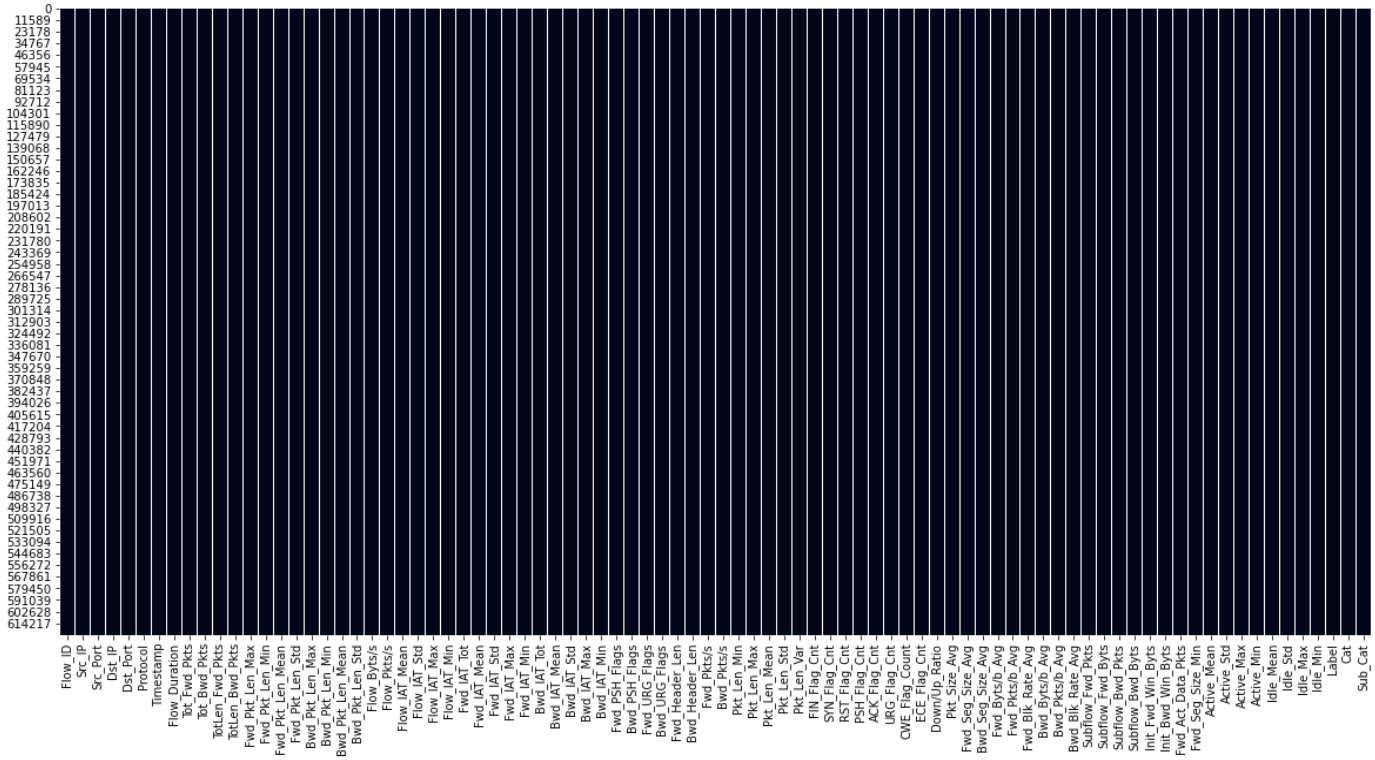


Figure 34: Résultats d'analyse de la totalité des données

❖ Analyse de fond

Pour l'analyse de fond nous avons débuté par visualiser la "variable target" et les résultats nous montrent qu'il est constitué de 583790 anomalies soit 93% du traffic réseau et 40073 de données normales soit 7% du traffic réseau. Ce qui dénote plus de présence d'anomalie que de trafic normal chose qui est normal vu que les données ont été collectées à partir de dispositifs victimes.

Ensuite, nous avons constaté que certaines caractéristiques sont représentées par leur valeur totale, moyenne, maximum, minimum et écart-type pour chaque flux. Ainsi, nous avons estimé que seules les moyennes suffisent pour représenter la valeur de ces caractéristiques et par conséquent nous avons supprimés les autres. De ce même constant les identifiants de flux, les adresses source et destination, le timestamp, les catégories et les sous catégories d'attaques sont supprimés car n'ayant pas de rapport avec notre sortie (Label).

Après la suppression de ces variables, nous aussi avons observé que les données restantes sont soit des flags ou des non-flags. Pour voir l'influence de ces variables restantes, une analyse sur la distribution des flags et des non flags est faite.

L’analyse montre que la distribution des flags par rapport aux anomalies et aux données normales n’a pas de très grande différence (voir figure 35 et 36, la tendance est la même avec les autres flags), elle est conforme sur tous les flux donc nous pouvons les supprimer.

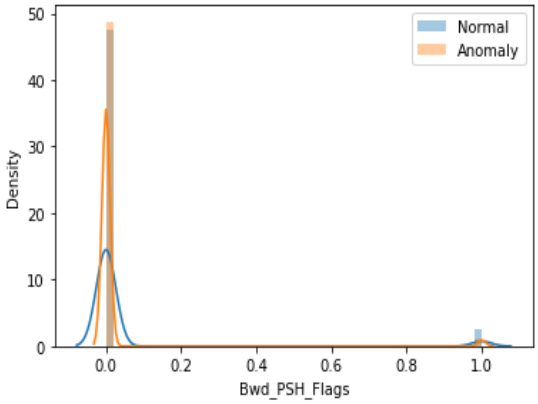


Figure 35: Distribution du flux par rapport à la transmission des flag PSH

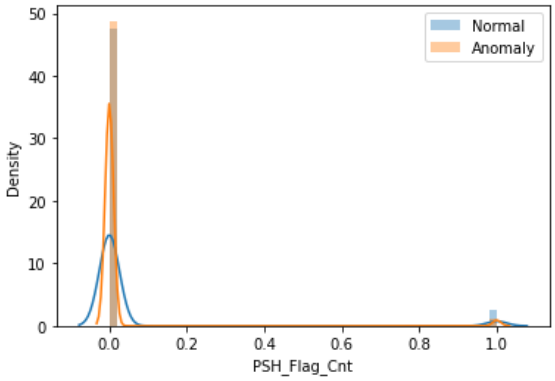


Figure 36: Distribution par rappport auu nombre de champs PSH

Quant aux non-flags l’analyse de la distribution montre que beaucoup de ces caractéristiques restantes ont une influence sur notre sortie.

