وزارة التعليم العالي والبحث العلمي

Centre Universitaire
Abdelhafid Boussouf - Mila

Abdelhafid Boussouf
University Center of Mila

المركز الجامعي

عبد الحفيظ بوالصوف – ميلة –

Institut Mathématique et Informatique

Département : Informatique

Année 2024

Thèse

Présentée

En vue de l'obtention du Diplôme de Master

Le Deep Learning pour un système de détection d'intrusions

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Sciences et Technologies de l'Information et de la Communication (STIC)

Par :

Amoura Aicha

Djimli Zahra

Devant le Jury

Président :	SELMANE Samir	C.U. Mila
Rapporteur :	MEGUEHOUT Hamza	C.U. Mila
Examinateur :	BENCHEIKH LEHOCINE Madjed	C.U. Mila

Certes, la science guide, dirige et sauve;
l'ignorance égare, trompe et ruine

Imâm Ali ibn Abi Talib

Dédicace

Je dédie ce travail.

À mes chers parents,

À mes chères sœurs et mon cher frère,

À toute la famille AMOURA,

À tous mes cher(e)s ami(e)s.

À tous ceux qui m'ont aidé de près ou de loin.

AMOURA Aicha

Dédicace

Je dédie ce travail.

À ma chère mère,

À mes chères sœurs et frères,

À toute la famille DJIMLI,

À tous mes cher(e)s ami(e)s.

À tous ceux qui m'ont aidé de près ou de loin.

DJIMLI Zahra

Remerciement

Tout d’abord, nous remercions Allah le Tout-Puissant de nous avoir accordé la volonté, le courage et la patience pour mener à bien ce travail.

Nous exprimons également notre sincère gratitude envers notre directeur de thèse, pour son suivi, ses conseils judicieux et pour le temps précieux qu’il nous a accordé tout au long de ce travail.

Nous tenons également à remercier les membres du jury pour l'honneur qu'ils nous ont fait-en examinant et évaluant notre modeste travail.

Enfin, nous souhaitons exprimer notre reconnaissance à tous ceux qui ont contribué, de près ou de loin, à la réalisation de ce travail, en particulier notre famille.

Résumé

Le domaine de l'Internet des objets est très prometteur, car chaque jour, de nombreux objets se connectent au réseau. Par conséquent, les risques d'intrusion augmentent et la détection d'intrusion devient un sujet d'actualité majeur.

Dans ce travail, nous avons exploité la puissance du Deep Learning pour la détection d'intrusions, surtout que cette dernière dispose de nombreuses datasets. Nous avons testé nos modèles sur le dataset CICIoMT2024, sur lequel ils n'avaient encore jamais été expérimentés. La puissance des modèles du Deep Learning, associée à la grande quantité de données huit millions d'instances, nous a contraints à utiliser des ressources puissantes de NVIDIA. Ce travail a permis de valider certains résultats et d'exploiter de nouvelles pistes de recherche.

Mots-clés : CICIoMT2024, Deep Learning, Détection d'intrusion, Internet des objets, Réseau de neurones convolutif, Réseaux de neurones récurrents.

Abstract

The field of the Internet of Things is very promising, as every day; many objects connect to the network. As a result, the risks of intrusion increase, making intrusion detection a major current issue. In this work, we have harnessed the power of Deep Learning for intrusion detection, especially given the availability of many datasets. We tested our models on the CICIoMT2024 dataset, which they had not been experimented before. The power of deep learning models coupled with the large amount of data —over height million instances— required us to use powerful NVIDIA resources. This work has validated certain results and opened new avenues of research.

Keywords: CICIoMT2024, Convolutional Neural Network, Deep Learning, Intrusion Detection, Internet of Things, Recurrent Neural Networks.

الملخص

إن مجال إنترنت الأشياء واعد للغاية، حيث يتم تتصل العديد من الأجهزة بالشبكة كل يوم. ونتيجة لذلك، تزداد مخاطر التسلل ويصبح اكتشاف التسلل مشكلة رئيسية هامة. في هذا العمل، استغللنا قوة التعلم العميق لاكتشاف التسلل، خاصة وأن هذا المجال يتوفر على العديد من مجموعات البيانات. اختبرنا نماذجنا على مجموعة بيانات CICIoMT2024، التي لم يتم اختبارها من قبل. إن قوة نماذج التعلم العميق، إلى جانب الكم الكبير من بيانات_ حوالي أكثر من ثمانية ملايين عينة، أجبرتنا على استخدام موارد NVIDIA القوية. وقد أتاح هذا العمل التحقق من صحة بعض النتائج واستكشاف سبل جديدة للبحث.

الكلمات المفتاحية: CICIoMT2024، التعلم العميق، كشف التطفل، إنترنت الأشياء، شبكة عصبونية التفافية، الشبكات العصبية المتكررة.

Abréviation

Abréviation	Libellé
AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DL	Deep Learning
DNN	Deep Neural Network
GAN	Generative Adversarial Network
HIDS	Host Intrusion Detection Systems
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IoHT	Internet of Health Things
IoMT	Internet of Medical Things
IoT	Internet of Things
KNN	K-Nearest Neighbour
LR	Logistic Regression
ML	Machine Learning
NIDS	Network Intrusion Detection Systems
RNN	Recurrent Neural Network
SVM	Support Vector Machine

Table des matières

Liste des figures

Liste des tableaux

Introduction Général

Introduction Général

À l'heure actuelle, Internet est de plus en plus utilisé, cette dépendance étant assurée par une connectivité quasi permanente à Internet. La connectivité est maintenue par des objets de la vie quotidienne tels que les Smartphones, les montres intelligentes, les ordinateurs, etc. Cependant, cette connectivité d'objets dépasse les tâches ordinaires du quotidien pour inclure des services publics ou privés. Vu l'extension de la connectivité et la diversité des objets connectés, le concept d'Internet des objets (Internet of Things) a émergé.

Le domaine de la santé est l'un des secteurs connectés à Internet via de nombreux objets, et étant donné son importance, un sous-axe a émergé : l'Internet des objets médicaux (Internet of Medical Things).

Dans ce travail, nous nous intéressons à l'utilisation du Deep Learning dans les systèmes de détection d'intrusion. Nous utilisons principalement un réseau de neurones convolutif, puis un réseau de neurones récurrents. Les expérimentations des deux modèles sont effectuées sur le dataset CICIoMT2024, réalisé par le célèbre centre canadien du cyber sécurité.

Nous avons constaté que Colab offre gratuitement 12 Go de RAM, 15 Go d'espace sur le Drive et l'accès à un GPU. Des ressources supplémentaires sont disponibles via des abonnements payants. Pour répondre à nos besoins de performance, nous avons souscrit à l'une des offres payantes de Google Colaboratory, augmentant notre capacité de traitement par GPU NVDIA A100. Cela nous a permis d'expérimenter notre modèle sur un nombre d'époques plus grand.

Nous débutons par un chapitre d'introduction au domaine de la détection d'intrusion et de l'Internet des objets. Ensuite, nous abordons l'apprentissage automatique et ses principales sous-catégories. Enfin, un dernier chapitre décrira notre contribution et les expérimentations réalisées. Nous terminons par une conclusion générale et les perspectives.

Chapitre 1
Systèmes de Détection d'Intrusion

Chapitre 1

Systèmes de Détection d'Intrusion

1- Introduction

Dans ce premier chapitre, il est nécessaire que nous abordions les systèmes de détection d'intrusion. Notre objectif n'est pas de nous approfondir dans la détection d'intrusion, en particulier dans le domaine du réseau, car ce n'est pas le principal sujet de notre étude. Il s'agit plutôt de donner au lecteur une idée claire et précise pour lui permettre de nous suivre lors des prochains chapitres.

Nous débuterons en définissant les systèmes de détection d'intrusion, leurs approches, familles et l'architecture de base. Ensuite, nous présentons de la détection d'intrusion dans le domaine médical, qui est le sujet de notre étude.

2- Systèmes de détection d'intrusion

Un système de détection d'intrusion (IDS) est une solution de sécurité réseau qui analyse le trafic et les dispositifs connectés afin de repérer les activités malveillantes, les comportements suspects ou les infractions aux politiques de sécurité [1]

Les professionnels de la cybersécurité sont alertés des violations par les systèmes de détection d'intrusion (SDI), qui inspectent les systèmes informatiques à la recherche de signes de non-conformité aux normes de sécurité [2].

Dans le contexte des réseaux, une intrusion est toute violation, qu'elle provienne de l'intérieur ou de l'extérieur du réseau [3]. Ainsi, toute violation peut être détectée et signalée par un système de détection d'intrusion (SDI) [4], améliorant ainsi le travail des usagers du réseau [5, 6].

Pour comprendre la détection d'intrusion, nous explorons ses stratégies et architectures. Ces systèmes protègent les réseaux et les systèmes informatiques des menaces. Nous examinons d'abord les approches principales de détection d'intrusion, puis les diverses familles basées sur l'architecture. Cette analyse offre une compréhension approfondie des méthodes sécurisant les environnements informatiques.

3- Approches de détection d'intrusion

En général, il existe quatre types d'approches pour détecter les intrusions dans les systèmes informatiques :

3.1- Basé sur signature (Signature Based Detection)

Ce concept ou cette famille de systèmes compare les signatures d'intrusion avec celles préalablement enregistrées dans la base de données lorsqu'une tentative d'attaque survient. En cas de correspondance, une alerte est déclenchée.

3.2- Basé sur les anomalies (Anomaly Base Detection)

La principale qualité des systèmes de détection d'anomalies réside dans leur aptitude à repérer les attaques inédites. Pour traiter la détection d'anomalies, différentes méthodes ont été considérées : la modélisation statistique, les réseaux de neurones, etc.

Ainsi, l'identification des anomalies implique de distinguer les événements probablement irréguliers en comparaison avec le fonctionnement normal du système[7].

3.3- Basé sur les spécifications (Spectification-Based Detection)

Ce type d'approche donne une alerte d'une mauvaise activité, quand il aperçoit une male utilisation du protocole, car il a des connaissances préalables des caractéristiques du protocole [8].

3.4- Hybrides

D'après [8], la première méthode présente un taux élevé de faux positifs (fausses alertes), tandis que la deuxième méthode est incapable de détecter de nouvelles intrusions qui ne sont pas répertoriées dans la base de données. Quant à la dernière méthode, elle ne parvient pas à identifier les attaques qui semblent être des utilisations légitimes du protocole.

En tenant compte des précédentes causes, les systèmes de détection d'intrusion hybrides sont utiles pour avoir des performances élevées.

4- Familles des systèmes de détection d'intrusion basées sur l'architecture

On a principalement trois catégories si l'on veut classer les systèmes de détection d'intrusion selon les données d'entrée et les sources de collecte.

4.1- Systèmes de détection d'intrusion basés sur l'hôte (Host Based – HIDS)

Ils interagissent avec le système d'exploitation et n'ont pas de visibilité sur le mouvement bas niveau du réseau. Dans la majorité des cas, ces systèmes s'appuient sur des données provenant des fichiers log et des audits du système. En termes simples, il s'agit généralement de logiciels qui surveillent uniquement les activités se déroulant localement sur le système [7].

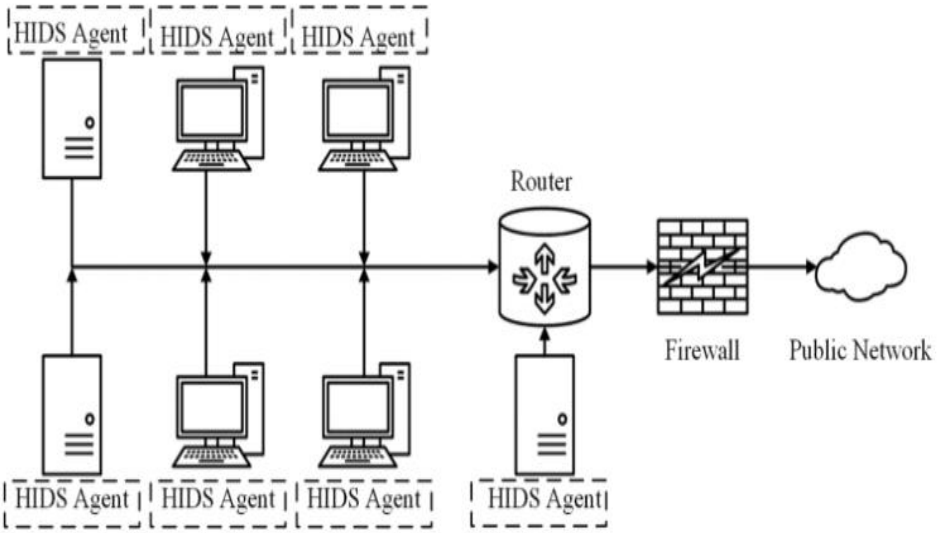

Figure 1.1 : Exemple d'un HIDS [9]

4.2- Systèmes d'intrusion détection basés sur les réseaux (Network IDS – NIDS)

Son but est d'observer les données qui passent à travers une liaison réseau spécifique. Parmi ses avantages, il est imperceptible aux attaquants et insensible à certaines attaques comme le "ping-of-death" [7].

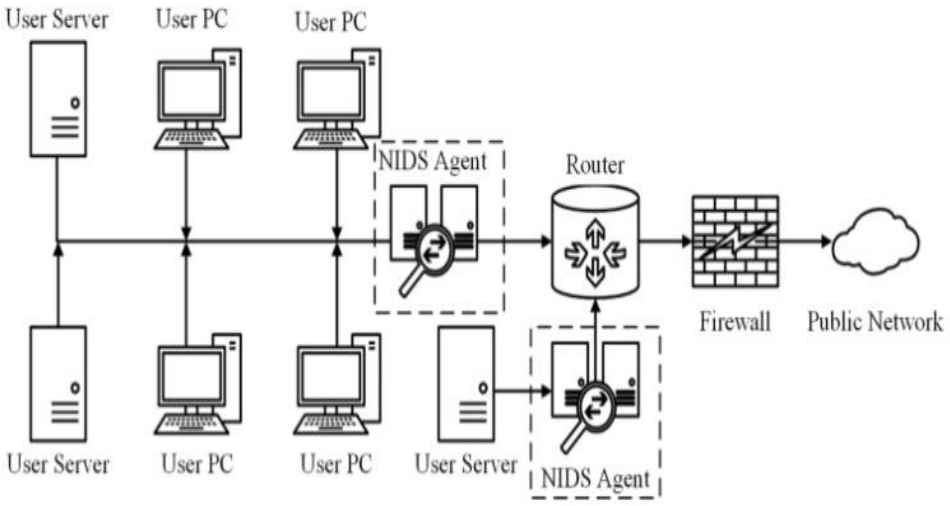

Figure 1.2 : Exemple d'un NIDS [9]

4.3- Systèmes de détection hybrides

Selon un article [10], l'actualité dans le domaine de la détection d'intrusion réside dans l'adoption de systèmes hybrides, combinant les fonctionnalités des HIDS et des NIDS. Ces systèmes hybrides offrent une certaine souplesse et considérés comme une bonne amélioration de la sécurité en comparaison aux deux autres solutions individuellement.

