الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – Guelma

Faculté des Mathématiques, d'Informatique et des Sciences de la matière

Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique

Option : Système d'information

Thème :

Vers des réseaux IoT sécurisés : surveillance des menaces cybernétiques basée sur intelligence artificielle

Encadré Par :

Dr. Ferrag Mohamed Amine

Présenté par :

Gheraibia Chaima

Devant le jury :

Dr.Benamira adel	**Président**
Dr.hannousse Abd Elhakim	**Examinateur**

Juin 2025

Remerciement

Tout d'abord, nous tenons à remercier le bon Dieu Le tout puissant de nous avoir donné la force et le courage de mener à bien ce modeste travail

Je remercie également ma famille pour leur sacrifices qu'elles ont faits pour que je termine mes études.

Je tiens à adresser mes remerciements à mon professeur ***Mr. MOHAMED AMINE FERRAG*** *pour son aide et ses conseils tout au long de la période de mon projet.*

Je tiens également à exprimer ma reconnaissance à membres du jury, pour leurs retours constructifs et leur engagement dans l'évaluation de ce travail.

Je remercie tous les enseignants à tous les niveaux, particulièrement ceux du département de l'informatique, pour tout le savoir que j'ai acquis grâce à eux durant ma formation.

Je ne saurais oublier de remercier mes collègues et amis pour leur soutien moral et leur compréhension tout au long de cette aventure académique.

Enfin, Je remercie toutes les personnes qui, de près ou de loin, ont contribué à la réussite de ce projet, que ce soit par leurs conseils, leurs critiques constructives ou simplement par leur présence bienveillante.

Dédicace

بسم الله الرحمان الرحيم

Arrivé au terme de mes études ; j'ai le plaisir de dédies ce modeste travail :

À moi-même, *qui a toujours résisté et persévéré. Merci à toi, moi-même, pour tout ce que tu as donné.*

À mon père*, qui est toujours présent dans mon cœur. Je suis diplômé, Papa, et quelle joie serait incomplète sans toi. J'aurais aimé que tu sois à mes côtés maintenant et que tu sois le premier à entendre parler de ma remise de diplôme. J'espère que mes sentiments te parviendront et que tu seras fier de ceux qui portent ton nom. Ils ont prouvé que tu es un excellent éducateur et un excellent enseignant.*

À la meilleur mère du monde *: ma chère maman pour ses encouragements, soutiens et ses prières pour que je réussisse dans ma vie, merci mère.*

À mon cher frère Abd Elmouize*, et mes sœur****s Amina*** *et* ***Chaima,*** *mon beau-frère* ***Hamza .***

A mes tantes « ***Saliha , Rachida*** *», Merci d'avoir été fières de moi ,*

et d'avoir été a mes cotes sans hésitation

À Toute La Famille ***GHERAIBIA et KEBAILI****.*

À mes amis, précieux joyaux de ma vie

CHAIMA

Résumé

Les systèmes de détection d'intrusion (IDS) jouent un rôle essentiel dans le renforcement de la sécurité des environnements numériques, notamment ceux liés à l'Internet des Objets (IoT), où les dispositifs connectés sont souvent vulnérables à des attaques sophistiquées. L'objectif de ce mémoire est de concevoir et d'évaluer un système intelligent capable de détecter efficacement les activités malveillantes dans un contexte IoT, en s'appuyant sur des techniques d'apprentissage profond.

Pour cela, nous avons développé un modèle de classification basé sur un réseau de neurones convolutif (CNN), entraîné sur le dataset Edge-IIoTset, un ensemble de données récent et représentatif des menaces ciblant les systèmes industriels connectés. Le pipeline de traitement comprend plusieurs étapes essentielles, notamment le nettoyage et la normalisation des données, le rééquilibrage des classes pour traiter les déséquilibres de distribution, la conception d'une architecture CNN optimisée, et l'analyse des performances obtenues.

Les résultats expérimentaux montrent que le modèle CNN proposé atteint des performances élevées en termes de précision, de rappel et de score F1, tout en maintenant un taux de détection élevé et un faible taux de fausses alertes. Cette étude met ainsi en évidence le potentiel des CNN comme solution efficace et robuste pour la détection d'intrusions dans les environnements IoT, tout en soulignant les pistes d'amélioration futures, notamment en matière de scalabilité et de confidentialité des données.

Mots-clés : Internet des Objets (IOT), cybersécurité, détection d'intrusion (IDS), apprentissage profond (DL), apprentissage automatique (ML), réseaux neuronaux convolutifs, Edge-IIoTset.

Abstract

Intrusion Detection Systems (IDS) play a vital role in strengthening the security of digital environments, particularly in the Internet of Things (IoT), where connected devices are often vulnerable to increasingly sophisticated attacks. This work aims to design and evaluate an intelligent IDS capable of effectively identifying malicious activities within IoT networks using deep learning techniques.

To achieve this, we developed a classification model based on a Convolutional Neural Network (CNN), trained on the Edge-IIoTset dataset—a recent and representative dataset of threats targeting industrial IoT systems. The proposed pipeline includes key steps such as data preprocessing and normalization, class re-balancing to address distribution imbalance, the design of an optimized CNN architecture, and performance analysis.

Experimental results demonstrate that the proposed CNN model achieves high performance in terms of accuracy, recall, and F1-score, while maintaining a high detection rate and a low false alarm rate. This study highlights the effectiveness and robustness of CNN-based approaches for intrusion detection in IoT environments and suggests future improvements in terms of scalability and data privacy.

Keywords : Internet of Things (IoT), cybersecurity, intrusion detection (IDS), deep learning (DL), machine learning (ML), convolutional neural networks (CNN), Edge-IIoTset.

ملخص

تلعب أنظمة كشف التسلل (IDS) دورًا أساسيًا في تعزيز أمان البيئات الرقمية، لا سيما تلك المرتبطة بإنترنت الأشياء (IoT)، حيث تكون الأجهزة المتصلة غالبًا عرضة لهجمات متطورة. الهدف من هذه الرسالة هو تصميم وتقييم نظام ذكي قادر على اكتشاف الأنشطة الخبيثة بفعالية في سياق إنترنت الأشياء (IoT)، بالاعتماد على تقنيات التعلم العميق.
لهذا الغرض، قمنا بتطوير نموذج تصنيف يعتمد على شبكة عصبية تلافيفية (CNN)، تم تدريبه على مجموعة بيانات Edge-IIoTset، وهي مجموعة بيانات حديثة وتمثيلية للتهديدات التي تستهدف الأنظمة الصناعية المتصلة. تشمل عملية المعالجة عدة خطوات أساسية، بما في ذلك تنظيف وتطبيع البيانات، وإعادة توازن الفئات لمعالجة اختلالات التوزيع، وتصميم بنية CNN المحسنة، وتحليل الأداء المحقق.
تُظهر النتائج التجريبية أن نموذج CNN المقترح يحقق أداءً عاليًا من حيث الدقة والاسترجاع ودرجة F1، مع الحفاظ على معدل اكتشاف مرتفع ومعدل منخفض من الإنذارات الكاذبة. تسلط هذه الدراسة الضوء على إمكانيات الشبكات العصبية التلافيفية (CNN) كحل فعال وموثوق للكشف عن التسللات في بيئات إنترنت الأشياء (IoT)، مع التأكيد على مجالات التحسين المستقبلية، لا سيما في ما يتعلق بالقدرة على التوسع وخصوصية البيانات.

الكلمات المفتاحية: إنترنت الأشياء (IoT)، الأمن السيبراني، كشف التسلل (IDS)، التعلم العميق (DL)، التعلم الآلي (ML)، الشبكات العصبية التلافيفية، Edge-IIoTset.

Table des matières

Liste des figures

Liste des tableaux

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

AI	Artificial Intelligence (Intelligence Artificielle)
CNN	Convolutional Neural Network (Réseau de Neurones Convolutif)
IDS	Intrusion Detection System (Système de Détection d'Intrusion)
IPS	Intrusion Prevention System (Système de Prévention d'Intrusion)
IoT	Internet of Things (Internet des Objets)
IIoT	Industrial Internet of Things (Internet Industriel des Objets)
DL	Deep Learning (Apprentissage Profond)
ML	Machine Learning (Apprentissage Automatique)
FL	Federated Learning (Apprentissage Fédéré)
NIDS	Network-based Intrusion Detection System (IDS basé sur le réseau)
HIDS	Host-based Intrusion Detection System (IDS basé sur l'hôte)
LSTM	Long Short-Term Memory (Mémoire à Long et Court Terme)
GRU	Gated Recurrent Unit (Unité Récurrente à Portes)
DNN	Deep Neural Network (Réseau de Neurones Profond)
AE	Autoencoder (Autoencodeur)
VAE	Variational Autoencoder (Autoencodeur Variationnel)
GAN	Generative Adversarial Network (Réseau Génératif Antagoniste)
SHAP	SHapley Additive exPlanations (explicabilité des modèles ML)
DoS	Denial of Service (Déni de Service)
DDoS	Distributed Denial of Service (Déni de Service Distribué)
MITM	Man In The Middle (Homme du Milieu)
SQLi	SQL Injection (Injection SQL)
Non-IID	Non-Independent and Identically Distributed (Non-indépendance et distribution non identique des données)
FN	False Negative (Faux Négatif)
FP	False Positive (Faux Positif)
TP	True Positive (Vrai Positif)
TN	True Negative (Vrai Négatif)

FPR	False Positive Rate (Taux de faux positifs)
TNR	True Negative Rate (Taux de vrais négatifs)
FANET	Flying Ad hoc NETwork (Réseau Ad Hoc de Drones)

INTRODUCTION GÉNÉRALE

Avec l'évolution des technologies de communication et l'expansion rapide d'Internet, de nombreuses applications sont devenues indispensables dans notre quotidien, telles que l'apprentissage à distance, le commerce électronique, les paiements en ligne, la messagerie instantanée ou encore les visioconférences. De nouvelles technologies émergent également, telles que les distributeurs intelligents, les systèmes domotiques, les véhicules autonomes ou les usines connectées. Cependant, cette omniprésence des systèmes informatisés s'accompagne d'un accroissement considérable des vulnérabilités en matière de cybersécurité. Les réseaux modernes et les systèmes d'information connectés sont aujourd'hui exposés à de véritables menaces, qu'elles soient intentionnelles ou accidentelles. Il ne se passe presque plus un jour sans qu'une nouvelle faille de sécurité ou une attaque informatique ne soit signalée : vol de données personnelles, fraudes bancaires, espionnage industriel, logiciels malveillants, ou attaques par déni de service (DDoS).

Les systèmes de détection d'intrusion (IDS – Intrusion Detection Systems) constituent l'une des principales lignes de défense contre les cyberattaques. Ils permettent de surveiller l'activité réseau et de détecter les comportements anormaux ou suspects. Toutefois, les méthodes classiques de détection, basées sur des signatures ou des règles fixes, se révèlent de plus en plus inefficaces face à des attaques nouvelles, polymorphes ou sophistiquées. Pour pallier ces limites, les chercheurs et les industriels se tournent désormais vers l'intelligence artificielle, en particulier l'apprentissage automatique (Machine Learning) et l'apprentissage profond (Deep Learning).

L'apprentissage profond, sous-domaine de l'intelligence artificielle, connaît un succès croissant grâce à sa capacité à apprendre directement à partir de grandes quantités de données, sans nécessiter une expertise humaine poussée pour l'extraction des caractéristiques. Parmi les modèles les plus performants figurent les réseaux de neurones convolutifs (CNN – Convolutional Neural Networks), qui ont été initialement développés pour le traitement d'images, mais qui trouvent aujourd'hui des applications prometteuses dans la détection d'anomalies et la classification de trafic réseau, notamment dans les environnements

complexes et hétérogènes comme l'Internet des Objets (IoT).C'est dans ce cadre que s'inscrit ce mémoire, dont l'objectif principal est de concevoir, implémenter et évaluer un système de détection d'intrusion basé sur un CNN, en s'appuyant sur un jeu de données réaliste et riche en scénarios d'attaques : le Edge-IIoTset. Ce dataset, conçu pour représenter fidèlement un environnement IoT industriel, offre une diversité de types d'attaques et de trafic légitime, ce qui en fait une référence pertinente pour évaluer les performances des modèles de détection.

Le présent mémoire est structuré comme suit :

Le première chapitre présente les fondements de l'intelligence artificielle, du Machine Learning et du Deep Learning, avec un focus particulier sur les réseaux neuronaux convolutifs.

Le deuxième chapitre passe en revue les travaux liés à la détection d'intrusions dans les environnements IoT, en comparant les approches classiques et intelligentes.

Le troisième chapitre est consacré à la conception, la mise en œuvre et l'évaluation expérimentale du système proposé, basé sur un CNN entraîné sur le dataset Edge-IIoTset.

La problématique

Avec l'expansion rapide de l'Internet des Objets (IoT) et la multiplication des dispositifs connectés dans des secteurs sensibles tels que l'industrie, la santé ou les transports, les systèmes informatiques deviennent de plus en plus vulnérables à des cybermenaces variées et sophistiquées. Dans ce contexte, les systèmes de détection d'intrusion (IDS) traditionnels montrent leurs limites face à la complexité des flux de données, à la diversité des attaques et à l'ampleur du volume à traiter. Dès lors, une question centrale se pose : comment concevoir un système de détection d'intrusion intelligent, capable d'exploiter la puissance de l'apprentissage profond, notamment des réseaux de neurones convolutifs (CNN), afin de détecter efficacement et avec précision les comportements malveillants dans les environnements IoT hétérogènes, tout en minimisant les faux positifs et en garantissant une évaluation rigoureuse à l'aide de jeux de données réalistes comme Edge-IIoTset ?

Le but de travail

L’objectif principal de ce travail est de concevoir, implémenter et évaluer un système de détection d'intrusion (IDS) basé sur un réseau de neurones convolutif (CNN), adapté aux environnements IoT. Pour cela, nous utilisons le jeu de données Edge-IIoTset, spécialement conçu pour représenter des scénarios d’attaques réalistes dans le contexte de l’Internet des Objets industriel.

Chapitre I

Définition et conception de base

Chapitre I : Définition et conception de base

I.1. INTRODUCTION

Dans un monde où l'hyperconnexion devient la norme, l'Internet des Objets (IoT) s'impose comme une révolution technologique majeure, modifiant profondément la manière dont les systèmes, les individus et les objets interagissent. Des capteurs médicaux aux véhicules autonomes, en passant par les maisons intelligentes et les infrastructures critiques, les objets connectés s'intègrent de plus en plus dans notre quotidien, générant des volumes massifs de données et multipliant les échanges réseau[1].

Mais cette transformation s'accompagne d'un revers préoccupant : l'exposition accrue aux risques de cyberattaques. Les dispositifs IoT, souvent déployés dans des environnements hétérogènes et contraints en ressources, manquent cruellement de mécanismes de sécurité robustes. La nature distribuée, la diversité des protocoles, et le manque de normalisation rendent la tâche de sécurisation particulièrement ardue. Face à ces défis, les systèmes de détection d'intrusion (IDS) constituent une solution de choix pour surveiller les comportements du réseau, identifier les anomalies et prévenir les intrusions potentielles[1].

Cependant, les approches classiques de détection, largement basées sur des règles prédéfinies ou des signatures connues, montrent rapidement leurs limites. Elles peinent à faire face à des menaces de plus en plus sophistiquées, adaptatives et parfois inconnues (zero-day attacks). Dans ce contexte évolutif, l'intelligence artificielle (IA), et plus précisément l'apprentissage automatique (ML) et l'apprentissage profond (DL), ouvre de nouvelles perspectives prometteuses. Ces méthodes permettent aux systèmes de sécurité de ne plus simplement réagir, mais d'apprendre, de s'adapter, et même d'anticiper les comportements anormaux[5].

Les réseaux de neurones convolutifs (CNN), au cœur du deep learning, se distinguent par leur capacité à extraire automatiquement des caractéristiques pertinentes à partir de données complexes, sans nécessiter une ingénierie manuelle des variables. Appliqués à la détection d'intrusion, ils permettent une analyse fine des flux réseau et améliorent significativement les

performances des IDS, notamment en termes de taux de détection et de réduction des faux positifs[46].

Ce premier chapitre pose les bases conceptuelles nécessaires à la compréhension de cette problématique. Il s'articule autour de cinq axes complémentaires. D'abord, nous introduisons les principes fondamentaux de l'IoT et ses principaux défis en matière de sécurité. Ensuite, nous explorons la genèse et l'évolution des systèmes de détection d'intrusion, avant d'examiner les apports successifs de l'intelligence artificielle, du machine learning et du deep learning dans ce domaine.

Ce cadre théorique constituera le socle à partir duquel nous développerons, dans les chapitres suivants, une solution de détection d'intrusion intelligente, adaptée aux contraintes spécifiques de l'IoT moderne.

I.2. INTERNET DES OBJETS

I.2.1. Définition d'internet des objets :

L'Internet des Objets (IoT) est un concept technologique en pleine expansion, bien qu'il ne bénéficie pas encore d'une définition universellement reconnue. Cette absence de consensus reflète l'évolution rapide et multidimensionnelle d'un domaine considéré aujourd'hui comme l'un des plus innovants et prometteurs dans le secteur des technologies de l'information.

Dans ce contexte, le terme « objet » désigne tout dispositif physique – capteurs, appareils ménagers ou infrastructures industrielles – capable de collecter, traiter et transmettre des données à travers un réseau, sans intervention humaine directe. Équipés de technologies embarquées, ces dispositifs interagissent de manière autonome avec leur environnement, contribuant ainsi à des processus de décision intelligents [38].

Ce système repose sur une infrastructure dans laquelle chaque unité possède une identité numérique propre et une présence active sur Internet. L'objectif principal est d'assurer une intégration fluide entre les mondes physique et virtuel, à travers des services et applications intelligents. Cette convergence est rendue possible grâce à la communication de machine à machine (M2M), qui constitue l'un des piliers techniques essentiels, en facilitant les échanges d'informations entre les objets connectés et les plateformes d'analyse hébergées à distance

(souvent dans le cloud)[38].

Selon le dictionnaire Oxford, ce paradigme peut être défini comme « l'interconnexion via Internet de dispositifs informatiques intégrés dans les objets du quotidien, leur permettant d'envoyer et de recevoir des données ». Cette définition met en lumière l'évolution des objets, qui passent d'un statut passif à celui d'acteurs numériques actifs, communicants et autonomes.

Ce réseau intelligent forme un écosystème complexe, souvent hiérarchisé, constitué de multiples couches et systèmes imbriqués. Par exemple, les dispositifs d'une maison intelligente peuvent être intégrés à un réseau urbain plus vaste, contribuant à l'émergence de villes intelligentes. Bien qu'en apparence structurée de manière simple, cette réalité technologique est beaucoup plus sophistiquée, et nécessite des approches rigoureuses en matière de sécurité, d'interopérabilité et de gouvernance des données[38].

I.2.2. Les composants de IoT:

Il y a cinq éléments clés de l'IoT, dont l'élément principal est l'objet connecté. Cet objet peut être conçu dès le départ pour être connecté ou être équipé d'une connectivité ajoutée par la suite. Il collecte et gère les données provenant de capteurs, tout en transmettant et recevant des instructions pour effectuer des opérations. Pour assurer ces fonctions, une source d'alimentation est généralement nécessaire, surtout si l'objet réalise un prétraitement des données[11]:

- **Les capteurs (Sensors):** sont des dispositifs électroniques qui transforment des grandeurs physiques comme la température, la pression ou le débit en signaux électroniques. Ils collectent ces données et les envoient à un serveur ou à une plate-forme IoT pour analyse, traitement et utilisation. Ces informations peuvent ensuite servir à la prise de décisions ou à l'automatisation de systèmes. Parmi les types de capteurs les plus courants, on trouve ceux qui mesurent la position, la proximité, le mouvement, l'accélération, la lumière, la température, l'humidité, le son, les vibrations, les substances chimiques, les gaz, le flux et la pression.
- **Les actionneurs (Actuators):** est un dispositif physique qui transforme une information

numérique en une action tangible dans le monde réel. Dans le contexte de l'Internet des Objets (IoT), un actionneur permet de contrôler ou de modifier l'état d'un objet physique en fonction des données reçues d'autres dispositifs connectés, par exemple: des écrans d'affichage, des alarmes, des caméras, des haut-parleurs,etc.

- **L'alimentation (Power / Energy):** l'un des plus grands défis des capteurs IoT est la gestion de leur consommation énergétique. Comme ils fonctionnent souvent sur batterie, leur autonomie est cruciale et peut s'étendre sur plusieurs années. Cela nécessite des solutions optimisées pour prolonger leur durée de vie tout en maintenant leurs performances.
- **Le réseau de capteurs (Wireless Sensor Network) :** un réseau de capteurs est un ensemble de petits dispositifs connectés capables d'échanger des données sans fil. Chaque capteur peut envoyer et recevoir des informations, mais pour assurer une communication efficace et rendre le système accessible et interopérable, ils doivent fonctionner de manière coordonnée au sein d'un réseau. Ce réseau permet ainsi de collecter, transmettre et analyser des données en temps réel, tout en optimisant la communication entre les capteurs.
- **La connectivité (Communication):** est un élément clé des objets IoT, leur permettant de communiquer avec leur environnement. Grâce à une antenne radiofréquence (RF), ces objets peuvent se connecter à des réseaux pour envoyer des informations telles que leur identité, leur état, des alertes ou encore les données captées. Ils peuvent également recevoir des commandes et des mises à jour en retour. Ce module de connectivité joue un rôle essentiel dans le fonctionnement et la gestion du cycle de vie de chaque objet connecté.