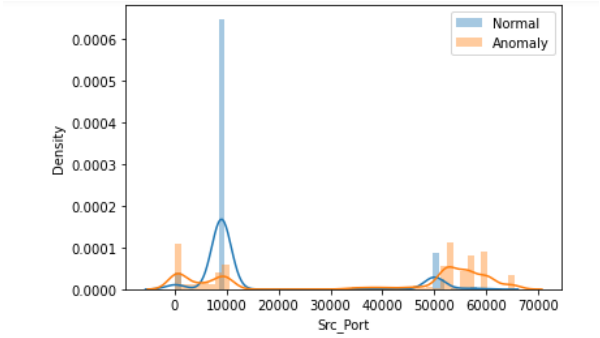


Figure 37: Distribution par rapport aux ports sources

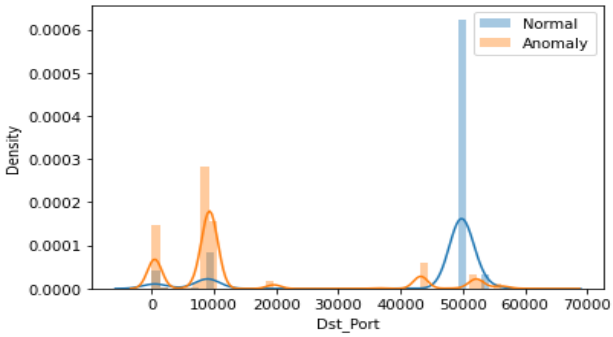


Figure 38: Distribution par rapport aux ports de destinations

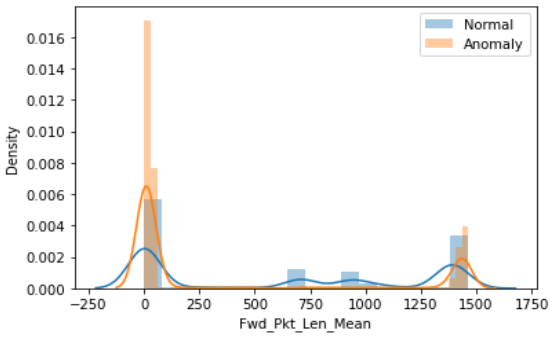


Figure 39: Distribution par rapport à la longueur moyenne des paquets transmis

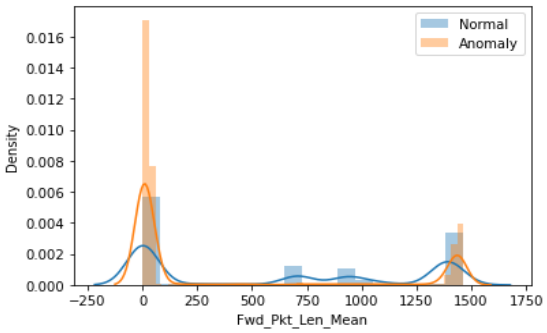


Figure 40: Distribution par rapport à la longueur moyenne des paquets

Cependant il existe toujours parmi eux des variables qui n'apportent aucun impact sur la sortie ou bien de façon moyenne mais celles-ci sont déterminées un peu plus loin avec le modèle baseline.

Enfin, nous avons aussi examiné la corrélation qui peut exister entre les features (figure 41). Cette analyse montre que les features qui ont une grande corrélation ne sont pas nombreux et la plus part sont moyennement corrélés.