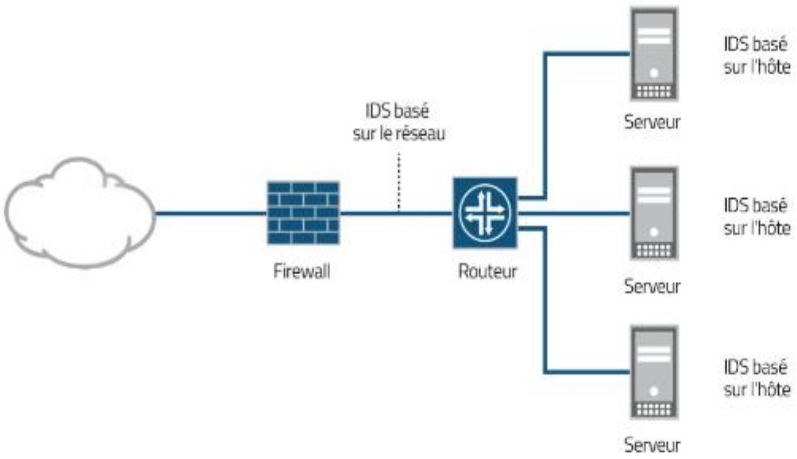

Figure 1.3 : Exemple de Systèmes de détection hybrides [11]

Tableau 1.1 : Comparaison entre les types des systèmes de détection d'intrusion [12]

	Avantage	Désavantages
HIDS	▪ Détecte les intrusions en surveillant les fichiers, les appels système et les événements, sans nécessité d'équipement supplémentaire.	▪ Besoin de l'installer sur chaque machine ▪ Détection d'attaques locales uniquement.
NIDS	▪ Détecte les intrusions en surveillant le trafic réseau. ▪ Besoin d'être placé sur le réseau (physiquement) ▪ Peut surveiller plusieurs systèmes en même temps.	▪ Difficile de détecter des intrusions provenant de contenu chiffré.

5- L'architecture de base d'un système de détection d'intrusion

Dans [6] l'auteure mentionne que le groupe "Intrusion Détections Working" de l'organisme d'élaboration de normes "Internet Engineering Task Force (IETF)[1]" a met un modèle standardisé d'un système de détection d'intrusion (Figure 1.4).

L'illustration de la figure 1.4 est extraite à partir d'une ressource originale d'IETF [13]. Elle représente divers termes et les relations qui les lient. Il est mentionné que ces termes seront combinés et décomposés en plusieurs instances, ou distingués en fonction du système de détection d'intrusion utilisé.

[1] www.ietf.org

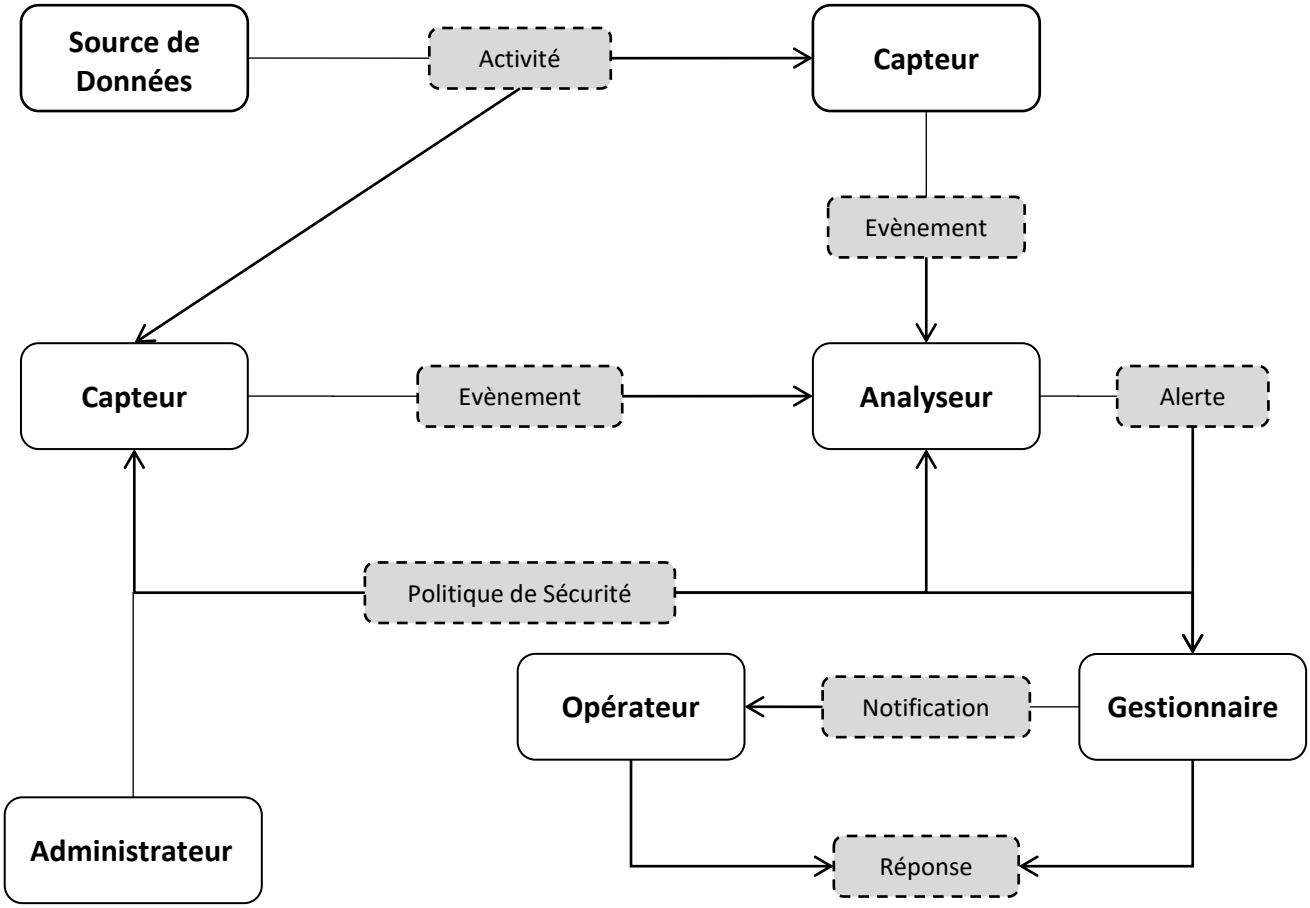

Figure 1.4: Modèle de base de la détection d'intrusions proposé par l'IDWG [13]

6- Détection d'intrusion et l'internet des objets médicaux

6.1- Internet des objets (Internet of Things – IoT)

D'une façon générale, le terme "Internet des objets" regroupe tous les objets connectés au réseau Internet. Avec l'arrivée de la 5G et la possibilité de se connecter à Internet presque partout, de nombreuses technologies sont désormais interconnectées, selon les experts en 2025 on peut arriver à 22 milliards d'objets. Cela, dans le but de transmettre des informations et d'automatiser des services[14, 15]. Selon notre constat, des objets sont connectés à Internet depuis longtemps. Cependant, l'Internet des objets a été mis en valeur après l'interconnexion d'une grande masse d'objets à Internet.

Donc l'IoT regroupe plusieurs objets connectés tels que : voitures, capteurs, etc., cette connectivité de milliards d'objets augmente en parallèle avec les avancées technologiques [16].

Le secteur de la santé est l'un des secteurs les plus encourageants pour l'usage des outils sous la catégorie d'internet des objets médicaux [17]

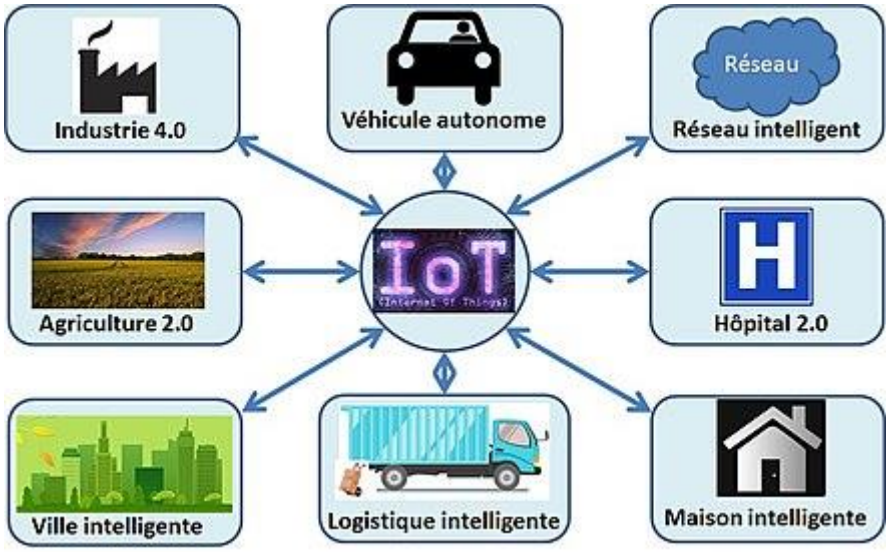

Figure 1.5 : Applications d'IoT dans la vie quotidienne [18]

6.2- Internet des objets médicaux

Parmi les sous-catégories de l'IoT, nous trouvons l'internet des objets médicaux (Internet of Medical Things – IoMT). L'intérêt de ces objets (équipements) est d'interconnecter les patients et les professionnels médicaux, via un réseau [19]. Les données collectées et transmîtés contribuent au suivi en temps réel, la réalisation d'un diagnostic approfondi et la prévention [15].

Dans [20], les auteurs citent plusieurs fonctionnalités d'objets IoMT, telles que le suivi des impulsions, la glycémie, la pression artérielle, les signaux physiologiques et d'autres fonctionnalités selon les technologies disponibles et les nécessités médicales. Selon ces mêmes auteurs, étant donné l'importance du domaine médical et les risques pour la vie des patients, la protection des informations des objets IoMT est primordiale. Car, comme tout autre environnement l'IoT contient des menaces de sécurité [21].

Figure 1.6 : Utilisations de l'Internet des objets dans le domaine médical [22]

6.3- Détection d'intrusion dans IoMT

Dans un article [17], un panorama des travaux sur l'IoMT est présenté. Nous avons consulté certains de ces ressources pour avoir une idée des travaux connexes à notre recherche.

Vu le grand nombre de données et les exigences de confidentialité dans le domaine médical, pour les auteures dans [23] l'apprentissage automatique constitue une réelle solution pour cette catégorie des systèmes IoMT. Les données de flux réseau et biométriques sont réunies et des méthodes d'apprentissage automatique sont utilisées dans les phases d'entrainement et validation.

Dans un travail basé sur l'apprentissage profond ainsi qu'une première dataset de Bluetooth classique et Bluetooth à basse consommation (Bluetooth Low Energy – BLE) [20].

Dans [24] les auteures présentent un système de détection et prévention des intrusions basé sur un apprentissage par renforcement. Ce système est dans le domaine médical, car selon les autres[2], il se base sur un protocole IEC 60870-5-104 vastement utilisé dans les systèmes de santé.

[2] Ils citent un article survey (https://doi.org/10.48550/arXiv.2102.05631)

Dans une autre étude [25], les données IoT dans le domaine de la santé sont produites par un outil open source nommé IoTFlock. Cet outil permet de générer du trafic réseau normal et des attaques. Ensuite, des approches d'intelligence artificielle sont employées pour l'apprentissage et les tests des systèmes IDS.

Dans l'article [26], les auteurs créent un dataset qui regroupe différentes failles de sécurité dans le domaine de l'Internet of Health Things (IoHT). Le principal objectif de ce travail est d'aider les responsables de la sécurité dans le secteur de la santé à prendre les précautions nécessaires. De plus, ils incitent les chercheurs en IA à contribuer à l'amélioration des systèmes de détection d'intrusion dans ce domaine.

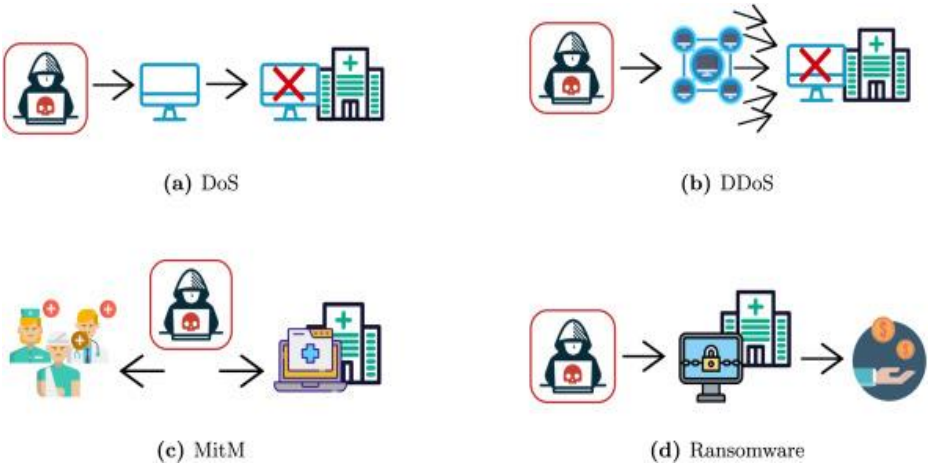

Figure 1.7 : Exemples de menaces à la sécurité de l'IoMT [27]

7- Les mesures d'évaluation d'IDS

Il existe plusieurs mesures d'efficacité pour les systèmes de détection d'intrusion [12] :

- Précision : Un IDS est considéré comme précis s'il parvient à détecter sans générer de faux positifs.
- Puissance : il s'agit de la rapidité du traitement des informations. À mesure que le système parvient à une détection en temps réel, il est considéré comme de plus en plus performant.

- Complétude : si le système à la possibilité d'identifié l'ensemble des attaques.
- Tolérance aux pannes : le système a l'obligation de surmonter des attaques de type déni de service, car principalement l'IDS est implémenté sur des outils prédisposés aux attaques.

8- Conclusion

Dans ce chapitre, nous avons présenté un aperçu de la détection d'intrusion et de l'Internet des objets. Nous avons également examiné l'Internet des objets médicaux. Dans le prochain chapitre, nous aborderons l'apprentissage profond.

Chapitre 2
Apprentissage Profond

Chapitre2

Apprentissage Profond

1- Introduction

Notre travail est basé sur l'une des méthodes de l'apprentissage automatique. Ainsi, ce chapitre sera consacré à l'apprentissage automatique, qui est l'un des plus grands et populaires axes de l'intelligence artificielle en raison de sa vaste utilisation dans de nombreuses disciplines : médecine, automobile, économie, etc.

Nous commençons par une introduction générale sur l'apprentissage automatique. Par la suite, nous donnerons un panorama des méthodes classiques, suivi d'une exploration sur la nouvelle sous-discipline de l'apprentissage automatique, le Deep Learning. Nous terminerons par l'utilisation de ce dernier dans la détection d'intrusion et enfin, une conclusion.

2- L'apprentissage automatique

Parmi les sous-catégories de l'intelligence artificielle le Machine Learning, appelé en français l'apprentissage automatique (au cours de ce chapitre et des chapitres suivants, nous préférerons utiliser les appellations anglaises).

Son principal avantage est de nous d'aider à comprendre les données (nombres, textes, etc.) de manière précise et rapide. Il repose sur l'entrainement de la machine pour apprendre, prédire ou identifier des motifs, tout cela sans programmation détaillée et spécifique [28].