I.2.3. les caractéristiques fonctionnelles d'internet des objets :

L'Internet des Objets (IoT) permet à une grande diversité de dispositifs, souvent très différents les uns des autres, de se connecter de manière dynamique, généralement via des connexions sécurisées. Bien que souvent invisibles pour l'utilisateur final, ces connexions rendent possible une communication fluide et continue entre les objets. Les principales fonctionnalités offertes par l'IoT sont illustrées dans la figure suivante :

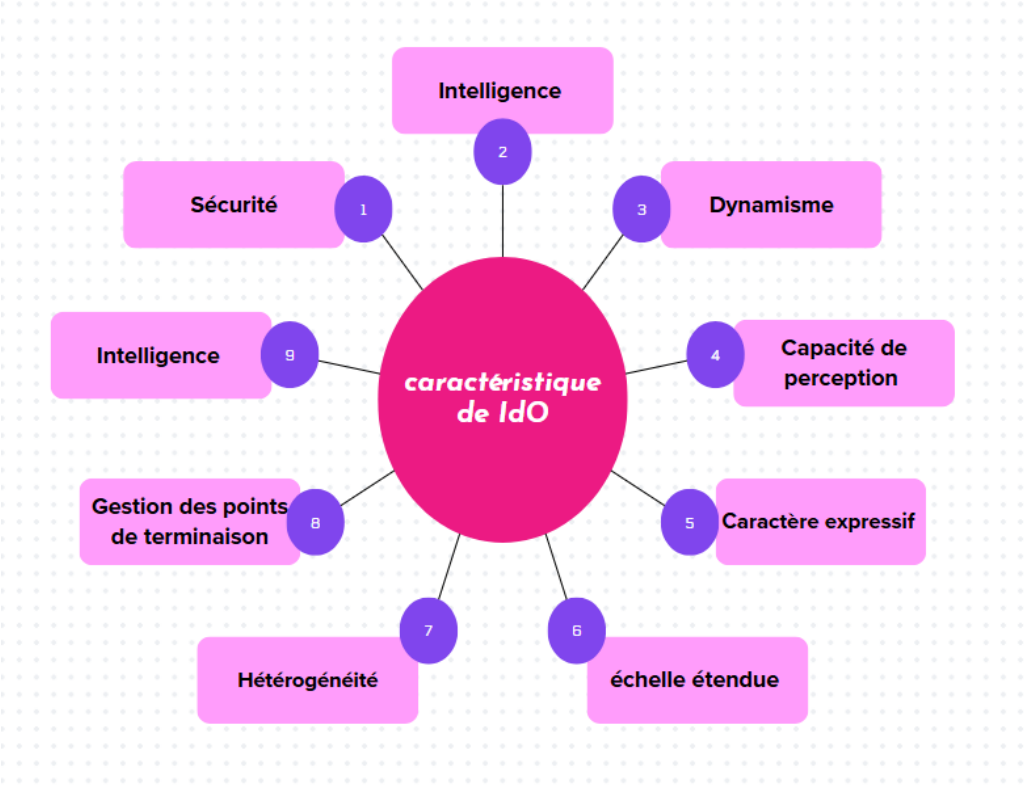


Figure I. 1 Caractéristique d'IoT

- **Interconnectivité :** L'IoT repose sur une interconnexion généralisée entre objets physiques et systèmes numériques. Chaque dispositif, qu'il s'agisse d'un capteur, d'un actionneur ou d'une machine, devient un nœud actif du réseau, capable de communiquer avec d'autres objets ou plateformes via des protocoles standards. Cette interconnectivité permet la création d'une infrastructure globale unifiée, facilitant l'échange de données à grande échelle [10].
- **Intelligence :** Les objets connectés ne se limitent pas à transmettre des données brutes. Ils intègrent des capacités de traitement local (edge computing) ou distant (cloud computing), leur permettant d'interpréter leur environnement et de prendre des décisions autonomes. Cette intelligence contextuelle favorise une adaptation dynamique aux conditions d'usage ou aux besoins de l'utilisateur [10]
- **Dynamisme :** Les systèmes IoT sont par nature évolutifs et réactifs. Les objets peuvent être ajoutés, déplacés, modifiés ou reconfigurés automatiquement selon leur contexte. Cette flexibilité est essentielle pour gérer des environnements complexes, hétérogènes et en constante évolution (ex. : villes intelligentes, logistique mobile, domotique) [10]

- **Capacité de perception :** les objets connectés sont équipés de capteurs qui leur permettent de collecter des données physiques et environnementales en temps réel (température, humidité, mouvement, qualité de l'air, etc.). Ces données forment la base des processus d'analyse, de prédiction et d'automatisation, constituant le lien fondamental entre le monde physique et le système numérique.[10]
- **Caractère expressif :** au-delà de la collecte de données, les objets doivent être capables de restituer des informations de manière compréhensible et utile pour l'utilisateur final ou pour d'autres systèmes. Cela se traduit par des interfaces interactives (GUI), des signaux visuels, sonores, ou par l'exécution directe d'actions (comme l'activation d'un dispositif). Ce caractère expressif renforce l'interaction homme-machine et améliore l'expérience utilisateur.[10]
- **Échelle étendue :** l'IoT est conçu pour fonctionner à très grande échelle, avec des milliards de dispositifs potentiellement connectés dans le monde. Cela implique des architectures capables de supporter des volumes massifs de données, une diversité de protocoles, et une gestion efficace de ressources réseau (bande passante, énergie, etc.).[10]
- **Hétérogénéité :** les objets IoT varient considérablement en termes de conception matérielle, de capacités logicielles, de protocoles de communication et de formats de données. Cette diversité exige des normes d'interopérabilité robustes afin de permettre une intégration fluide entre dispositifs de fabricants et de technologies différents.[10]
- **Gestion des points de terminaison :** un système IoT performant repose sur une gestion centralisée ou distribuée des points de terminaison, c'est-à-dire des objets qui collectent, traitent ou reçoivent des données. Cette gestion inclut l'identification unique des dispositifs, leur authentification, leur mise à jour logicielle, leur maintenance à distance, et la supervision de leur fonctionnement pour garantir la fiabilité du réseau.[10]
- **Sécurité et confiance numérique :** la sécurité est une exigence critique dans les systèmes IoT. Elle concerne la protection des communications, l'intégrité des données, la confidentialité des informations collectées et la résilience du réseau face aux attaques (cyberintrusions, détournements, etc.). Des mécanismes tels que le chiffrement,

l'authentification forte, la segmentation du réseau et la détection d'anomalies doivent être intégrés dès la conception pour instaurer une confiance durable dans les services basés sur l'IoT [10]

I.2.4. Architecture de l'IoT :

L'architecture de l'Internet des Objets (IoT) peut être représentée comme une structure en couches, généralement organisée en trois niveaux technologiques distincts, chacun jouant un rôle clé dans le fonctionnement global du système.
Voici une représentation schématique de l'architecture en trois couches utilisées dans l'IoT :

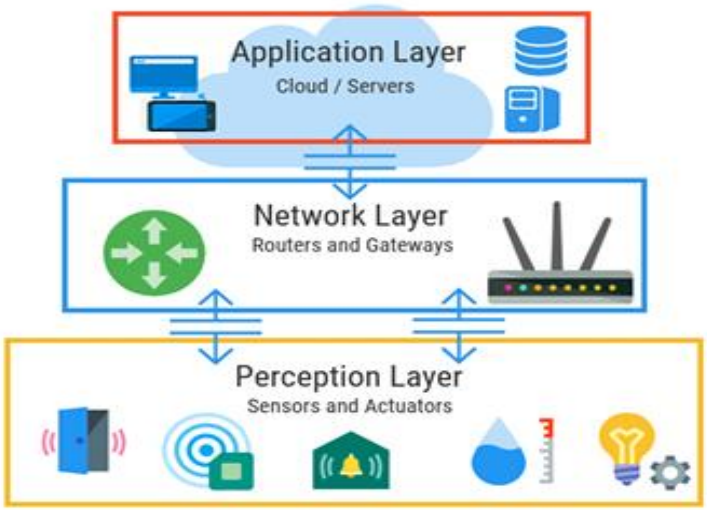


Figure I. 2: Architecture en 3 couches [12]

- **La couche de perception:** est le premier niveau de fonctionnement de l'IoT, chargé de collecter les informations sur l'environnement. Elle repose sur différents dispositifs comme les cartes à puce, les étiquettes RFID et les réseaux de capteurs, qui permettent d'identifier et de suivre des objets en temps réel, où qu'ils se trouvent [12] . Grâce au système RFID, chaque objet reçoit un identifiant unique appelé EPC (Electronic Product Code), qui permet de l'identifier précisément. En plus de cet identifiant, d'autres informations comme le fabricant, la catégorie du produit, ainsi que ses dates de fabrication et d'expiration peuvent être associées. Cette couche joue donc un rôle clé dans la traçabilité et la gestion des objets connectés [12] .
- **La couche réseau:** assure le transport des données collectées par les capteurs vers Internet ou d'autres systèmes connectés. Pour cela, elle utilise des ordinateurs, des

réseaux filaires ou sans fil, ainsi que divers équipements de communication. Son rôle principal est de garantir un transfert fiable et sécurisé des informations, en s'appuyant également sur des mécanismes de la couche de transport pour assurer une transmission efficace [12] .

- **La couche application:** est chargée d'analyser les données reçues et de prendre des décisions intelligentes pour automatiser certaines actions. Elle permet la connexion, l'identification et le contrôle des objets et appareils IoT. Grâce à des technologies avancées comme le cloud computing, elle traite ces informations afin d'optimiser la gestion des objets connectés, en décidant quoi faire et quand le faire de manière intelligente et efficace [12] .

I.2.5. Exigences de sécurité dans l'Internet des Objets (IoT) :

En raison des nombreuses vulnérabilités inhérentes aux systèmes IoT, la mise en place de mécanismes de sécurité robustes s'avère indispensable. Pour garantir un fonctionnement fiable et protégé, un système IoT doit répondre aux exigences fondamentales de sécurité traditionnelles, telles que décrites ci-dessous [12] :

- **Authenticité :** les données échangées dans un réseau IoT doivent pouvoir être vérifiées afin de garantir leur origine. Il est crucial de s'assurer que les informations proviennent bien de dispositifs ou d'étiquettes électroniques authentifiés. Cela permet d'éviter les usurpations d'identité entre les objets connectés.
- **Intégrité :** l'intégrité des données garantit qu'elles n'ont pas été modifiées, altérées, copiées ou remplacées au cours de leur transmission. Dans un environnement où les capteurs, actionneurs et plateformes communiquent en continu, préserver l'authenticité des informations est fondamental pour éviter les manipulations malveillantes.
- **Confidentialité :** la confidentialité concerne la protection des données sensibles – qu'elles soient personnelles, commerciales ou techniques. Le système doit empêcher toute divulgation non autorisée, notamment en utilisant des techniques de chiffrement et des politiques de contrôle d'accès adaptées aux contraintes de l'IoT.
- **Protection de la vie privée :** outre la confidentialité technique, l'IoT doit aussi garantir la vie privée des utilisateurs, en empêchant l'exploitation abusive de leur identité, de leurs

habitudes ou de leurs préférences. Cette exigence devient d'autant plus critique dans les environnements domestiques ou médicaux.

- **Disponibilité :**les services IoT doivent rester accessibles aux utilisateurs autorisés en toutes circonstances. Les attaques par déni de service (DoS), qui visent à rendre les dispositifs inaccessibles, représentent une menace majeure. Il est donc indispensable d'implémenter des mécanismes de résilience pour garantir une disponibilité continue des systèmes.

I.2.6. Vulnérabilités dans internet des objets

Avec l'essor rapide de l'Internet des Objets, la sécurité des dispositifs connectés devient une préoccupation centrale. Bien qu'ils apportent confort, efficacité et innovation dans de nombreux domaines, ces objets restent vulnérables en raison de leur conception légère, de leur connectivité permanente et de leur déploiement dans des environnements souvent peu sécurisés. Identifier et comprendre ces failles est indispensable pour protéger les systèmes et les individus qui les utilisent au quotidien .Voici certains des dangers potentiels auxquels les objets connectés sont confrontés [34] :

- **Sécurité physique insuffisante :** la majorité des dispositifs IoT sont déployés dans des environnements ouverts, souvent sans aucune supervision. Cette absence de protection physique rend ces appareils particulièrement vulnérables aux manipulations externes. Un individu malveillant peut ainsi facilement accéder à l'appareil, le démonter, en extraire des données sensibles, dupliquer le micrologiciel (firmware) ou encore exposer les algorithmes cryptographiques qu'il utilise. Ces actions peuvent compromettre non seulement l'intégrité de l'appareil, mais aussi celle du réseau dans lequel il est intégré.
- **Limites énergétiques :** les objets connectés sont, dans la grande majorité des cas, alimentés par des batteries ou des sources d'énergie à faible autonomie. Cette contrainte énergétique représente une faille exploitable. Un attaquant peut inonder le dispositif de requêtes, qu'elles soient valides ou corrompues, afin d'en épuiser rapidement l'énergie. Ce type d'attaque, souvent silencieuse, perturbe la disponibilité du service et empêche son fonctionnement normal.

- **Authentification faible :** les limites en matière de calcul et de mémoire propres aux dispositifs IoT compliquent l'implémentation de protocoles d'authentification avancés. Cette situation facilite les attaques visant à insérer des nœuds malveillants, falsifier des données ou intercepter des communications. De plus, les clés d'authentification sont fréquemment mal protégées, stockées sans chiffrement ou transférées sans précaution, ce qui les expose à des pertes, des corruptions ou des vols. Même les systèmes d'authentification les plus robustes deviennent inefficaces en l'absence d'une gestion rigoureuse des clés.
- **Chiffrement inadapté :** dans les systèmes critiques tels que les réseaux électriques intelligents, les usines automatisées ou les bâtiments connectés, la protection des données est primordiale. Le chiffrement constitue une barrière de sécurité indispensable. Toutefois, les contraintes matérielles des dispositifs IoT limitent la puissance et l'efficacité des algorithmes cryptographiques embarqués. Un attaquant compétent peut exploiter ces failles pour contourner les mesures de sécurité, accéder à des données confidentielles ou modifier le comportement du système.
- **Ports non sécurisés :** de nombreux objets connectés présentent des ports de communication ouverts par défaut, sans utilité directe pour leur fonctionnement. Ces ports deviennent des points d'entrée vulnérables par lesquels un acteur malveillant peut établir une connexion, analyser le système ou exploiter des failles non corrigées. Cette mauvaise configuration initiale augmente considérablement la surface d'attaque.
- **Contrôle d'accès défaillant :** la gestion des identifiants dans l'écosystème IoT est souvent négligée. Plusieurs dispositifs sont livrés avec des identifiants par défaut que les utilisateurs ne sont pas incités à modifier. Par ailleurs, les systèmes de gestion cloud associés n'imposent généralement pas de politiques de mot de passe robustes. En conséquence, de nombreux utilisateurs conservent des privilèges élevés, ce qui facilite l'accès non autorisé aux dispositifs. Cette défaillance compromet la sécurité des données et menace l'intégrité du réseau.

I.2.7. Domaines d'application de IOT

- **Santé:** les objets connectés jouent un rôle essentiel dans le domaine médical. Ils

permettent de suivre les signes vitaux des patients à distance (comme la fréquence cardiaque ou la tension), facilitant une détection précoce des anomalies. Ils servent aussi à gérer les équipements, les stocks médicaux et le respect des traitements.[35]

- **Industrie et fabrication :**

Le domaine industriel constitue l'un des principaux terrains d'application de l'Internet des Objets (IoT), avec une connectivité étendue entre les dispositifs, que ce soit à l'intérieur ou à l'extérieur des sites de production. En interne, les projets d'automatisation et de contrôle basés sur l'IoT sont de plus en plus nombreux, intégrant des solutions complètes de « smart factory ». Ces solutions comprennent notamment la surveillance des chaînes de production, l'utilisation d'équipements portables (wearables), la réalité augmentée sur les lignes de fabrication, ainsi que le contrôle à distance d'automates programmables industriels (PLC) et de systèmes de contrôle qualité automatisés.[35].

À l'extérieur des installations, les applications typiques incluent la gestion à distance des machines connectées et la surveillance d'équipements dispersés. Plusieurs études de cas mettent en évidence que les principaux moteurs de l'adoption de l'IoT par les fabricants d'équipements d'origine (OEM) sont la réduction des temps d'arrêt et la réalisation d'économies de coûts.[35].

Un exemple marquant de cette transformation est l'introduction de véhicules autonomes dans les usines. Grâce à la convergence de technologies comme la robotique, les capteurs, les caméras 3D, la connectivité 5G, les logiciels avancés et l'intelligence artificielle, des essaims de robots mobiles circulent désormais librement sur les sites de production. Ces véhicules autonomes peuvent être coordonnés de manière fine pour exécuter des tâches répétitives de manière fiable, avec un minimum d'intervention humaine. Ils contribuent ainsi à améliorer la productivité, la sécurité, la collecte de données et à réduire les coûts. Ils sont particulièrement utiles pour la manipulation et le transport de composants. Cela permet également aux opérateurs humains de se concentrer sur des tâches à plus forte valeur ajoutée, comme l'assemblage ou le contrôle de qualité. Par exemple, le fabricant italien Automotive Systems Manufacturer Faurecia (ASMF) utilise des robots mobiles développés par Mobile Industrial Robots pour optimiser l'efficacité de sa logistique [35].

- **Villes intelligentes :**

Grâce aux capacités offertes par l'Internet des Objets (IoT), de nombreuses villes à travers le monde deviennent progressivement plus intelligentes et interconnectées sur le plan numérique. En exploitant les vastes quantités de données générées par les dispositifs IoT déployés dans divers systèmes urbains (infrastructures, transports, sécurité, etc.), les autorités locales peuvent améliorer la qualité de vie des citoyens.

Les villes intelligentes sont en mesure de prendre des décisions plus éclairées, fondées sur des données en temps réel concernant les besoins en infrastructures, les exigences en matière de mobilité, la sécurité publique ou encore la criminalité. Une étude récente révèle que l'utilisation d'applications urbaines intelligentes permet d'améliorer certains indicateurs de qualité de vie (tels que la criminalité, les embouteillages ou la pollution) de l'ordre de 10 à 30 %.L'intégration des technologies IoT dans la vie quotidienne, que ce soit à domicile, dans les moyens de transport ou dans les espaces publics urbains, contribue à rendre l'expérience de vie plus fluide, plus confortable et plus efficace. À travers l'automatisation des tâches courantes et l'amélioration des services de santé et de bien-être, l'Internet des Objets offre la promesse d'une meilleure qualité de vie[35].

- **l'entretien des véhicules et la notion de propriété automobil**e

L'Internet des Objets (IoT) joue un rôle clé dans l'évolution des technologies d'entretien prédictif des véhicules. Grâce à des capteurs connectés et à des outils de communication en temps réel, il est désormais possible de surveiller en continu l'état de fonctionnement des différents composants d'un véhicule. Ces dispositifs collectent des données de performance, les transmettent au cloud, puis les analysent afin d'évaluer les risques de dysfonctionnements potentiels, qu'ils soient matériels ou logiciels. Une fois les données traitées, le conducteur est alerté des réparations ou entretiens nécessaires, lui permettant d'agir en amont pour éviter pannes imprévues ou accidents. Ainsi, la connectivité IoT contribue à réduire considérablement les arrêts non planifiés et à améliorer la sécurité routière[35].

Par ailleurs, l'IoT transforme en profondeur la conception même de la propriété automobile.

Selon une étude récente menée par Tony et James [46], la possession individuelle de voitures pourrait diminuer de 80 % d'ici 2030. Cette tendance est déjà perceptible, notamment en milieu urbain, où de plus en plus de citadins choisissent de ne pas posséder de véhicule personnel. À la place, ils privilégient des alternatives telles que les plateformes de covoiturage, les services de location courte durée ou encore les systèmes de partage de vélos et de trottinettes connectés, proposés par des entreprises comme Uber, DiDi ou Alibaba. Cette évolution est également soutenue par l'amélioration continue des réseaux de transports publics, rendant la mobilité urbaine plus flexible, durable et accessible [35].

Cette figure présente les différents domaines dans lesquels l'IoT trouve ses applications :

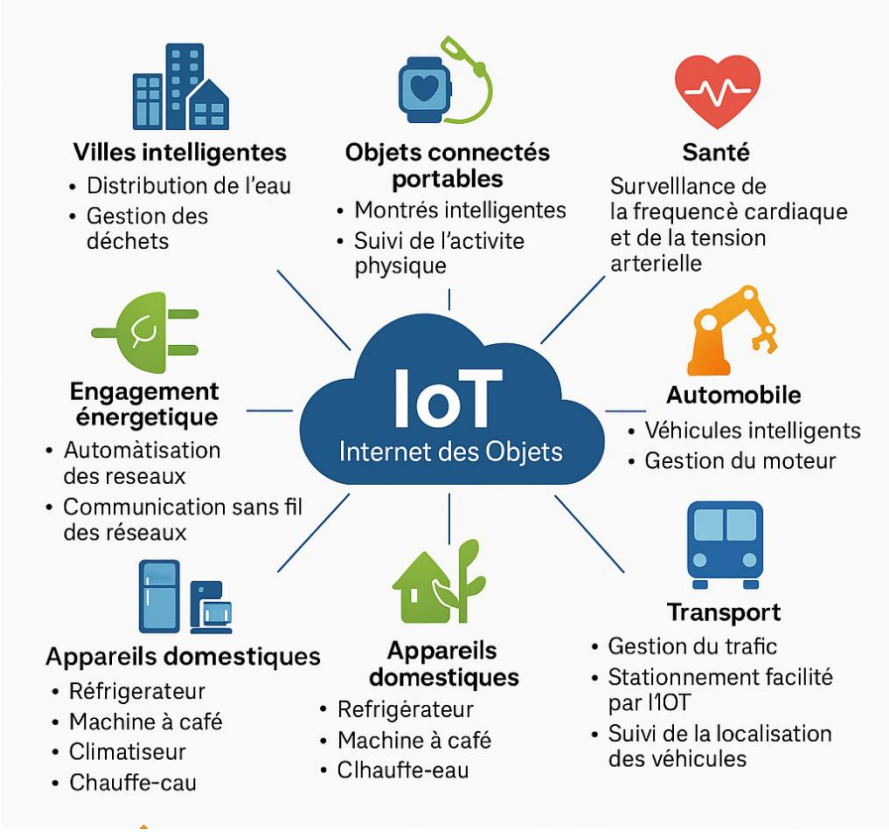


Figure I. 3: Domaines d'application

I.3. SYSTÈME DE DÉTECTION D'INTRUSION

I.3.1. Définition Système de Détection d'Intrusion :

Un système de détection d'intrusion (Intrusion Detection System – IDS) définit un ensemble de composants matériels et logiciels interconnectés à finalité unique de surveillance, en temps réel et de manière cachée, du trafic sur un réseau ou de l'activité d'un système informatique pour identifier toute activité jugée suspecte ou non autorisée qui impacte la confidentialité (accès exclusif aux données), l'intégrité (protection contre les modifications frauduleuses) ou la disponibilité (préservation de l'accès aux services) des ressources informatiques. L'IDS se superpose à la mécanique de contrôle d'accès des utilisateurs et de surveillance du réseau, détectant des comportements malfaisants qui ne sont pas toujours détectés par les mécanismes classiques de sécurité. Il procède à une dynamique d'évaluation des événements pour différencier l'usage normal du système des signes d'une attaque. Il constitue de ce fait, aujourd'hui, un module structurant de l'architecture de sécurité du système d'information pour anticiper la prévention et limiter les impacts des menaces numériques[7][29].

I.3.2. Fonctionnement et rôle d'un Système de Détection d'Intrusion :

La collecte d'informations à partir de points critiques d'un réseau constitue l'un des principes fondamentaux de la détection des intrusions. Ces données sont ensuite analysées à l'aide de règles prédéfinies afin d'identifier la présence éventuelle d'activités malveillantes et de classifier les attaques en cours. Ce processus est communément désigné sous le nom de système de détection d'intrusion (IDS) [36].

Les sources de données au sein des réseaux renferment généralement des informations précieuses, telles que les modifications apportées à des fichiers sensibles, l'exécution de programmes inhabituels ou encore le trafic réseau. Une fois ces données extraites, elles sont soumises à une étape de traitement et d'analyse — appelée analyse de l'information — qui permet de détecter d'éventuelles intrusions. Pour ce faire, on utilise fréquemment des

techniques d'exploration de données, des méthodes d'appariement de motifs ainsi que des approches d'apprentissage intégré.

La réponse à une intrusion détectée repose sur les résultats de l'analyse, et peut inclure des actions post-traitement telles que l'archivage des données pour un usage ultérieur, la reconfiguration des routeurs, ou d'autres mesures équivalentes [36].

Positionné en aval du pare-feu, le système de détection d'intrusion joue un rôle complémentaire essentiel. Tandis que le pare-feu filtre le trafic entrant selon des règles statiques, l'IDS surveille en temps réel les flux de données à la recherche de comportements anormaux. Lorsqu'une activité suspecte est identifiée, l'IDS peut agir conjointement avec le pare-feu pour bloquer la connexion et renforcer ainsi la sécurité du système global [36].

Le fonctionnement typique d'un système de détection d'intrusion est illustré dans la figure suivante :

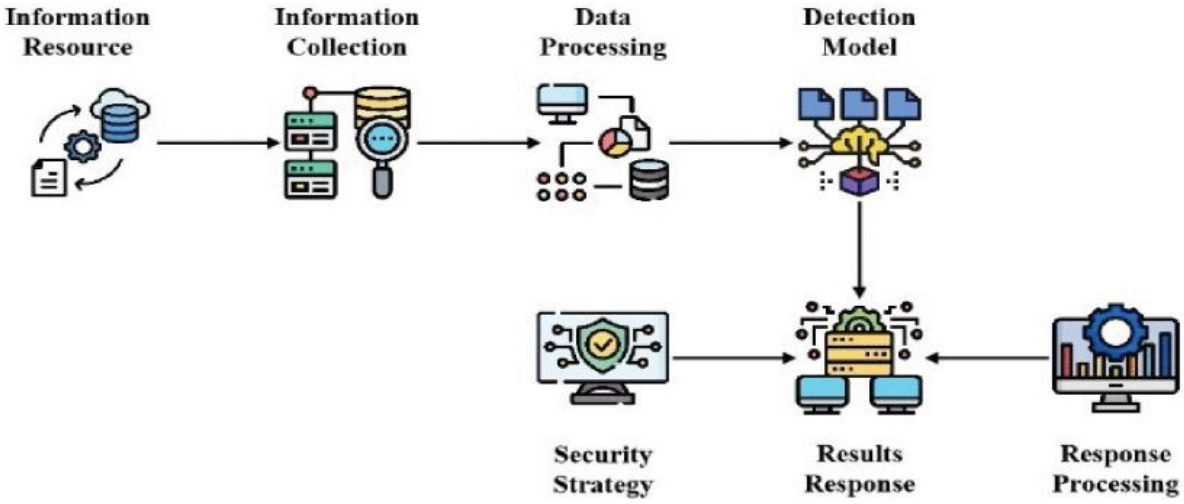


Figure I. 4: Le fonctionnement de détection d'intrusion[36].

I.3.3. Méthodes de détection dans les systèmes IDS

Les dispositifs de détection d'intrusion (IDS) reposent principalement sur deux approches d'analyse complémentaires [15] :

I.3.3.1. La détection basée sur signatures:

la détection par signatures consiste à reconnaître des modèles d'attaques connus pour lesquels un ensemble de signatures a été défini dans une bibliothèque. Quand un événement réseau est identifié comme un de ces modèles, une alerte est lancée automatiquement. Cette technique

est particulièrement efficace pour détecter des attaques de même type récurrents, et elle a généralement un faible taux de faux positifs. Sa principale faille réside dans son incapacité à détecter les attaques zero-day, c'est-à-dire nouvelles ou inédites donc non encore présentes dans la base de données. Elle dépend donc fortement de l'historique des menaces et est de ce fait vulnérable à des techniques d'attaque encore inconnues.