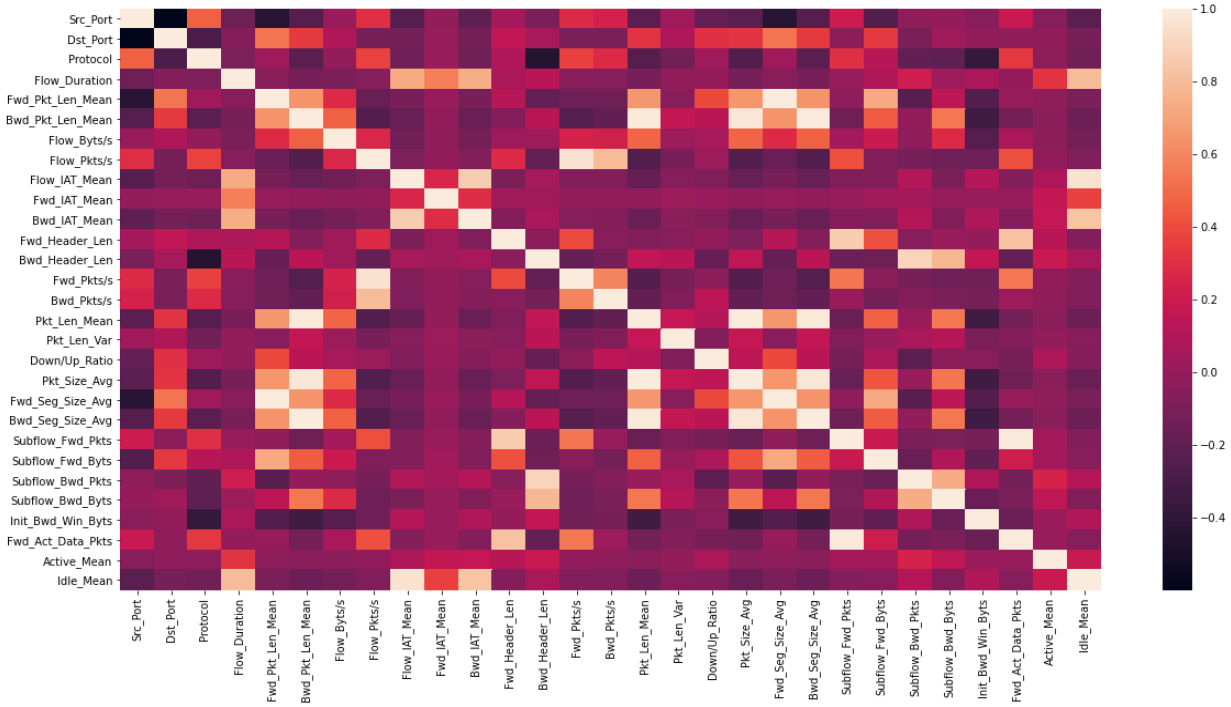


Figure 41: Résultat de la corrélation entre les caractéristiques

Après cette première analyse pour la compréhension des données, nous allons faire un prétraitement pour préparer les données à l'application des modèles.

5.1.2 **Préparation des données**

Après tout le travail de compréhension et d'extraction des caractéristiques nous avons obtenu un nouveau Dataset avec 29 features (voir tableau 2) sur lequel nous allons faire un prétraitement des données. Le prétraitement consiste à séparer les données pour la phase d'entrainement et pour la phase de test grâce à la méthode *train_test_split* du module sklearn. Les données d'entrainement représentent 80% de l'ensemble et le reste réservé au test. Après cela on sépare les données afin de faciliter la convergence. On obtient ainsi les données suivantes ***: x_train, y_ train***, pour l'apprentissage et ***x_test, y_test*** pour les tests. Les tailles de ces données sont visualisées dans la figure (42). Ensuite un encodage est effectué sur le label avec 0 qui représente anomalie et 1 normale. A présent les données sont prêtes pour tester les modèles de machine learning.

Numéro	**Carctéristiques**		**Numéro**	**Caractéristiques**
1	Source_Port		16	Pkt_Len_Mean
2	Destination_Port		17	Pkt_Len_Var
3	Protocol		18	Down/Up_Ratio
4	Flow_duration		19	Pkt_Size_Avg
5	Fwd_Pkts_Len_Mean		20	Fwd_Seg_Size_Avg
6	Bwd_Pkst_Len_Mean		21	Bwd_Seg_Size_Avg
7	Flow_Byts/s		22	Subflow_Fwd_Byts
8	Flow_Pkts/s		23	Subflow_Bwd_Byts
9	Flow_IAT_Mean		24	Subflow_Fwd_Pkts
10	Fwd_IAT_Mean		25	Subflow_Bwd_Pkts
11	Bwd_IAT_Mean		26	Init_Bwd_Win_Byts
12	Fwd_Header_Len		27	Fwd_Act_Data_Pkts
13	Bwd_Header_Len		28	Active_Mean
14	Fwd_Pkts/s		29	Idle_Mean
15	Bwd_Pkts/s			

Tableau 2: Dataset après extraction des caractéristiques

Le tableau ci-après présente la distribution des données de notre sortie après séparation en données d'entrainenment et de test.

Label	**Données d'entrainement**	**Données de test**
Anomaly	468618	117092
Normal	32008	8065

Tableau 3: Distribution de la sortie par rapport aux données d'entrainement et de test.

5.2 Evaluation des Résultats

5.2.1 Les critères d'évaluation

❖ **Matrice de confusion**

La performance d'un algorithme de machine Learning est surtout basée sur la capacité à prédire avec efficacité les résultats attendus. Une matrice de confusion ou tableau de contingence est un tableau qui visualise la performance d'un algorithme de classification testé par rapport à la classification réelle. En effet, elle représente la comparaison des valeurs réelles et de la répartition par classe des prédictions correctes et incorrectes. Chaque colonne du tableau contient une classe prédite par l'algorithme et les lignes des classes réelles. En outre elle nous permet de savoir les erreurs commises et surtout leur type.

La matrice de confusion est composée de 4 valeurs terminologiques que sont :

- **TP (Vrais Positifs) :** résultat où la prédiction et la valeur réelle sont positives. Il correspond ici à la valeur où les anomalies sont correctement prédites.
- **TN (Vrais Négatifs) :** résultat où la prédiction et la valeur réelle sont négatives. C'est équivaut au cas où les données normaux sont correctement prédites.
- **FP (Faux Positifs) :** résultat où la prédiction est positive et la valeur réelle est négative. C'est le cas où les données normales sont classées comme anomalies.
- **FN (Faux Négatifs) :** résultat où la prédiction est négative et la valeur réelle est positive. Il définit ici le cas où les anomalies sont classées normales.

❖ **Les métriques**

La matrice de confusion nous permet de calculer des indicateurs de mesures de performance appelés métriques.

- **Précision** : pourcentage de prédictions correctes parmi les prédictions positives.

$$Précision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

- **Rappel /Sensibilité (Recall):** pourcentage de vrais positifs correctement identifiés.

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

- **Justesse (accuracy) :** pourcentage de prédictions correctes sur l'ensemble des données.

$$Justesse = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

- **F1Score :** pourcentage représentant la moyenne harmonique des valeurs de précision et du rappel. Il nous fournit un équilibre entre ces deux valeurs pour mieux évaluer les performances de notre classification.

$$F1Score = 2 * \frac{Precision * Rappel}{Precision + Rappel}$$

5.2.2 Visualisation des résultats

a. Pour les algorithmes du Machine Learning

Nous avons modélisés les données en utilisant les algorithmes par des modèles Machine Learning de classification. D'abord, pour ajuster notre prétraitement nous avons choisi un modèle de base, le Décision- Tree. Ensuite, après ajustement de notre Dataset nous avons testé d'autres modèles de classification le Random Forest et l'AdaBoost.