2.1- Types d'apprentissage

L'apprentissage automatique est divisé en 3 types :

- **L'apprentissage supervisé :** Cette méthode d'apprentissage utilise des données (Dataset) annotées, incluant un ensemble de cas et leur résultat respectif. On considère qu'un modèle d'apprentissage est efficace lorsque la différence entre ses prédictions et les résultats des cas de test est minimale.[29]

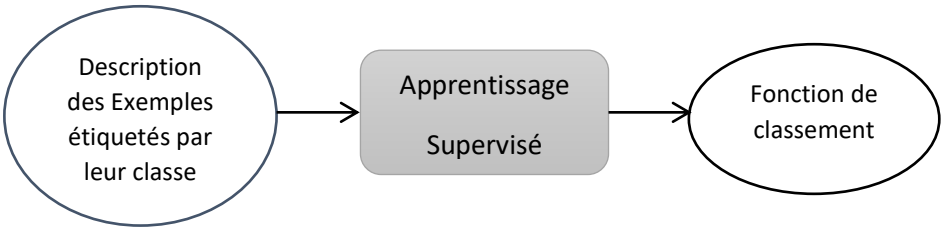

Figure 2.1 :L'apprentissage supervisé. [30]

- **L'apprentissage non supervisé :** La principale différence entre cette catégorie d'apprentissage et la précédente réside dans l'absence de données annotées. Ainsi, les algorithmes de cette famille cherchent à comprendre les données et à proposer une structuration [31].
- **Apprentissage par renforcement :** Il s'agit d'une autre famille de méthodes, peu exploitée, mais qui commence à attirer l'attention des chercheurs. Cette approche simule la capacité humaine à apprendre à partir des expériences et des erreurs, en renforçant les actions qui permettent d'atteindre un objectif donné [32].

3- Les approches classiques de l'apprentissage automatique

3.1- Les réseaux neuronaux (Neural networks)

La prise de décision dans ce type d'approches, d'algorithmes ou de modèles s'inspire du fonctionnement des neurones biologiques de notre cerveau, afin de résoudre des problèmes d'identification, d'évaluation, etc. [33].

3.2- Machines à vecteurs de support (Support Vector Machine)

Ce type d'approche a émergé dans les années 90, fondé sur la théorie statistique de Vapnik-Chervonenkis. Appartenant à la famille des approches supervisées, elle se concentre sur les problèmes de discrimination et de régression. Son succès s'explique par sa simplicité, son faible nombre de paramètres, sa capacité à gérer une grande quantité de données et ses bons résultats [34].

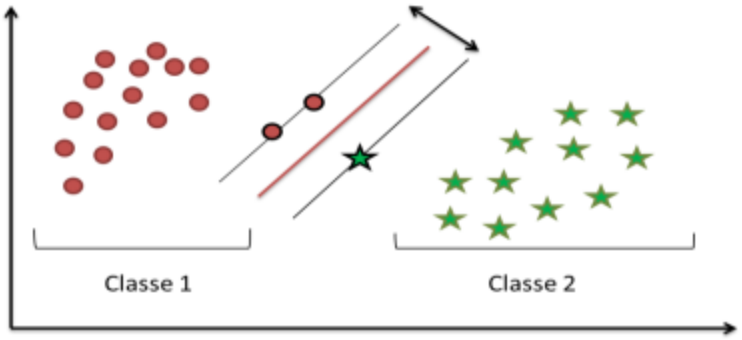

Figure 2.2 : Exemple d'une classification SVM [35]

3.3- La régression logistique

Ce modèle statistique linéaire vise à identifier les relations entre un ensemble de variables qualitatives X et Y. En optimisant ses coefficients, il permet d'estimer la probabilité qu'un événement se produise (en dépassant un seuil) dans un intervalle allant de 0 à 1 [36].

Ce champ d'étude des méthodes classiques est largement couvert et expliqué dans la littérature. Il existe également de nombreuses autres approches, telles que les K plus proches voisins (Figure 2.3) [37] et les arbres de décision (Figure 2.4) [38], etc. Par conséquent, nous ne nous attarderons pas sur l'explication de ces méthodes.

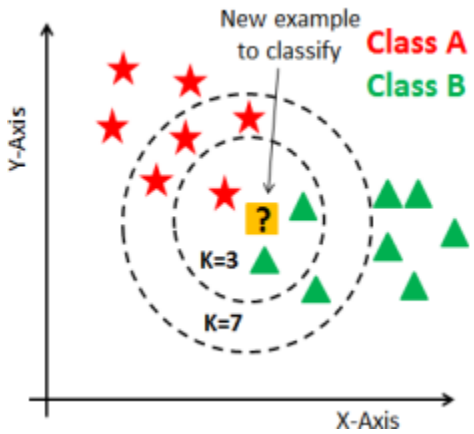

Figure 2.3 : Exemple de K-plus proche voisins (KPPV) [35]

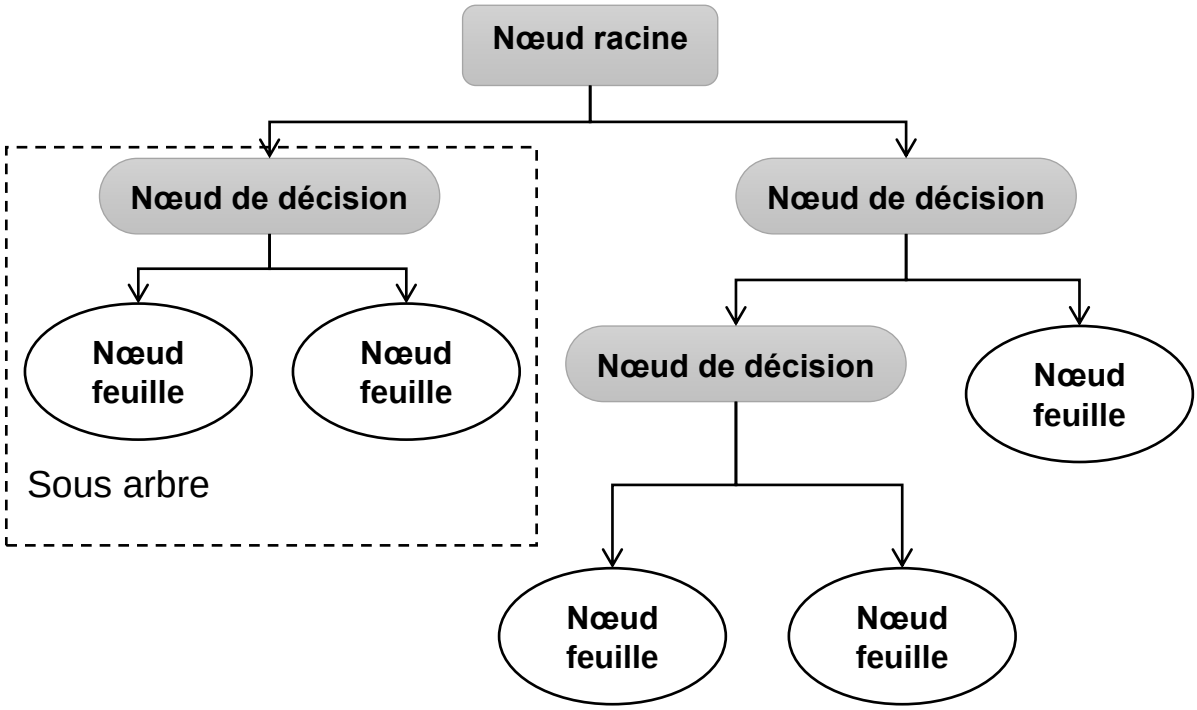

Figure 2.4 : Structure d'un arbre de décision [35]

4- L'apprentissage profond (Deep Learning - DL)

Dans le titre précédent, nous avons discuté des approches classiques, y compris les réseaux de neurones basés sur les neurones biologiques. De même, l'apprentissage profond, souvent désigné par son appellation anglaise Deep Learning est une famille d'approche d'apprentissage automatique inspirée par le fonctionnement de notre cerveau et les réseaux de neurones. Il est néanmoins classé comme un domaine à part entière dans l'apprentissage automatique (Figure 2.5)[39].

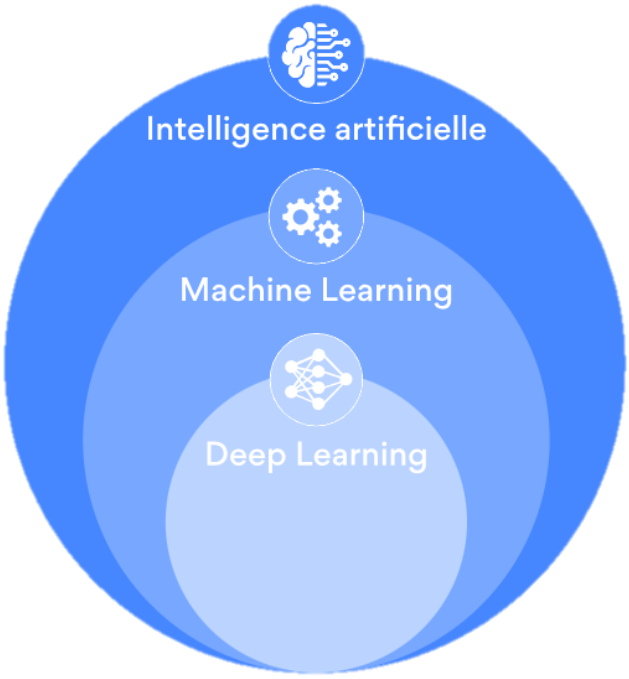

Figure 2.5 : Hérarchie entre l'IA, la Machine Learning et le Deep Learning [39]

Les modèles du Deep Learning sont plus récents par rapport aux approches classiques. Le traitement des données s'effectue de manière hiérarchique, passant d'une couche à l'autre, où chaque couche intermédiaire reçoit les informations de la couche précédente et les transmet à la couche suivante. Leur succès repose sur leur efficacité dans le traitement des données et leur capacité à traiter des domaines complexes tels que le traitement d'images, le diagnostic médical, la traduction automatique, etc.[8, 39].

5- Principe et fonctionnement du Deep Learning

La figure 2.6 offre un aperçu standard, compréhensible et simple de l'architecture globale du Deep Learning. Comme mentionné auparavant, le Deep Learning est essentiellement un réseau de neurones. La figure montre qu'il comprend trois types de couches de neurones (nœuds) : une couche d'entrée (verte), des couches cachées (bleues) et une couche de sortie (orange). Les couches adjacentes sont reliées par des connexions pondérées (lignes noires) [8].

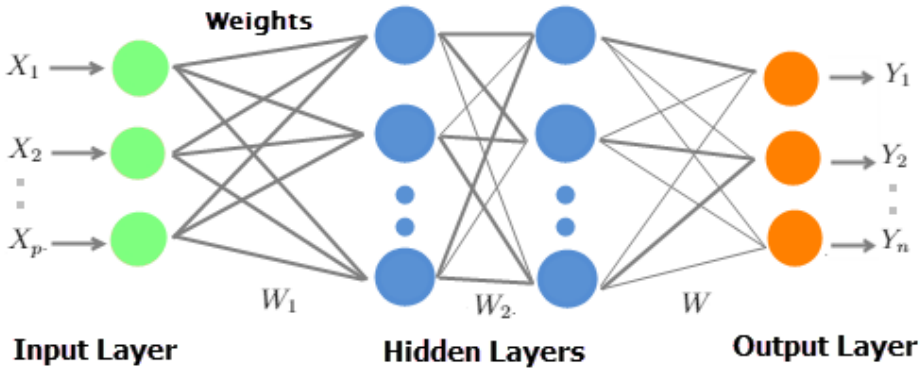

Figure 2.6 : L'architecture d'un modèle Deep Learning. [8]

6- Méthodes du Deep Learning

Cet axe d'apprentissage automatique inclut plusieurs méthodes, chacune pouvant comporter plusieurs architectures ou modèles. Dans cette section, nous présentons un aperçu des principales méthodes du Deep Learning.

6.1- Réseau de neurones profonds (Deep Neural Network – DNN)

Il s'agit de la description générale du Deep Learning ou nous le considérons comme la méthode classique du DL qui se distingue par son architecture comportant au moins deux couches, contrairement à un réseau neuronal simple. Son principal objectif est de découvrir des relations entre les inputs et les outputs, via des formules mathématiques simples et complexes [40] .

6.2- Réseau de neurones à convolution (Convolutional Neural Network – CNN)

Comme toutes les architectures du Deep Learning, le CNN comprend une couche d'entrée, une couche de sortie et plus qu'une couche intermédiaire [41]. Ses principales couches sont :

- ✓ La couche de convolution.
- ✓ La couche de pooling.
- ✓ La couche entièrement connectée.

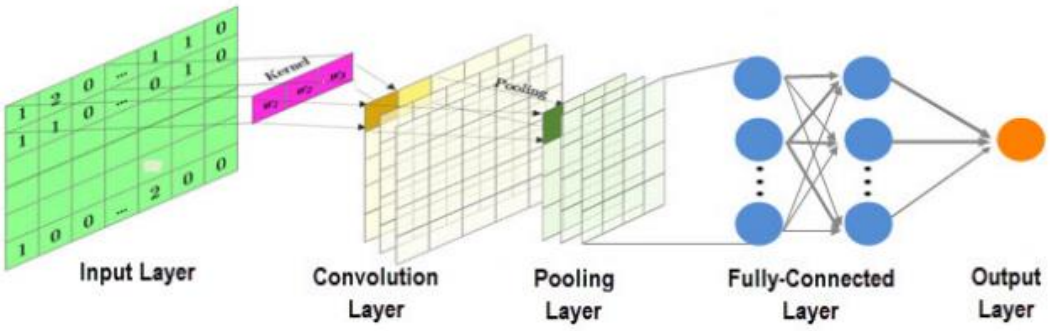

Figure 2.7 : Architecture de CNN [8]

6.3- Réseaux Neuronaux Récurrents (Recurrent Neural Network – RNN)

Ce type est l'un des modèles est l'un des plus important dans le domaine du DL. Il a la capacité d'influencer la sortie en se basant sur des prédictions antérieures grâce à la mémorisation des opérations précédentes. Parmi ses utilisations les plus célèbres, on peut citer la recherche vocale de Google et Siri[3]. Il est également souvent utilisé dans des taches de traitement du langage naturel (ex. traduction automatique, génération de texte, etc.)[42, 43]. La figure 2.8 illustre l'architecture générale d'un modèle RNN.