Une illustration du mécanisme de détection basé sur les signatures est fournie ci-dessous :

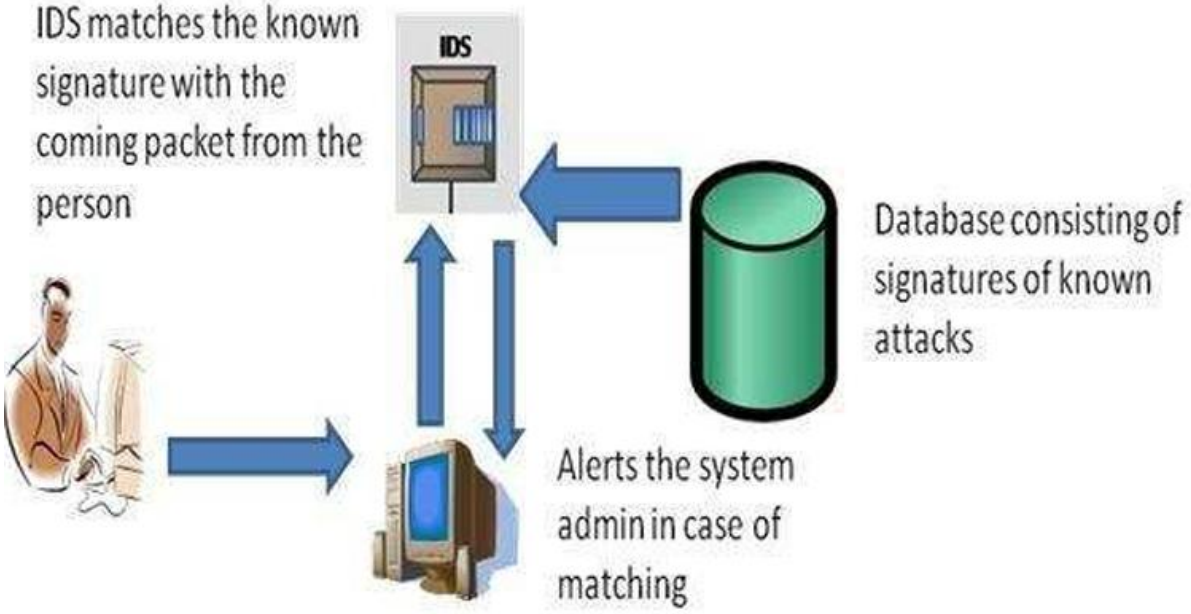


Figure I. 5: La détection basée sur signatures [27]

I.3.3.2. Détection basée sur les anomalies :

la détection des anomalies repose sur une analyse comportementale de tout le système. L'idée consiste à produire un modèle de fonctionnement normal à partir de l'observabilité des flux données, et ce que ce soit à l'intérieur ou à l'extérieur du réseau, considéré comme valeur de référence. Tout écart significatif au modèle est ainsi interprété comme une anomalie, pouvant correspondre à une tentative d'intrusion. Cette approche est la plus efficace capable de découvrir des attaques non connues, y compris celles dites zero-day. L'inconvénient principal consiste cependant en la difficulté à modéliser le comportement normal dans certains environnements, souvent très dynamiques, entraînant pour ce seul motif un taux très élevé de faux positifs. Par ailleurs, l'incapacité à analyser les données chiffrées ouvre la voie à des attaques furtives, qui ne seront pas détectées.

La figure suivante illustre la détection d'anomalies comme méthode alternative :

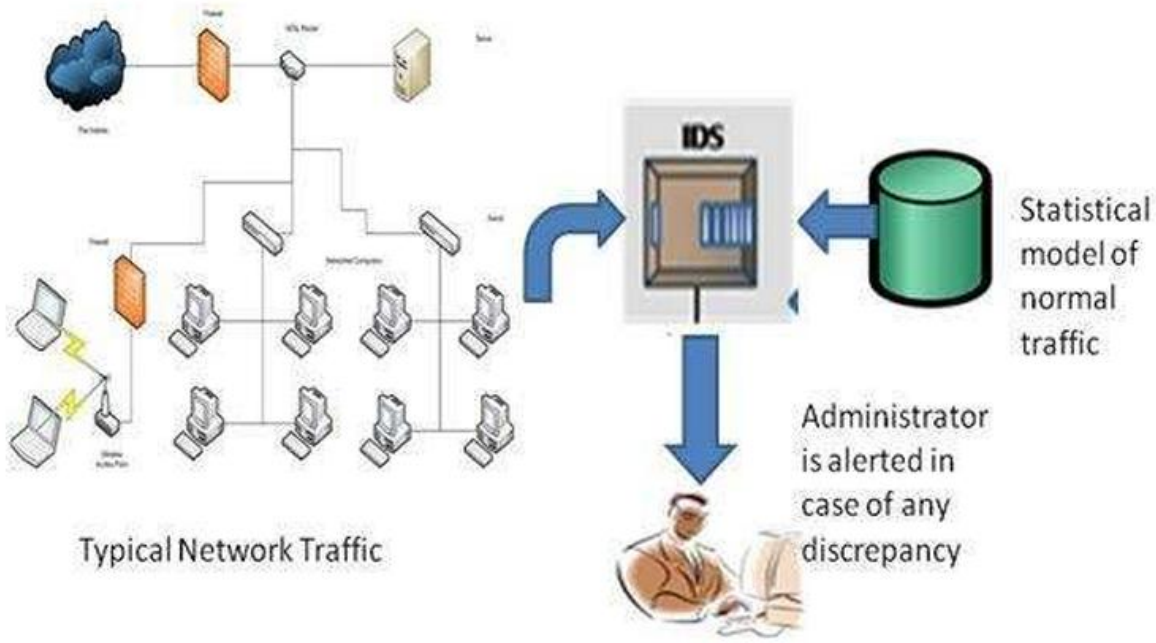


Figure I. 6: La détection basée sur les anomalie[27]

I.3.4. Les types de système détection d'intrusion

Il existe plusieurs types de systèmes de détection d'intrusion (IDS), chacun adapté à un environnement spécifique et à des besoins de sécurité particuliers. Les deux principales catégories sont [15]:

I.3.4.1. Système de détection d'intrusion basé sur l'hôte (HIDS) :

Le HIDS (système d'intrusion sur hôte) est un système qui s'installe directement au niveau d'un appareil (serveur, ordinateur, etc.) et permet de surveiller les activités internes de cette machine. Il fait l'analyse des fichiers logs, des événements systèmes, des zones mémoire, ainsi que de l'action des utilisateurs afin de repérer tout comportement suspect ou anormal.
Son aptitude à observer finement les attaques ciblant un hôte spécifique le rend très précis, et en fait un outil adapté dans les environnements critiques. Par contre, le HIDS possède plusieurs limites. Il est notamment susceptible aux attaques de falsification (tampering), ce

qui requiert de renforcer son au moins en termes de protection. Sa consommation significative de ressources préjudicie aux performances générales de l'appareil hôte, et même si le HIDS permet un dispositif de prévention, celui-ci demeure limité dans la mesure où la plupart des menaces ne sont pas détectées avant même qu'elles n'aient été exécutées sur la machine.[15]

I.3.4.2. Système de détection d'intrusions basé sur le réseau (NIDS) :

le NIDS est un système de détection d'intrusions installé au niveau d'un réseau, généralement à des points de contrôle (passerelles, routeurs), afin de surveiller l'ensemble du trafic entrant et sortant du réseau entre différents hôtes. Il procède à l'inspection des paquets qui circulent sur le réseau et les compare à une base de signatures d'attaques connues ou de modèles « comportementaux » afin de diagnostiquer toute anomalie dans le fonctionnement des applications.

Le principal avantage d'un réseau de type NIDS est sa capacité à offrir une vision d'ensemble du trafic du réseau, sans affecter les machines surveillées. En effet, il fonctionne de manière passive et son utilisation ne peut pas être détectée par un attaquant.

Il faut néanmoins en souligner quelques limites. En période de trafic intense, le NIDS est susceptible de ne pas traiter tout le trafic en temps réel et donc pourrait ne pas détecter d'attaques discrètes. Par ailleurs, il se retrouve démuni pour l'interprétation de données chiffrées ou encapsulées (comme dans le cas de transports VPN), ce qui lui en limite l'efficacité face à certaines menaces sophistiquées recourant au chiffrement.[15]

Ce schéma met en contraste les approches NIDS et HIDS en matière de surveillance réseau :

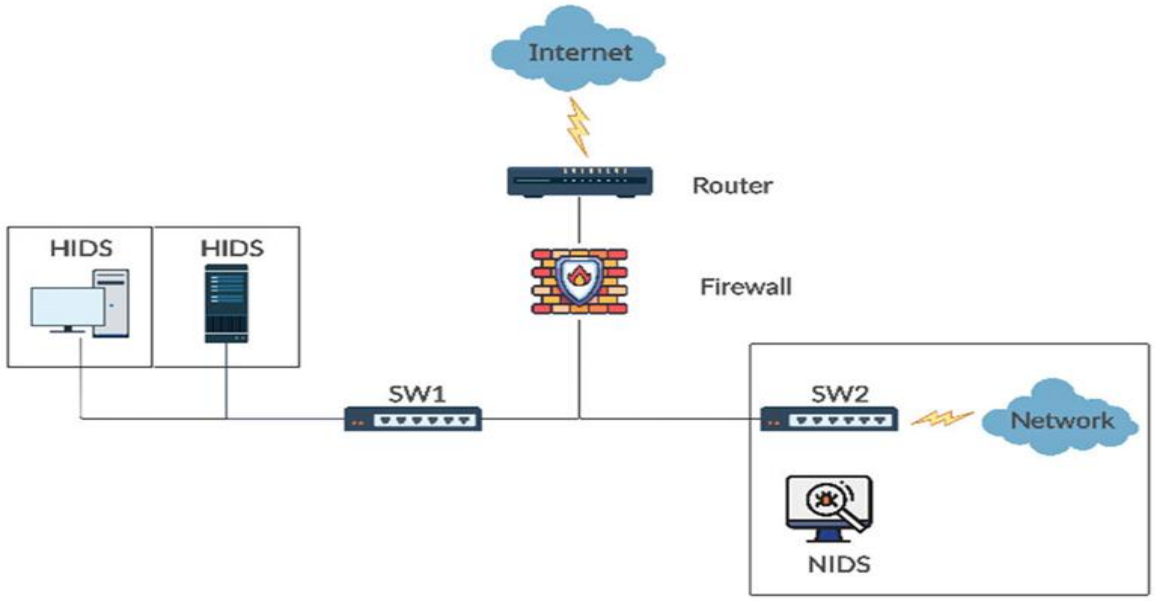


Figure I. 7: schéma générale de NIDS et HIDS [36]

I.4. DEEP LEARNING

I.4.1. Intelligence artificielle :

L'intelligence artificielle (IA) désigne un domaine de l'informatique qui vise à développer des systèmes ou des machines capables d'imiter certaines fonctions cognitives humaines telles que l'apprentissage, le raisonnement, la perception, la prise de décision ou la résolution de problèmes. Elle regroupe un ensemble de techniques et d'approches, allant des systèmes experts aux algorithmes d'apprentissage automatique (machine learning), permettant aux machines d'analyser des données, d'adapter leur comportement et d'améliorer leurs performances sans intervention humaine explicite[38]. Cette figure explore les divers champs d'application de l'intelligence artificielle :

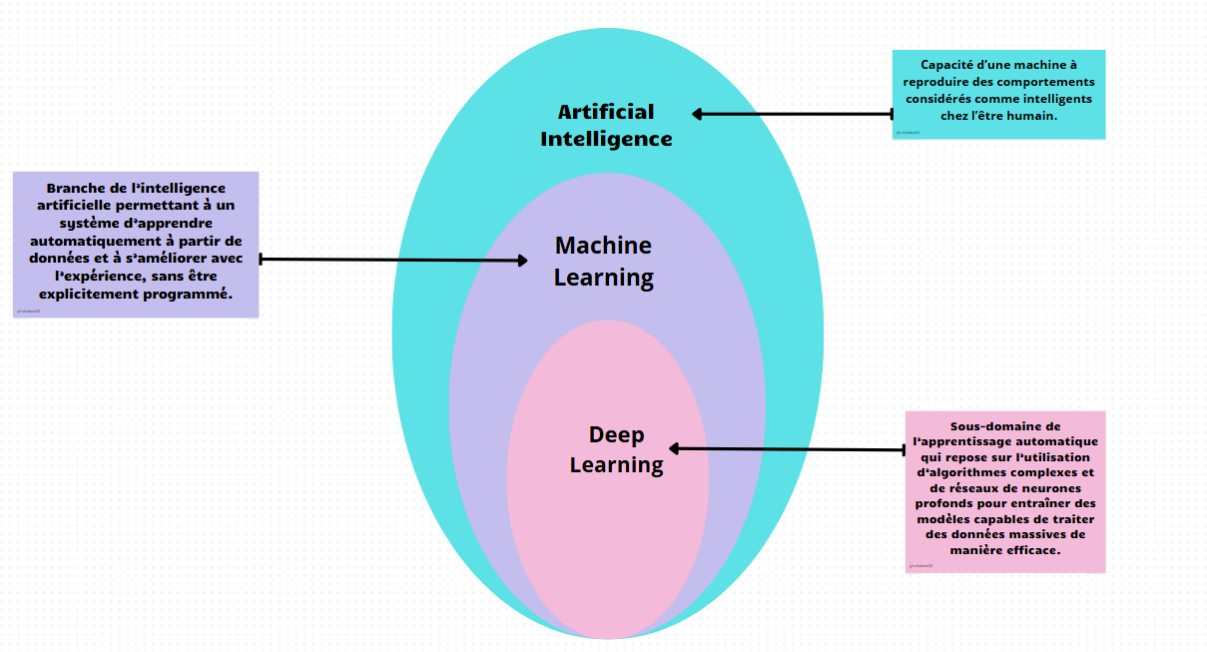


Figure I. 8: les domaine de intelligence artificielle

I.4.2. L'apprentissage automatique:

L'apprentissage automatique (machine learning) constitue un sous-domaine de l'informatique et représente l'une des branches fondamentales de l'intelligence artificielle. Il désigne un ensemble de techniques permettant aux programmes informatiques d'apprendre automatiquement à partir de données et de détecter des motifs ou des structures cachées sans avoir été explicitement programmés à cet effet . Les algorithmes d'apprentissage automatique sont conçus pour analyser des ensembles de données, en tirer des enseignements et ainsi formuler des prédictions ou prendre des décisions. Avant de pouvoir effectuer de telles prédictions, un algorithme doit passer par une phase d'apprentissage supervisé, au cours de laquelle il est exposé à un grand nombre d'exemples annotés (parfois plusieurs milliers), c'est-à-dire des données d'entrée accompagnées de la réponse attendue. Cette phase permet à l'algorithme de modéliser la relation entre les variables d'entrée et les sorties attendues. [27]

Une fois cette phase d'apprentissage complétée, le modèle est en mesure de généraliser ses connaissances à de nouvelles données non vues auparavant. Par exemple, dans le domaine médical, un algorithme de machine learning peut être entraîné à partir de séries temporelles représentant la fréquence cardiaque de patients. En apprenant la relation entre l'heure de la journée et la fréquence cardiaque, le système peut ensuite prédire la valeur normale attendue à

un instant donné. En comparant cette prédiction à la fréquence cardiaque réelle mesurée, il devient alors possible de détecter des anomalies physiologiques et ainsi contribuer à une surveillance proactive de l'état de santé du patient [27].

Les principaux types de systèmes d'apprentissage automatique sont synthétisés dans la figure suivante :

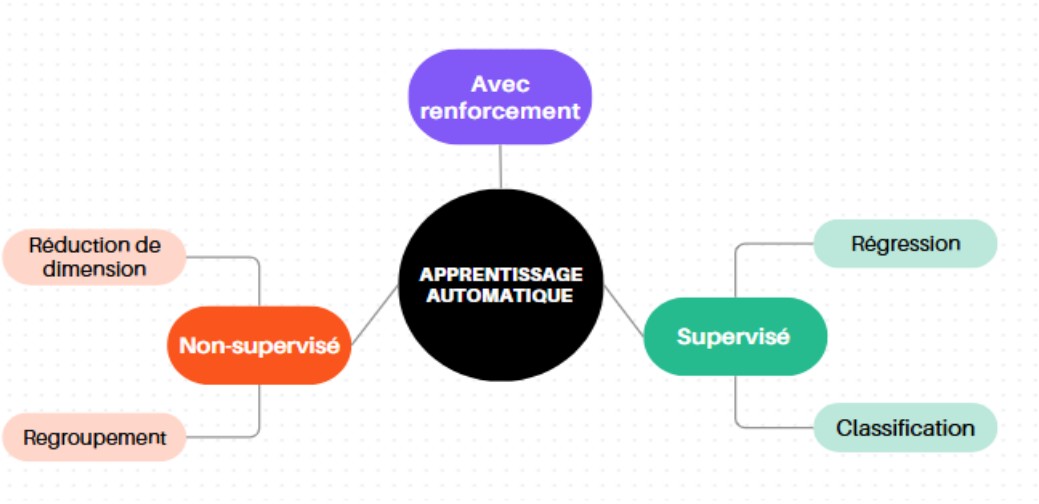


Figure I. 9: Types des systèmes de l'apprentissage automatique

I.4.2.1. Types des systèmes de l'apprentissage automatique

I.4.2.1.1. L'apprentissage supervisé :

L'apprentissage supervisé est une variété d'apprentissage machine qui consiste à former un modèle à partir de données étiquetées, ce qui signifie que chaque entrée est associée au bon résultat. Dans l'apprentissage supervisé, le modèle apprend en comparant ses prédictions avec les bonnes réponses données dans les exemples de formation. Au fil du temps, le modèle change et s'ajuste pour réduire ses erreurs et améliorer sa précision. Le but de l'apprentissage supervisé est que le modèle prédise aussi précisément que possible les nouvelles données non vues lorsqu'elles sont affichées. Par exemple, si un modèle est formé pour reconnaître des chiffres écrits à la main, son apprentissage pourrait lui permettre d'identifier correctement de nouveaux nombres manuscrits qu'il n'a jamais vus ou rencontrés auparavant.

L'apprentissage supervisé peut être utilisé de diverses façons, y compris la classification et la régression de l'apprentissage supervisé. Cela en fait une approche extrêmement importante dans le domaine de l'intelligence artificielle et du minage de données supervisé.[28]

Un concept important à comprendre dans l'apprentissage machine supervisé est d'apprendre une classe à partir d'exemples. Ceci est fait en fournissant les exemples modèles de la classe avec l'étiquette correcte. Ainsi, par exemple, si le modèle doit apprendre à classer les images de chats et de chiens, il apprendra en montrant des exemples étiquetés de chats et de chiens. Le modèle apprend à identifier les caractéristiques pertinentes de chaque classe et applique ensuite ces connaissances pour classer les nouvelles images d'entrée.[28]

I.4.2.1.2. L'apprentissage non supervisé :

L'apprentissage non supervisé est une branche de l'apprentissage automatique qui traite des données non étiquetées. Contrairement à l'apprentissage supervisé, où les données sont étiquetées avec une catégorie ou un résultat spécifique, les algorithmes d'apprentissage non supervisé ont pour tâche de trouver des modèles et des relations dans les données sans aucune connaissance préalable de la signification des données. Les algorithmes d'apprentissage automatique non supervisés trouvent des modèles et des données cachés sans aucune intervention humaine, c'est-à-dire que nous ne donnons pas de sortie à notre modèle. Le modèle de formation n'a que des valeurs de paramètres d'entrée et découvre les groupes ou les modèles par lui-même.[28]

I.4.2.1.3. L'apprentissage par renforcement:

L'apprentissage par renforcement (RL) est une branche de l'apprentissage automatique qui se concentre sur la façon dont les agents peuvent apprendre à prendre des décisions par essais et erreurs pour maximiser les récompenses cumulatives. RL permet aux machines d'apprendre en interagissant avec un environnement et en recevant des commentaires basés sur leurs actions. Cette rétroaction prend la forme de récompenses ou de pénalités.[28]

I.4.2.2. Les algorithme d'apprentissage automatique

Les algorithmes d'apprentissage automatique les plus couramment utilisés pour les systèmes de détection d'intrusion (IDS) incluent des approches supervisées et non supervisées telles que

la régression logistique, les machines à vecteurs de support (SVM), les k-plus proches voisins (KNN), les arbres de décision. Ces méthodes permettent d'identifier efficacement les activités anormales ou malveillantes à partir des données du réseau.

I.4.2.2.1. arbre de décision (Decision tree) :

L'arbre de décision est un algorithme d'apprentissage supervisé fondamental, couramment employé pour des tâches de classification et de régression. Il s'appuie sur une structure arborescente composée de nœuds, de branches et de feuilles. Chaque nœud interne représente un test portant sur une caractéristique spécifique des données, chaque branche correspond à une issue possible de ce test, et chaque feuille indique une prédiction, soit une classe dans le cas de la classification, soit une valeur dans le cadre de la régression. Le processus de construction de l'arbre consiste à diviser récursivement l'espace des données en sous-groupes homogènes à l'aide de règles de décision optimales, sélectionnées automatiquement en fonction de critères comme l'entropie ou l'indice de Gini. Une fois l'arbre construit, une phase d'élagage peut être appliquée afin de supprimer les branches peu informatives, réduisant ainsi le risque de surapprentissage. Grâce à sa lisibilité et à sa capacité à modéliser des relations complexes sans nécessiter de transformation préalable des données, l'arbre de décision constitue également la base de méthodes d'ensemble plus avancées, telles que la Forêt Aléatoire (Random Forest) et le gradient boosting (par exemple XGBoost[37].

I.4.2.2.2. K-plus proches voisins (k-Nearest Neighbors) :

L'algorithme des k plus proches voisins (k-Nearest Neighbors, ou KNN) est l'un des algorithmes d'apprentissage supervisé les plus simples et intuitifs. Il repose sur le principe de similarité entre les caractéristiques pour attribuer une classe à une observation donnée. Plus précisément, KNN prédit la classe d'un échantillon en identifiant les k instances les plus proches dans l'espace des caractéristiques, généralement à l'aide d'une mesure de distance telle que la distance euclidienne. L'étiquette de classe est ensuite déterminée à partir des classes majoritaires parmi ces k voisins. Le choix du paramètre k est crucial pour les performances du modèle : une valeur trop faible peut rendre le modèle sensible au bruit et conduire à un surapprentissage, tandis qu'une valeur trop élevée risque d'entraîner une perte de précision par généralisation excessive, voire une mauvaise classification. Le KNN est

apprécié pour sa simplicité, mais il peut être coûteux en calcul lors de la phase de prédiction, surtout sur de grands ensembles de données[37].

I.4.2.2.3. Le Support Vector Machine (SVM) :

Le Support Vector Machine (SVM) est un algorithme d'apprentissage supervisé qui repose sur le principe de la séparation optimale par un hyperplan à marge maximale dans un espace de caractéristiques à n dimensions. Il est utilisé pour résoudre à la fois des problèmes linéaires et non linéaires. Dans le cas des problèmes non linéaires, des fonctions noyau (kernel functions) sont employées afin de projeter les données d'un espace d'entrée de faible dimension vers un espace de caractéristiques de dimension plus élevée, où une séparation linéaire devient possible. L'algorithme cherche alors à identifier l'hyperplan optimal qui maximise la marge entre les classes, en s'appuyant uniquement sur un sous-ensemble d'échantillons appelés vecteurs de support. Dans le cadre des systèmes de détection d'intrusion réseau (NIDS), l'utilisation de SVM permet d'améliorer l'efficacité et la précision de la classification, en distinguant de manière fiable les activités normales des comportements malveillants[37].

.

I.4.3. L'apprentissage profond :

L'apprentissage profond, ou deep learning, est une branche avancée de l'apprentissage automatique qui repose sur des réseaux de neurones artificiels pour apprendre directement à partir des données brutes. Ces réseaux, inspirés du fonctionnement du cerveau humain, sont capables de réaliser diverses tâches complexes comme la classification, la reconnaissance d'images ou le traitement du langage naturel [29].

Le deep learning révolutionne la manière dont les machines comprennent, apprennent et interagissent avec des données complexes. En imitant les réseaux neuronaux du cerveau humain, il permet aux ordinateurs de découvrir de façon autonome des motifs cachés et de prendre des décisions éclairées à partir de grandes quantités de données non structurées [28]. Cette technique est aujourd'hui largement utilisée dans des domaines aussi variés que la biologie, la physique, l'astronomie ou l'économie, où elle facilite l'extraction d'informations à partir de données massives.

Ce tableau compare les approches d'apprentissage automatique et d'apprentissage profond selon plusieurs critères :

	Apprentissage automatique	Apprentissage profond
Volume de données	Fonctionne bien avec des ensembles de données de taille moyenne.	Nécessite de grandes quantités de données pour obtenir de bonnes performances.
Temps d'entraînement	Relativement rapide à entraîner.	.Temps d'entraînement long en raison de la complexité du modèle.
Ressources matérielles	Peut fonctionner avec des processeurs standards (CPU)	Requiert généralement des processeurs graphiques (GPU) pour l'entraînement
Traitement des données	Requiert souvent un prétraitement manuel des caractéristiques.	Effectue automatiquement l'extraction de caractéristiques à partir des données brutes.