❖ **Evaluation de Decision-Tree**

Le tableau ci-dessous montre le résultat de notre modèle de base, Decision-Tree.

	precision	Recall	f1_score	support	Matrice de confusion
0	1.00	1.00	1.00	117992	
1	1.00	0.99	1.00	8065	
					[[117060 32]
accuracy			1.00	125157	[48 8017]]
Macro avg	1.00	1.00	1.00	125157	
Weighted avg	1.00	1.00	1.00	125157	

Tableau 4: Résultats de la modélisation avec Decision-Tree

Les résultats semblent montrer une parfaite prédiction avec une précision de 100% et une justesse de 100%. Cependant, il faut s'assurer que le modèle n'a pas fait du surapprentissage et regarder comment il va se comporter avec de nouvelles données qu'il n'a pas vu durant la phase d'apprentissage. Pour cela nous avons aussi déroulé l'entrainement avec des données de validation. La courbe ci-dessous montre la tendance du score avec les données d'entrainement et avec celles de validation.

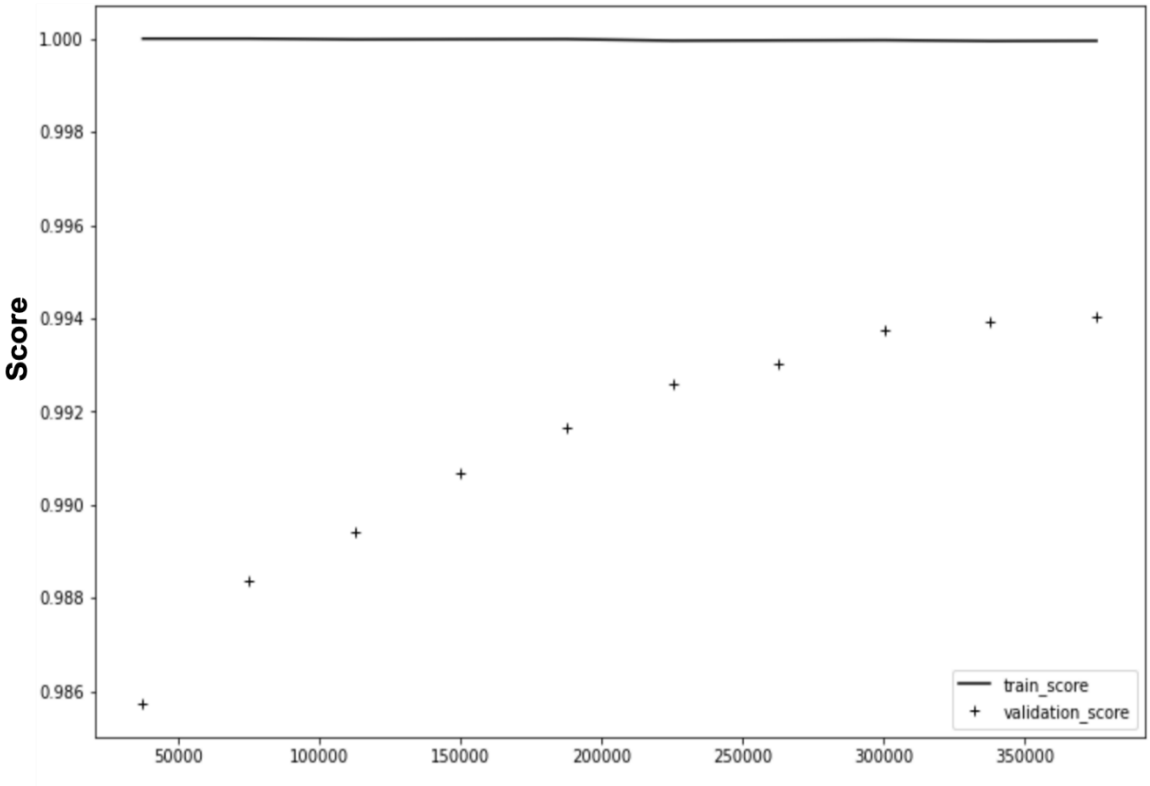


Figure 42: Résultats de la modélisation avec Decision-Tree

On observe un écart entre les deux, ce qui signifie que le modèle peut ne pas bien prédire de nouvelles données. Ainsi, pour améliorer ce résultat on peut ajuster encore notre prétraitement. Pour cela on a regardé quels sont les features les plus important dans la prédiction avec notre modèle de base. La figure suivante montre le niveau d'importance dans la prédiction de chaque feature.