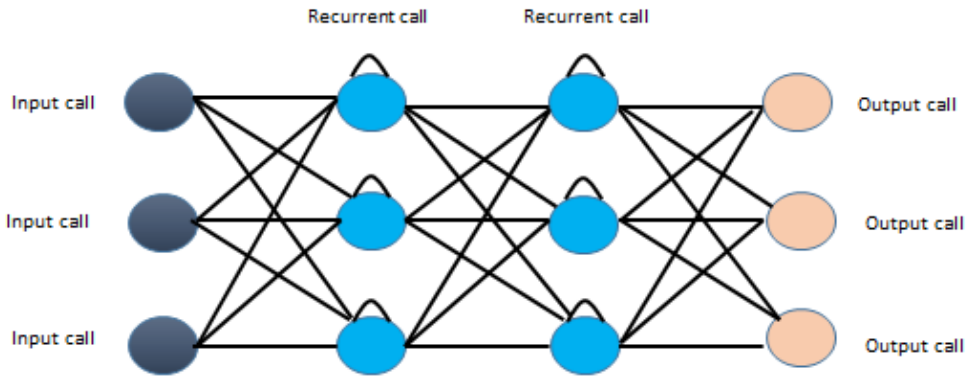

Figure 2.8 : L'architecture de modèle RNN [44]

Nous avons montré les principales grandes familles du Deep Learning, mais il existe plein d'autres modèles.

[3] Un assistant intelligent d'Appel

7- Deep Learning dans la détection d'intrusion

Divers travaux se sont penchés sur la détection d'intrusions basée sur le deep learning.

D'après [8], l'utilisation du Deep Learning dans la détection d'intrusion a commencé en 2011 avec le travail de [45]. Ils ont combiné le Deep Learning et les SVM pour une classification binaire et à cinq classes. Cependant, le Deep Learning est utilisé pour la réduction des caractéristiques.

Le tableau 2.1 présente une sélection d'études portant sur la détection de l'intrusion basée sur des modèles DL, les ensembles de données et les mesures de performance.

Tableau 2.1 : Travaux antérieurs connexes pour la détection d'intrusion basé sur le deep learning [8]

Méthode	Dataset	Mesures de performances	Année	Auteurs
DBN-SVM	NSL-KDD	ACC = 90%	2011	salama et al.
DNN-DBN	Vehicular Network communication	DR = 99% - 99.46% FPR = 1-2% ROC = _	2016	kang et al.
DBN	NSL-KDD	ACC = 97.5%	2015	Alom et al.
CNN	NSL-KDD	ACC = 97.88% - 99.46% DR = 68.66% FPR = 27.9%	2018	Wu et al.
RNN	DARPA	DR = 95.37% FPR = 2.1%	2015	Kim et al.
RNN-LSTM	CTU-13	DR = 97.96% FPR = 2.27%	2016	Torres et al.
CNN-RNN	KDD'99	ACC = 99.99% PR = 99.99% RC = 99.99%	2017	vinayak et al.

		F1 = 99.99%		
Deep auto-encoder	NSL-KDD	ACC = 99.99% PR = 84.6% RC = 92.8% specificity =80.7%	2018	Sandee et al.

Un article Survey [44] de référence cités plus 900 fois donne un bon panorama de l'utilisation du Deep Learning pour la détection d'intrusion.

8- Conclusion

Dans ce second chapitre, nous avons défini l'apprentissage automatique ainsi que ses sous-domaines. Pour conclure, nous avons exploré l'utilisation de l'apprentissage automatique dans le domaine de la détection d'intrusion. Le prochain chapitre détaillera notre contribution à l'exploitation de la puissance du Deep Learning dans les systèmes de détection d'intrusion.

Chapitre 3
Conception et Réalisation

Chapitre3

Conception et Réalisation

1- Introduction

Dans ce troisième chapitre, nous présentons plus en détail le dataset utilisé. Ensuite, nous montrons l'architecture de notre modèle, les expériences réalisées, et nous concluons par une discussion.

2- Environnement du travail

Dans cette première partie, nous présentons les outils utilisés, en commençant par le langage de programmation Python.

2.1- Python

Nous avons choisi Python en raison de sa large adoption dans le domaine de l'apprentissage automatique. Selon le classement TIOBE[4], Python est en tête des langages de programmation depuis l'année 2022 (Figure 3.1). De plus, le classement IEEE de 2023 le place également au premier rang (Figure 3.2).

Jun 2024	Jun 2023	Change	Programming Language	Ratings	Change
1	1		Python	15.39%	+2.93%
2	3	↑	C++	10.03%	-1.33%
3	2	↓	C	9.23%	-3.14%
4	4		Java	8.40%	-2.88%

Figure 3.1 : Classement TIOBE du Juin 2024 des langages de programmation [46]

Figure 3.2 : Classement IEEE des langages de programmation[47]

[4]https://www.tiobe.com/about-us/

2.2- Google Colaboratory

Notre travail sur le Deep Learning et le large dataset utilisé, nous ont obligés à utiliser une machine d'une grande capacité de traitement. Ainsi, nous avons utilisé Google Colaboratory[5].

Connu sous le nom de Colab, cet outil gratuit nécessite un compte Google et permet d'utiliser Jupyter Notebook pour l'écriture et l'exécution de code Python. Il est équipé de nombreuses bibliothèques pertinentes, notamment Keras, TensorFlow, PyTorch, etc. [48]

Nous avons constaté que Colab offre gratuitement 12 Go de RAM, 15 Go d'espace sur le Drive et l'accès à un GPU. Des ressources supplémentaires sont disponibles via des abonnements payants.

Pour répondre à nos besoins de performance, nous avons souscrit à l'une des offres payantes de Google Colaboratory, augmentant notre capacité de traitement par **GPU NVIDIA A100**. Cela nous a permis d'expérimenter notre modèle sur un nombre d'époques plus grand.

3- CICIoMT 2024 dataset

Nous avons choisi comme dataset le CIC-IoMT2024, réalisé par le célèbre institut canadien de cybersécurité (CIC)[6]. Il contient 18 types d'attaques différentes exécutées sur 40 objets du domaine de la santé [17].

Le tableau 3.1 regroupe les features utilisées dans le processus d'apprentissage et d'évaluation du Deep Learning.

[5] https://colab.research.google.com/

[6] https://www.unb.ca/cic/datasets/iomt-dataset-2024.html

Tableau 3.1 : La liste de features du dataset

#	Nom de features	#	Nom de features
1	Header Length	**24**	IRC
2	Duration	**25**	TCP
3	Rate	**26**	UDP
4	Srate	**27**	DHCP
5	Drate	**28**	ARP
6	Fin flag number	**29**	ICMP
7	Syn flag number	**30**	IPv
8	Rst flag number	**31**	LLC
9	Psh flag number	**32**	Tot sum
10	Ask flag number	**33**	Min
11	Ece flag number	**34**	Max
12	Cwr flag number	**35**	AVG
13	Syn count	**36**	Std
14	Ack count	**37**	Tot size
15	Fin count	**38**	IAT
16	Rst count	**39**	Number
17	IGMP	**40**	Radius
18	HTTPS	**41**	Magnitude
19	HTTP	**42**	Variance
20	Telnet	**43**	Covariance
21	DNS	**44**	Weight
22	SMTP	**45**	Protocol Type
23	SSH		

Le tableau 3.2 tiré [17] montre plusieurs informations sur les features.

Tableau 3.2: Description de chaque features

	Mean	Std	Min	25%	50%	75%	Max
Header_Length	29962.4717	282363.394	0	2.17	108	19421	9896704
Protocol Type	8.047203	6.304832	0	1.05	6	17	17
Duration	64.63691	7.853066	0	64	64	64	255
Rate	15744.49	40008.55	0	6.422730	133.1418	19759.20	2097152
Srate	15744.49	40008.55	0	6.422730	133.1418	19759.20	2097152
Drate	0	0	0	0	0	0	0
fin_flag_number	0.05123298	0.03415862	0	0	0	0	1
syn_flag_number	0.1572115	0.3368789	0	0	0	0	1
rst_flag_number	0.03951838	0.1392995	0	0	0	0	1
psh_flag_number	0.02217887	0.09653540	0	0	0	0	1
ack_flag_number	0.09566387	0.2521415	0	0	0	0	1
ece_flag_number	0.00000291	0.0005036	0	0	0	0	0.2
cwr_flag_number	0.00000192	0.00037966	0	0	0	0	0.2
ack_count	0.02797713	0.17825565	0	0	0	0	11.2
syn_count	0.2938792	0.60364144	0	0	0	0	10.7
fin_count	0.08557531	0.56123908	0	0	0	0	151.74
rst_count	65.8712672	498.281211	0	0	0	0	9576.5
HTTP	0.00086767	0.0284215	0	0	0	0	1
HTTPS	0.00559028	0.06034021	0	0	0	0	1
DNS	0.00015156	0.0052478	0	0	0	0	1
Telnet	0.0000125	0.00089169	0	0	0	0	0.1
SMTP	0.0000125	0.00089171	0	0	0	0	0.1
SSH	0.0000261	0.00271903	0	0	0	0	1
IRC	0.0000125	0.00089234	0	0	0	0	0.11
TCP	0.41487102	0.48990278	0	0	0	1	1
UDP	0.31084808	0.45956644	0	0	0	1	1
DHCP	0.00000326	0.00098659	0	0	0	0	0.6
ARP	0.00076075	0.01928389	0	0	0	0	1
ICMP	0.27351348	0.44404871	0	0	0	0.99	1
IGMP	0.00000447	0.0012065	0	0	0	0	0.7
IPv	0.99923925	0.01928389	0	1	1	1	1
LLC	0.99923925	0.01928389	0	1	1	1	1
Tot sum	636.01138	991.6091	42	441	525	567	23467
Min	55.1201614	69.0993765	42	42	50	54	1514
Max	72.3325073	133.609047	42	43.77	50	54	1514
AVG	60.5865132	88.0720219	42	42.093330	50	54	1514
Std	6.06053694	38.0578569	0	0	0	0	721.15
Tot size	60.5890934	87.8768798	42	42.24	50	54	1514
IAT	84683677.9	17819169.6	-1.28	846791734	84696417	84696902.6	169470846.2
Number	9.49908609	0.84157738	1	9.5	9.5	9.5	15
Magnitue	10.4382451	3.15807323	9.17	9.1749773	10	10.3923048	55.03
Radius	8.56000977	53.8045034	0	0	0	0	1020.23
Covariance	2370.54475	19758.8155	0	0	0	0	520437.89
Variance	0.09074362	0.2329791	0	0	0	0	1
Weight	141.527342	21.6618865	1	141.55	141.55	141.55	244.60

Notre dataset contient plus de 8 millions d'instances (8.775.013). Ces dernières sas divisées en deux parties "entrainement" et "test", qui contiennent respectivement 7.160.831 et 1.614.182. Pour plus de détail, le tableau 3.3 donne le nombre d'instances pour chaque classe.

Tableau 3.3 : Nombre des instances (cas ou exemples) de chaque type d'attaque du dataset

Classe	Catégorie	Attaque	Nombre
Benign	-	-	230339
Attaque	SPOOFING	ARP Spoofing	17791
	RECON	Recon VulScan	3207
		OS Scan	20666
		Port Scan	106603
		Ping Sweep	926
	MQTT	Maltformed Data	6877
		DDOS Publish	36039
		DDOS Connect	214952
		DOS Publish	52881
		DOS Connect	15904
	DOS	DOS-TCP	462480
		DOS-UDP	704503
		DOS-ICMP	514724
		DOC-SYN	540498
	DDOS	DDOS-TCP	987063
		DDOS-UDP	1998026
		DDOS-ICMP	1887175
		DDOC-SYN	974359

4- Préparation du dataset

N'importe quel modèle d'apprentissage automatique nécessite une phrase préalable de prétraitement et une phase primordiale, car elle a une influence sur les résultats du modèle.

Nous avant au total 72 fichiers d'extension CSV répartis en deux dossiers (**Train** et **Test**) : 51 fichiers (1,73 Go) dans le dossier Train et 21 fichiers (393 Mo) dans le dossier test.

4.1- Duplication et réorganisation du dataset

Nous avant dupliquer et réorganiser ce dataset, en trois autres datasets.

4.1.1- Dataset 1 (2 classes)

Sous les deux dossiers racines **Train** et **Test**, nous avons créé deux sous-dossiers (**Attack** et **Benign**). Chaque sous-dossier regroupe les fichiers CSV et porte le nom de la classe correspondante.

Ensuite, à l'aide de code Python que nous avons développé, nous avons combiné les fichiers des sous-dossiers **Attack** et **Benign** pour obtenir deux fichiers CSV : **train_2classes** et **test_2classes**. Chaque fichier CSV contient une colonne "Type" qui indique la classe (Attack ou Benign)[7] des instances.

Enfin, les fichiers **train_2classes** et **test_2classes** sont regroupés dans un seul fichier nommé **dataset1.csv**. Le tableau 3.4 et la figure 3.3 illustrent des détails sur ce dataset.

[7] Pour les trois jeux de données, les noms des classes sont extraits des noms des dossiers parent des fichiers CSV

Tableau 3.4 : Statistiques du Dataset 1

	Les classes concernées	Nombre d'instances	
		Train	Test
Dataset 1 (2 Classes)	Benign	192732	37607
	Attack	6968099	1576575

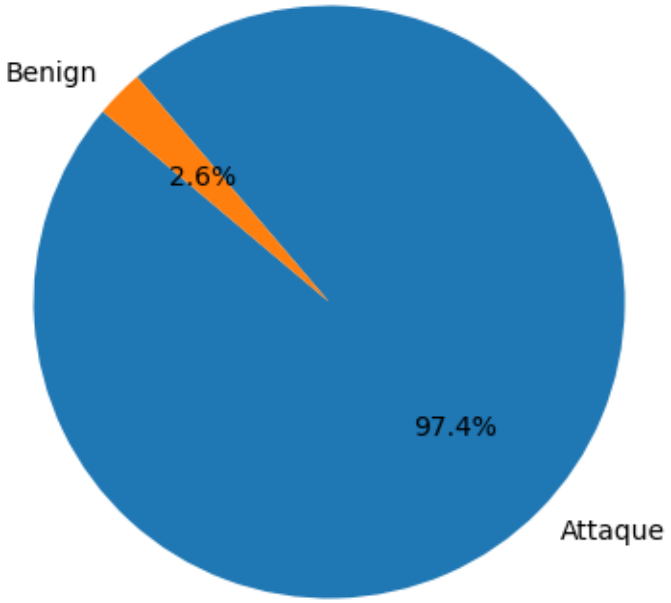

Figure 3.3 : Diagramme circulaire des pourcentages du Dataset 1

4.1.2- Dataset 2 (6 classes)

Pour le deuxième dataset dans les deux dossiers racines **Train** et **Test**, nous avons créé six sous-dossiers (**DoS**, **DDoS**, **MQTT**, **Recon**, **Spoofing** et **Benign**). Chaque sous-dossier regroupe les fichiers CSV et porte le nom de la classe correspondante.

Nous avons combiné les fichiers de chacun de six sous-dossiers pour obtenir deux fichiers CSV : **train_6classes** et **test_6classes**. Chaque fichier CSV contient une colonne "Type" qui indique la classe (DoS, DDoS, MQTT, Recon, Spoofing ou Benign) des instances.

Les deux fichiers **train_6classes** et **test_6classes** sont regroupés dans un seul fichier nommé **dataset2.csv**. Pour plus de détail, le tableau 3.5 et la figure 3.4 illustrent le nombre de données dans ce dataset.