Tableau I. 1: comparaison entre apprentissage automatique et profond

I.4.3.1. méthodes d'apprentissage profond

I.4.3.1.1. Réseaux de Neurones Récurrents (RNN) :

Les réseaux de neurones récurrents (RNN – Recurrent Neural Networks) constituent une catégorie particulière de réseaux de neurones artificiels conçue spécifiquement pour le traitement des données séquentielles, où l'ordre et le contexte des éléments ont une importance cruciale. Contrairement aux réseaux de neurones classiques (feedforward), qui traitent les données dans un flux unidirectionnel de l'entrée vers la sortie, les RNN intègrent des connexions récurrentes leur permettant de mémoriser et d'exploiter des informations issues des étapes précédentes de la séquence. L'unité fondamentale d'un RNN est la couche récurrente, composée de cellules récurrentes. Chaque cellule maintient un état interne, appelé état caché (hidden state), qui évolue à chaque pas temporel et est transmis à l'itération

suivante. Cet état agit comme une mémoire contextuelle, permettant au réseau de modéliser les dépendances temporelles et les motifs au fil de la séquence. À chaque étape temporelle, l'entrée courante est combinée à l'état caché précédent. Ce mélange est ensuite traité via une fonction d'activation, typiquement la tangente hyperbolique (tanh) ou la fonction ReLU (Rectified Linear Unit), afin de générer le nouvel état caché. Ainsi, l'état caché encode à la fois l'entrée actuelle et le contexte antérieur. Dans le cadre des systèmes de détection d'intrusion (IDS), les réseaux de neurones récurrents (RNN) sont généralement utilisés pour des tâches de classification supervisée ainsi que pour l'extraction automatique de caractéristiques pertinentes à partir des données séquentielles[39].

I.4.3.1.2. réseau de neurones convolutifs (CNN) :

Le terme réseau de neurones convolutifs (CNN) désigne une architecture de réseau neuronal reposant sur une opération mathématique appelée convolution. Contrairement aux réseaux neuronaux classiques qui utilisent des multiplications matricielles générales, les CNN intègrent au moins une couche où cette opération est remplacée par une convolution, permettant ainsi une meilleure exploitation des relations spatiales locales dans les données. Cette architecture est particulièrement efficace pour l'extraction automatique de caractéristiques pertinentes à partir de données localement connectées, notamment dans le traitement d'images. Les résultats issus des filtres convolutifs (ou noyaux) sont ensuite transmis à des fonctions d'activation non linéaires, telles que ReLU, qui introduisent de la non-linéarité dans l'espace des caractéristiques et permettent au modèle d'apprendre des représentations plus complexes et abstraites. Cette capacité à générer des motifs d'activation variés facilite l'identification de structures discriminantes dans les données[38].

La topologie d'un CNN se compose généralement de plusieurs étapes successives, incluant des couches convolutives, des fonctions d'activation non linéaires et des couches de sous-échantillonnage (ou pooling) destinées à réduire la dimensionnalité tout en conservant les informations essentielles. Ce schéma hiérarchique favorise une modélisation efficace et robuste des caractéristiques à différents niveaux d'abstraction[38].

Les réseaux de neurones convolutifs tirent leur efficacité de l'intégration de différents types de couches spécialisées, chacune jouant un rôle spécifique dans le traitement et

l'apprentissage des données. Ces couches incluent notamment :

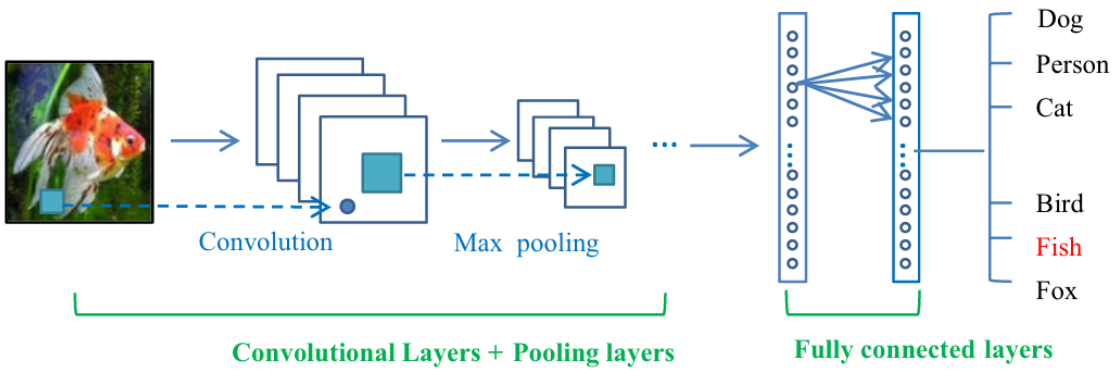


Figure I. 10: architecture générale de CNN

- **les couches de convolution (Convolutional Layers):**

Les couches de convolution constituent l'élément central de l'architecture des réseaux de neurones convolutifs (CNNs). Leur rôle principal est d'extraire automatiquement des caractéristiques discriminantes à partir des données d'entrée, notamment les images. Chaque couche convolutionnelle est composée d'un ensemble de filtres (ou noyaux) (kernels), représentés sous forme de matrices de poids, qui sont initialement attribués de manière aléatoire, puis ajustés tout au long du processus d'apprentissage afin de capter les motifs pertinents dans les données.

L'opération de convolution consiste à faire glisser chaque noyau sur l'ensemble de l'image d'entrée (ou du volume de caractéristiques issu d'une couche précédente), selon un pas défini, en calculant à chaque position le produit scalaire entre les éléments du noyau et ceux de la région locale de l'image. Le résultat de cette opération est une carte de caractéristiques, qui reflète la présence et la localisation d'un motif donné dans l'image. L'ajout éventuel d'un remplissage (padding) permet de préserver la taille spatiale de l'entrée ou de mieux capter les bords de l'image[40].

Chaque noyau agit donc comme un détecteur spécifique d'une caractéristique donnée (par exemple, une bordure verticale ou un motif texturé). Grâce à la connectivité locale, chaque neurone est uniquement connecté à une région restreinte du volume d'entrée, ce qui permet de

réduire considérablement le nombre de paramètres à apprendre et donc la complexité du modèle. Par ailleurs, la partageabilité des poids (weight sharing) signifie qu'un même filtre est appliqué à l'ensemble de l'entrée, ce qui contribue également à alléger le modèle et à accélérer l'apprentissage[40].

En sortie de chaque couche convolutionnelle, une fonction d'activation non linéaire (la fonction ReLU (Rectified Linear Unit)),

$$f(x) = \max(0.x)$$

renvoie la valeur d'entrée x lorsque x>0, et renvoie 0 lorsque $x \leq 0$, est appliquée afin d'introduire de la non-linéarité et de permettre au réseau de modéliser des relations complexes. Ce mécanisme favorise l'émergence de représentations hiérarchiques de plus en plus abstraites à mesure que l'on progresse dans les couches du réseau.

Ainsi, les couches de convolution permettent aux CNNs de construire efficacement des représentations compactes et robustes des données d'entrée, tout en limitant les besoins en mémoire et en temps de calcul grâce à une structure optimisée[40].

- **les couches de regroupement (Pooling Layers):**

La couche de pooling (ou sous-échantillonnage) a pour fonction principale de réduire la dimension spatiale des cartes de caractéristiques générées par les couches de convolution, tout en conservant les informations essentielles. Cette réduction permet de diminuer la complexité computationnelle du réseau et de limiter le risque de surapprentissage. Le pooling agit en sélectionnant des régions locales dans les cartes de caractéristiques, sur lesquelles une opération statistique est appliquée à l'aide d'un noyau (ou filtre) et d'un pas (stride) définis à l'avance[40].

Parmi les techniques de pooling les plus couramment utilisées, on retrouve le max pooling, qui retient la valeur maximale dans une région donnée, le min pooling, qui sélectionne la valeur minimale, et le average pooling, qui calcule la moyenne. Des variantes globales comme le global average pooling (GAP) ou le global max pooling peuvent également être employées pour réduire les cartes de caractéristiques à une dimension unique par canal.

Cependant, bien que cette opération permette une simplification efficace des représentations, elle présente certaines limites. En particulier, le pooling peut entraîner une perte

d'informations fines, notamment celles relatives à la localisation précise des caractéristiques dans l'image, ce qui peut affecter la performance globale du modèle CNN dans certaines tâches[40].

- **les couches entièrement connectées (Fully Connected Layers) :**

À la fin de l'architecture d'un réseau de neurones convolutif (CNN), on retrouve généralement une ou plusieurs couches entièrement connectées, dans lesquelles chaque neurone est connecté à l'ensemble des neurones de la couche précédente. Ces couches assurent la phase finale de classification, en exploitant les caractéristiques extraites par les couches de convolution (convolutional layers) et de sous-échantillonnage (pooling layers).

La dernière couche entièrement connectée applique généralement une fonction d'activation Softmax (Softmax activation function), qui permet d'attribuer une probabilité à chaque classe possible, facilitant ainsi la prise de décision du modèle dans les tâches de classification à plusieurs classes[40].

Dans certaines architectures CNN modernes, ces couches entièrement connectées peuvent être remplacées par des couches de mise en commun globales (Global Pooling Layers), telles que la mise en commun moyenne globale (Global Average Pooling). Cette alternative permet de réduire significativement le nombre de paramètres, limitant ainsi le risque de surapprentissage (overfitting) tout en conservant de bonnes performances prédictives[40].

I.5. Conclusion :

Ce premier chapitre a permis de poser les fondements théoriques nécessaires à la compréhension de ce travail. Nous avons tout d'abord introduit les notions clés de l'intelligence artificielle (IA), de l'apprentissage automatique (Machine Learning) et de l'apprentissage profond (Deep Learning), en mettant en évidence leur rôle croissant dans la cybersécurité. Une attention particulière a été accordée aux réseaux neuronaux convolutifs (CNN), qui se sont imposés comme une architecture performante pour l'analyse de données complexes et la détection d'anomalies.

Chapitre II

État de art

Chapitre II : État de art

II.1. Introduction

La complexité croissante des cybermenaces dans les environnements IoT a conduit les chercheurs à développer des solutions de détection d'intrusion (IDS) de plus en plus intelligentes et autonomes. Parmi celles-ci, les réseaux de neurones convolutifs (CNN) ont émergé comme une approche efficace, capable d'extraire automatiquement des caractéristiques discriminantes à partir du trafic réseau, sans nécessiter d'ingénierie manuelle poussée.

Ce chapitre présente une synthèse des travaux récents ayant appliqué des architectures CNN à la détection d'intrusions dans des contextes IoT. L'objectif est de situer notre propre étude dans l'état de l'art actuel, en mettant en évidence les choix d'architecture, les jeux de données utilisés et les résultats obtenus dans la littérature. En outre, cette revue prépare le terrain pour une évolution future de notre approche vers l'apprentissage fédéré (Federated Learning), afin de répondre aux défis liés à la confidentialité, la scalabilité et la décentralisation dans les réseaux IoT réels.

II.2. Travaux de recherche récents sur la détection d'intrusions :

Ces dernières années, la détection d'intrusions a connu un fort essor, notamment avec l'émergence des technologies IoT et la sophistication croissante des cyberattaques. De nombreux chercheurs ont proposé des systèmes IDS intelligents, exploitant les avancées de l'apprentissage automatique et profond pour analyser le trafic réseau en temps réel et identifier les comportements anormaux. Cette section présente une revue structurée des approches les plus récentes, en mettant l'accent sur les modèles à base de réseaux de neurones convolutifs (CNN), leur intégration avec l'apprentissage fédéré, ainsi que les alternatives non basées sur les CNN.

II.2.1. Systèmes de détection d'intrusion basés sur les réseaux de neurones convolutifs (CNN) et approches hybrides :

les réseaux de neurones convolutifs (CNN) ont démontré une efficacité remarquable dans le domaine de la détection d'intrusions, notamment grâce à leur capacité à extraire

automatiquement des caractéristiques discriminantes à partir de données réseau. Utilisés seuls ou en combinaison avec d'autres architectures telles que les réseaux récurrents (LSTM, GRU), les CNN permettent de concevoir des systèmes IDS robustes, capables de détecter des menaces complexes dans des environnements IoT/IIoT. Cette section examine les travaux récents mettant en œuvre des architectures CNN pures ou hybrides, en soulignant leurs performances, leurs avantages et leurs limites. Parmi ces travaux, on peut citer :

Okey et al [43] ont proposé un système de détection d'intrusion basé sur l'apprentissage par transfert et l'architecture des réseaux de neurones convolutifs (CNN). Le système utilise cinq modèles CNN pré-entraînés, à savoir VGG16, VGG19, Inception, MobileNet et EfficientNet. L'entraînement est effectué sur deux jeux de données : CIC-IDS2017 et CSE-CICIDS2018. Avant l'apprentissage, les auteurs appliquent un prétraitement incluant le traitement du déséquilibre, la réduction de dimensionnalité et la transformation des vecteurs de caractéristiques en images à l'aide du Quantile Transformer. Trois modèles (InceptionV3, MobileNetV3Small et EfficientNetV2B0) sont sélectionnés pour former un modèle d'ensemble appelé ELETL-IDS, en utilisant une méthode d'agrégation par moyenne. Le système proposé atteint une précision, un rappel, une exactitude et un F-score de 100 %. Le coefficient de corrélation de Matthews (MCC) est utilisé pour valider les résultats, avec une valeur AUC-ROC atteignant 0,9996. Le système est jugé léger, efficace et adapté aux environnements IoT dans le cloud.

Arsalan et al [41] ont développé un algorithme de réseau de neurones convolutif unidimensionnel (1DCNN) peu coûteux en ressources pour la classification des cyberattaques. Le modèle proposé permet de classer neuf types d'attaques avec une exactitude de 99,90 %. Plusieurs autres métriques de performance ont été évaluées afin de valider l'efficacité du système. Une comparaison a également été effectuée avec des approches récentes de l'état de l'art. Les résultats obtenus montrent que la méthode proposée peut apporter une contribution significative à la mise en place de systèmes de détection d'intrusion sécurisés dans le contexte de l'IIoT.

Kilichev et al [45] ont proposé un cadre basé sur l'apprentissage profond pour la détection et la classification automatique des cyberattaques. Ce cadre intègre plusieurs optimisations, telles que la réduction de dimensionnalité, l'ajustement des hyperparamètres et l'ingénierie des caractéristiques. Le système repose sur une version améliorée du réseau de neurones convolutif (CNN), appelée IIDNet (Intelligent Intrusion Detection Network), conçue pour améliorer l'efficacité de détection. Une optimisation des couches au niveau architectural est appliquée à l'aide de l'algorithme LBIID (Learning-Based Intelligent Intrusion Detection), afin d'augmenter les performances de détection du modèle. Les expériences menées sur le jeu de données de référence UNSW-NB15 ont montré que le modèle IIDNet atteint une précision de 95,47 %, tout en réduisant significativement le temps d'entraînement et en assurant une bonne scalabilité. Le modèle surpasse ainsi plusieurs systèmes de détection d'intrusion existants.

Deshmukh et al [46] introduit un cadre de détection intelligent basé sur un CNN amélioré appelé IIDNet. Le modèle intègre diverses optimisations, notamment la réduction de la dimension, l'ingénierie des caractéristiques et l'ajustement des hyperparamètres. En s'appuyant sur l'algorithme LBIID et le jeu de données UNSW-NB15, le système atteint une précision de 95,47 %. Il offre également un temps d'entraînement réduit et une bonne évolutivité, surpassant plusieurs modèles de détection classiques.

Sinha et al [43] ont proposé un cadre sécurisé avancé basé sur une architecture hybride LSTM-CNN afin d'optimiser la détection d'intrusion en temps réel dans les environnements IoT. Le modèle intègre des couches LSTM pour capturer les dépendances temporelles, ainsi que des couches CNN pour extraire les caractéristiques spatiales. L'ensemble permet d'identifier efficacement les menaces. Le jeu de données utilisé, BoT-IoT, comprend plusieurs types d'attaques, notamment DDoS, botnet, reconnaissance et exfiltration de données.
Les résultats obtenus montrent que le modèle atteint une exactitude de 99,87 %, une précision de 99,89 % et un rappel de 99,85 %, avec un faible taux de faux positifs de 0,13 %. Le modèle proposé surpasse les performances de plusieurs architectures classiques telles que CNN, RNN, LSTM standard, BiLSTM et GRU. En présence d'attaques adversariales, il maintient

une exactitude de 90,2 %, démontrant ainsi sa robustesse.

Une analyse d'importance des caractéristiques à l'aide de SHAP a révélé que la taille des paquets, la durée des connexions et le type de protocole sont des indicateurs clés pour la détection des menaces. Ces résultats suggèrent que le modèle hybride LSTM-CNN est prometteur pour renforcer la sécurité des dispositifs IoT tout en minimisant les taux d'alarme erronée.

Danish Javeed et al [22],ont introduit un modèle d'apprentissage fédéré horizontal intégrant les réseaux de neurones convolutionnels (CNN) et les réseaux de neurones à mémoire long terme bidirectionnelle (BiLSTM) pour la détection efficace des intrusions. L'objectif étant de surmonter les limites des techniques existantes pour améliorer l'efficacité de la détection des intrusions dans le cadre de l'apprentissage fédéré à destination de l'IoT. Pour ce faire, le CNN réalise l'extraction des caractéristiques spatiales en déterminant des motifs locaux pouvant caractériser une intrusion, pendant que le BiLSTM capture les dépendances temporelles tout en apprenant les motifs temporels sous forme séquentielle dans les données. Le système de détection d'intrusion proposé fonctionne sur le modèle de confiance zéro, conservant les données sur les dispositifs locaux et n'échangeant avec le serveur centralisé de l'apprentissage fédéré que les poids obtenus. Le serveur FL procède alors à l'agrégation des mises à jour pour optimiser la qualité du modèle d'apprentissages global. Les résultats expérimentaux réalisés sur les jeux de données CICIDS2017 et Edge-IIoTset montrent la pertinence de l'approche par rapport à des systèmes d'IDS s'appuyant sur des approches d'apprentissage profond centralisées et fédérées.

Shuroog Alsaleh et al [26] proposent un modèle semi-décentralisé basé sur l'apprentissage fédéré (FL) pour un système de détection d'intrusion léger, adapté aux capacités des dispositifs IoT. Le modèle proposé repose sur le clustering des dispositifs IoT (les clients FL) et l'attribution d'un chef de cluster pour chaque groupe, qui agit au nom des clients FL. Ainsi, le nombre de dispositifs IoT communiquant avec le serveur est réduit, ce qui permet de diminuer la surcharge de communication. De plus, le clustering améliore le processus d'agrégation, chaque cluster envoyant les poids moyens du modèle au serveur pour agrégation

lors d'un seul tour d'apprentissage fédéré. L'attaque par déni de service distribué (DDoS) est la principale préoccupation de leur modèle IDS, car elle peut facilement survenir dans les dispositifs IoT aux capacités limitées. Le modèle proposé est configuré avec trois techniques d'apprentissage profond : LSTM, BiLSTM et WGAN, utilisant le jeu de données CICIoT2023. Les résultats expérimentaux montrent que le BiLSTM offre de meilleures performances et est adapté aux dispositifs IoT à ressources limitées, en raison de la taille réduite du modèle. Nous testons le modèle pré-entraîné semi-décentralisé basé sur l'apprentissage fédéré sur trois jeux de données : BoT-IoT, WUSTL-IIoT-2021 et Edge-IIoTset et les résultats montrent que notre modèle obtient les meilleures performances dans la plupart des classes, en particulier pour les attaques DDoS.

Heng et al [44] ont présenté approche hybride CNN-LSTM pour la cybersécurité dans les environnements industriels intelligents. Le modèle est conçu pour capturer à la fois les motifs spatiaux et les dépendances temporelles afin de détecter des anomalies réseau en temps réel. En utilisant un jeu de données IIoT à grande échelle, le modèle atteint une précision binaire de 71 % et un rappel de 99 % pour le trafic normal. Bien qu'il excelle dans la détection des attaques à grand volume comme les DDoS, sa performance sur les attaques rares reste limitée, signalant la nécessité d'améliorations futures.

II.2.2. Systèmes de détection d'intrusion basés sur les CNN intégrés à l'apprentissage fédéré (Federated Learning FL) :

L'apprentissage fédéré (Federated Learning - FL) représente une avancée majeure pour la mise en œuvre d'IDS dans des environnements sensibles à la confidentialité, tels que l'IoT. Il permet d'entraîner des modèles localement sur les dispositifs en périphérie sans centraliser les données. Lorsqu'il est combiné aux réseaux de neurones convolutifs (CNN), le FL offre un compromis intéressant entre précision, efficacité et respect de la vie privée. Cette section présente les principales contributions associant CNN et FL pour la détection d'intrusions, en mettant en évidence les bénéfices liés à la distribution des modèles et les défis liés aux données hétérogènes, aux attaques adversariales ou à la latence. Pour illustrer cette tendance, plusieurs publications récentes peuvent être mentionnées :

Meryem Janati Idrissi et al [19], ont proposé Fed-ANIDS, un système de détection d'intrusions (NIDS) mis en œuvre avec l'AD (apprentissage par détection d'anomalies) et le FL (apprentissage fédéré) pour répondre aux enjeux de confidentialité liés aux modèles centralisés. Le processus de détection d'intrusion repose sur le calcul d'un score d'intrusion par approche par erreur de reconstruction pour le trafic normal à l'aide de plusieurs modèles AD, autoencodeurs simples, autoencodeurs variationnels et autoencodeurs adversariaux. Les auteurs évaluent Fed-ANIDS à l'aide de divers paramètres et jeux de données connues comme USTC-TFC2016, CIC-IDS2017, CSE-CIC-IDS2018. Ils montrent que la méthode proposée fournit de bonnes performances selon plusieurs critères, tout en garantissant la confidentialité des données mises en commun par les clients distribués. Leurs résultats font apparaître que les modèles autoencodeurs l'emportent sur leurs modèles GAN (réseaux antagonistes génératifs) concurrents et offrent une bonne précision de détection et faibles fausses alertes. En outre, la prise en compte du cadre FL (FedProx), qui est une généralisation et une reparamétrisation de la méthode FL standard (FedAvg), contribue à de bonnes performances. Le code est en accès libre à Github.

Thein et al [21] ont développé une méthode de détection d'intrusion (IDS) basée sur l'apprentissage fédéré (FL) pour répondre aux défis d'hétérogénéité des données et d'attaques par empoisonnement dans des systèmes IoT, où des clients malveillants peuvent manipuler les données locales ou les modèles à l'échelle globale. Afin de résoudre ces problèmes, un modèle d'apprentissage fédéré personnalisé a été développé, intégrant une recherche aléatoire pour l'optimisation des hyperparamètres. Les données des clients sont protégées grâce à une agrégation sécurisée des modèles locaux. Il est démontré que la méthode pFL-IDS permet de détecter efficacement les attaques par empoisonnement tout en préservant la performance du modèle global par rapport aux méthodes traditionnelles. Le mécanisme de détection repose sur l'analyse de la similarité cosinus entre les modèles locaux et un centre modélisant les modèles non empoisonnés, ce qui permet d'identifier les clients malveillants et de limiter leur impact sur le modèle global. Les résultats montrent que pFL-IDS offre des performances robustes dans un scénario d'attaques par empoisonnement, sans compromettre les taux de détection.

Othmane Friha et al [25] ont développé un système de détection d'intrusion basé sur l'apprentissage fédéré, nommé FELIDS, destiné à sécuriser les infrastructures IoT agricoles. Le système FELIDS protège la confidentialité des données grâce à l'apprentissage local, où les dispositifs bénéficient des connaissances de leurs pairs en partageant uniquement les mises à jour de leurs modèles avec un serveur d'agrégation qui produit un modèle de détection amélioré. Afin de prévenir les attaques sur les IoT agricoles, le système FELIDS utilise trois classificateurs d'apprentissage profond : les réseaux de neurones profonds, les réseaux de neurones convolutionnels et les réseaux de neurones récurrents. Les auteurs étudient les performances du système IDS proposé sur trois sources différentes : CSE-CIC IDS2018, MQTTset et InSDN. Les résultats montrent que le système FELIDS surpasse les versions classiques/centralisées de l'apprentissage machine (non-apprentissage fédéré) en protégeant la confidentialité des données des dispositifs IoT et en obtenant la meilleure précision dans la détection des attaques.

Robert Akinie et al [16] ont proposé un cadre hybride serveur-périphérie s'appuyant sur l'apprentissage fédéré (FL) permettant de décharger le pré-entraînement vers un serveur central tout en permettant un réglage léger au niveau des dispositifs périphériques. Au final, cette approche permet de réduire l'utilisation de la mémoire jusqu'à 42%, de diminuer les temps d'entraînement jusqu'à 75%, et d'atteindre une précision compétitive en détection d'intrusion (IDS) allant jusqu'à 99,2%. L'analyse de la scalabilité montre également une dégradation des performances sous-jacentes minimale en fonction du nombre de clients, suggérant la faisabilité du cadre proposé pour les réseaux CAV et autres applications IoT.