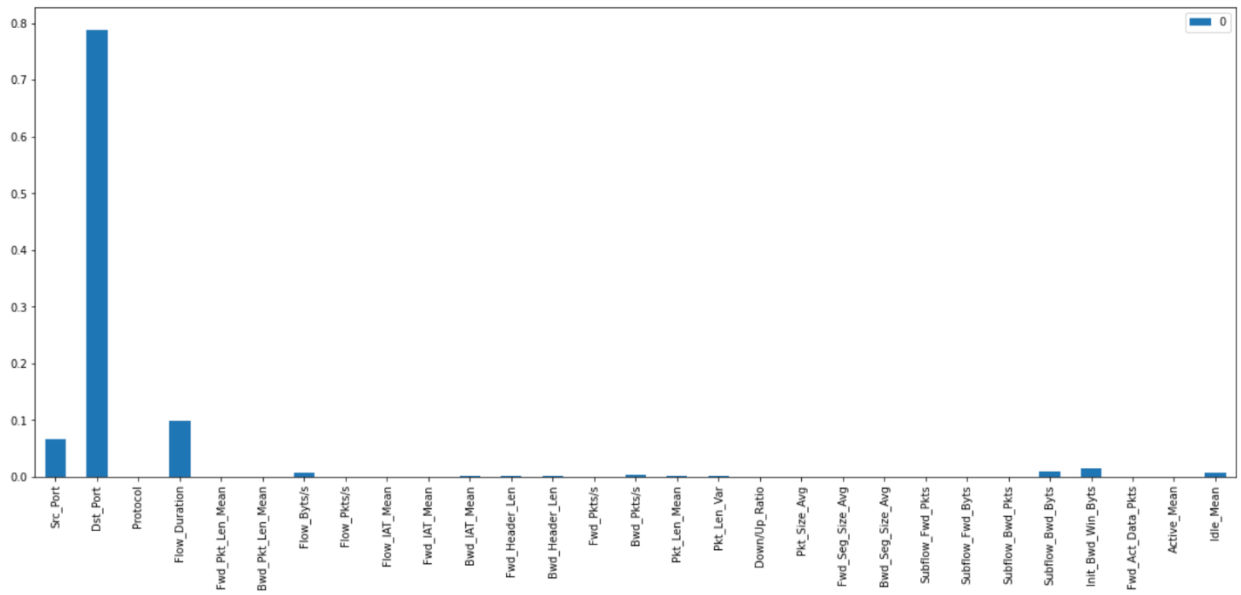


Figure 43: Histogramme d'évaluation des caractéristiques importantes utilisées par Decision-Tree

Cette histogramme nous montre que les caractéristiques les plus importantes sont comme par exemple les adresses sources, adresses destination, la durée de flux et d'autres avec un taux un peu faible. Ce qui veut dire qu'on peut encore ajuster et supprimer les features les moins importants et tester avec les modèles avec le reste des features e.i. les plus importants

❖ Evaluation des modèles : Random Forest et Adaboost

Plusieurs modèles de classification ont été testés : Random Forest, SVM, KNN, et Adaboost. Cependant, Random Forest et AdaBoost offrent les meilleurs résultats. Nous avons donc visualisé et analysé ces deux modèles.

Pour modéliser ces deux modèles nous avons utilisés les caractéristiques importantes trouvées lors de l'évaluation avec le modèle de base, le Decision-Tree. Les résultats sont présentés dans les tableaux 4 (Random Forest) et 5 (Adaboost).

	precision	recall	f1_score	support	Matrice de confusion
0	1.00	1.00	1.00	117992	
1	1.00	0.99	0.99	8065	[[117082 10]
accuracy			1.00	125157	[88 7977]]
Macro avg	1.00	0.99	0.99	125157	
Weighted avg	1.00	1.00	1.00	125157	

Tableau 5: Résultats de la modélisation avec Random-Forest

	precision	recall	f1_score	Support	Matrice de confusion
0	1.00	1.00	1.00	117992	
1	0.99	0.95	0.97	8065	[[117031 61]
accuracy			1.00	125157	[384 7681]]
Macro avg	0.99	0.98	0.98	125157	
Weighted avg	1.00	1.00	1.00	125157	

Tableau 6: Résultats de la modélisation avec Adaboost

Selon les résultats observés dans les tableaux ci-dessus le Radom Forest offre plus de performance avec un f1_score de 99% par rapport à AdaBoost qui a 97%. Cependant, AdaBoost est préférable puisqu'il se comporte mieux avec de nouvelles données (voir figure 46). Pour le Radom Forest, la courbe de validation est éloignée par rapport à celle de l'apprentissage, ce qui dénote une tendance au sur apprentissage.