Tableau 3.5 : Statistiques du Dataset 2

	Les classes concernées	Nombres d'instances	
		Train	Test
Dataset 2 (6 Classes)	Benign	192732	37607
	Dos	1805529	416676
	DDos	4779859	1066764
	MQTT	262938	63715
	Recon	103726	27676
	Spoofing	16047	1744

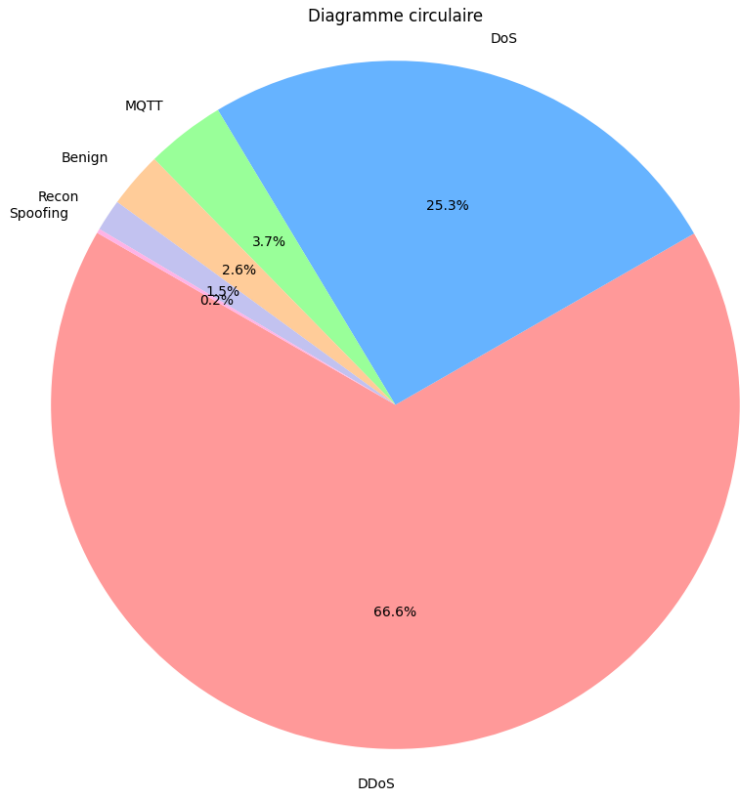

Figure 3.4 : Diagramme circulaire des pourcentages du Dataset 2

4.1.3- Dataset 3 (19 classes)

Dans ce dataset, nous avons créé dix-neuf sous-dossiers dans les dossiers racines Train et Test. Le dossier "Benign" contient un seul fichier (Benign.csv). Les autres dossiers contiennent les fichiers CSV des différents types d'attaques, nommés selon leurs catégories respectives (par exemple DoS-ICMP, DDoS-TCP, VulScan, DoSconnect, Spoofing, etc.).

Nous avons combiné les fichiers de chacun des dix-neuf sous-dossiers pour obtenir deux fichiers CSV : **train_19classes** et **test_19classes**. Chaque fichier CSV contient une colonne "Type" qui indique les dix-neuf classes des instances.

Les deux fichiers **train_19classes** et **test_19classes** sont regroupés dans un seul fichier nommé **dataset3.csv**. Pour plus de détail, le tableau 3.6 et la figure 3.5 montrent la répartition des données dans ce dataset.

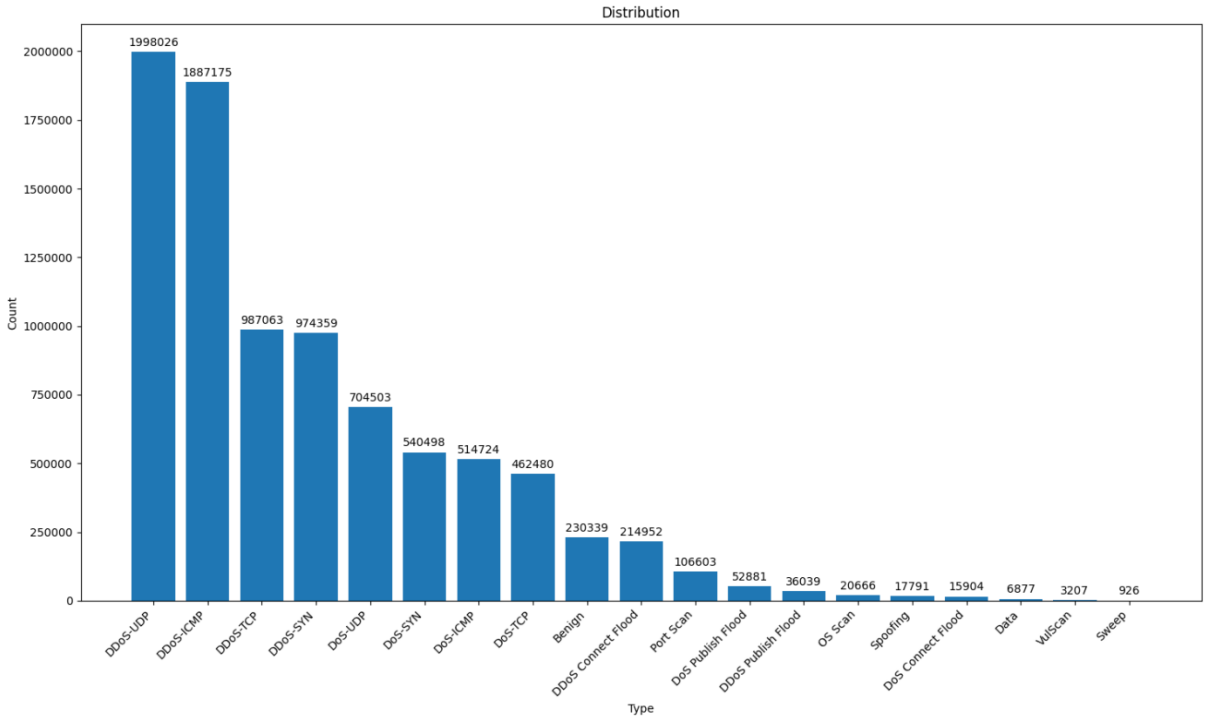

Figure 3.5 : Distribution des classes dans le Dataset 3

Tableau 3.6 : Statistique du Dataset 3

	Les classes concernées	Nombre d'instances	
		Train	Test
Dataset3 (19 Classes)	Benign	192732	37607
	DoS_ICMP	416292	98432
	DoS_TCP	380384	82096
	DoS_UDP	566950	137553
	DoS_SYN	441903	98595
	DDoS_ICMP	1537476	346999
	DDoS_TCP	804465	182598
	DDoS_UDP	1635956	362070
	DDoS_SYN	801962	172397
	MQTT-Maltformed-Data	5130	1747
	MQTT-DoS-Connect_Flood	12773	3131
	MQTT-DoS-Publish_Flood	44376	8505
	MQTT-DDoS-Connect_Flood	173036	41916
	MQTT-DDoS-Publish_Flood	27623	8416
	Recon-OS_Scan	16832	3834
	Recon-Port_Scan	83981	22622
	Recon-VulScan	2173	1034
	Recon-Ping_Sweep	740	186
	ARP_Spoofing	16047	1744

4.2- Nettoyage des Datasets

- ✓ Suppression des colonnes dont toutes les valeurs sont égales à 0. Après cette opération, la colonne "Drate" a été supprimée.
- ✓ Dans certains jeux de données, les fichiers CSV peuvent contenir un nombre différent de features. Pour cela, nous avons vérifié le nombre de features.
- ✓ Vérification des valeurs 'NAN' (Not A Number) ou 'INF' (Infinite Value).

4.3- Encodage

La dernière colonne "Type" est une colonne qui regroupe des classes (catégories). Ainsi, pour les besoins de traitement, elle nécessite une transformation. Pour cela, nous avant utilisé la technique de **Label Ecnoding**.

Pour la colonne "Type", qui représente la classe de chaque instance, nous avons utilisé la technique de "Label Encoding". La figure 3.6 montre un exemple de l'encodage dans le Dataset 1

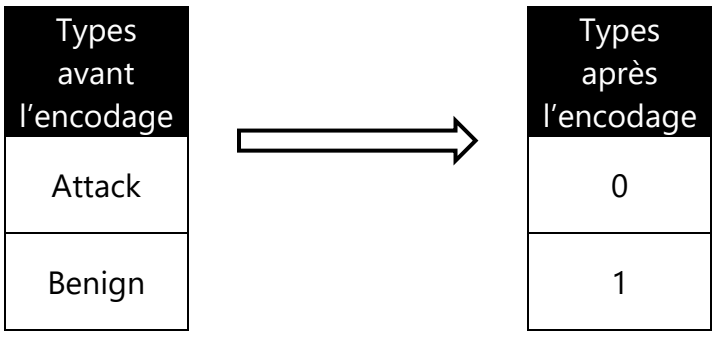

Figure 3.6 : Exemple de Label Encoding dans Dataset1

4.4 - Normalisation (Min-Max)

L'objectif de la normalisation est de réduire l'échelle entre les valeurs. La normalisation Min-Max est montrée par la formule suivante :

$$x' = \frac{x - \mathbf{Min}(x)}{\mathbf{Max}(x) - \mathbf{Min}(x)} [49]$$

- ✓ x′ est la valeur normalisée ;
- ✓ x est la valeur originale ;
- ✓ Min(x) est la valeur minimale dans l'ensemble de données ;
- ✓ Max(x) est la valeur maximale dans l'ensemble de données.

4.5- Division des Datasets (Train, Validation et Test)

Les données d'entraînement sont divisées en deux parties : une partie est utilisée pour l'entraînement du modèle et une autre pour la validation. Ensuite, le modèle sera testé uniquement sur l'ensemble de test.

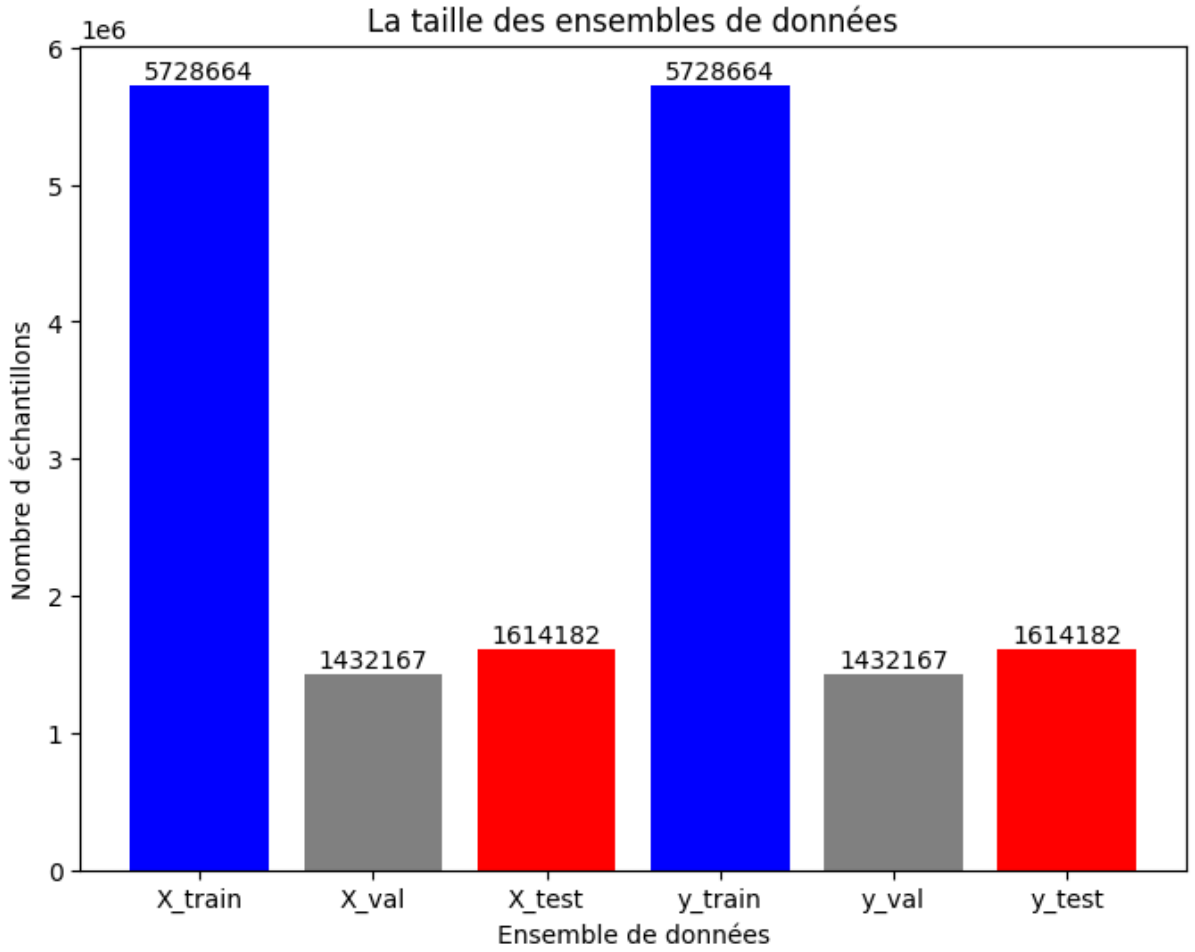

Figure 3.7: L'ensemble des données (Train, Validation et Test)

5- Modèle proposé

Dans notre travail, nous avons implémenté deux modèles Deep Learning : réseau de neurone convolutif et le réseau de neurone récurrent. Les deux modèles ont été évalués sur les trois datasets :

- ✓ Une classification binaire (deux classes) sur le Dataset 1 ;
- ✓ Une multi-classification (six classes) sur le Dataset 2 ;
- ✓ Une multi-classification (dix-neuf classes) sur le Dataset 3.

5.1- Paramètres globaux

Les architectures des modèles proposées partagent certaines propriétés et paramètres que nous détaillons ci-dessous :

- ✓ Les couches d'entrée ont des dimensions égales au nombre de features.
- ✓ Nous avons utilisé la fonction ReLU (RectifiedLinear Unit).
- ✓ Les couches de sortie ont les mêmes dimensions que le nombre de classes.
  - Classification binaire → fonction d'activation " Sigmoid"
  - Classification multi-classes → fonction d'activation "Softmax"
- ✓ La technique du "dropout" et "EarlyStopping" ont été utilisées, pour éviter le problème de sur-apprentissage (Overfitting).
- ✓ La fonction de perte (Loss function) :
  - Classification binaire → "binary-cross-entropy"
  - Classification multi-classes → "categorical-cross-entropy"
- ✓ L'optimiseur "Adam" a été utilisé avec un taux d'apprentissage (Learning Rate) de 0.001. Aussi, "ReduceLROnPlateau" pour l'ajustement du Learning Rate.
- ✓ En utilisant « reshape », nous préparons nos données pour les rendre compatibles avec l'entrée d'un modèle CNN.
- ✓ Différentes tailles de Batch_size.

- ✓ Le nombre des "Epochs" change selon nos expériences

5.2- Un modèle de détection d'intrusion basé sur le réseau de neurones convolutif CNN

- ✓ Nous avons utilisés des couches de convolution 1D
- ✓ Nous avons commencé par nombres varié de couches convolution et de filtres (32, 64, etc.) avec des "Kernel" de (3, 6, etc.)
- ✓ L'utilisation de couches "MaxPooling1D".
- ✓ L'utilisation de couches "Flatten".
- ✓ En dernier des couches "Dense".