II.2.3. Approches de détection d'intrusion ne reposant pas sur les réseaux de neurones convolutifs (non-CNN) :

En parallèle des modèles CNN, d'autres approches d'apprentissage automatique et profond sont utilisées pour la détection d'intrusions. Ces méthodes incluent notamment les autoencodeurs (AE, VAE, AAE), les réseaux de neurones profonds (DNN), les machines à vecteurs de support (SVM), les forêts aléatoires (Random Forest), ainsi que les techniques de

clustering comme K-Means++. Certaines de ces approches sont particulièrement adaptées aux scénarios non supervisés ou à forte contrainte de ressources. Cette section explore ces solutions non-CNN, en analysant leurs résultats, leur applicabilité aux données IoT, ainsi que leurs limites face aux menaces émergentes. Ces approches sont mises en œuvre dans plusieurs travaux, notamment :

M. A. Ferrag et al[3], proposent une analyse prospective des attaques de type "poisoning" dans un contexte d'apprentissage fédéré en périphérie (federated edge learning), destiné aux environnements IoT renforcés par des jumeaux numériques et les réseaux 6G. Leur travail s'intéresse plus particulièrement à l'impact potentiel des acteurs malveillants sur le processus d'apprentissage et le développement de modèles fédérés dans ces environnements innovants.Les auteurs démontrent que des attaques de type "data poisoning" peuvent être menées avec succès aussi bien dans des configurations centralisées que dans des cadres fédérés, compromettant sérieusement la précision des modèles entraînés. Afin d'évaluer l'ampleur de cette menace, des expériences ont été menées sur un nouveau jeu de données de cybersécurité dédié aux applications IoT, en testant trois réseaux de neurones profonds (DNN), sous des distributions de données IID (indépendantes et identiquement distribuées) et non-IID (non-indépendantes et non identiquement distribuées).Les résultats révèlent une chute significative de la précision des modèles à la suite de ces attaques. Pour les données IID, la précision diminue de 94,93 % à 85,98 %, tandis que pour les données non-IID, elle chute de 94,18 % à seulement 30,04 %, mettant en évidence la vulnérabilité accrue des systèmes fédérés dans des environnements distribués.

Olanrewaju-George et al [14]. testent des modèles profonds en apprentissage supervisé ou non, via apprentissage fédéré (FL), pour concevoir un système de détection d'intrusions (IDS) pour dispositifs IoT. Les performances des modèles formés avec FL sont comparées à celles des modèles formés sans FL, avec le jeu de données N-BaIoT composé de 9 dispositifs IoT. Pour optimiser au mieux la performance des modèles profonds, une optimisation de leurs hyperparamètres est réalisée par recherche aléatoire des valeurs. Plusieurs métriques de performances sont utilisées pour juger de la qualité des prédictions. Les résultats montrent que

le modèle AutoEncoder (AE) non supervisé entraîné via FL est le modèle qui présente les meilleures performances globales d'après tous les critères à l'issue de l'évaluation des modèles FL et non FL sur les 9 dispositifs IoT.

Ozlem Cevizet al. [23] ont examiné comment l'apprentissage fédéré, associé à l'apprentissage avec peu d'exemples (Few-shot Learning – FSL), peut considérablement diminuer la quantité de données requises pour détecter les intrusions dans les réseaux aériens ad hoc (FANETs). Les chercheurs ont mis au point une méthode qu'ils ont appelée FSFL-IDS (Few-shot Federated Learning-based IDS), qui allie les concepts de l'apprentissage fédéré (FL) et de l'apprentissage avec peu d'exemples pour surmonter les défis spécifiques à la détection d'intrusion dans ces environnements. Cela inclut des préoccupations telles que la confidentialité, la consommation d'énergie, les coûts de communication et la perte de paquets. Leur approche permet non seulement de réduire le temps nécessaire à l'entraînement des modèles, tant locaux que globaux, mais aussi de diminuer la taille des échantillons requis. Cela se traduit par une charge de calcul allégée, moins de communications et une meilleure autonomie énergétique. De plus, grâce à l'intégration de FSL, qui nécessite moins de données pour l'apprentissage, le système de détection proposé est plus résistant face aux connexions de communication instables, qui sont typiques des FANETs.

Maxime Gourceyraud et al [24], ont proposé une architecture IDS s'appuyant sur l'apprentissage non supervisé pour réduire l'intrusion de l'étiquetage des données. Le système a déployé un cadre d'apprentissage fédéré pour assurer l'apprentissage collaboratif entre les appareils. Pour aller au-delà des modèles existants en matière de confidentialité fédérée dans le clustering, ils ont développé une technique d'initialisation K-Means++ fédérée. En changeant du one-serveur modèle à un paradigme fédéré, les auteurs ont expliqué que cela ne perturbe pas le type de performance du système initial, ce qui garantit une performance adéquate du système avec suffisamment de sécurité de la confidentialité.

II.3. Analyse critique des travaux existants :

Les études récentes montrent que les modèles d'apprentissage profond, en particulier les réseaux de neurones convolutifs (CNN), permettent d'améliorer significativement la détection d'intrusions dans les environnements IoT. Toutefois, la majorité de ces travaux reposent sur une architecture centralisée, ce qui soulève des préoccupations liées à la confidentialité des données, à la scalabilité du système et à la latence. De plus, certains modèles proposés présentent une complexité architecturale élevée ou nécessitent des ressources computationnelles importantes, ce qui les rend peu adaptés aux dispositifs en périphérie (edge devices).

Par ailleurs, l'apprentissage fédéré émerge comme une solution prometteuse pour répondre à ces contraintes, en permettant l'entraînement collaboratif des modèles tout en préservant la confidentialité. Néanmoins, cette approche reste encore en développement et confronte plusieurs défis, notamment la stabilité de la convergence, l'hétérogénéité des nœuds et la difficulté d'intégration avec des modèles simples. Dans ce contexte, notre projet adopte une approche centralisée, en s'appuyant sur un CNN simple et efficace, tout en envisageant, à terme, une extension vers une architecture fédérée qui combine performance, légèreté et respect de la vie privée.

Le tableau suivant récapitule les principales études existantes liées à la détection d'intrusion dans les environnements IoT :

Réf	Année	Modèle de Menace	Solution d'Atténuation	Mode d'Apprentissage	Modèle ML	Jeux de Données	Avantages(+)	Problèmes Ouverts(-)
[43]	2023	DDoS, DoS, Infiltration, Bot, Man-in-the-middle	Transfer Learning avec CNN optimisé et ensemble learning (ELETL-IDS)	Apprentissage supervisé avec transfert de connaissance	CNN (VGG, Inception, EfficientNet)	CIC-IDS2017, CSE-CICIDS2018	+Précision jusqu'à 100%, robuste, léger, conversion en images + optimisation hyperparamètres	-Conversion tabulaire-image coûteuse. -sur-apprentissage possible avec ensembles profonds
[3]	2023	Attaques par Empoisonnement (Label flipping, Backdoor, DDoS), surtout en Non-IID	Étude anticipative : analyse des stratégies, modèles d'attaques et sélection de clients malveillants	Apprentissage Fédéré en périphérie (Federated Edge Learning) et Centralisé	CNN, RNN, DNN	Edge-IIoTset	+Évaluation complète de plusieurs scénarios d'attaques. + Analyse réaliste dans des environnements 6G/IoT. +Utilisation de plusieurs modèles DL et distribution des données	-Manque de mécanismes de défense intégrés dans l'étude. -Performance critique avec données Non-IID
[19]	2023	Intrusions réseau (DoS, DDoS, Injection SQL, reconnaissance, etc.)	Fed-ANIDS: Détection d'anomalies basée sur l'apprentissage fédéré (FL)	Apprentissage fédéré (FedProx, FedAvg)	Autoencodeurs (AE, VAE, AAE)	USTC-TFC2016, CIC-IDS2017, CSE-CIC-IDS2018	Haute précision, préservation de la confidentialité des données	Hétérogénéité des données, latence et surcharge de communication
[45]	2024	Cybermenaces dans les EVCS IoT	Architecture hybride CNN-LSTM-GRU	Apprentissage supervisé	CNN, LSTM, GRU (ensemble)	Edge-IIoTset	+Précision exceptionnelle (100% binaire, 97.44% multiclasses). +modèle robuste et adaptable	-Besoin d'extension vers blockchain. -systèmes prédictifs, et intégration smart grid

[46]	2024	Attaques multiples	Modèle IIDNet (CNN amélioré) avec tuning de paramètres	Apprentissage supervisé	CNN optimisé (IIDNet)	UNSW-NB15	+Haute précision (95.47%). +faible temps d'entraînement. +bonne scalabilité	-Pas d'application à un contexte industriel spécifique. -manque de diversité des attaques
.[4]	2024	Attaques Zero-day , données Non-IID , attaques adversariales (e.g., FGSM, BIM, DeepFool)	Système FedGenID avec 3 modèles : cGAN génératif, Critique local, Classificateur global	Apprentissage Fédéré (Federated Learning)	Conditional GAN (cGAN), CNN	Edge-IIoTset (2022)	- Détection améliorée des attaques (jusqu'à +10%). - Résilience aux attaques adversariales Préservation de la confidentialité. - Gestion des données déséquilibrées et distribuées	– Coût computationnel élevé – Instabilité potentielle des GANs – Validation complexe des données synthétiques – Partage de modèle pouvant présenter un risque
[5]	2024	Vulnérabilités de l'apprentissage en périphérie (Edge Learning) dans l'IoT 6G (8 attaques)	Revue complète des attaques (backdoor, adversarial, Sybil, etc.) et des solutions de défense (DP, HE, Blockchain, etc.)	Centralisé / fédéré / distribué	CNN, GAN, RNN, SVM…	Plusieurs datasets :NSL-KDD (une version améliorée du KDD Cup 99), UNSW-NB15, CICIDS2017, TON_IoT, Bot-IoT, IoTID20, Edge-IIoTset.	+ Large couverture des vulnérabilités + Classification détaillée des attaques et défenses + Propositions de futures solutions intégrant sécurité, confidentialité et efficacité	- Pas d'évaluation expérimentale directe - Besoin d'études concrètes pour valider les mécanismes dans des réseaux 6G réels

[21]	2024	Attaques de type "Poisoning" (label flipping, model poisoning)	Système IDS basé sur l'apprentissage fédéré personnalisé (pFL-IDS) avec détection des clients malveillants	Apprentissage fédéré personnalisé	1D-CNN (réseaux neuronaux convolutifs unidimensionnels(	N-BaIoT	+Haute précision , +réduction des effets des clients malveillants, + adaptation aux données hétérogènes	-Difficulté à gérer la non-homogénéité des données , - nécessité d'améliorer les techniques de détection des attaques
[22]	2024	Attaques sur IoT (DoS, DDoS, MITM, Injection SQL, etc.)	Système IDS basé sur FL avec modèle zéro confiance	Apprentissage fédéré horizontal	CNN-BiLSTM	CICIDS2017, Edge-IIoTset	+Haute précision, + préservation de la confidentialité, + robustesse contre les menaces évolutives	-Latence de communication, - surcharge réseau, - gestion des données non homogènes
[41]	2024	Cyber-attaques dans l'IIoT	1D CNN pour classification	Apprentissage profond (DL) Apprentissage supervisé	CNN	Edge-IIoTset	+Très haute précision (99,90 %). +faible coût computationnel, entraînement rapide (3 epochs). +9 attaques classifiées	-Déséquilibre de classes. -couverture limitée des attaques (exclut port scan, SQLi…). -nécessité d'extension et de généralisation

[42]	2025	DDoS, Botnet, Reconnaissance, Data Exfiltration, Adversarial Attacks	Architecture hybride LSTM-CNN	Apprentissage supervisé	LSTM-CNN	BoT-IoT	+Haute précision (99.87%). +robuste aux attaques adversariales. +faible taux de faux positifs (0.13%). + SHAP pour l'interprétabilité	-Complexité computationnelle. -modèle encore sensible à certains types d'attaques évolutives
[44]	2025	DoS, SQL injection, Ransomware, autres attaques IIoT	Modèle CNN-LSTM pour IIoT	Apprentissage supervisé	CNN-LSTM	Edge-IIoTset	+Bonne performance sur DDoS et classes majoritaires. + réduction du taux de faux négatifs +recall de 99%	-Difficultés pour attaques rares. -performance faible en multiclasses (71%). -besoin d'optimisation future
[14]	2025	Cyberattaques sur dispositifs IoT	Apprentissage Fédéré avec AutoEncoder (non supervisé) pour la détection DNN (supervisé) pour la classification	Apprentissage profond Supervisé et non supervisé (AE + DNN) avec et sans FL	AE, DNN	N-BaIoT (9 dispositifs IoT)	+ Préservation de la confidentialité des données. + Bonne performance globale (surtout avec FL). + AE en FL donne les meilleurs résultats FPR.	- Coût de calcul élevé pour FL. - Défis liés aux données non-IID. - Vulnérabilité potentielle aux attaques d'empoisonnement.

[16]	2025	Cyberattaques sur les véhicules autonomes (DoS, DDoS, Injection SQL, etc.)	Détection d'intrusions basée sur l'apprentissage fédéré)FL(	Apprentissage fédéré hybride (server-edge)	CNN + MLP (Fine-Tuning)	NSL-KDD	Haute précision 99.2%() réduction de la consommation mémoire ,(%42) réduction du temps d'entraînement (%75)	Contraintes de ressources , latence et surcharge de communication , hétérogénéité des réseaux
[23]	2025	Attaques de routage sur les FANETs (sinkhole, blackhole, flooding)	Détection d'intrusions basée sur l'apprentissage fédéré et few-shot learning (FSFL-IDS)	Apprentissage fédéré avec few-shot learning	Deep Neural Networks (DNN) et Convolutional Neural Networks (CNN)	FANET dataset	+Réduction des coûts de communication et de calcul , +préservation de la confidentialité , + meilleure adaptation aux réseaux dynamiques	-Hétérogénéité des données , -latence et surcharge de communication, - besoin d'amélioration de la détection des attaques complexes
[24]	2025	Attaques réseau (DoS, DDoS, attaques modernes)	Système IDS basé sur apprentissage fédéré avec K-Means++ fédéré	Apprentissage non supervisé et fédéré	K-Means++ amélioré	UNSW-NB15, CIC-IDS2017	+Réduction des besoins en étiquetage, + préservation de la confidentialité , +performances comparables aux modèles centralisés	-Sous-optimalité dans l'agrégation , -besoin de clusters supplémentaires pour des performances équivalentes

[25]	2025	Attaques réseau (DoS, DDoS, Injection SQL, etc.)	FELIDS: Système IDS basé sur apprentissage fédéré avec trois classificateurs DL (DNN, CNN, RNN)	Apprentissage fédéré	DNN, CNN, RNN	CSE-CIC-IDS2018, MQTTset, InSDN	+Haute précision , +préservation de la confidentialité , +meilleure performance par rapport aux modèles centralisés	+Hétérogénéité des données , +latence et surcharge de communication , +besoin d'amélioration pour les attaques complexes
[26]	2025	Attaques DDoS principalement dans les réseaux IoT	Modèle FL semi-décentralisé avec clustering des clients	Apprentissage fédéré semi-décentralisé	BiLSTM, LSTM, WGAN	CICIoT2023, BoT-IoT, WUSTL-IIoT-2021, Edge-IIoTset	+Haute précision , +réduction de la surcharge de communication, +gestion de l'hétérogénéité des données	-Complexité des modèles, -besoin d'amélioration pour les attaques complexes, -surcharge de communication

Tableau II. 1 : Travaux antérieurs connexes pour la détection d'intrusion

II.4. Conclusion :

Dans ce chapitre, nous avons présenté une revue structurée des travaux récents portant sur la détection d'intrusion dans les environnements IoT à l'aide de réseaux de neurones convolutifs (CNN). Ces approches ont démontré une forte capacité à détecter les attaques réseau complexes grâce à l'extraction automatique de caractéristiques discriminantes, tout en surpassant généralement les méthodes .d'apprentissage automatique classiques

Nous avons également mis en évidence les limites des modèles centralisés, notamment en termes de confidentialité des données, de scalabilité et de diversité des sources de données dans les systèmes IoT. Ces constats ont conduit à l'émergence de l'apprentissage fédéré (Federated Learning), comme solution prometteuse pour entraîner des modèles collaboratifs sans partage direct des

données. Plusieurs études récentes intègrent les CNN ou d'autres architectures profondes dans des cadres fédérés, démontrant leur efficacité même en présence de .données hétérogènes ou de menaces adversariales

Cette synthèse des travaux existants prépare le terrain pour notre propre contribution, qui repose d'abord sur un modèle CNN centralisé performant, avant d'envisager son extension vers un cadre fédéré dans lesétudes prochaines.

Chapitre III

Conception et Évaluation des Performances

Chapitre III : Conception et Évaluation des Performances

III.1. Introduction

Face à la croissance exponentielle des objets connectés (IoT) et de leurs équivalents industriels (IIoT), les réseaux deviennent de plus en plus exposés à des attaques sophistiquées et ciblées[1]. Cette évolution rapide soulève de nouveaux défis en matière de cybersécurité, notamment dans des contextes où la connectivité, la scalabilité et la fiabilité sont des exigences critiques[11].

Dans ce contexte, les systèmes de détection d'intrusion (IDS) basés sur l'apprentissage automatique se positionnent comme des approches prometteuses pour renforcer la sécurité des réseaux IoT/IIoT. Toutefois, nous observons que leur efficacité reste étroitement liée à la qualité et à la représentativité des données utilisées pour leur entraînement, ainsi qu'à la capacité des modèles à traiter des flux réseau massifs, hétérogènes et dynamiques.

Dans ce chapitre, nous présentons la conception et l'évaluation d'un système de détection d'intrusion basé sur un réseau de neurones convolutif (CNN), entraîné de manière centralisée sur le jeu de données Edge-IIoTset, un dataset réaliste et complet proposé récemment par [1]. Ce dernier couvre une large gamme de capteurs, de protocoles de communication, et de types d'attaques, organisés autour d'un banc de test à sept couches. Il constitue aujourd'hui l'un des rares ensembles de données publics adaptés tant à l'apprentissage centralisé qu'au deep learning, ce qui le rend particulièrement pertinent pour les recherches avancées en cybersécurité IoT/IIoT.

L'approche que nous avons adoptée repose sur la mise en œuvre progressive d'un modèle CNN 1D, entraîné sur Google Colab à l'aide d'un notebook que nous avons développé. Celui-ci intègre toutes les étapes nécessaires à la chaîne de traitement : ingestion des données, prétraitement, entraînement, évaluation et visualisation des résultats. Nous avons veillé à ce que chaque étape soit reproductible, rigoureuse, et alignée sur les meilleures pratiques en matière de machine learning.

Les sections qui suivent détaillent successivement les différentes composantes de notre approche : l'acquisition et le traitement des données, la construction et la configuration du modèle CNN, la stratégie d'apprentissage adoptée, ainsi que l'analyse des performances

obtenues sur le jeu de données Edge-IIoTset.

À travers cette contribution, nous cherchons à démontrer comment l'intelligence artificielle, lorsqu'elle est bien encadrée par des données pertinentes et une méthodologie rigoureuse, peut offrir des solutions concrètes aux enjeux critiques de la cybersécurité dans les environnements connectés.

III.2. Conception d'un IDS basé sur CNN

Cette section détaille les choix techniques et les étapes de développement du système de détection d'intrusion (IDS) basé sur un réseau de neurones convolutif (CNN). L'implémentation est réalisée en mode centralisé sur le dataset Edge-IIoTset, dans l'environnement Google Colab.

III.2.1. Architecture Générale du Système

Dans ce travail, nous avons conçu une architecture complète pour un système de détection d'intrusion basé sur un réseau de neurones convolutif (CNN), dont le pipeline suit les étapes implémentées dans notre code expérimental. Le processus commence par l'importation du jeu de données DNN-EdgeIIoT-dataset.csv, issu du dataset Edge-IIoTset. Nous avons ensuite effectué un nettoyage initial des données en supprimant les colonnes inutiles (par exemple les adresses IP, ports, timestamps) et en éliminant les valeurs manquantes ou infinies. Une analyse exploratoire des classes d'attaques a révélé un fort déséquilibre dans la distribution des échantillons. Pour y remédier, nous avons mis en œuvre une méthode de rééquilibrage manuelle combinant le sous-échantillonnage des classes majoritaires et le sur-échantillonnage des classes minoritaires, afin que chaque classe contienne exactement 10 000 échantillons.. Après cette phase de préparation, nous avons appliqué une transformation de type one-hot encoding sur les labels multiclasses, puis nous avons normalisé l'ensemble des caractéristiques numériques avec un StandardScaler. Les données ont ensuite été divisées en ensembles d'entraînement et de test selon une répartition 80/20. Pour rendre les données compatibles avec une architecture CNN unidimensionnelle, nous avons effectué un remodelage (reshape) des tableaux d'entrée. L'architecture CNN que nous avons développée comprend une succession de couches Conv1D, MaxPooling1D, Dropout, Flatten, et Dense, avec des fonctions d'activation ReLU dans les couches intermédiaires et Softmax en sortie.

L'entraînement a été réalisé avec l'optimiseur Adam, un batch size de 32 et jusqu'à 100 époques, avec l'utilisation de callbacks EarlyStopping et ModelCheckpoint pour prévenir le surapprentissage. Enfin, le modèle a été évalué sur le jeu de test à l'aide de métriques telles que l'accuracy, le F1-score, la matrice de confusion et le rapport de classification, permettant de mesurer l'efficacité de notre système dans la détection multi-classes d'attaques dans un environnement IoT/IIoT. Le diagramme suivant résume l'architecture globale de notre système IDS basé sur CNN :

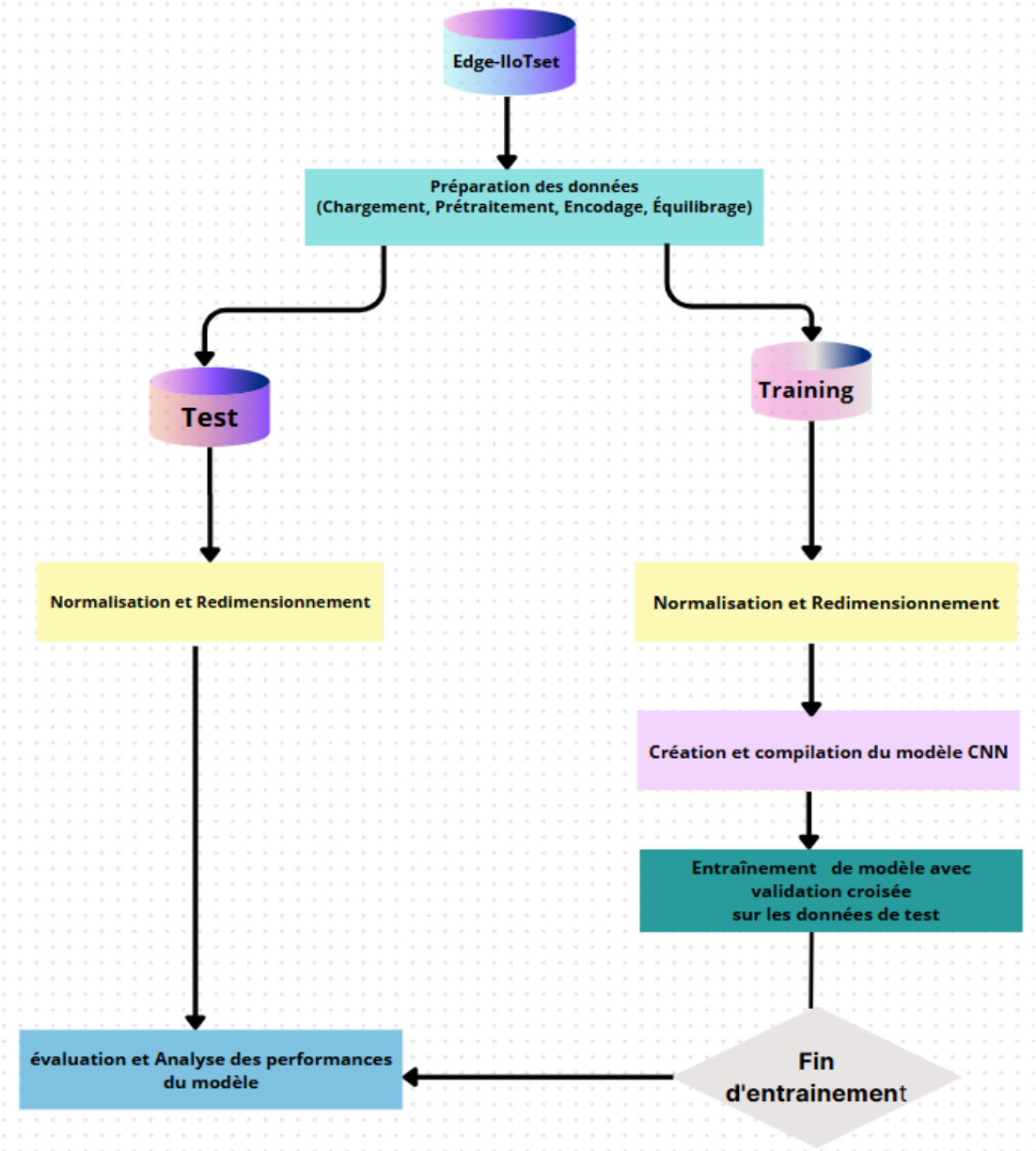


Figure III. 1: Schéma global de notre IDS basée sur le deep Learning (CNN)

III.2.2. Acquisition de données :

L'ensemble de données choisi pour cette étude est Edge-IIoTset[1], est un dataset récent et réaliste dédié à la recherche en cybersécurité dans les environnements IoT/IIoT , destiné à être utilisé dans des systèmes de détection d'intrusions (IDS) en modes d'apprentissage centralisé et fédéré. Les données ont été générées à partir d'un testbed sophistiqué organisé en sept couches , incorporant plus de dix types de capteurs IoT réels (capteurs de température et humidité, ultrasoniques, capteurs de flamme, capteurs de son, capteurs de pH, etc.) et plusieurs dispositifs IIoT via le protocole Modbus TCP/IP.