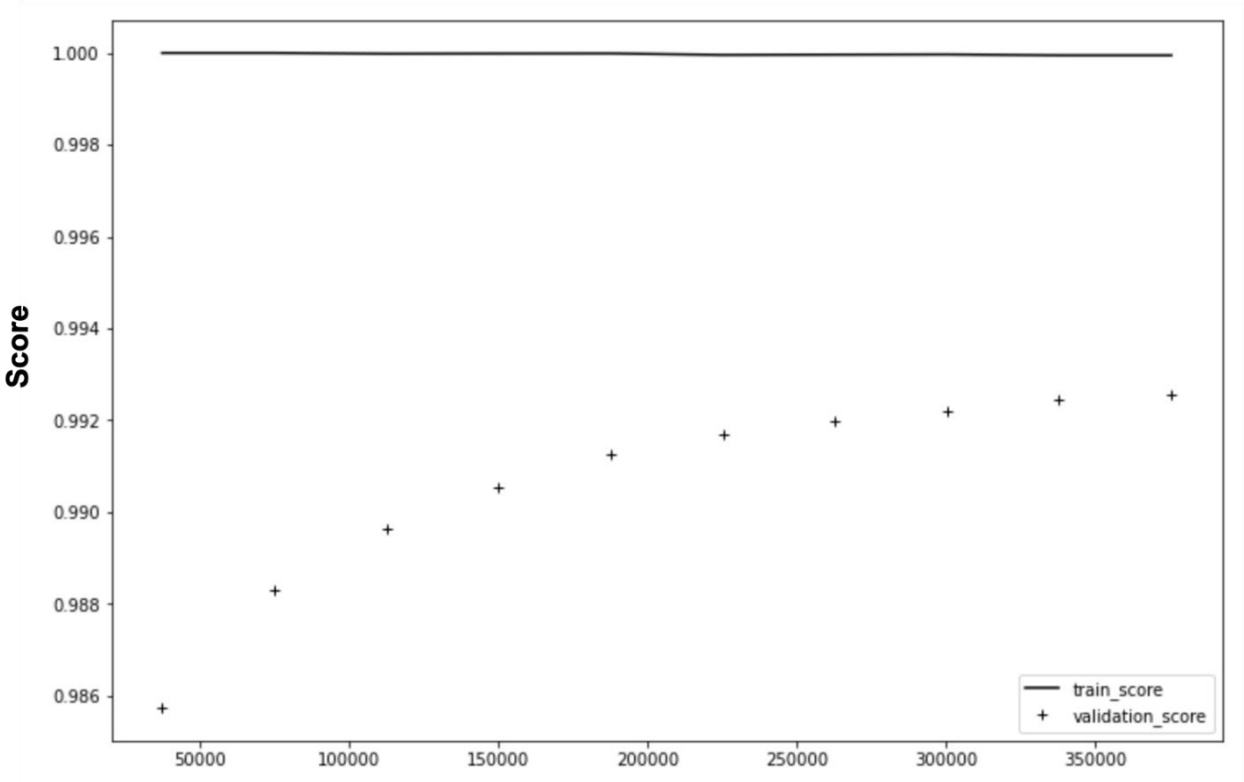


Figure 44: Résultats de la modélisation avec Random-Forest

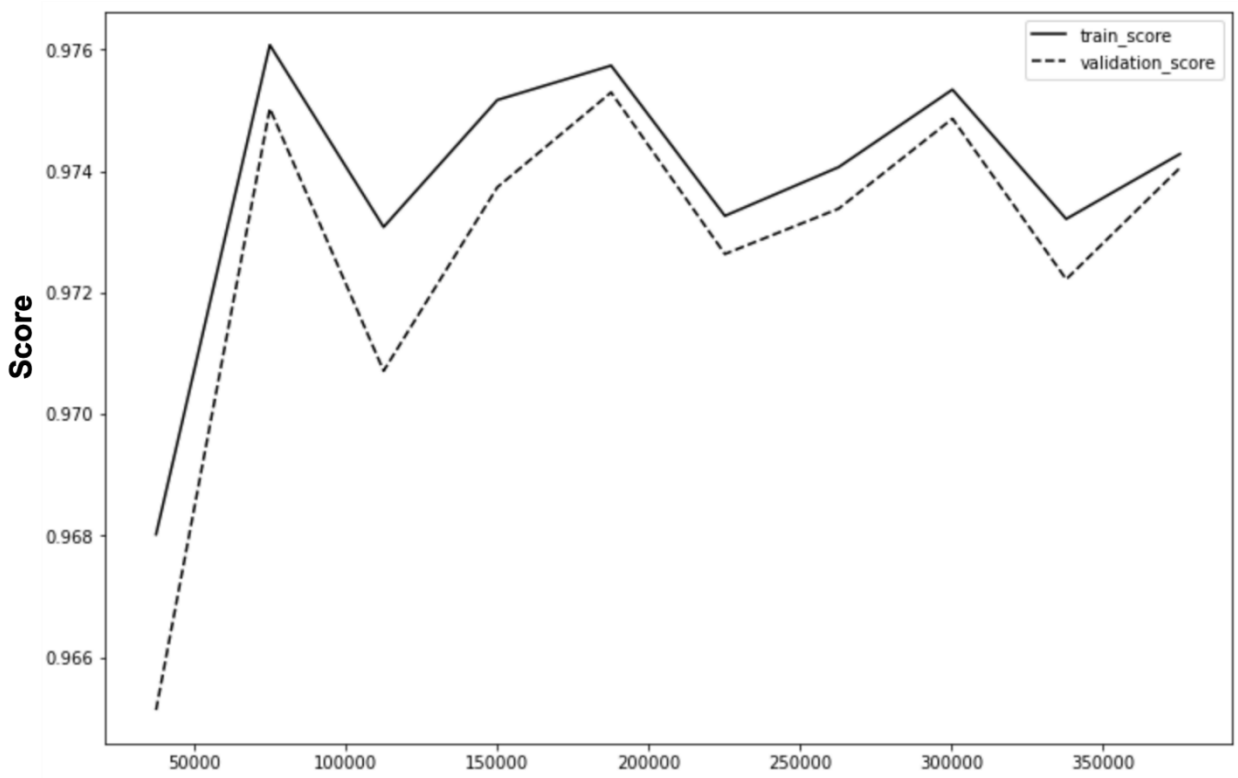


Figure 45: Résultats de la modélisation avec AdaBoost

Dans la suite, nous allons développer un modèle de réseau de neurone et comparer ses performances avec le meilleur modèle qu'on a à la première étape c'est à dire l'AdaBoost.

b. Evaluation des résultats avec les réseaux de neurones

Le réseau de neurone qu'on a conçu est réseau de neurone de classification qui est composé de 4 couches avec 2 couches cachés, une couche d'entrée de 29 features et une couche de sortie. Sur la couche de sortie nous avons utilisé *Sigmoïd* comme fonction d'activation et *ReLu* pour les autres couches. Le tableau suivant donne le résumé des différentes couches. Après chaque couche nous avons ajouté une couche de *Dropout* pour prévenir l'overfitting ou l'underfitting.

```
Model: "sequential_34"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_95 (Dense)             (None, 20)                600
_________________________________________________________________
dropout_40 (Dropout)         (None, 20)                0
_________________________________________________________________
dense_96 (Dense)             (None, 10)                210
_________________________________________________________________
dropout_41 (Dropout)         (None, 10)                0
_________________________________________________________________
dense_97 (Dense)             (None, 1)                 11
=================================================================
Total params: 821
Trainable params: 821
Non-trainable params: 0
```

Figure 46: Résumé de la composition du modèle de réseau de neurones

Nous avons déroulé le modèle sur 100 *epoch* et visualiser l'erreur et la justesse aussi bien sur les données d'entrainement que sur les données de validation. Le résultat montre que le n'a pas fait de l'overfitting et commence avoir des valeurs stables à partir de la 30ème epoch et le reste jusqu'à peu près la 70ème *epoch*. Le meilleur compromis du modèle on peut ainsi l'avoir entre ces deux valeurs d'*epoch.*