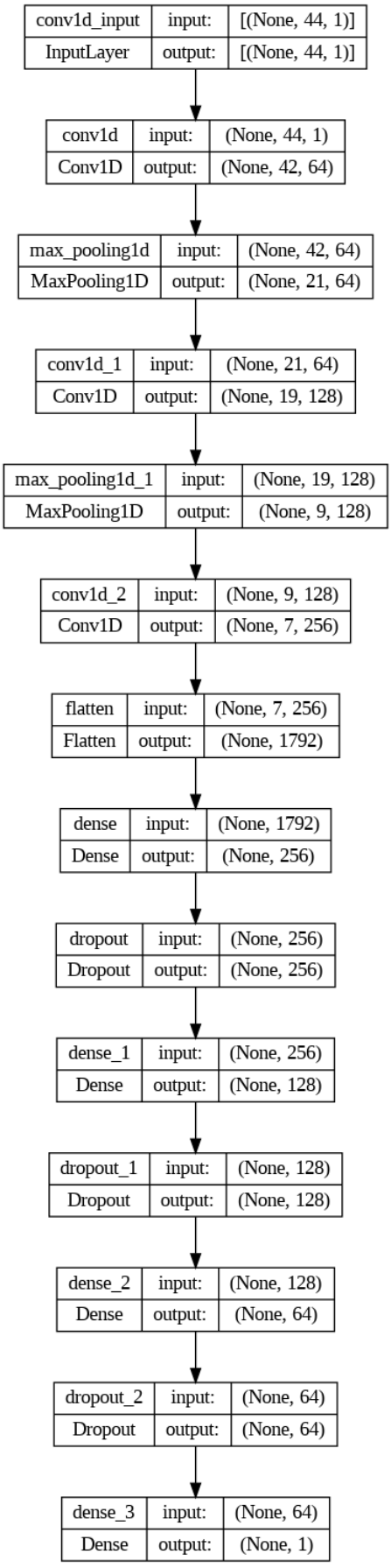

Figure 3.8 : L'architecture de modèle CNN proposée pour le Dataset 1

6- Toutes les Expérimentations et résultats (CNN)

6.1- Expérimentations sur le Dataset 1

6.1.1- Expérimentation 1

Dans cette expérimentation, nous avons défini ces paramètres :

- ✓ Les 3 couches de convolution 1D ont des filtres de différentes tailles (64,128 et 256), kernel_size = 3 et fonction d'activation ReLU.
- ✓ 2 couches MaxPooling1D avec pool_size = 2.
- ✓ 3 couches Dense avec 256, 128 et 64 neurones, toutes les couches avec une activation ReLU et une régularisation L2 (0.001)
- ✓ Des couches Dropout (0.2) pour la régularisation.
- ✓ Le modèle est entraîné sur les données d'entraînement (X_train et y_train) avec un batch_size de 64 et sur 5 epochs.

Les résultats obtenus sont décrits ci-dessous :

Tableau 3. 7 : Métriques de performance du modèle 1_Dataset 1

Métriques	Résultats
Accuracy	0.996
Recall	0.944
Precision	0.905
F1-Score	0.924

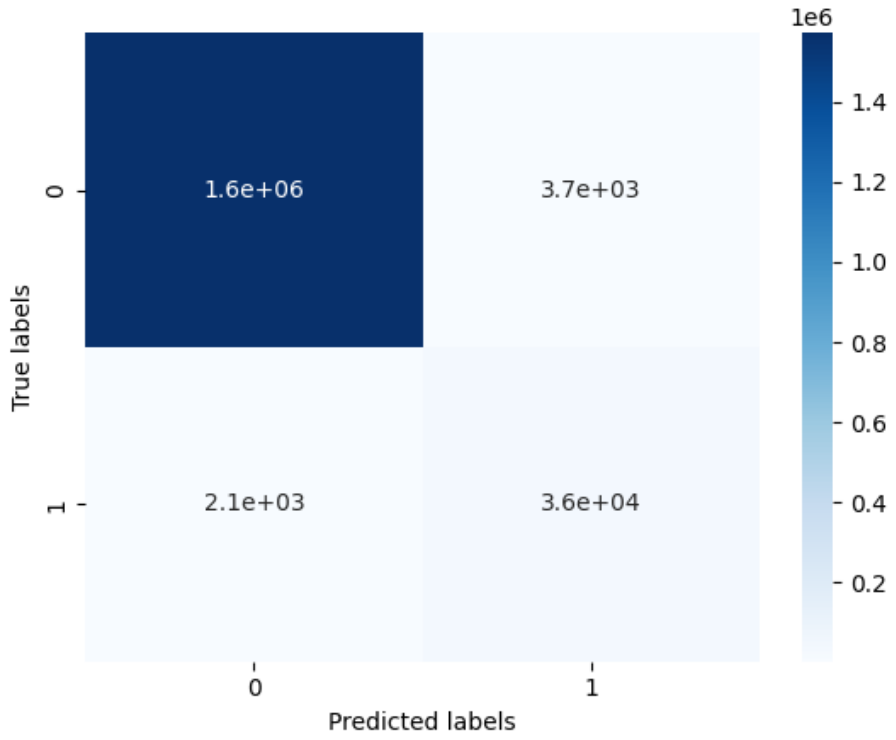

Figure 3.9 : Matrice de confusion du modèle 1_Dataset 1

6.1.2- Expérimentation 2

Dans cette expérimentation, nous avons défini les mêmes paramètres que ceux de l'expérimentation précédente avec Dataset 1.

Nous avons appliqué un sous-échantillonnage aléatoire pour équilibrer les classes majoritaires dans l'ensemble d'entraînement.

Les résultats obtenus sont présentés ci-dessous :

Tableau 3. 8 : Métriques de performance du modèle 2_Dataset 1

Métriques	Résultats
Accuracy	0.996
Recall	1.0
Precision	0.868
F1-Score	0.929

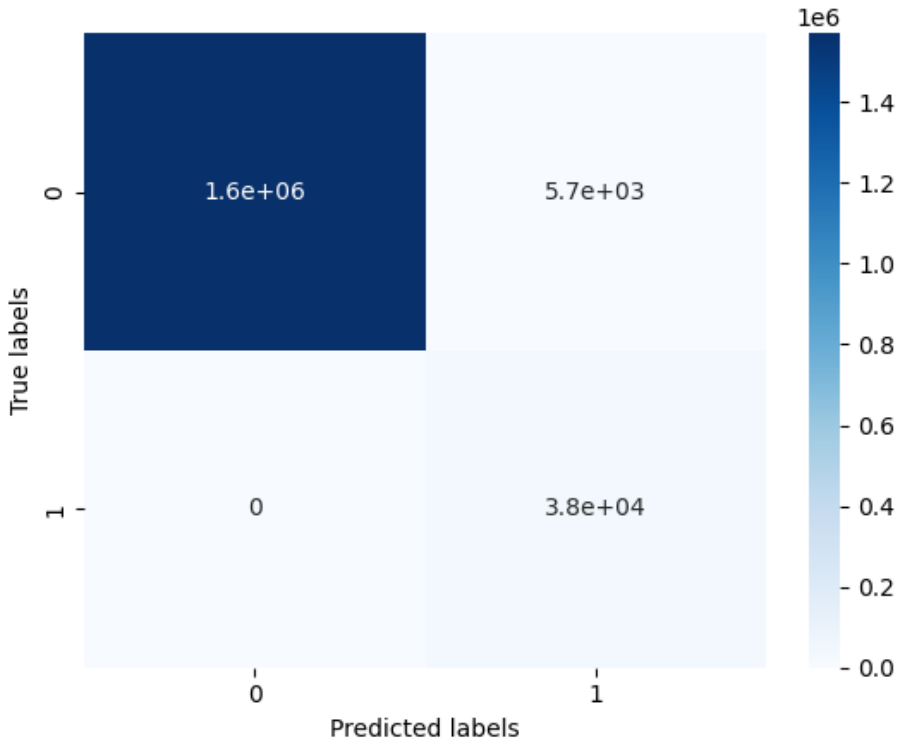

Figure 3.10 : Matrice de confusion du modèle 1_Dataset 1

6.2- Expérimentations sur le Dataset 2

6.2.1- Expérimentation 1

Dans cette expérimentation, nous avons défini les mêmes paramètres de l'expérimentation précédente de Dataset 1. Mais elle diffère un peu dans :

- ✓ En utilise la fonction **to_categorical** de Keras pour convertir les étiquettes de classe en un format binaire catégoriel (one-hot encoding).

Les résultats obtenus sont décrits ci-dessous :

Tableau 3. 9 : Métriques de performance du modèle 1_Dataset 2

Métriques	Résultats
Accuracy	0.782
Recall	0.782
Precision	0.771
F1-Score	0.753

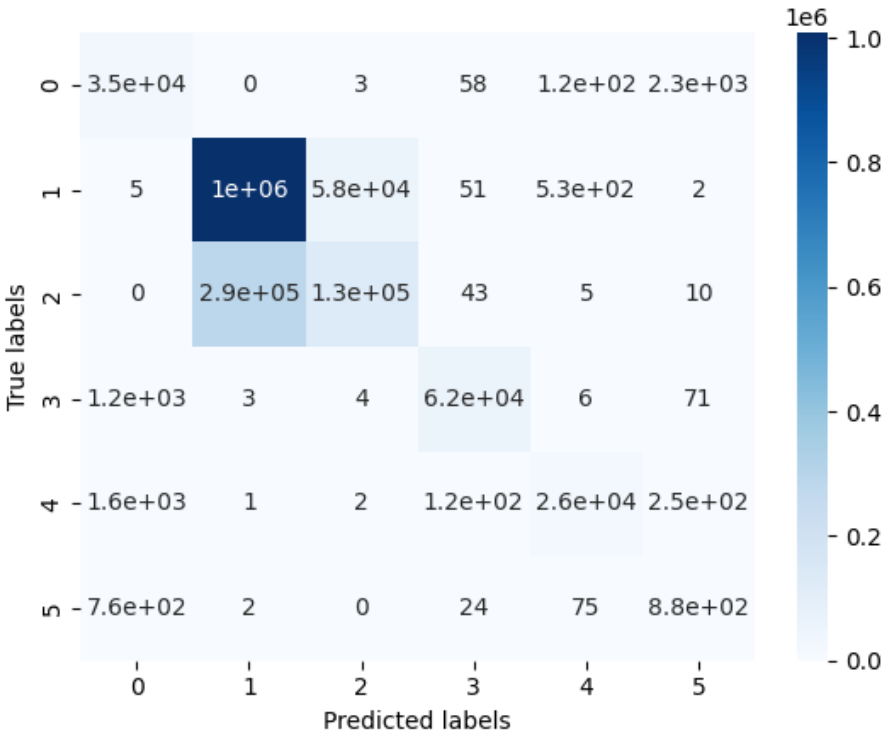

Figure 3.11 : Matrice de confusion du modèle 1_Dataset2

6.2.2- Expérimentation 2

L'architecture du CNN utilisée n'est pas très différente de l'architecture précédente. Elle diffère cependant dans certains aspects, comme :

- ✓ Les 3 couches de convolution ont des nombres de filtres variés de différentes tailles (32, 64 et 128).
- ✓ Dropout après chèque couche MaxPlooing1D.
- ✓ Une couche Dense avec 128 neurones, une activation ReLU et une régularisation L2 (0.001), suivies de couche Dropout (0.5) pour la régularisation.
- ✓ Le modèle est entraîné sur les données d'entraînement (X_train et y_train) avec un batch_size de 64 et sur 15 epochs.

Tableau 3. 10 : Métriques de performance du modèle 2_Dataset 2

Métriques	Résultats
Accuracy	0.779
Recall	0.779
Precision	0.802
F1-Score	0.724

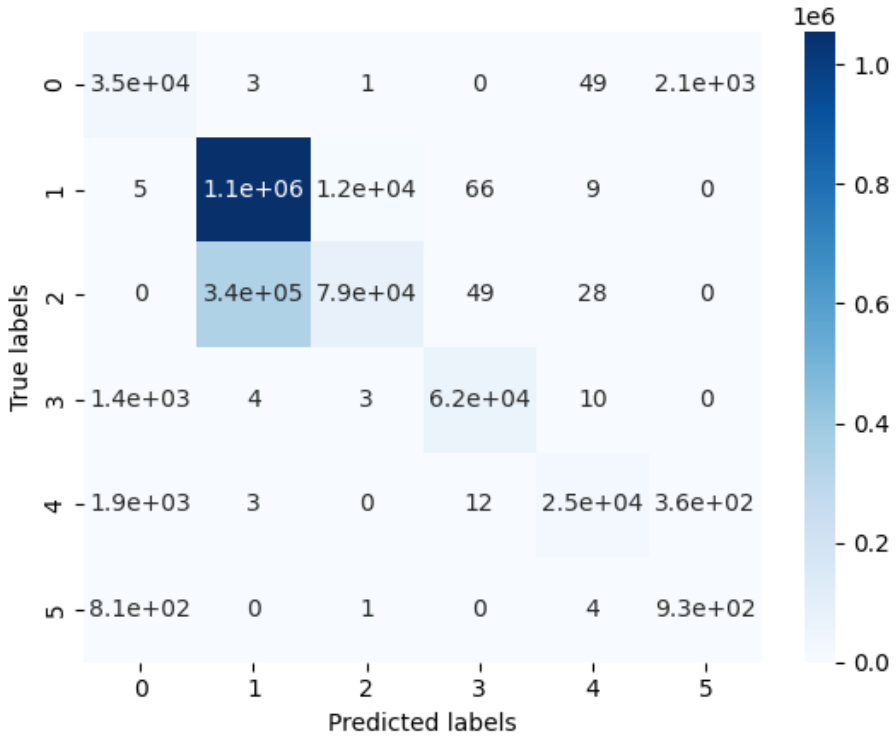

Figure 3.12 : Matrice de confusion du modèle 2_Dataset 2

6.2.3- Expérimentation 3

Dans cette expérience, nous avons utilisé les mêmes paramètres de l'expérimentation 2 (Dataset 2). Cependant, elle diffère sur certains points :

- ✓ L'ajout des couches Dense avec 128, 64 et 32 neurones, toutes les couches avec une activation ReLU et une régularisation L2 (0.001), suivies de couches Dropout (0.5) pour la régularisation.

✓ Nous avons utilisé 90% des données de Train pour l'entraînement et les 10% restantes pour la validation. Avec un batch_size = 64 et 10 epochs.