L'ensemble de données se compose de trois grandes parties : trafic normal capturé depuis les capteurs IoT et IIoT, trafic d'attaque comprenant quatorze types d'attaques réparties en cinq catégories principales (DoS/DDoS, collecte d'informations, Man-in-the-Middle, injection, et Malware), et ensembles sélectionnés pour l'apprentissage automatique et profond. Les données sont disponibles à la fois sous forme de fichiers PCAP (trafic brut) et CSV (flux extraits et enrichis). L'extraction de 61 caractéristiques pertinentes parmi 1176 initialement trouvées a été réalisée à l'aide des outils Zeek et TShark, en combinant des informations issues du trafic réseau, des ressources système et des journaux.

Les ensembles CSV destinés à l'apprentissage supervisé ont été nettoyés (suppression des colonnes inutiles, suppression des doublons et des valeurs manquantes), encodés (variables catégorielles transformées) et standardisés pour être exploitables par les modèles de Machine Learning. L'ensemble "DNN-EdgeIIoT-dataset.csv" contient les données adaptées pour les approches de Deep Learning, tandis que "ML-EdgeIIoT-dataset.csv" est destiné aux algorithmes classiques de Machine Learning.

Une illustration détaillée de taxonomie des attaques ainsi que les différentes sous-catégories d'attaques est présentée dans la figure III.2 et leur description succincte est donnée dans le tableau III.1

Catégorie Principale	Types d'Attaques Incluses	Description
Attaques de Déni de Service (DoS/DDoS)	- TCP SYN Flood - UDP Flood - HTTP Flood - ICMP Flood	Rendre les serveurs IoT inaccessibles en les saturant avec des paquets de requêtes.
Attaques de Collecte d'Informations	- Port Scanning - OS Fingerprinting - Vulnerability Scanner	Identifier les points faibles et les vulnérabilités des réseaux IoT.
Attaques Man-in-the-Middle (MITM)	- ARP Spoofing - DNS Spoofing	Intercepter les communications entre les dispositifs IoT et les serveurs.
Attaques par Injection	- SQL Injection - Cross-Site Scripting (XSS) - Uploading Attack (Webshell Injection)	Insérer des scripts ou des requêtes malveillantes pour modifier ou voler des données.
Attaques Malware et Intrusions	- Backdoor attack - Password cracking - Ransomware attack	Installer des logiciels malveillants pour obtenir un accès non autorisé et contrôler le système.

Tableau III. 1: Les attaques classées en cinq grandes catégories

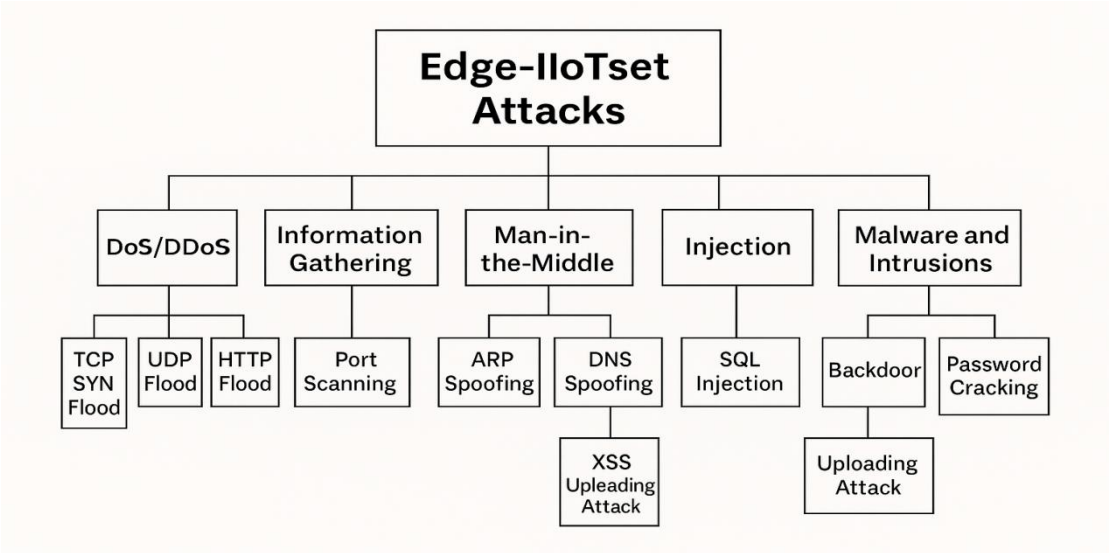


Figure III. 2: Taxonomie des attaques

- Pour télécharger et ingérer l'ensemble de données Edge-IIoTset dans notre environnement Google Colab, nous avons d'abord configuré l'accès à l'API Kaggle en téléchargeant le fichier d'authentification kaggle.json. Ce fichier contient les informations nécessaires pour authentifier notre compte Kaggle et permettre l'accès au dataset. Une fois l'authentification configurée, nous avons téléchargé le dataset directement depuis Kaggle. Le dataset a été récupéré sous forme de fichier compressé, que nous avons ensuite extrait pour rendre les fichiers accessibles dans notre environnement Colab. Une fois extraites, les données ont été prêtes à être utilisées pour le prétraitement et l'entraînement du modèle de détection d'intrusions basé sur CNN. Cette procédure a permis d'intégrer efficacement les données dans notre pipeline d'analyse, garantissant ainsi leur disponibilité pour les étapes suivantes du projet.

-

III.2.3. Prétraitement des Données et Ingénierie des Caractéristiques :

Avant d'entamer la phase d'entraînement de notre modèle CNN, un processus rigoureux de préparation des données a été mis en œuvre afin de garantir la qualité, la cohérence et la compatibilité du jeu de données avec les exigences de l'apprentissage profond.

Nous avons travaillé à partir du fichier **DNN-EdgeIIoT-dataset.csv**, conçu spécifiquement pour les approches de deep learning. Ce fichier contient 61 caractéristiques extraites à partir de données réseau, système et journaux, après une phase de sélection réalisée par les auteurs du dataset.

a) Sélection et Nettoyage des Colonnes:

La première étape consiste à examiner et filtrer les colonnes du dataset afin d'exclure les attributs non pertinents. Par exemple, les adresses IP, les numéros de ports et les informations textuelles complexes comme les payloads ou requêtes brutes sont retirés. Ces champs ne contribuent pas directement à la détection des intrusions et peuvent introduire du bruit ou des biais, ce qui pourrait nuire à la capacité d'apprentissage du modèle. En éliminant ces colonnes, nous réduisons la dimensionnalité du jeu de données, ce qui diminue la charge computationnelle et limite le risque de surapprentissage (overfitting).

Nous avons également supprimé les lignes comportant des valeurs manquantes, aberrantes ou des doublons. Ces anomalies peuvent fausser les calculs et dégrader la qualité du modèle si elles ne sont pas traitées. Enfin, nous avons mélangé (randomisé) les échantillons pour éviter tout biais lié à l'ordre des données, assurant ainsi une distribution représentative lors de la division en ensembles d'entraînement et de test.

Le tableau ci-dessous illustre la distribution des classes avant et après cette phase de nettoyage :

IoT Trafic	Type de trafic (Attack_type)	Nombre d'échantillons avant nettoyage	Nombre d'échantillons Après nettoyage
Normal	Normal	1 615 643	1363998
Attaques	DDoS_UDP	121 568	121567
	DDoS_ICMP	116 436	67939
	SQL_injection	51 203	50826
	Password	50 153	49933
	Vulnerability_scanner	50 110	50026
	DDoS_TCP	50 062	50 062
	DDoS_HTTP	49 911	48544
	Uploading	37 634	36807
	Backdoor	24 862	24026
	Port_Scanning	22 564	19977
	XSS	15 915	15066
	Ransomware	10 925	9689
	MITM	1 214	358
	Fingerprinting	1 001	853

Tableau III. 2: distribution des attaque avant et après le nettoyage

Nous remarquons également une forte inégalité dans la distribution des classes, notamment entre le trafic normal et les attaques rares comme MITM ou Fingerprinting. Ce déséquilibre devra être traité ultérieurement, par des techniques de rééchantillonnage ou de pondération.

b) Encodage des Variables Catégorielles:

Certaines colonnes du dataset, telles que http.request.method, mqtt.protoname ou dns.qry.name.len, contiennent des valeurs catégorielles. Or, les modèles de réseaux de neurones ne peuvent traiter que des entrées numériques. Pour résoudre cette incompatibilité, nous avons appliqué un encodage one-hot — une méthode couramment utilisée dans l'apprentissage profond [1]. Cette approche transforme chaque catégorie en un vecteur binaire où une seule position est activée, ce qui permet au modèle de distinguer clairement chaque catégorie sans introduire d'ordre hiérarchique artificiel. Cet encodage facilite l'apprentissage des relations entre les catégories par le réseau CNN.

c) Normalisation et Redimensionnement:

Les caractéristiques numériques présentent souvent des échelles très différentes (par exemple, des temps, des tailles de paquets ou des compteurs), ce qui peut ralentir voire empêcher la convergence du modèle. Pour remédier à cela, nous avons appliqué une normalisation Min-Max qui ramène toutes les valeurs dans l'intervalle [0,1], comme recommandé dans de nombreux travaux en deep learning [2]. Cette homogénéisation permet d'éviter qu'une variable à grande échelle domine l'apprentissage, assurant ainsi une progression plus stable et rapide des algorithmes d'optimisation.

Enfin, pour que les données soient compatibles avec une architecture de réseau convolutif unidimensionnel (1D-CNN), nous avons procédé au redimensionnement (reshape) des données en tenseurs tridimensionnels de forme (nombre d'échantillons, nombre de caractéristiques, 1). Cette mise en forme est nécessaire car les couches Conv1D attendent une entrée avec cette structure, qui leur permet d'exploiter efficacement la structure locale et séquentielle des données.

III.2.4. Architecture du Modèle CNN :

Dans le but de concevoir un système de détection d'intrusion (IDS) efficace pour les environnements IoT/IIoT, nous avons développé un modèle basé sur un réseau de neurones convolutif unidimensionnel (1D-CNN). Ce type d'architecture est particulièrement bien adapté à l'analyse de séquences temporelles et de flux réseau, car il permet d'extraire automatiquement des caractéristiques discriminantes à partir de vecteurs de caractéristiques normalisés.

L'architecture du modèle CNN que nous avons adoptée repose sur une structure séquentielle, construite à l'aide de l'API **Sequential()** de **Keras**. Elle est constituée des couches suivantes :

- ***Première couche convolutive :*** cette couche utilise un filtre de taille 3 sur les données d'entrée pour détecter des motifs locaux dans les séries temporelles. Le nombre de filtres est fixé à 64, ce qui permet au modèle d'extraire des caractéristiques variées à partir des données. La fonction d'activation ReLU (Rectified Linear Unit) est utilisée pour introduire de la non-linéarité et ainsi améliorer la capacité du modèle à capturer des relations complexes dans les données.
- ***MaxPooling1D(2) :*** après chaque convolution, une couche de MaxPooling est appliquée avec un facteur de réduction de 2. Cette opération permet de réduire la taille de la représentation des données, tout en préservant les informations les plus pertinentes. Elle réduit également la complexité computationnelle du modèle et aide à éviter le surapprentissage.
- ***Deuxième couche convolutive (Conv1D) :*** une deuxième couche convolutive est ajoutée avec 128 filtres de taille 3. Elle permet au modèle d'extraire des caractéristiques plus complexes après la réduction de la dimensionnalité effectuée par la première couche. Elle est suivie d'une autre couche de MaxPooling (MaxPooling1D(2)) pour une nouvelle réduction de la taille des données.
- ***Dropout(0.5) :*** un mécanisme de Dropout est appliqué avec un taux de 50 % pour éviter le surapprentissage. Cela consiste à désactiver aléatoirement la moitié des neurones pendant l'entraînement, forçant ainsi le modèle à apprendre des représentations plus robustes et à éviter de se sur-adapter aux données d'entraînement.

- ***Aplatir les résultats de la convolution (Flatten()) :*** après les couches convolutives et de pooling, les données doivent être converties en une forme unidimensionnelle pour être traitées par des couches denses. La fonction Flatten aplatit la sortie de la dernière couche de MaxPooling en un vecteur.
- ***Couches denses :*** une première couche dense est ajoutée avec 64 neurones et une fonction d'activation ReLU. Cette couche permet de créer des combinaisons complexes des caractéristiques extraites par les couches convolutives. Une autre couche de Dropout (50 %) est appliquée pour éviter le surapprentissage à ce stade.
- ***Couche de sortie (Dense) :*** la couche finale est une couche dense avec num_classes neurones, correspondant au nombre de classes dans la tâche de classification (par exemple, les classes des attaques ou le comportement normal). La fonction d'activation softmax est utilisée pour transformer les sorties en probabilités pour chaque classe, ce qui est essentiel dans le cadre de la classification multi-classes

L'architecture complète du modèle CNN est décrite dans le tableau suivant :

couches	formes de sortie	paramètres
Couche d'entrée	(None, 1, 67)	0
conv1d_2 (Conv1D)	(None, 94, 64)	256
max_pooling1d_2 (MaxPooling1D)	(None, 47, 64)	0
conv1d_3 (Conv1D)	(None, 45, 128)	24.704
max_pooling1d_3 (MaxPooling1D)	(None, 22, 128)	0
dropout_2 (Dropout)	(None, 22, 128)	0
flatten_1 (Flatten)	(None, 2816)	0
dense_2 (Dense)	(None, 64)	180.288
dropout_3 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 15)	975

Tableau III. 3: l'Architecture du Modèle CNN

III.2.5. Stratégie d'Entraînement

Pour garantir un apprentissage efficace et une bonne généralisation de notre modèle basé sur un réseau de neurones convolutif (Convolutional Neural Network – CNN), nous avons mis en œuvre une stratégie d'entraînement structurée, combinant une préparation équilibrée des

données, des techniques de régulation durant l'entraînement, et une configuration optimale des paramètres d'apprentissage (hyperparamètres).

Dans un premier temps, nous avons divisé notre ensemble de données équilibré en deux sous-ensembles : 80 % pour l'entraînement (training set) et 20 % pour les tests (test set), selon un découpage stratifié (stratified split), afin de préserver la proportion des 15 classes dans chaque portion. Cela permet de maintenir une distribution cohérente entre les phases d'apprentissage et d'évaluation.

Compte tenu de la forte déséquilibration du dataset Edge-IIoTset, nous avons appliqué un rééquilibrage manuel pour fixer un maximum de 10 000 échantillons par classe. Cette opération a impliqué du sous-échantillonnage (downsampling) pour les classes majoritaires (par exemple, le trafic normal) et du sur-échantillonnage (up-sampling) pour les classes minoritaires (telles que XSS, MIM ou Fingerprinting), en dupliquant des exemples existants. Cette étape est cruciale pour garantir une apprentissage équitable entre toutes les classes.

L'entraînement du modèle a été encadré par deux mécanismes de régulation (callbacks) essentiels pour éviter le surapprentissage (overfitting) :

- ✓ **Le rappel d'arrêt anticipé (EarlyStopping):** qui surveille la fonction de perte sur les données de validation (validation loss) et interrompt automatiquement l'entraînement après 5 époques sans amélioration. Cette méthode restaure également les meilleurs poids enregistrés pendant l'entraînement.
- ✓ **Le rappel de sauvegarde du meilleur modèle (ModelCheckpoint):** qui permet d'enregistrer uniquement le modèle ayant obtenu la meilleure performance sur le jeu de validation. Le fichier sauvegardé est ensuite utilisé pour l'évaluation finale.

L'entraînement a été effectué avec les hyperparamètres suivants :

- ✓ **Taille du batch (batch size) :** fixée à 64, ce qui constitue un bon compromis entre vitesse de calcul et stabilité de l'optimisation.
- ✓ **Nombre d'époques (epochs) :** défini à 30, en laissant la possibilité à l'EarlyStopping d'interrompre l'entraînement plus tôt si nécessaire.
- ✓ **Fonction de coût (loss function):** categorical crossentropy, adaptée à notre tâche de classification multi-classes.
- ✓ **Optimiseur (optimizer) :** Adam, choisi pour sa capacité à adapter dynamiquement les

taux d'apprentissage et pour sa convergence rapide dans les architectures CNN.

III.3. Configuration expérimentale et méthodologie

III.3.1. Préparation du dataset Edge-IIoTset

Dans le cadre de notre travail, nous avons utilisé le jeu de données Edge-IIoTset, une ressource récemment proposée pour l'étude de la cybersécurité dans les environnements IoT et IIoT. Ce dataset a été conçu pour refléter de manière réaliste les interactions réseau entre objets connectés industriels, en simulant à la fois du trafic normal et divers scénarios d'attaques. Il se distingue par son ampleur et sa richesse : il contient plus de 80 millions de paquets réseau, répartis sur plusieurs types de protocoles (HTTP, Modbus, MQTT, DNS, etc.), et couvre un large éventail d'attaques réparties en 15 classes, dont une classe "Normal" et 14 classes d'attaques appartenant à des familles variées telles que DDoS, injection, backdoor, scanning, malware, ou attaque de type homme du milieu (MITM).

Afin d'exploiter ce dataset dans le cadre de notre modèle basé sur un CNN, nous avons utilisé la version tabulaire prétraitée intitulée DNN-EdgeIIoT-dataset.csv, mise à disposition par les auteurs sur Kaggle. Cette version agrège les flux réseau sous forme de lignes représentant des connexions individuelles, avec pour chaque ligne un ensemble de 70 caractéristiques numériques extraites automatiquement à partir des paquets bruts. Une colonne supplémentaire encode la classe cible (étiquette) correspondant au type de trafic ou d'attaque associé à chaque échantillon.

Nous avons importé ce fichier dans notre environnement de travail sur Google Colab, en utilisant l'API Kaggle pour automatiser le téléchargement. L'ensemble des données a ensuite été chargé dans un DataFrame Pandas, ce qui nous a permis de les manipuler aisément pour les étapes suivantes du pipeline : nettoyage, encodage, normalisation, équilibrage, et découpage. Cette préparation a constitué une étape essentielle pour garantir la qualité des données fournies au modèle et assurer un entraînement efficace. Figure III.3 expliquer distribution des attaques. La figure ci-dessous présente la répartition des types d'attaques dans le dataset :

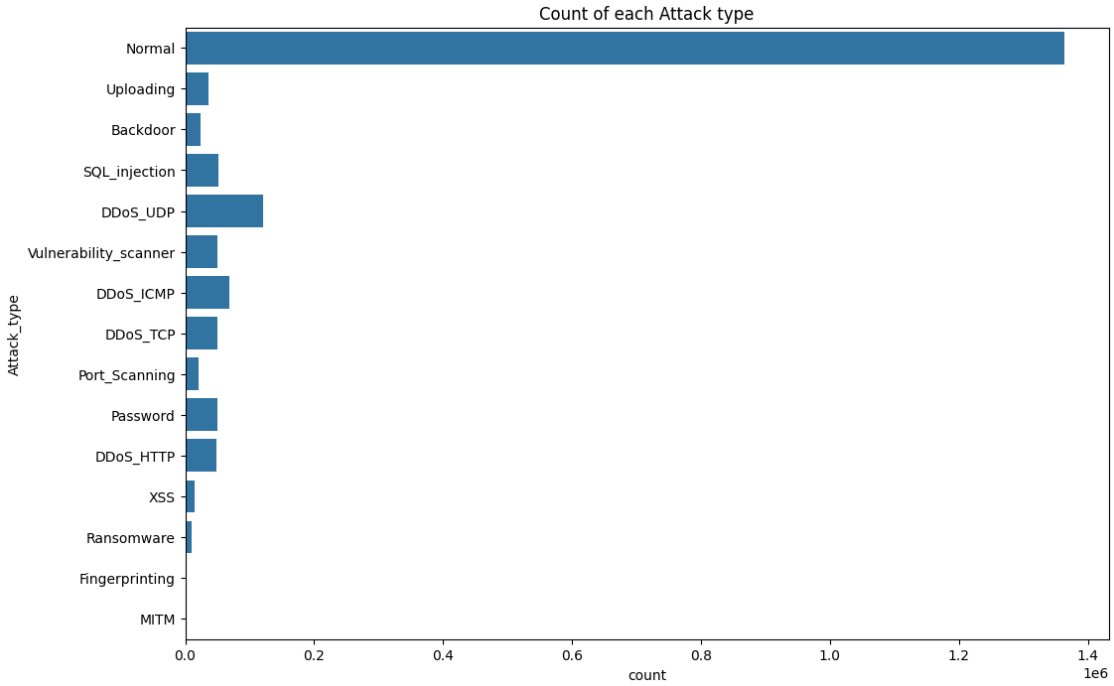


Figure III. 3: distribution des attaques

III.3.2. Division et Équilibrage des Données

✓ **la Séparation Stratifiée :**

les données sont séparées en deux ensembles distincts : un ensemble d'entraînement et un ensemble de test. Cette séparation est effectuée de manière stratifiée, ce qui signifie que la répartition des différentes classes (normal ou attaque) dans le dataset est préservée dans les deux ensembles. La stratification garantit que chaque ensemble contient une proportion équivalente des différentes catégories, ce qui est crucial pour que le modèle soit capable de bien apprendre à partir de toutes les classes présentes dans le dataset. En d'autres termes, les échantillons de chaque classe sont répartis de manière égale entre les ensembles d'entraînement et de test, évitant ainsi un biais qui pourrait se produire si certaines classes étaient sur-représentées ou sous-représentées dans l'un des ensembles.

Cette séparation est effectuée à l'aide de la fonction train_test_split de Scikit-learn, où la variable cible y (qui contient les types d'attaque) est utilisée pour effectuer la stratification. Cela permet de garantir que la distribution des classes dans les ensembles d'entraînement et de test reste cohérente avec la distribution initiale des classes dans l'ensemble de données complet.

✓ **Procédure d'Équilibrage des Classes :**

Une fois la séparation des données effectuée, le code applique une procédure d'équilibrage des classes pour gérer le déséquilibre entre les différentes classes. Le déséquilibre des classes se produit lorsque certaines classes, comme le trafic normal, sont beaucoup plus fréquentes que d'autres, comme les attaques DDoS ou Man-in-the-Middle, dans le dataset.

Pour résoudre ce problème, le code utilise une méthode qui permet d'équilibrer les classes en ajustant le nombre d'échantillons pour chaque classe. La stratégie utilisée consiste à sous-échantillonner (down-sampling) les classes majoritaires et à sur-échantillonner (up-sampling) les classes minoritaires, de manière à atteindre un nombre cible d'échantillons pour chaque classe. Par exemple, un nombre minimum d'échantillons par classe est défini (ici, 10 000 échantillons par classe), et les classes qui n'ont pas assez d'échantillons sont suréchantillonnées, tandis que les classes ayant trop d'échantillons sont sous-échantillonnées.

Ainsi, après l'équilibrage, chaque classe (qu'il s'agisse d'attaques ou de trafic normal) aura un nombre d'échantillons égal, ce qui aide le modèle à apprendre à partir d'une représentation équitable de toutes les classes. Cela permet de mieux prédire les attaques rares et d'éviter un biais vers la classe majoritaire.

Une fois l'équilibrage effectué, les données sont prêtes à être utilisées pour l'entraînement du modèle, assurant ainsi une meilleure détection des attaques dans le contexte des systèmes IoT/IIoT.

III.3.3. Environnement de Développement

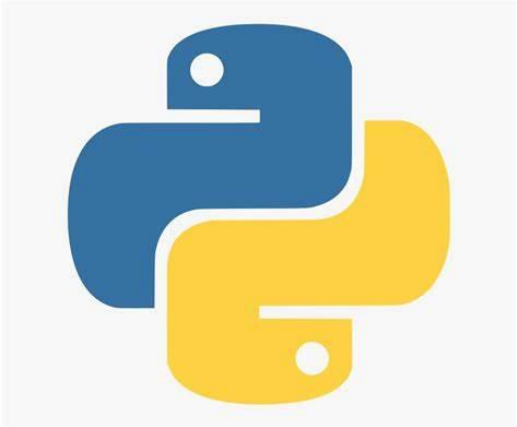
L'entraînement de notre modèle de détection d'intrusion a été réalisé dans un environnement de calcul hébergé sur :

✓ **Google Colab:**

Google Colab, abréviation de Google Colaboratory, est un environnement de développement en ligne basé sur Jupyter Notebook, proposé gratuitement par Google. Il permet d'écrire, d'exécuter et de partager du code Python directement depuis un

navigateur, sans nécessiter l'installation préalable de bibliothèques ou d'outils spécifiques sur la machine locale. Google Colab offre un accès gratuit à des ressources de calcul puissantes, notamment des unités de traitement graphique (GPU) et des unités de traitement tensoriel (TPU), ce qui en fait une solution particulièrement adaptée aux tâches d'apprentissage automatique et de calcul intensif. Grâce à son infrastructure cloud, il facilite également l'expérimentation, la collaboration et la reproductibilité des travaux en science des données et intelligence artificielle[30]. Plus précisément, nous avons exploité un GPU NVIDIA Tesla T4, doté de 16 Go de mémoire, ce qui a permis d'accélérer considérablement les calculs liés aux couches convolutives et aux opérations de rétropropagation. L'utilisation de l'accélération matérielle a été déterminante pour réduire les temps d'entraînement, en particulier lorsque le volume de données est important et que l'architecture comporte plusieurs millions de paramètres.