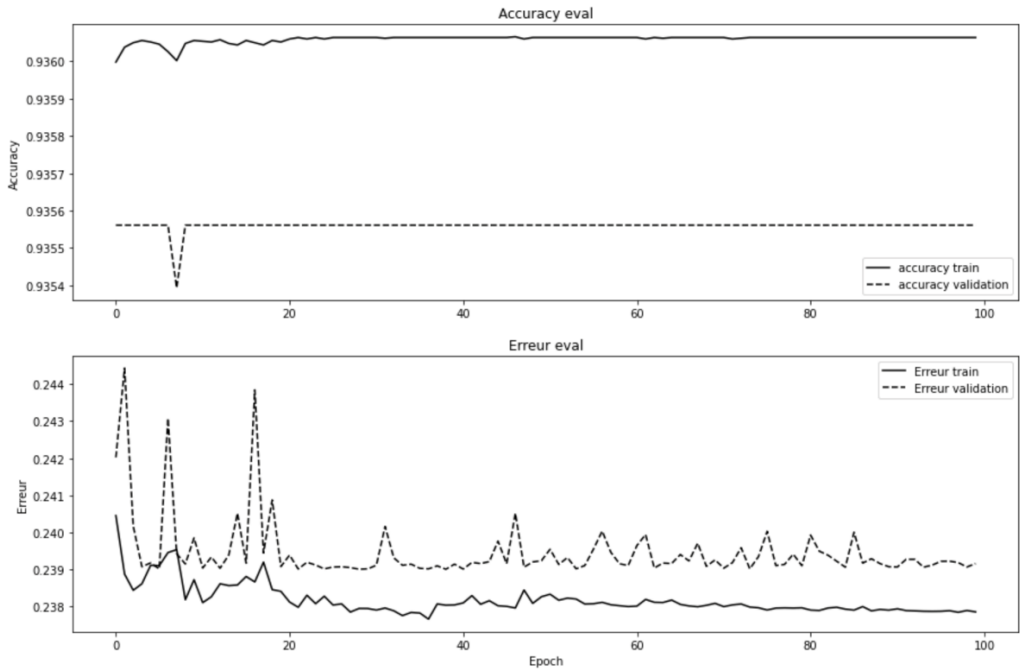


Figure 47: Performance du réseau de neurone par rapport aux données d'entrainement et de validation

Ce modèle offre une très bonne précision ~93% qui est largement suffisant pour une détection d'anomalie satisfaisante. Cependant, d'après les résultats précédent, le AdaBoost offre la meilleure performance avec ~97% de précision.

6 Conclusion

Dans ce chapitre, nous avons fait une implémentation de modèles de classification pour la détection d'anomalie dans les réseaux de smart home. Nous avons testé et comparé trois performants modèles : Random Forest, AdaBoost et un modèle de réseau de neurone. L'AdaBoost offre le meilleur résultat avec 97% de précision dans les prédictions tout en se comportant très bien avec des données nouvelles.

Conclusion générale

L'objectif principal de notre mémoire était la détection d'anomalie dans un réseau IoT avec comme domaine d'application une maison intelligente. Dans cette optique, afin de bien répondre à la problématique et à l'objectif principal que nous pensons atteindre, nous nous sommes penchés sur une classification des données tout en s'appuyant de l'Intelligence artificielle et de la puissance que nous offrent ses différentes branches.

En dehors de l'introduction générale, notre mémoire est divisé en 3 chapitres. Dans le premier chapitre nous avons présenté de manière générale l'état de l'art d'IoT. Ensuite, dans le deuxième chapitre nous avons une étude théorique sur la classification des données ainsi que les algorithmes et modèles des différentes branches de l'IA tels que le Machine Learning, les réseaux de neurones artificielles et le Deep Learning. Enfin, dans le troisième chapitre nous avons fait une réalisation d'un classificateur intelligent pour la détection d'anomalie dans un réseau d'une maison intelligente en appliquant les modèles du Machine Learning, des RNA et du Deep Learning.

Nous avons implémenté plusieurs modèles de classification et ceux qui offrent les meilleurs restent le Random Forest, l'AdaBoost et le réseau de neurones. Le Random Forest offre de bonne performance mais ne se comporte pas bien avec de nouvelles données. L'AdaBoost et le réseau de neurone prédisent bien les nouvelles données mais l'AdaBoost offre la meilleure performance avec 97% contre 93% pour le réseau de neurone. Néanmoins, dans les deux cas les pourcentages précision sont suffisamment grands pour satisfaire le besoin de détection d'anomalies dans les réseaux d'objets intelligents de smart home que nous avions comme objectif dans ce mémoire.

En définitive, nous pouvons dire que l'utilisation de l'IA pour la détection d'anomalie dans les réseaux IoT est une solution sérieuse à envisager vu qu'il offre de très bonne performance en termes de classification. Cependant, ce travail montre qu'il reste des choses à faire pour tirer le maximum d'avantage de l'AI pour répondre à ce type de problème. En effet, nous avons constaté que certains modèles pourraient s'améliorer avec plus de données en entrée. Plus encore, nous n'avons pas pu exploiter les hyperparamètres des différents modèles afin d'avoir les meilleures performances qu'on peut en leur tirer. Nous pouvons aussi dire que dans ce travail nous avons pu déterminer les paramètres de flux réseau qui sont les plus intéressants à regarder pour détecter des

anomalies dans un réseau. Ainsi, on peut envisager à l'avenir d'étudier à une plus grande échelle ces paramètres et confirmer leur importance dans la détection d'anomalies.