Tableau 3. 11 : Métriques de performance du modèle 3_Dataset 2

Métriques	Résultats
Accuracy	0.779
Recall	0.779
Precision	0.803
F1-Score	0.722

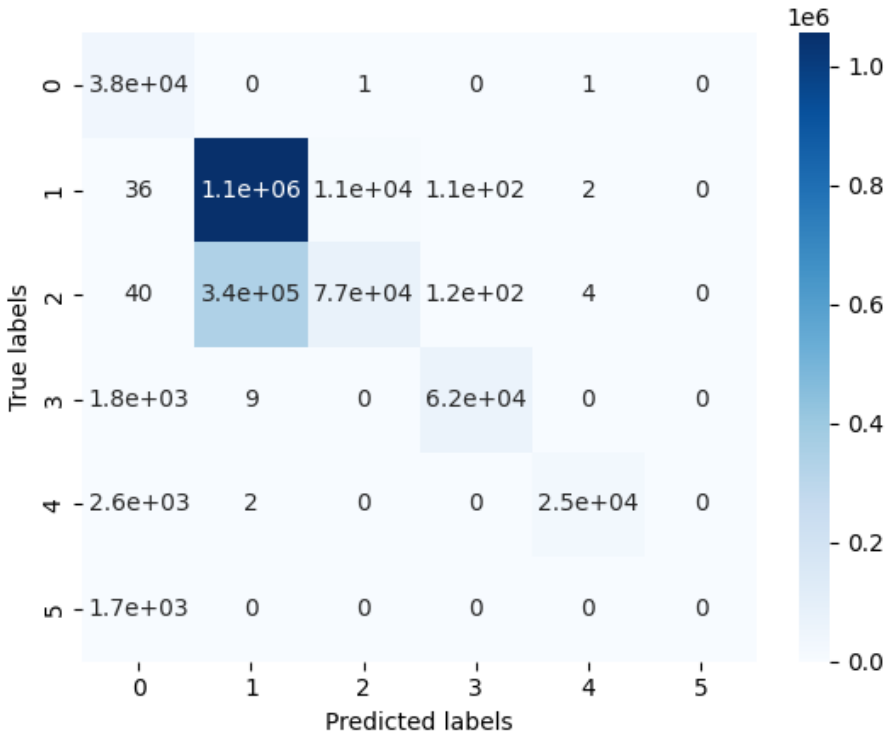

Figure 3.13 : Matrice de confusion du modèle 3_Dataset 2

6.2.4- Expérimentation 4

Nous avons utilisé les mêmes paramètres de l'expérimentation précédente, mais avec un batch size de 128 sur 5 époques.

Tableau 3. 12 : Métriques de performance du modèle 4_Dataset 2

Métriques	Résultats
Accuracy	0.780
Recall	0.780
Precision	0.799
F1-Score	0.727

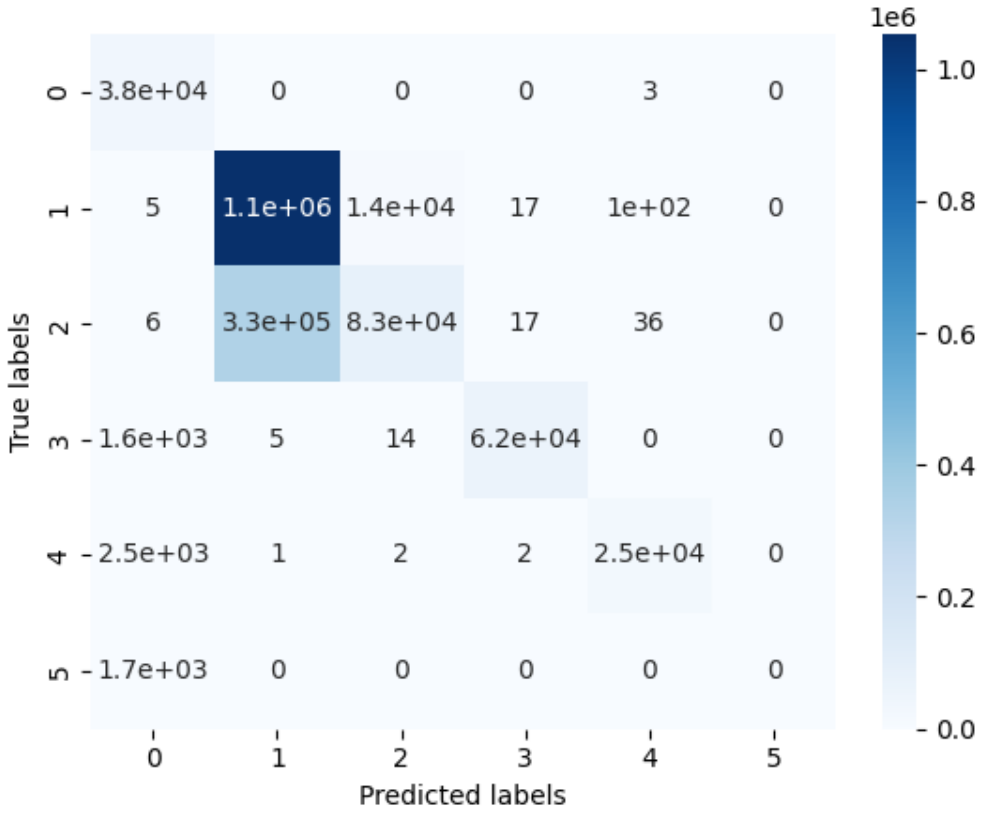

Figure 3.14 : Matrice de confusion du modèle 4__Dataset 2

6.2.5- Expérimentation 5

Mêmes paramètres des expérimentations précédentes 3 et 4, mais avec un batch size de 128 sur 20 époques.

Tableau 3. 13 : Métriques de performance du modèle 5_Dataset 2

Métriques	Résultats
Accuracy	0.775
Recall	0.775
Precision	0.797
F1-Score	0.716

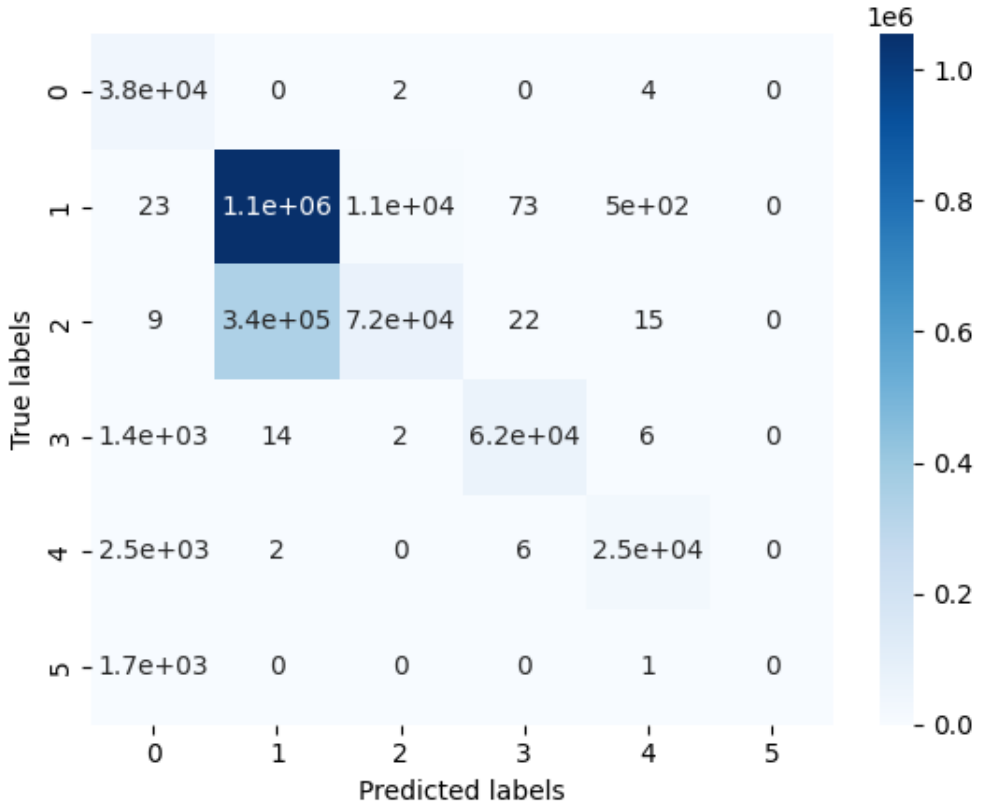

Figure 3.15 : Matrice de confusion du modèle 5_Dataset 2

6.2.6- Expérimentation 6

Mêmes paramètres que dans l'expérimentation précédente : 3, 4 et 5, mais nous avons ajouté 3 couches de "BatchNormalization".

Tableau 3. 14 : Métriques de performance du modèle 6_Dataset2

Métriques	Résultats
Accuracy	0.775
Recall	0.775
Precision	0.797
F1-Score	0.716

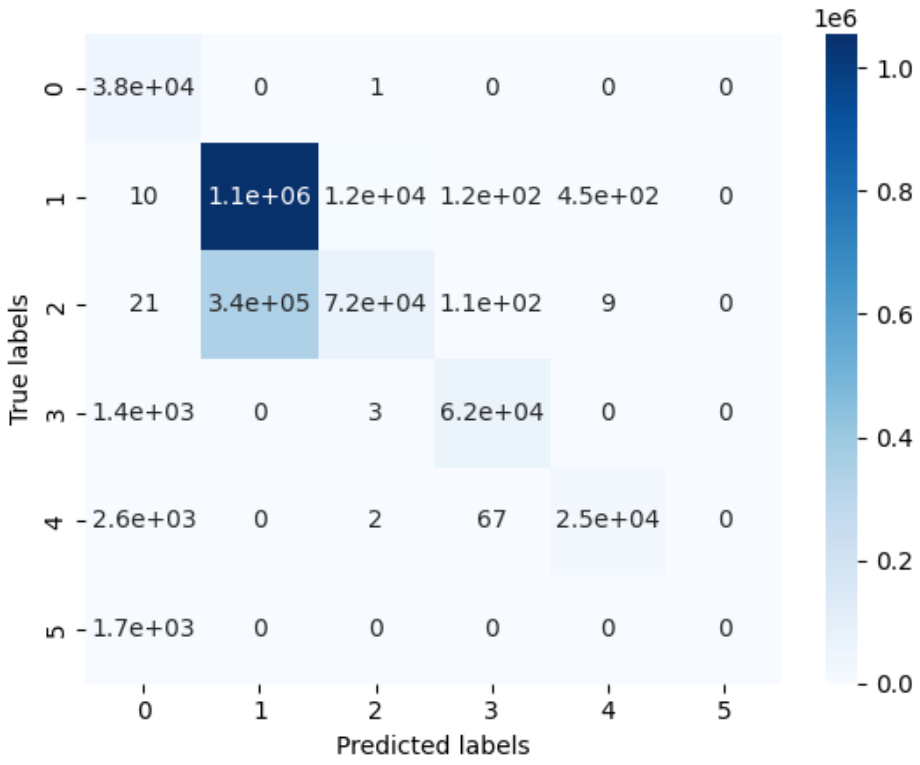

Figure 3.16 : Matrice de confusion du modèle 6_Dataset 2

6.3- Expérimentations sur le Dataset 3

6.3.1- Expérimentation 1

L'architecture du modèle est similaire à l'expérimentation 2 avec du Dataset 2.

Tableau 3.15: Métriques de performance du modèle 1_Dataset3

Métriques	Résultats
Accuracy	0.767
Recall	0.767
Precision	0.758
F1-Score	0.693

6.3.2- Expérimentation 2

Mêmes paramètres utilisés sauf que le "batch size" est égal à 64 et les nombres d'époques est 5.

Tableau 3.16: Métriques de performance du modèle 2_Dataset3

Métriques	Résultats
Accuracy	0.760
Recall	0.760
Precision	0.615
F1-Score	0.671

7- Expérimentations et résultats (RNN)

7.1- Expérimentation sur le Dataset 1

Dans cette expérimentation, nous avons utilisé le modèle RNN avec quelques paramètres :

- ✓ Utilisation de trois couches LSTM (Long Short-Term Memory) avec (64, 32) nombre de neurones.
- ✓ Des couches de dropout (0.2) entre elles pour prévenir le sur-apprentissage.
- ✓ Epochs = 5.
- ✓ Batch size = 64.

Tableau 3.17: Métriques de performance du modèle RNN_Dataset 1

Métriques	Résultats
Accuracy	0.995
Recall	0.900
Precision	0.923
F1-Score	0.911

7.2- Expérimentation sur le Dataset 3

Dans cette expérimentation, nous avons utilisé les mêmes paramètres que dans l'expérimentation précédente.

Tableau 3.18: Métriques de performance du modèle RNN_Dataset 3

Métriques	Résultats
Accuracy	0.777
Recall	0.777
Precision	0.728
F1-Score	0.698

8- Résultats et Discussion générale

Nous avons implémenté 2 modèles de Deep Learning (CNN et RNN). Ces modèles ont été formés et testés sur 3 sous-ensembles de données différentes du CICIoMT2024 Dataset.

Pour avoir les meilleurs paramètres qui conduisent aux bons résultats, nous avons relancé l'apprentissage plusieurs fois. Les paramètres principaux fixés pour l'apprentissage sont : batch-size, epoch, fonction d'activation, optimiseurs, etc.

Vu les spécifications des données que nous avons utilisées, les temps d'exécution étaient considérablement longs, surtout pour les modèles les plus complexes. La figure montre les meilleurs résultats des modèles proposés.

Tableau 3.19: Résultats d'un Modèle CNN sur les 3 datasets

		Modèle CNN
Dataset 1	Accuracy	0.996
	Recall	1.0
	Precision	0.868
	F1-Score	0.929
Dataset 2	Accuracy	0.782
	Recall	0.782
	Precision	0.771
	F1-Score	0.753
Dataset 3	Accuracy	0.767
	Recall	0.767
	Precision	0.775
	F1-Score	0.717

Pour Dataset 1, notre modèle a obtenu :

- ✓ L'accuracy de 0.996 indique que le modèle a correctement classé 99.6% des instances dans le dataset. Cette métrique mesure la proportion totale des prédictions correctes (vrais positifs et vrais négatifs) sur l'ensemble des instances. Montrant une très bonne performance globale.
- ✓ Le recall de 1.0 indique que Le modèle identifie toutes les instances positives sans n'en manquer aucune, ce qui est crucial pour minimiser les faux négatifs.
- ✓ La précision de 0.868 signifie que parmi toutes les instances classées comme positives par le modèle, 86.8% étaient effectivement positives. Cette métrique

est importante dans des contextes où il est essentiel de minimiser les faux positifs Bien que le score de précision soit légèrement inférieur à celui du recall, il reste élevé.

- ✓ Le F1-Score de 0.929 est la moyenne harmonique de la precision et du recall. Il fournit une mesure équilibrée des performances du modèle, surtout lorsqu'il y a un déséquilibre entre le nombre de classes positives et négatives. Un F1-Score de 0.929 indique que le modèle a une bonne balance entre la capacité à détecter les instances positives (recall) et la précision des prédictions positives (précision).

Pour Dataset 2, notre modèle CNN a obtenu :

- ✓ L'accuracy de 0.782 indique que le modèle a correctement classé 78.2% des instances dans le dataset. Cette métrique, qui mesure la proportion totale des prédictions correctes (vrais positifs et vrais négatifs) sur l'ensemble des instances, montre une performance modérée.
- ✓ Le recall de 0.782 signifie que le modèle identifie 78.2% des instances positives. Cela montre que le modèle réussit à détecter une bonne proportion des instances positives, mais qu'il en manque encore environ 21.8%.
- ✓ La precision de 0.771 indique que parmi toutes les instances classées comme positives par le modèle, 77.1% étaient effectivement positives. Cette métrique est importante dans des contextes où il est essentiel de minimiser les faux positifs.
- ✓ Un F1-Score de 0.753 montre que le modèle a une balance modérée entre la capacité à détecter les instances positives (recall) et la précision des prédictions positives (precision). Cependant, comme le F1-Score est inférieur à 0.8, il indique que le modèle pourrait bénéficier d'améliorations pour mieux gérer les déséquilibres entre les classes positives et négatives.