✓ Définition de langage de programmation :



Python est un langage de programmation interprété, open-source, et multiparadigme, conçu pour être simple, lisible et facile à apprendre. Créé par Guido van Rossum en 1991, Python favorise la programmation impérative, fonctionnelle et orientée objet, tout en offrant une syntaxe claire et concise qui facilite le développement rapide d'applications. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire et d'un système de gestion des exceptions, ce qui le rend adapté à une grande variété de domaines, tels que le développement web, la science des données, l'intelligence artificielle, l'apprentissage automatique, et l'automatisation.

Python est également apprécié pour sa vaste bibliothèque standard et son écosystème riche en bibliothèques tierces, comme NumPy, Pandas, TensorFlow et Scikit-learn, qui permettent de répondre à des besoins spécifiques dans des domaines variés.[31]

Sur le plan logiciel, nous avons construit notre pipeline avec les bibliothèques Python suivantes :

✓ **Panda:**

Pandas est une package open-source pour le langage de programmation Python, conçue pour la manipulation et l'analyse de données. Elle fournit des structures de données puissantes et flexibles, comme les DataFrames (tableaux à deux dimensions) et les Series (tableaux à une dimension), qui permettent de manipuler facilement des données tabulaires ou chronologiques [32]

. Pandas est largement utilisée pour des tâches telles que le nettoyage, le filtrage, l'agrégation, la transformation et la visualisation des données. Grâce à ses fonctionnalités avancées, elle est devenue un outil incontournable dans les domaines de la science des données, de l'apprentissage automatique et de l'analyse

✓ **NumPy:**

NumPy (Numerical Python) est une bibliothèque open-source pour le langage de programmation Python, conçue pour effectuer des calculs scientifiques et mathématiques de manière efficace. Elle fournit des outils puissants pour manipuler des tableaux multidimensionnels (arrays) et des matrices, ainsi qu'une large collection de fonctions mathématiques de haut niveau pour effectuer des opérations sur ces structures [33]

✓ **Scikit-learn:**

Scikit-learn, également appelée sklearn, est une bibliothèque open-source de machine learning pour Python. Elle est largement utilisée pour effectuer des tâches d'apprentissage supervisé (comme la classification et la régression) et non supervisé (comme le clustering et la réduction de

dimensionnalité). Scikit-learn propose une interface simple et cohérente pour implémenter et comparer des modèles d'apprentissage automatique. Elle permet aux utilisateurs de choisir parmi une variété de classes d'algorithmes, de les paramétrer, de les entraîner, et de les évaluer sur des ensembles de données.

✓ TensorFlow:

TensorFlow est une bibliothèque open-source développée par Google pour le machine learning et le deep learning. Créée initialement en 2011 sous le nom de DistBelief, elle a été rendue publique en 2015. TensorFlow permet de concevoir, d'entraîner et de déployer des modèles d'apprentissage automatique, notamment des réseaux de neurones, en utilisant une architecture basée sur des graphes de flux de données. Les nœuds de ces graphes représentent des opérations mathématiques, tandis que les connexions entre eux transportent des tenseurs, qui sont des matrices ou des vecteurs multidimensionnels.

Cette bibliothèque est compatible avec plusieurs plateformes, telles que les CPU, GPU et TPU (Tensor Processing Units), et peut être utilisée sur des appareils locaux, des serveurs cloud ou des appareils mobiles. TensorFlow propose des API de haut niveau, comme Keras, pour simplifier le développement de modèles, tout en offrant des API de bas niveau pour un contrôle plus précis. Elle est largement utilisée dans des domaines tels que la reconnaissance d'images, le traitement du langage naturel, les systèmes de recommandation et les applications d'intelligence artificielle avancées. Grâce à son caractère modulaire et à son soutien par Google, TensorFlow est devenu un outil incontournable pour les chercheurs et les développeurs travaillant sur des projets d'apprentissage automatique.

✓ Keras:

Keras est une bibliothèque open-source écrite en Python, conçue pour simplifier le développement et le

prototypage rapide de modèles de deep learning. Créée en 2015 par François Chollet dans le cadre du projet ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), Keras est une API de haut niveau qui s'appuie sur des frameworks de machine learning tels que TensorFlow, JAX et PyTorch. Elle permet de construire et d'entraîner facilement des réseaux de neurones convolutifs, récurrents ou entièrement connectés grâce à une interface intuitive et modulaire.

Keras est conçue pour être accessible aux débutants tout en offrant une grande flexibilité pour les chercheurs et les développeurs. Elle propose des outils pour créer des couches de neurones, des fonctions d'activation, des optimiseurs et des schémas de régularisation, tout en permettant l'entraînement distribué sur plusieurs GPU ou TPU. Les modèles développés avec Keras peuvent être déployés sur diverses plateformes, telles que les appareils mobiles, les navigateurs web ou les environnements cloud. Grâce à sa simplicité et à son efficacité, Keras est devenue l'une des bibliothèques les plus populaires pour le deep learning, utilisée dans des domaines tels que la reconnaissance d'images, le traitement du langage naturel et les systèmes de recommandation.

Afin d'assurer la reproductibilité des résultats, nous avons fixé :

- ✓ **La graine aléatoire (random seed) :** est un mécanisme utilisé pour contrôler l'initialisation des processus aléatoires dans le programme. permet de garantir que ces choix sont reproductibles, c'est-à-dire que chaque fois que le modèle est entraîné avec la même graine, il produira les mêmes résultats. Cela est particulièrement important pour les chercheurs et les développeurs qui souhaitent tester un modèle dans des conditions identiques et comparer les performances sur des jeux de données constants.
- ✓ **Le versioning (gestion de versions):** consiste à suivre et à gérer les différentes versions des logiciels et des dépendances utilisées dans un projet. Cela permet de s'assurer que l'environnement de travail reste cohérent et stable au fil du temps, même si des mises à jour ou des modifications sont effectuées sur les outils utilisés, assurant ainsi la reproductibilité et la compatibilité du code.

L'ensemble de ces choix nous a permis de créer un environnement stable, transparent et réplicable, adapté au développement d'un modèle de détection d'intrusion performant dans un

contexte IoT.

III.3.4. Métriques d'évaluation :

Pour évaluer la performance de notre modèle CNN dans la tâche de détection d'intrusion multi-classe, nous avons utilisé un ensemble de métriques standards en classification supervisée, permettant de mesurer à la fois l'exactitude globale du modèle et sa capacité à différencier correctement les différentes classes, y compris les plus rares.

- **Exactitude (Accuracy) :** L'exactitude mesure la proportion de prédictions correctes (positives et négatives) par rapport à l'ensemble des échantillons. Elle est définie par :

$$Accuracy = \frac{\sum_{i=1}^{c} TP_i}{\sum_{i=1}^{c}(TP_i + +FP_i + FN_i)}$$

C : nombre total de classes.

TP (True Positives) : attaques correctement détectées comme attaques.

TN (True Negatives) : trafic normal correctement identifié.

FP (False Positives) : faux positifs (trafic normal classé à tort comme attaque).

FN (False Negatives) : attaques non détectées.

- **Précision (Precision) :** la précision représente la proportion de classifications correctes parmi toutes les instances prédites comme des attaques. Elle est donnée par :

$$Precision = \frac{TP_i}{TP_i + FP_i}$$

- **Rappel (Recall) :** le rappel indique la proportion des attaques correctement identifiées par rapport au nombre total réel d'attaques. Il est calculé comme suit :

$$Recall = \frac{TP_i}{TP_i + FN_i}$$

- **Score F1 (F1-Score) :** le score F1 représente la moyenne harmonique entre la précision et le rappel. Il est défini par la formule :

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

- **Le taux de fausses alarmes (False Alarm Rate – FAR) :** correspond à la proportion de trafic normal incorrectement identifié comme une attaque par rapport à l'ensemble du trafic réellement normal. Il est calculé comme suit :

$$FAR = \frac{FP_{Normal}}{FP_{Normal} + TN_{Normal}}$$

- **Matrice de confusion (matrice d'erreur) :** est une représentation tabulaire spécifique qui permet de visualiser les performances d'un algorithme d'apprentissage automatique appliqué à un problème de classification.

III.4. Résultats Expérimentaux

III.4.1. Performance de Formation et de Validation :

Afin d'évaluer la capacité d'apprentissage du modèle CNN que nous avons conçu, nous avons analysé l'évolution de ses performances tout au long de l'entraînement. Cette analyse repose sur le suivi de deux métriques clés : la précision (accuracy) et la fonction de perte (loss), mesurées sur les ensembles d'entraînement et de validation au fil des époques. En observant ces courbes, nous pouvons juger de la convergence du modèle, de sa stabilité pendant l'apprentissage, ainsi que de sa capacité de généralisation sur des données inédites. Ces éléments sont déterminants pour garantir l'efficacité du système dans un contexte réel de détection d'intrusion en environnement IoT.

La perte sur le jeu d'entraînement passe de 0.1906 à 0.1138, tandis que la perte sur le jeu de validation diminue de 0.1333 à 0.1046, indiquant que le modèle apprend efficacement à réduire l'erreur et à généraliser sur de nouvelles données. La précision sur le jeu d'entraînement augmente de 92.10% à 94.45%, et celle sur le jeu de validation atteint 94.75% à la fin de l'entraînement. Ces courbes suggèrent que le modèle n'est ni sur-appris ni sous-appris. En effet, l'absence de divergence notable entre les courbes d'entraînement et de validation montre que le modèle n'a pas mémorisé les détails des données d'entraînement, ce qui aurait été un signe de sur-apprentissage, et qu'il n'est pas resté bloqué dans un état de faible performance, ce qui aurait indiqué un sous-apprentissage. Ainsi, le modèle a bien convergé, apprenant de manière efficace les caractéristiques discriminantes tout en conservant

une bonne capacité de généralisation, ce qui le rend adapté à un déploiement dans des environnements IoT réels.

III.4.2. Évaluation de l'ensemble de test :

Après la phase d'entraînement, le modèle CNN a été évalué sur l'ensemble de test, constitué de données entièrement inédites, afin de mesurer sa capacité de généralisation. Cette étape est cruciale pour valider la robustesse et l'efficacité du système de détection d'intrusion dans des conditions réalistes, proches d'un déploiement en environnement IoT.

Les résultats finaux obtenus sont les suivants :

- ***Accuracy sur le jeu de test :*** 94,74 %
- ***Loss sur le jeu de test :*** 0,1045

Ces résultats indiquent que le modèle maintient un excellent niveau de performance sur des données qu'il n'a jamais vues auparavant. Le taux de précision élevé confirme que la majorité des échantillons sont correctement classés, tandis que la faible valeur de la fonction de perte traduit une bonne concordance entre les prédictions du modèle et les étiquettes réelles.

De plus, la stabilité des performances entre les ensembles de validation et de test (accuracy de 94,75 % en validation contre 94,74 % en test) montre que le modèle ne souffre ni de surapprentissage ni de dégradation significative face à de nouvelles données. Cela valide la stratégie d'entraînement adoptée, incluant le prétraitement des données, le rééquilibrage des classes, et le choix des hyperparamètres.

Ces performances confirment l'efficacité du CNN proposé comme solution de détection d'intrusion dans des systèmes IoT, avec une forte capacité à reconnaître des comportements normaux et malveillants dans des environnements complexes et dynamiques.

III.4.3. Matrice de confusion et rapport de classification :

nous avons analysé la matrice de confusion ainsi que le rapport de classification détaillant la précision, le rappel et le F1-score pour chaque classe. Ces métriques sont essentielles pour comprendre comment le modèle distingue les attaques et le trafic normal, particulièrement face à un déséquilibre des classes typique des environnements IoT.

Figure III.4 (Matrice de Confusion) présente la matrice de confusion du modèle. Elle montre les valeurs de vrais positifs (TP), faux positifs (FP), faux négatifs (FN) et vrais négatifs (TN)

pour chaque classe. Le modèle démontre une excellente performance pour certaines attaques fréquentes, telles que les attaques DoS/DDoS (Classe 4), avec un nombre élevé de vrais positifs et un faible taux de faux positifs, indiquant que le modèle parvient à identifier ces attaques de manière fiable. En revanche, pour des attaques moins fréquentes, comme la Classe 5 (par exemple, des attaques peu courantes), la matrice montre une augmentation des faux négatifs, suggérant que le modèle a du mal à détecter ces types d'attaques.

Une matrice de confusion est utilisée ici pour illustrer la performance du modèle de classification :

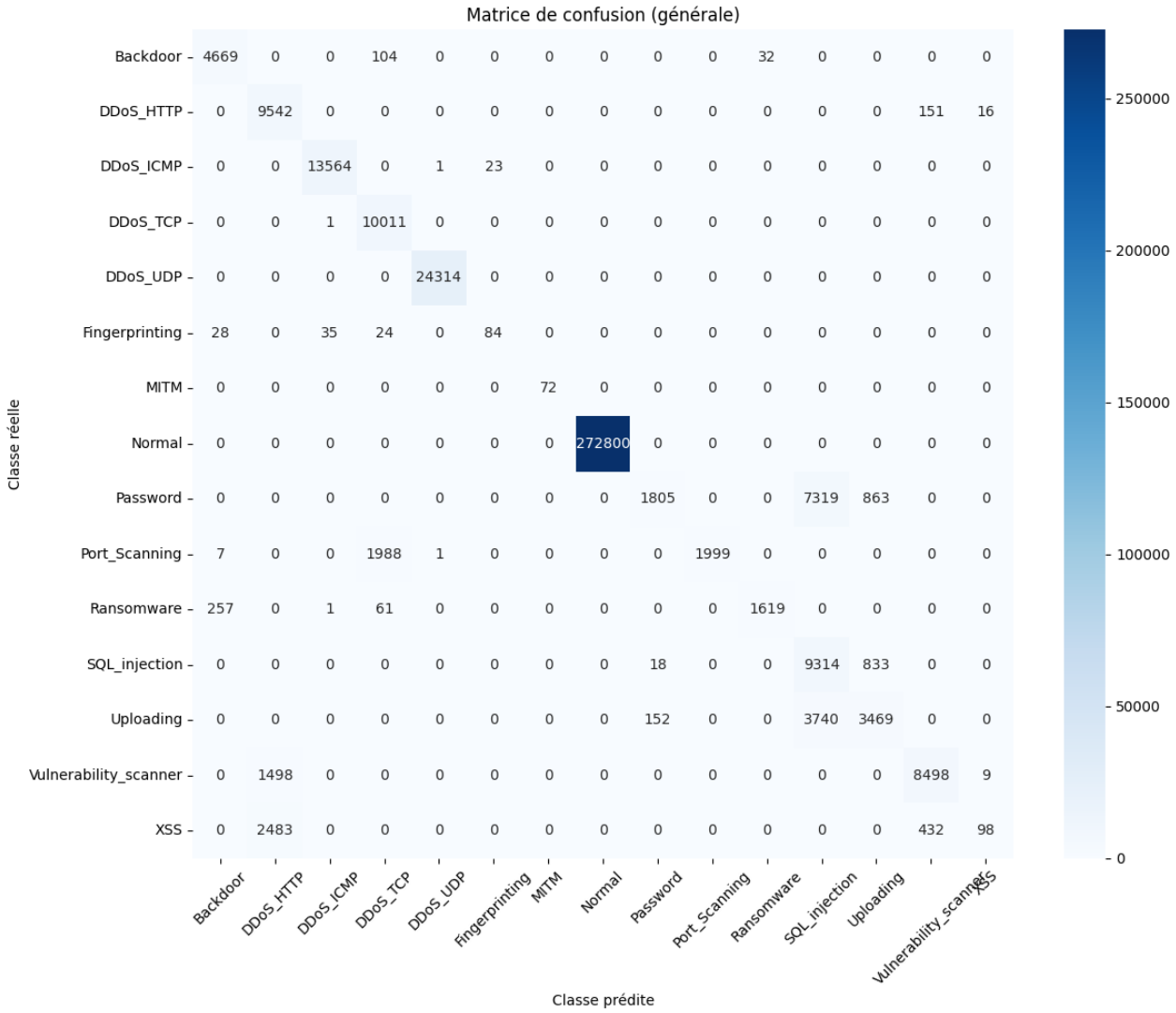


Figure III. 4: Matrice de confusion

Le tableau III.4 (Rapport de Classification) présente les performances détaillées du modèle CNN pour chaque classe d'attaque et pour le trafic normal, en termes de précision, rappel et

F1-score. Cette évaluation fine permet d'analyser la capacité du modèle à détecter correctement les différentes formes de trafic dans un environnement IoT.

Les résultats montrent que pour les classes possédant un grand nombre d'exemples, notamment la classe "Normal" (classe 7), le modèle atteint des performances remarquables, avec une précision, un rappel et un F1-score de 100 %. Cela traduit une excellente capacité à distinguer le trafic légitime des flux malveillants, ce qui est essentiel dans les systèmes de détection d'intrusion.

De même, les attaques fortement représentées comme DDoS_ICMP (classe 2), DDoS_UDP (classe 4) ou encore MIM (classe 6) sont détectées avec une précision et un rappel parfaits, atteignant un F1-score de 1.00, ce qui confirme la fiabilité du modèle pour les classes majoritaires.

En revanche, pour certaines classes minoritaires ou plus subtiles, les performances sont nettement inférieures. Par exemple, la classe "Fingerprinting" (classe 5) ne dépasse pas 0.49 de rappel malgré une précision de 0.79, ce qui donne un F1-score modeste de 0.60. De manière encore plus marquée, la classe "XSS" (classe 14) affiche un F1-score extrêmement faible de 0.06, avec un rappel de seulement 0.03, illustrant une grande difficulté du modèle à identifier ce type d'attaque.

Ces résultats soulignent l'importance de l'équilibrage des classes dans l'entraînement du modèle. Alors que certaines classes majeures sont bien détectées, le faible F1-score pour les classes rares suggère qu'une amélioration est nécessaire, possiblement par l'utilisation de techniques comme SMOTE ou par un ajustement des hyperparamètres.

Dans l'ensemble, la matrice de confusion et le rapport de classification fournissent une évaluation détaillée de la performance du modèle. Les résultats confirment que le modèle est efficace pour détecter les attaques courantes, mais qu'il nécessite des améliorations pour traiter les attaques rares, particulièrement en ce qui concerne le rappel et le F1-score des classes minoritaires.

Le tableau suivant résume les principales métriques de classification obtenues après l'évaluation du modèle :

Classe	Nbr Classe	Precision	Recall	F1-score	Nombre d'échantillons
Backdoor	0	0.94	0.97	0.96	4805
DDoS_HTTP	1	0.71	0.98	0.82	9790
DDoS_ICMP	2	1.00	1.00	1.00	13588
DDoS_TCP	3	0.82	1.00	0.90	10012
DDoS_UDP	4	1.00	1.00	1.00	24314
Fingerprinting	5	0.79	0.49	0.60	171
MIM	6	1.00	1.00	1.00	72
Normal	7	1.00	1.00	1.00	272800
Password	8	0.91	0.18	0.30	9987
Port_Scanning	9	1.00	0.50	0.67	3995
Ransmware	10	0.98	0.84	0.90	1938
SQL_injection	11	0.46	0.92	0.61	10165
Uploading	12	0.67	0.47	0.55	7361
Vulnerability	13	0.94	0.85	0.89	10005
XSS	14	0.80	0.03	0.06	3013

Tableau III. 4: Rapport de Classification

III.4.4. Courbes d'apprentissage :

Afin de mieux comprendre le comportement du modèle pendant l'entraînement, nous avons analysé les courbes d'apprentissage, en particulier celles de la précision (accuracy) et de la fonction de perte (loss), sur les ensembles d'entraînement et de validation. Ces courbes nous permettent d'évaluer la convergence du modèle ainsi que la présence éventuelle de sous-apprentissage (underfitting) ou de surapprentissage (overfitting).

La Figure III.6 (courbe de précision) illustre l'évolution de l'accuracy au fil des époques. Nous observons une progression rapide dès les premières itérations, traduisant la capacité du modèle à apprendre efficacement les motifs discriminants présents dans les données. La précision atteint environ 94,45 % sur le jeu d'entraînement et 94,75 % sur le jeu de validation,

avant de se stabiliser. La proximité entre les deux courbes suggère une bonne généralisation, avec peu de signes de surapprentissage.

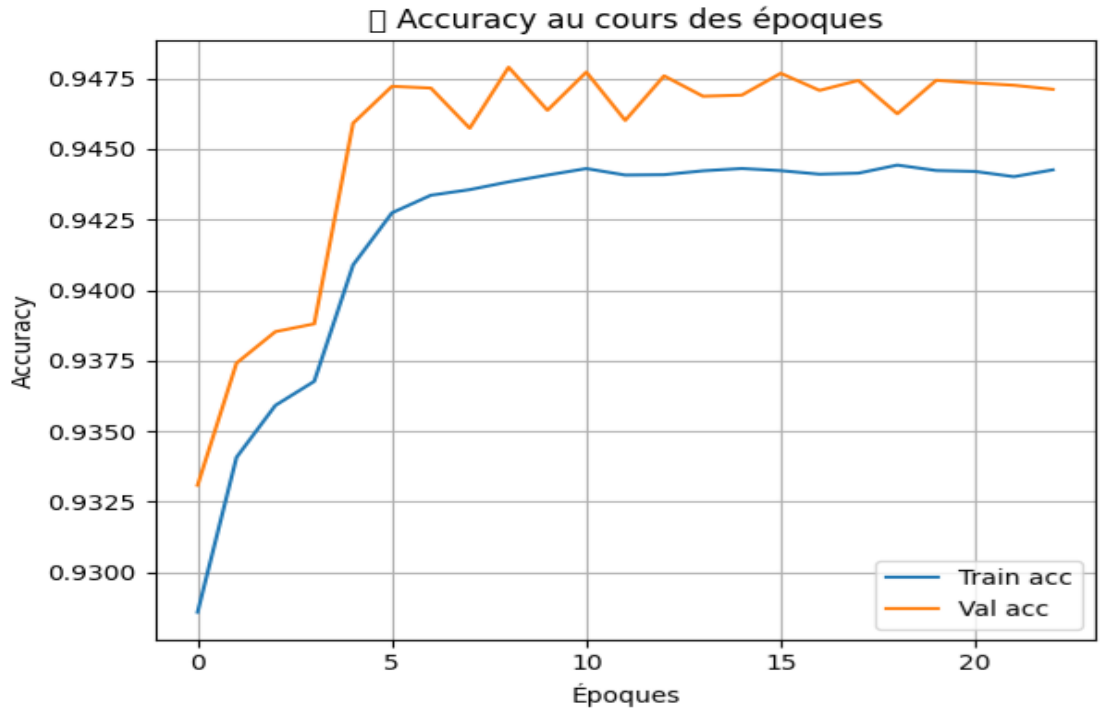


Figure III. 5: courbe de précision

En complément, la figure III.7 (le courbe de perte) montre une diminution progressive de la fonction de coût au cours de l'entraînement. La perte (loss) sur l'ensemble d'entraînement descend jusqu'à environ 0,112, tandis que La perte (loss) sur l'ensemble de validation atteint un plateau autour de 0,105. Le parallélisme des deux courbes, sans divergence notable, indique que l'apprentissage est stable et efficace, sans dérive vers une spécialisation excessive du modèle.

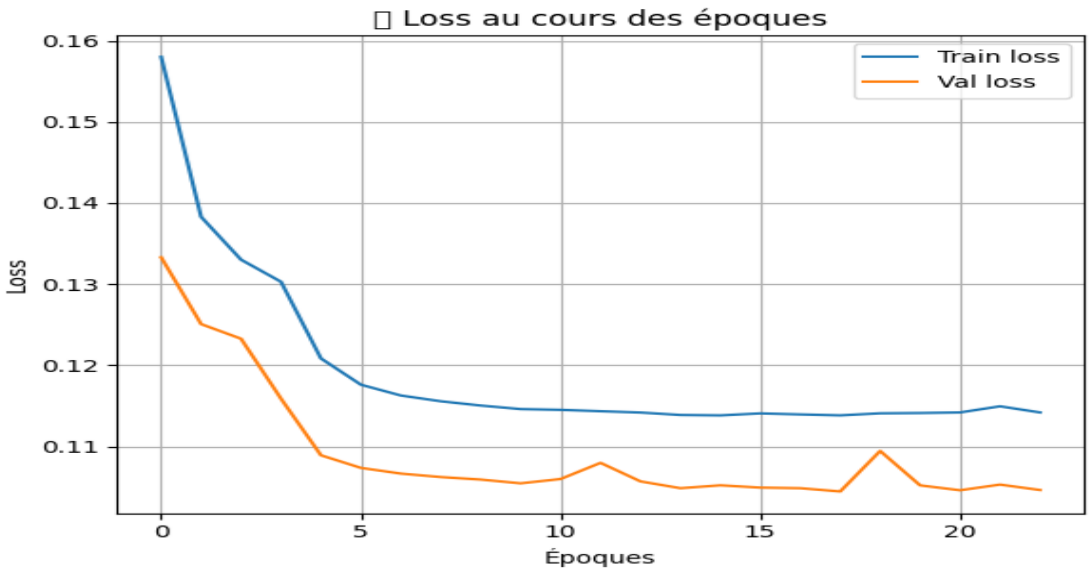


Figure III. 6: courbe de perte

Dans l'ensemble, l'analyse de ces courbes confirme que notre modèle CNN a convergé correctement, avec une configuration architecturale et des hyperparamètres pertinents. L'absence de surapprentissage comme de sous-apprentissage témoigne de la robustesse de notre approche. Cela renforce la pertinence de notre système pour une intégration dans des environnements IoT réels, où la stabilité et la fiabilité des performances sont des critères essentiels.