REFERENCES

1. P. Suresh1 J. Vijay Daniel2 ,Dr.V.Parthasarathy4 “A state of the art review on the Internet of Things (IoT)”, International Conference on Science, Engineering and Management Research (ICSEMR 2014), consulté 10/10/21
2. https://www.itu.int/en/ITU-D/Regional-Presence/Africa/Documents/PRIDA%202020%20-%20ONLINE%20Capacity%20building%20%26%20digital%20services/FR_Workshop_Slides.pdf, consulté 10/10/21
3. Pallavi Sethi and Smruti R. Sarangi “Internet of Things: Architectures, Protocols, and Applications” , Journal of Electrical and Computer Engineering/2017, consulté 10/10/21
4. https://www.omnisci.com/technical-glossary/fog-computing, consulté 18/09/21
5. https://www.c2s-bouygues.com/insights/article-IoT-les-grandes-etapes-de-la-realisation-dun-projet, consulté 20/10/21
6. https://ryax.tech/fr/IoT-et-big-data-comprendre-la-relation-entre-ces-deux-technologies, May 2021, consulté 10/10/21
7. Keyur K Patel1 , Sunil M Patel2 “Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges”, International Journal of Engineering Science and Computing, May 2016, consulté 12/10/21
8. Saleh Imad “Internet des Objets: Concept, Enjeux, Défis et Perspectives, ” Février 2018, consulté 16/10/21
9. Pierre-Louis Lussan, “Les 10 types de cyberattaques les plus courants”, Septembre 2029, consulté 14/09/21
10. Bambang Susilo and Riri Fitri Sari, “Intrusion Detection in IoT Networks Using Deep Learning Algorithm”, Mai 2020, consulté 04/11/21
11. Mangayarkarasi Ramaiah, Vanmathi Chandrasekaran, Vinayakumar Ravi, Neeraj Kumar “An intrusion detection system using optimized deep neural network architecture”, Janvier 2021, consulté 06/11/21
12. Sarhan, Mohanad, Siamak Layeghy, Nour Moustafa, and Marius Portmann. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. In Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th

EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings (p. 117). Springer Nature, consulté 27/10/21.

13. Victor Garcia-Font, Carles Garrigues and Helena Rifà-Pous “A Comparative Study of Anomaly Detection Techniques for Smart City Wireless Sensor Networks”, June 2016, consulté 06/10/21.
14. Ibrahim Alrashdi, Raed Alharthi, Ali Alqazzaz, Mohamed Zohdy, Esam Alouf, Hua Ming “Ad-IOT: Anomaly detection of IoT cyberattacks in smart city using machine learning”, 2019, consulté 28/10/21.
15. Djellel Eddine Difallah, Philippe Cudré-Mauroux and Sean A. McKenna “Scalable Anomaly Detection for Smart City Infrastructure Networks”, November/ December 2013, consulté 28/10/21.
16. Dimitrios Serpanos , Marilyn Wolf “Internet-of-Things (IoT) Systems:Architectures, Algorithms, Methodologies”, consulté 05/10/21.
17. (PDF) (en anglais seulement) Examen des systèmes d’exploitation, des plates-formes et des applications de l’Internet des objets (researchgate.net), consulté 10/10/21.
18. Victor Garcia-Font *, Carles Garrigues and Helena Rifà-Pous, “A Comparative Study of Anomaly Detection Techniques for Smart City Wireless Sensor Networks”, consulté 10/10/21.
19. Pierre-Louis Lussan “Data Classification : Qu’est-ce que c’est, pourquoi en préoccuper, comment la mettre en œuvre “, Mai 2020, consulté 15/11/21.
20. Peff Petters et Adrien Rahmati-Georges “Définition de la classification des données”, Février 2019, consulté 15/10/21.
21. Wolfgang Ertel “Introduction to Intelligence Artificielle”, second edition, Springer 2017, consulté 09/10/21.
22. https://www.futura-sciences.com/tech/questions-reponses/intelligence-artificielle-sont-pionniers-intelligence-artificielle-4907/, consulté 12/11/21.
23. Mohammed Ali Al-Garadi, Amr Mohamed, Abdulla Al-Ali, Xiaojiang Du, Mohsen Guizani “A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security”, consulté 29/10/21.
24. Isabelle Martelly, « Jean-Gaël Barbara, La naissance du neurone », Histoire de la recherche contemporaine, Tome I-N°2 | 2012, 190, consulté 06/10/21.

25. Nouamane Cherkaoui "Les neurosciences ont inspiré l'intelligence artificielle : le modèle neuronal et les concepts – [2/2], Mis à jour le 21/09/20", consulté 10/10/21.
26. https://fr.wikipedia.org/wiki/Perceptron, consulté 11/10/21.
27. http://eric.univ-lyon2.fr/~ricco/cours/slides/reseaux_neurones_perceptron.pdf, consulté 12/11/21.
28. Hugo Welter "Deep learning : définition, fonctionnement et applications", publication originale le 24 juin 2021, mise à jour le 29 novembre 2021, consulté 12/12/21.
29. Réseaux de neurones convolutifs | SpringerLink (en anglais), consulté 17/12/21.
30. https://fr.wikipedia.org/wiki/R%C3%A9seau_neuronal_convolutif, consulté 17/12/21.
31. Yann MORERE, "Les réseaux de neurons récurrents", Septembre 2001, consulté 20/11/21.
32. https://datavalue-consulting.com/deep-learning-reseaux-neurones-recurrents-rnn, Août 25, 2021 | IA/Data Science, consulté 21/12/21.
33. https://the-learning-machine.com/article/dl/autoencoder?gclid=CjwKCAiAiKuOBhBQEiwAId_sKwzXCQhS2gSWCzve0xKc24DJIzr6BSt_lQJx6PcQeLYpneHhGfX7eRoCqPkQAvD_BwE,consulté 25/12/21.
34. Matthew Stewart, "Comprehensive Introduction to Autoencoders", Avril 2019, consulté 27/12/21.
35. Abeshu, A., & Chilamkurti, N. (2018). Deep Learning: The Frontier for Distributed Attack Detection in Fog-to-Things Computing. IEEE Communications Magazine, 56(2), 169–175. doi:10.1109/mcom.2018.1700332, consulté 05/11/21.
36. Ian J. Goodfellow , Jean Pouget-Abadie† , Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair , Aaron Courville, Yoshua Bengio, "Generative Adversarial Nets", consulté 03/01/22.