Sur Dataset3, notre modèle a obtenu :

- ✓ L'accuracy de 0.767 indique que le modèle a correctement classé 76.7% des instances dans le dataset. Cette métrique, qui mesure la proportion totale des prédictions correctes (vrais positifs et vrais négatifs) sur l'ensemble des instances, montre une performance modérée.
- ✓ Le recall de 0.767 signifie que le modèle identifie 76.7% des instances positives. Cela montre que le modèle réussit à détecter une bonne proportion des instances positives, mais qu'il en manque encore environ 23.3%.
- ✓ La precision de 0.775 indique que parmi toutes les instances classées comme positives par le modèle, 77.5% étaient effectivement positives. La precision, bien qu'un peu supérieur au recall, montre que le modèle génère encore un nombre significatif de faux positifs.
- ✓ Un F1-Score de 0.717 montre que le modèle a une balance modérée entre la capacité à détecter les instances positives (recall) et la précision des prédictions positives (precision).

Les résultats montrent que le modèle CNN performe exceptionnellement bien sur le Dataset 1, avec des scores élevés dans toutes les métriques clés. L'accuracy et le recall particulièrement élevés sont indicatifs d'un modèle bien entraîné et performant. Cependant, pour les Dataset 2 et Dataset 3, le modèle CNN a des performances modérées. Avec une accuracy, un recall et une precision autour de 0.78, le modèle n'est ni particulièrement performant ni totalement inadéquat. Il pourrait être utile dans des applications où une performance modérée est acceptable.

9- Comparaison

Les auteurs [44] ont testé un modèle DNN

Les mesures de performance des 3 modèles DNN, CNN et RNN pour les 3 datasets ont été illustrées dans le tableau 3.20.

Tableau 3.20: Comparaison des résultats entre les méthodes DL

		Modèle CNN	Modèle RNN	Modèle DNN
Dataset 1	Accuracy	0.996	0.995	0.996
	Recall	1.0	0.900	0.952
	Precision	0.868	0.923	0.962
	F1-Score	0.929	0.911	0.957
Dataset 2	Accuracy	0.782	/	0.780
	Recall	0.782	/	0.760
	Precision	0.771	/	0.768
	F1-Score	0.753	/	0.733
Dataset 3	Accuracy	0.767	0.777	0.776
	Recall	0.767	0.777	0.601
	Precision	0.775	0.728	0.738
	F1-Score	0.717	0.698	0.579

Pour le Dataset 1, les modèles CNN et DNN montrent des performances similaires en termes d'accuracy (0.996 pour les deux). Le modèle RNN a un recall légèrement inférieur (0.900) par rapport au CNN (1.000) et au DNN (0.952). En termes de précision et de F1-Score, le DNN présente de meilleures performances que le CNN et le RNN.

Pour Dataset 2, CNN et DNN ont des accuracies similaires (0.782 et 0.780), mais CNN a de meilleures recall et precision.

Pour Dataset 3, DNN montre la meilleure performance en accuracy (0.776), suivie de près par RNN (0.777) et CNN (0.767). CNN et RNN ont des performances assez similaires en recall et precision, tandis que DNN a une recall plus élevée mais une precision légèrement inférieure. En F1-Score, CNN a le score le plus élevé, suivi par DNN et RNN.

Les performances varient selon le dataset et la métrique évaluée. CNN montre généralement de bonnes performances en F1-Score et en recall sur plusieurs datasets, tandis que DNN obtient souvent des résultats solides en termes d'accuracy et de precision. RNN, bien que moins représenté ici, présente des performances compétitives en recall sur certains datasets.

10- Conclusion

Ce dernier chapitre a fourni les détails concernant nos trois ensembles de données d'expérimentation. Ensuite, nous avons examiné plusieurs expérimentations visant à ajuster les paramètres afin d'optimiser le modèle.

Conclusion et Perspectives

Nous avons travaillé sur un sujet d'actualité comportant plusieurs défis : l'utilisation du Deep Learning, le manque de ressources matérielles, la disponibilité de nombreux travaux existants, ce qui rend la tâche d'apporter de la nouveauté très difficile et l'utilisation du récent dataset.

Nous avons implémenté deux modèles, CNN et RNN, puis les avons testés pour la première fois sur le dataset CICIoMT2024, qui n'avait jusqu'alors été évalué qu'avec un modèle DNN.

Dans nos futurs travaux, nous envisageons d'améliorer nos modèles pour le CICIoMT2024 et d'explorer d'autres types de datasets, tels que CICEVSE2024, en utilisant des algorithmes d'apprentissage profond non supervisés, comme les auto_encodeurs et d'autres algorithmes intéressants.

Références

[1] Entreprise IBM, « Qu'est-ce qu'un système de détection d'intrusion (IDS) ? », *IBM*, www.ibm.com/fr-fr/topics/intrusion-detection-system, 8 décembre 2023.

[2] Traian Toma, « Les systèmes de détection des intrusions », Université de Montréal, Québec, Canada, 2024.

[3] Lumír Honus, « Design, implementation and simulation of intrusion detection system for wireless sensor networks », Master, Faculté d'informatique de l'univarsité Masarykova, Brno, 2009.

[4] Md. Safiqul Islam, Razib Hayat Khan, et Dewan Muhammad Bappy, « A Hierarchical Intrusion Detection System in Wireless Sensor Networks », *IJCSNS*, suède, Aout 2010.

[5] Hossein Jadidoleslamy, « A High-Level Architecture for Intrusion Detection on Heterogeneous Wireless Sensor Networks: Hierarchical, Scalable and Dynamic Reconfigurable », *Scientific Research*, Rasht, Iran, 2011.

[6] Idres Louiza, « Système de détection d'intrusion hybride et hiérarchique pour les réseaux de capteurs sans fil », Thése de Master, Mouloud Mammeri, Tizi-Ouzou, 2012.

[7] Manu Bijone, « A Survey on Secure Network: Intrusion Detection & Prevention Approaches », *American Journal of Information Systems*, Inde, 2016.

[8] Hamouda Djallel, « Intrusion Detection System for Cyber Security », Thése de Master, Université 8 Mai 1945, Guelma, 2020.

[9] P. Technlogy, « HIDS vs NIDS », www.linkedin.com/pulse/hids-vs-nids-pro-resources-consulting, 28 avril 2023.

[10] Asadullah Momand, Sana Ullah Jan, et Naeem Ramzan, « A Systematic and Comprehensive Survey of Recent Advances in Intrusion Detection Systems Using Machine Learning: Deep Learning, Datasets, and Attack Taxonomy », *Journal of Sensors*, paisley, 28 février 2023.

[11] IoT Industriel, « Système de détection d'intrusion (IDS) », *Blog*, www.iotindustriel.com/glossaire-iiot/systeme-de-detection-dintrusion-ids/, mai 2024.

[12] Mohammed BELGUIDOUM, « Détection des attaques de déni de service par une approche basée Deep learning », Master, Université Larbi, Tébessa, 2022.

[13] M. A. Erlinger et M. Wood, « Intrusion Detection Message Exchange Requirements », Internet Engineering Task Force, Atlanta, mars 2007.

[14] Entreprise Oracle, « Qu'est-ce que l'Internet of Things (IoT) ? », *Oracle*, www.oracle.com/fr/internet-of-things/what-is-iot/, 2024.

[15] Nikita Griffin, « Qu'est-ce que l'IoMT (Internet des Objets Médicaux) ? », *Fiberroad Technology*, www.fiberroad.com/fr/resources/articles/what-is-iomt-internet-of-medical-things/, 5 octobre 2022.

[16] Entreprise Amazone, « Qu'est-ce que l'Internet des objets (IoT) ? – L'internet des objets expliqué – AWS », *Amazon Web Services, Inc.*, www.aws.amazon.com/fr/what-is/iot/, 2023.

[17] Sajjad Dadkhah, Euclides Carlos Pinto Neto, Raphael Ferreira, Reginald Chukwuka Molokwu, Somayeh Sadeghl, et Ali Ghorbani, « CICIoMT2024: Attack Vectors in Healthcare devices-A Multi-Protocol Dataset for Assessing IoMT Device Security », *Preprints.org*, Fredericton, New Brunswick, Canada, 16 février 2024.

[18] J. Rio, « Qu'est-ce Que L'Internet Des Objets (IoT) Et Comment L'utiliser ? », *Blog RingCentral*, www.ringcentral.com/fr/fr/blog/iot/, 28 octobre 2021.

[19] Entreprise splunk, « Qu'est-ce que l'Internet des objets médicaux (IoMT) ? », *Splunk*, www.splunk.com/fr_fr/data-insider/what-is-the-internet-of-medical-things-iomt.html, mars 2024.

[20] Mohammed Zubair *et al.*, « Secure Bluetooth Communication in Smart Healthcare Systems: A Novel Community Dataset and Intrusion Detection System », *Sensors*, Basel, Switzerland, 2022.

[21] Mohammad Wazid, Ashok Kumar Das, Joel J. P. C. Rodrigues, Sachin Shetty, et Youngho Park, « IoMT Malware Detection Approaches: Analysis and Research Challenges », *IEEE Access*, USA, 2019.

[22] P. de Veyan, « Sécurisation des objets connectés de santé », *Veyan*, www.veyan.fr/securite-des-objets-connectes-dans-le-domaine-de-la-sante/, 27 mai 2021.

[23] A. A. Hady, A. Ghubaish, T. Salman, D. Unal, et R. Jain, « Intrusion Detection System for Healthcare Systems Using Medical and Network Data: A Comparison Study », *IEEE Access*, 2020.

[24] P. Radoglou-Grammatikis *et al.*, « Modeling, Detecting, and Mitigating Threats Against Industrial Healthcare Systems: A Combined Software Defined Networking and Reinforcement Learning Approach », *IEEE Transactions on Industrial Informatics*, mars 2022.

[25] Faisal Hussain, Ghalib Shah, Ivan Miguel Pires, et Usama Ubaid, « A Framework for Malicious Traffic Detection in IoT Healthcare Environment », *Sensors*, Pakistan, 2021.

[26] Ahmed, M, Byreddy, S, Nutakki, A, Sikos, L.F, et Haskell-Dowland, P, « ECU-IoHT: A dataset for analyzing cyberattacks in Internet of Health Things », *Ad Hoc Networks*, Perth, Australie, 2021.

[27] Mireya Lucia Hernandez-Jaimes, Alfonso Martinez-Cruz, Kelsey Alejandra Ramírez-Gutiérrez, et Claudia Feregrino-Uribe, « Artificial intelligence for IoMT security: A review of intrusion detection systems, attacks, datasets and Cloud–Fog–Edge architectures », *Internet of Things*, Puebla, Mexico, Aout 2023.

[28] Red Hat, « L'apprentissage automatique, ou machine learning, qu'est-ce que c'est ? », *Entreprise Red Hat*, www.redhat.com/fr/topics/ai/what-is-machine-learning, 8 mars 2024.

[29] Pitpitt, « Apprentissage supervisé — DataFranca », *datafranca.org*, www.datafranca.org/wiki/Apprentissage_supervis%C3%A9, 8 février 2024.

[30] Mohammed Boughaba et Boukhris Brahim, « L`apprentissage profond (Deep Learning) pour la classification et la recherche d'images par le contenu », Thése de Master, Kasdi Merbah, Ouargla, 2016.

[31] Study smarter, « Apprentissage non supervisé: IA, Algorithmes », *Équipe éditoriale StudySmarter*, www.studysmarter.fr/resumes/informatique/big-data/apprentissage-non-supervise/, mai 2024.
[32] Entreprise Amazone, « Qu'est-ce que l'apprentissage par renforcement ? - L'apprentissage par renforcement expliqué », *Amazon Web Services*, www.aws.amazon.com/fr/what-is/reinforcement-learning/, 2023.
[33] Entreprise IBM, « Que sont les réseaux neuronaux ? », *IBM*, www.ibm.com/fr-fr/topics/neural-networks, 16 mai 2024.
[34] Entreprise journaldunet, « Machine à vecteurs de support (SVM) : définition et cas d'usage », *Journal du net*, www.journaldunet.fr/intelligence-artificielle/guide-de-l-intelligence-artificielle/1501879-machine-a-vecteurs-de-support-svm-definition-et-cas-d-usage/#les-machines-a-vecteurs-de-support-svm-cest-quoi-, 2023.
[35] FADEL Ouissal, « Combinaison de classifieurs pour la reconnaissance des anomalies mammaires », Thése de Master, 8 Mai 1945, Guelma, 2023.
[36] J. Robert, « La régression logistique, qu'est-ce que c'est ? », *Formation Data Science | DataScientest.com*, www.datascientest.com/regression-logistique-quest-ce-que-cest, 4 novembre 2020.
[37] Entreprise IBM, « Qu'est-ce que l'algorithme des k plus proches voisins ? », *IBM*, www.ibm.com/fr-fr/topics/knn, 13 juillet 2022.
[38] Entreprise IBM, « Qu'est-ce qu'un arbre de décisions », *IBM*, www.ibm.com/fr-fr/topics/decision-trees, 20 octobre 2022.
[39] Delphine Lacour, « Deep learning : c'est quoi ? | NordVPN », *NordVPN*, www.nordvpn.com/fr/blog/apprentissage-profond/, 15 février 2023.
[40] Goodfellow, Ian, Bengio, Yoshua, et Courville, Aaron, « Deep Learning », Cambridge, Massachusetts, USA, 2016.
[41] Entreprise Math works, « Introduction aux réseaux neuronaux convolutifs (CNN) | 3 choses à savoir », *MathWorks*, www.fr.mathworks.com/discovery/convolutional-neural-network.html, mai 2024.
[42] A. A. Abid, « Recurrent Neural Network Tutorial (RNN) », *datacamp*, www.datacamp.com/tutorial/tutorial-for-recurrent-neural-network, mars 2022.
[43] Mohamed Abd Elmoumen et DJABALLAH, « Système de prédiction de la consommation d'énergie basé Deep Learning », Thése de Master, 8 Mai 1945, Guelma, 2021.
[44] Mohamed Amine Ferrag, Leandros Maglaras, Sotiris Moschoyiannis, et Helge Janicke, « Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study », *Journal of Information Security and Applications*, Guelma, Algeria, février 2020.
[45] Mostafa A. Salama, Heba F. Eid, Rabie A. Ramadan, Ashraf Darwish, et Aboul Ella Hassanien, « Hybrid intelligent intrusion detection scheme », Le Caire, Egypte, 2011.
[46] TIOBE Organisation, « TIOBE Index », *TIOBE*, www.tiobe.com/tiobe-index/, juin 2024.

[47] IEEE Magazine, « The Top Programming Languages 2023 », *IEEE Spectrum*, www.spectrum.ieee.org/the-top-programming-languages-2023, 29 août 2023.
[48] S. Prud'homme, « Initiation au Deep Learning avec Google Colab », *Moov AI*, www.moov.ai/fr/blog/deep-learning-avec-google-colab, 24 janvier 2019.
[49] Henderi, Tri Wahyuningsih, et Efana Rahwanto, « Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer », *International Journal of Informatics and Information System*, Indonésie, 1 mars 2021.