III.4.5. Comparaison avec l'apprentissage automatique traditionnel :

Dans le cadre de l'étude comparative réalisée à partir du dataset Edge-IIoTset, les auteurs de l'article [1] ont évalué les performances de quatre algorithmes classiques d'apprentissage automatique, dans le but de fournir une référence expérimentale pour la détection d'intrusions dans les réseaux IoT. La méthodologie suivie repose sur un prétraitement rigoureux des données et une sélection des attributs les plus pertinents, avant l'application des algorithmes suivants : Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM) et K-Nearest Neighbors (KNN).

Dans le prolongement de cette approche, nous avons entraîné et évalué deux modèles profonds sur le même ensemble de données : Un modèle basé sur un réseau de neurones convolutif (CNN) et une version hybride CNN-LSTM, combinant l'extraction spatiale et la mémoire temporelle. les résultats obtenus sont illustrées dans la figure III.7 :

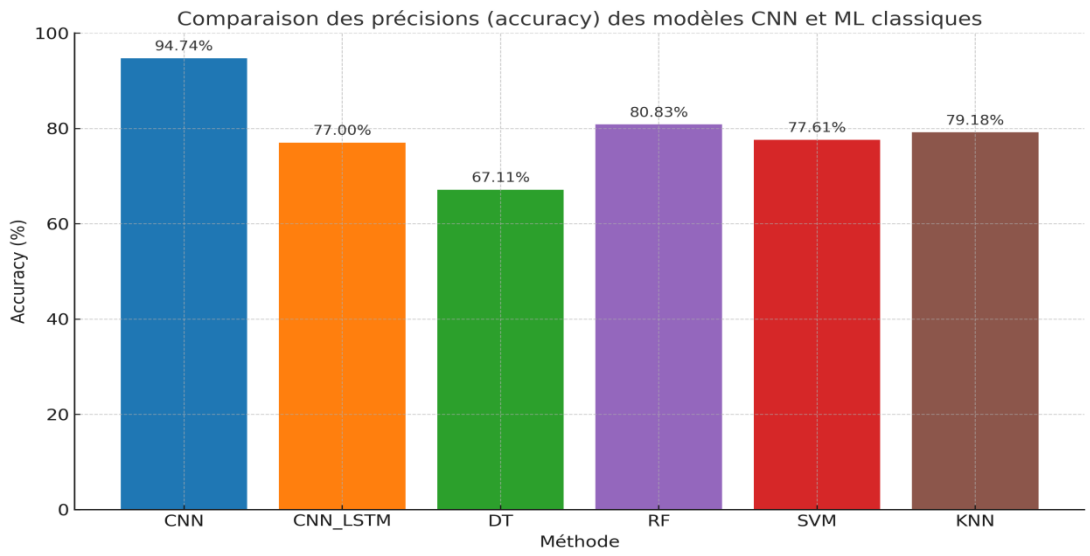


Figure III. 8: comparaison de précision (accuracy) des modèle CNN et ML classiques

Cette comparaison met en évidence une supériorité nette du modèle CNN, qui surpasse largement les algorithmes classiques avec une précision de près de 95 %. Ce résultat confirme l'efficacité du CNN pour extraire automatiquement des motifs discriminants complexes à partir de flux réseau multivariés. En revanche, le modèle CNN-LSTM, bien qu'intégrant une composante séquentielle via une couche de mémoire, n'améliore pas significativement la performance globale. Avec une précision de 77 %, il se situe au même niveau que certains modèles traditionnels, ce qui pourrait s'expliquer par une complexité excessive ou un manque d'optimisation de la partie LSTM, peu adaptée dans ce cas aux données statiques ou peu séquentielles du dataset..

III.5. Discussion

III.5.1. Impact des choix de prétraitement :

es étapes de prétraitement que nous avons appliquées ont joué un rôle déterminant dans les performances globales de notre système de détection basé sur un réseau de neurones convolutif. Étant donné la nature brute, multidimensionnelle et hétérogène des données issues du dataset Edge-IIoTset, un traitement rigoureux et adapté était indispensable pour permettre au modèle d'apprendre de manière efficace et stable.

Tout d'abord, nous avons procédé à une sélection de colonnes pertinentes, en supprimant les champs non informatifs ou redondants. Cette réduction de la dimensionnalité a permis de

limiter le bruit dans les données d'entrée, tout en conservant les caractéristiques discriminantes essentielles à la détection des comportements malveillants.

Ensuite, nous avons appliqué un encodage des variables catégorielles telles que les protocoles (HTTP, MQTT, DNS) via la méthode one-hot encoding, afin de les rendre exploitables par le réseau de neurones. Cette transformation a permis de préserver la nature nominale de ces informations tout en évitant toute hiérarchisation artificielle, ce qui aurait pu biaiser l'apprentissage du modèle.

La normalisation des données (min-max scaling) a également été une étape cruciale, permettant de ramener toutes les caractéristiques sur une échelle comparable. Cela a facilité la convergence de l'optimiseur en réduisant l'influence disproportionnée des attributs à grande valeur absolue sur la fonction de coût.

Enfin, les données ont été reformatées sous forme de tenseurs unidimensionnels afin d'être compatibles avec l'entrée d'un réseau CNN 1D. Ce format a permis d'exploiter efficacement la structure séquentielle et contextuelle des données réseau, tout en maintenant une faible complexité computationnelle.

Dans l'ensemble, ces choix de prétraitement ont grandement contribué à la stabilité de l'apprentissage, à la qualité de la généralisation observée sur les données de validation et de test, et à la réduction du surapprentissage. Nous considérons donc que le pipeline de préparation des données constitue un élément fondamental du bon fonctionnement de notre système.

III.5.2. Forces et Limites de l'Approche CNN :

L'utilisation d'un réseau de neurones convolutif (CNN) dans le cadre de la détection d'intrusion en environnement IoT nous a permis de tirer parti de plusieurs avantages inhérents à cette architecture, tout en prenant conscience de ses limitations.

- **Points forts :**

L'un des principaux atouts de l'approche CNN réside dans sa capacité à apprendre automatiquement des caractéristiques discriminantes à partir des données d'entrée, sans nécessiter d'ingénierie manuelle complexe. Grâce aux couches de convolution, notre modèle a pu capturer des motifs locaux pertinents au sein des séquences de données réseau, facilitant

ainsi la détection précise d'activités anormales ou malveillantes.

Nous avons également constaté que le CNN est bien adapté au traitement de données temporelles ou séquentielles, comme celles générées dans les communications réseau IoT. Le faible nombre de paramètres associé à une structure relativement simple (1D-CNN) a permis d'obtenir des temps d'entraînement raisonnables tout en maintenant une excellente précision globale de 94,74 % sur l'ensemble de test.

Par ailleurs, l'architecture s'est révélée robuste face au surapprentissage, comme en témoignent la stabilité des courbes d'apprentissage et la faible différence entre les performances en validation et en test. Cela confirme la pertinence de l'architecture choisie et des stratégies de régularisation mises en place (dropout, early stopping, etc.).

- **Limitations :**

Malgré ses bonnes performances, notre approche CNN présente certaines limites. En particulier, le modèle a montré des difficultés à détecter les classes d'attaque rares, comme l'atteste le faible score F1 pour les classes Password, XSS ou Fingerprinting. Cette faiblesse peut s'expliquer par la sensibilité des CNN aux déséquilibres de classes, une problématique récurrente dans les jeux de données en cybersécurité.

De plus, les CNN sont généralement moins efficaces pour modéliser des relations à long terme ou des dépendances complexes entre les événements. Dans certains cas, une attaque peut se manifester par des comportements échelonnés dans le temps, que le CNN ne parvient pas à relier. Une solution potentielle serait de combiner le CNN avec un réseau récurrent (RNN) ou un LSTM, capables de mieux capturer les dépendances temporelles.

Enfin, bien que notre modèle soit performant sur une machine de développement, son déploiement en temps réel sur des dispositifs IoT à faibles ressources pourrait nécessiter des optimisations supplémentaires (compression de modèle, quantification, inférence embarquée, etc.).

III.5.3. Considérations de Scalabilité et de Déploiement dans les Réseaux IoT Réels :

L'objectif ultime d'un système de détection d'intrusion tel que celui que nous avons

développé est son intégration dans un environnement IoT opérationnel, caractérisé par des contraintes spécifiques telles que la limitation des ressources matérielles, la variabilité du trafic réseau, et la nécessité de traitements en temps réel. Dans ce contexte, il est crucial d'évaluer la scalabilité de notre approche ainsi que sa faisabilité de déploiement dans des conditions réelles.

Notre modèle CNN, bien qu'entraîné dans un environnement de calcul disposant de ressources importantes (Google Colab avec GPU), présente une architecture relativement légère, composée d'un nombre modéré de couches et de paramètres. Cette compacité rend le modèle compatible avec une inférence rapide, ce qui constitue un atout pour un traitement embarqué ou en périphérie (edge computing). Néanmoins, les dispositifs IoT sont souvent contraints en mémoire, puissance de calcul et autonomie énergétique, ce qui impose des optimisations supplémentaires avant tout déploiement.

Pour répondre à ces contraintes, plusieurs pistes d'amélioration peuvent être envisagées. Tout d'abord, des techniques de compression de modèles telles que le pruning (élagage des connexions inutiles) ou la quantization (réduction de la précision numérique) permettraient de réduire la taille du modèle sans perte significative de performance. De plus, la distillation de connaissances pourrait être utilisée pour transférer les compétences d'un modèle complexe vers un modèle plus simple, mieux adapté aux plateformes embarquées.

Une autre voie prometteuse consiste à envisager un déploiement distribué, dans lequel l'inférence serait effectuée au niveau de passerelles edge ou de serveurs fog, plus puissants que les objets eux-mêmes, tout en restant proches des sources de données. Cela permettrait de maintenir une faible latence tout en assurant une bonne capacité de traitement.

Enfin, dans une perspective plus avancée, nous envisageons l'intégration de notre solution dans un cadre d'apprentissage fédéré (Federated Learning). Ce paradigme permettrait à chaque nœud IoT de contribuer à l'apprentissage collectif en entraînant localement une copie du modèle, sans avoir à partager ses données sensibles. Les poids des modèles seraient ensuite agrégés de manière sécurisée au niveau central. Cette approche garantit à la fois l'évolutivité, la confidentialité des données et la résilience du système face aux évolutions du trafic ou des menaces.

En résumé, bien que notre modèle CNN ait démontré une efficacité notable en environnement

expérimental, sa généralisation vers des réseaux IoT réels nécessite une réflexion approfondie sur les aspects liés à la légèreté du modèle, à la répartition de la charge de calcul, et à l'apprentissage collaboratif. Ces considérations seront au cœur de nos travaux futurs pour assurer un déploiement robuste, adaptable et sécurisé.

III.6. Résumé du Chapitre et Travaux Futurs

Dans ce chapitre, nous avons présenté la conception, la mise en œuvre et l'évaluation expérimentale de notre système de détection d'intrusion basé sur un réseau de neurones convolutif (CNN), appliqué au jeu de données Edge-IIoTset, spécifiquement conçu pour des environnements IoT et IIoT. Nous avons détaillé les différentes étapes du pipeline, depuis la préparation et l'équilibrage des données jusqu'à l'entraînement du modèle, en passant par le choix de l'architecture et des hyperparamètres.

L'évaluation du modèle a mis en évidence des performances très satisfaisantes, avec une précision globale de 94,74 % sur les données de test. Les courbes d'apprentissage ont montré une convergence stable, sans surapprentissage, et les résultats du rapport de classification ont confirmé la capacité du modèle à distinguer efficacement la plupart des classes, notamment le trafic normal et les attaques DDoS. La robustesse de l'architecture CNN et la rigueur du prétraitement ont largement contribué à cette réussite.

Cependant, malgré ces performances globalement élevées, notre système a montré des limitations dans la détection des classes d'attaque rares, telles que les attaques XSS, Password ou Fingerprinting. Ces faiblesses sont dues à la forte déséquilibration du dataset, qui pénalise l'apprentissage sur les classes peu représentées. Pour remédier à cela, nous recommandons l'utilisation de techniques d'oversampling comme SMOTE, l'ajustement des poids de classes dans la fonction de perte, ou encore l'intégration de classifieurs spécialisés pour ces attaques ciblées.

Par ailleurs, ce travail ouvre plusieurs perspectives. Dans une optique de déploiement réel, il sera nécessaire d'optimiser le modèle pour qu'il soit utilisable sur des dispositifs à faibles ressources, notamment à travers des techniques de compression (pruning, quantization) ou de distillation de connaissances (knowledge distillation). De plus, l'adoption d'un apprentissage fédéré (Federated Learning) constituerait une évolution naturelle de notre approche,

permettant à chaque nœud IoT d'apprendre localement, sans partage de données sensibles, tout en participant à un modèle global collaboratif. Cette approche renforcerait à la fois la confidentialité, la scalabilité, et la résilience du système face à l'évolution continue des attaques.

En conclusion, notre étude a démontré la faisabilité et l'efficacité d'un système de détection basé sur un réseau convolutif, tout en identifiant des axes d'amélioration essentiels pour permettre une intégration efficace et sécurisée dans les réseaux IoT réels.

CONCLUSION GÉNÉRALE

La transformation numérique des systèmes connectés, alimentée par la généralisation des technologies intelligentes et la multiplication des dispositifs interconnectés, a profondément modifié le paysage de la cybersécurité. Les réseaux de l'Internet des Objets (IoT), en particulier, deviennent des cibles privilégiées pour les cyberattaques en raison de leur structure décentralisée, de leur diversité technologique, et de leurs contraintes en ressources. Dans ce contexte, il est devenu crucial de développer des solutions de détection d'intrusion (IDS) capables de répondre aux exigences de ces environnements complexes et dynamiques.

Ce mémoire s'inscrit dans cette démarche en proposant la conception, l'implémentation et l'évaluation d'un système IDS basé sur un réseau de neurones convolutif (CNN), appliqué au jeu de données Edge-IIoTset, conçu pour refléter les spécificités des infrastructures IoT industrielles. En adoptant une méthodologie structurée comprenant le nettoyage et la transformation des données, l'équilibrage des classes, la définition d'une architecture CNN adaptée et l'analyse des résultats expérimentaux, nous avons mis en évidence l'efficacité du modèle pour la détection des intrusions, avec un bon compromis entre précision et complexité.

Les performances obtenues soulignent la pertinence du deep learning, et plus spécifiquement des CNN, dans le renforcement de la sécurité des réseaux IoT. Néanmoins, des enjeux restent à relever, notamment sur les plans de l'extensibilité, de la consommation énergétique et de la résistance face aux attaques sophistiquées. Par ailleurs, l'adoption de paradigmes émergents comme l'apprentissage fédéré constitue une orientation prometteuse, permettant d'améliorer la protection des données sensibles tout en optimisant les performances globales.

En définitive, cette étude contribue à une meilleure compréhension de l'intégration des approches d'intelligence artificielle dans les systèmes de sécurité IoT, et ouvre des perspectives intéressantes pour des solutions plus robustes, intelligentes et respectueuses de la vie privée.

REFERENCES

[1] Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access, 10, 40281-40306.

[2] Ferrag, M. A., Friha, O., Maglaras, L., Janicke, H., & Shu, L. (2021). Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis. IEEE Access, 9, 138509-138542.

[3] Ferrag, M. A., Kantarci, B., Cordeiro, L. C., Debbah, M., & Choo, K. K. R. (2023, May). Poisoning attacks in federated edge learning for digital twin 6g-enabled iots: An anticipatory study. In 2023 IEEE International Conference on Communications Workshops (ICC Workshops) (pp. 1253-1258). IEEE.

[4] Hamouda, D., Ferrag, M. A., Benhamida, N., Seridi, H., & Ghanem, M. C. (2024). Revolutionizing intrusion detection in industrial IoT with distributed learning and deep generative techniques. Internet of Things, 26, 101149.

[5] Ferrag, M. A., Friha, O., Kantarci, B., Tihanyi, N., Cordeiro, L., Debbah, M., ... & Choo, K. K. R. (2023). Edge learning for 6G-enabled Internet of Things: A comprehensive survey of vulnerabilities, datasets, and defenses. IEEE Communications Surveys & Tutorials.

[6]Zhou, J., Wang, Y., Liu, X., & Zhang, H. (2025). The Internet of Things under Federated Learning:A Review of the Latest Advances and Applications. Journal of Network and Computer Applications, 213, 103567.

[7]Hamouda, D. (2020). Un système de détection d'intrusion pour la cybersécurité (Mémoire de Master, Université de 8 Mai 1945 – Guelma, Algérie).

[8]Bououdina, S. (2020). Un apprentissage fédéré avec une sélection de clients pour la détection d'intrusions réseau (Mémoire de Maîtrise, École de technologie supérieure, Université du Québec, Montréal).

[9]Ayed, M. A. (2022). Apprentissage fédéré pour la détection des intrusions (Mémoire de Maîtrise, École de technologie supérieure, Université du Québec, Montréal).

[10] Lakhwani, K. (2020). Internet of Things (IoT). New Delhi, India : BPB Publications.

[11]Benmessaoud, A., & Bouziane, M. (2024). Apprentissage automatique pour la détection

d'intrusions dans l'IoT (Mémoire de master). Université Ibn Khaldoun de Tiaret.

[12]Abbassi, Y., & Benlahmer, H. (2021, mars). Un aperçu sur la sécurité de l'internet des objets (IoT). Communication présentée au Colloque sur les Objets et systèmes Connectés (COC'2021), Marseille, France.

[13] Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Alazab, M., Bhattacharya, S., Maddikunta, P. K. R., & Gadekallu, T. R. (2022). Federated Learning for Intrusion Detection System: Concepts, Challenges and Future Directions. Computer Communications, 190, 1–10.

[14]Olanrewaju-George, B., & Pranggono, B. (2025). Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models. Cyber Security and Applications, 3, 100068.

[15]Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Bhattacharya, S., Maddikunta, P. K. R., & Gadekallu, T. R. (2021). Federated Learning for Intrusion Detection System: Concepts, Challenges and Future Directions. arXiv preprint, arXiv:2106.09527.

[16] Akinie, R., Gyimah, N. K., Bhavsar, M., & Kelly, J. (2025). Fine-Tuning Federated Learning-Based Intrusion Detection Systems for Transportation IoT. arXiv preprint, arXiv:2502.06099.

[17] Torre, D., Chennamaneni, A., Jo, J., Vyas, G., & Sabrsula, B. (2025). Toward Enhancing Privacy Preservation of a Federated Learning CNN Intrusion Detection System in IoT: Method and Empirical Study. ACM Transactions on Software Engineering and Methodology, 34(2), Article 53.

[18]Benhassine, Z., & Benhamou, K. (2022). Sélection des clients pour l'apprentissage fédéré dans l'Internet des objets. Mémoire de Master, Université Saad Dahlab de Blida 1.

[19]M. J. Idrissi, H. Alami, A. El Mahdaouy, A. El Mekki, S. Oualil, Z. Yartaoui, and I. Berrada, "Fed-ANIDS: Federated learning for anomaly-based network intrusion detection systems," Expert Systems with Applications, vol. 229, p. 121964, Jul. 2023,

[20]M. Moshawrab, M. Adda, A. Bouzouane, H. Ibrahim, et A. Raad, « Reviewing Federated Learning Aggregation Algorithms; Strategies, Contributions, Limitations and Future Perspectives », Electronics, vol. 12, no. 10, p. 2287, mai 2023, doi:

10.3390/electronics12102287

[21]Thein, T.T.; Shiraishi, Y.; Morii, M,"Personalized federated learning-based intrusion detection system: Poisoning attack and defense.",Future Generation Computer Systems, 153 (2024): 182-192.

[22]Javeed, D.; Saeed, M.S.; Adil, M.; Kumar, P.; Jolfaei, A. "A Federated Learning-Based Zero Trust Intrusion Detection System for Internet of Things.",Ad Hoc Networks, Volume 162, September 2024, Article Number 103540.

[23]Ceviz, O.; Sen, S.; Sadioglu, P."Distributed Intrusion Detection in Dynamic Networks of UAVs using Few-Shot Federated Learning.",EAI SecureComm 2024.

[24]Gourceyraud, M.; Ben Salem, R.; Neal, C.; Cuppens, F.; Boulahia Cuppens, N. "Federated Intrusion Detection System Based on Unsupervised Machine Learning." arXiv preprint, 2025.

[25]Friha, O.; Ferrag, M.A.; Shu, L.; Maglaras, L.; Choo, K.K.R.; Nafaa, M.,"FELIDS: Federated Learning-based Intrusion Detection System for Agricultural Internet of Things.",Journal of Parallel and Distributed Computing, Volume 165, March 2022, Pages 17-31.

[26]Alsaleh, S.; Menai, M.E.B.; Al-Ahmadi, S."A Heterogeneity-Aware Semi-Decentralized Model for a Lightweight Intrusion Detection System for IoT Networks Based on Federated Learning and BiLSTM.",Sensors, Volume 25, Issue 4, February 2025, Article Number 1039

[27]Repalle, S. A., & Kolluru, V. R. (2017). Intrusion detection system using ai and machine learning algorithm. International Research Journal of Engineering and Technology (IRJET), 4(12), 1709-1715.

[28]GeeksforGeeks. (n.d.). Machine Learning. GeeksforGeeks. https://www.geeksforgeeks.org/machine-learning/

[29]Bououdina, S. (2020). Un apprentissage fédéré avec une sélection de clients pour la détection d'intrusions réseau (Doctoral dissertation, École de technologie supérieure).

[30]DataScientest. (2023). *Google Colab : tout savoir*. https://datascientest.com/google-colab-tout-savoir

[31]Python Software Foundation. (2024). *Tutoriel Python – Documentation Python 3.12.3*. https://docs.python.org/fr/3/tutorial/

[32]Gallic, E. (s.d.). *Pandas en Python*. https://egallic.fr/Enseignement/Python/pandas.html

[33]Data Transition Numérique. (s.d.). *NumPy en Python : à quoi sert-il et comment l'utiliser ?*. https://www.data-transitionnumerique.com/numpy-python/

[34]Sasi, T., Lashkari, A. H., Lu, R., Xiong, P., & Iqbal, S. (2024). A comprehensive survey on IoT attacks: Taxonomy, detection mechanisms and challenges. Journal of Information and intelligence, 2(6), 455-513.

[35]Mouha, R.A. (2021) Internet of Things (IoT). Journal of Data Analysis and Information Processing, 9, 77-101. https://doi.org/10.4236/jdaip.2021.92006

[36]Liao, H., Murah, M. Z., Hasan, M. K., Aman, A. H. M., Fang, J., Hu, X., & Khan, A. U. R. (2024). A survey of deep learning technologies for intrusion detection in Internet of Things. IEEe Access, 12, 4745-4761.

[37]Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150.

[38]Serardi, S. (2023). Vers un système de détection d'intrusion dans l'Internet des Objets (Doctoral dissertation, Université Ibn Khaldoun).

[39]Azam, Z., Islam, M. M., & Huda, M. N. (2023). Comparative analysis of intrusion detection systems and machine learning-based model analysis through decision tree. IEEE Access, 11, 80348-80391.

[40]Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of big Data, 8, 1-74.

[1] Arsalan, M., Mubeen, M., Bilal, M., & Abbasi, S. F. (2024, August). 1D-CNN-IDS: 1D CNN-based intrusion detection system for IIoT. In 2024 29th International Conference on Automation and Computing (ICAC) (pp. 1-4). IEEE.

[2] Sinha, P., Sahu, D., Prakash, S., Yang, T., Rathore, R. S., & Pandey, V. K. (2025). A high performance hybrid LSTM CNN secure architecture for IoT environments using deep learning. Scientific Reports, 15(1), 9684.

[3] Okey, O. D., Melgarejo, D. C., Saadi, M., Rosa, R. L., Kleinschmidt, J. H., & Rodríguez, D. Z. (2023). Transfer learning approach to IDS on cloud IoT devices using optimized CNN.

IEEE Access, 11, 1023-1038.

[4] Heng, P. Y., & Yusoff, Y. (2025). Intrusion Detection System using Convolutional Neural Network for Industrial Internet of Things Security. International Journal of Innovative Computing, 15(1), 95-107.

[5] Kilichev, D., Turimov, D., & Kim, W. (2024). Next–generation intrusion detection for iot evcs: Integrating cnn, lstm, and gru models. Mathematics, 12(4), 571.

[6] Deshmukh, A., & Ravulakollu, K. (2024). An efficient CNN-based intrusion detection system for IoT: Use case towards cybersecurity. Technologies, 12(10), 203.