الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية التكنولوجيا

قسم: الإعلام الآلي

Mémoire de Master

Spécialité : **Réseaux Informatiques** et **Systèmes Répartis**

Thème

Deep Learning pour la sécurité des réseaux internet des objets

Présenté par :

AMARNI Fatima Zohra

HAMRI Sara

Dirigé par :

Dr. ADJIR Noureddine

Année universitaire 2022-2023

Abstract

The main objective of our work is to address the issue of intrusion detection in IOT networks using deep learning techniques. We use deep learning models (LSTM, GRU, and LSTM-GRU) for intrusion detection in IOT networks to enhance their security. We first have performed data preprocessing steps and evaluate multiple intrusion detection models on the Bot-IoT dataset. The experimental results demonstrate the effectiveness of our deep learning approach for intrusion detection in IOT networks.

Keywords : IOT network security, intrusion detection, Deep Learning, data preprocessing, Bot-IoT.

Résumé

L'objectif principal de notre travail est d'aborder la problématique de la détection d'intrusions dans les réseaux IDO en utilisant des techniques de Deep Learning. Nous avons utilisé des modèles de deep learning (LSTM, GRU et LSTM-GRU) pour la détection d'intrusions dans les réseaux IDO pour améliorer leur sécurité. Nous avons tout d'abord effectué des étapes de prétraitement des données et évalué plusieurs modèles de détection d'intrusions sur l'ensemble de données Bot-IoT. Les résultats expérimentaux ont montré l'efficacité de notre approche de Deep Learning pour la détection d'intrusions dans les réseaux IDO.

Mots clés : Sécurité des réseaux IDO, Détection d'intrusions, Deep Learning, Prétraitement des données, Bot-IoT.

الملخص

الهدف من عملنا هو معالجة مسألة كشف الاختراق في شبكات إنترنت الأشياء باستخدام تقنيات التعلم العميق. إستخدمنا نماذج التعلم العميق ($LSTM, GRU$ و $LSTM-GRU$) للكشف عن الاختراقات في شبكات إنترنت الأشياء و تحسين أمنها. لتحقيق هذا الهدف ، قمنا أولا بإجراء خطوات المعالجة المسبقة للبيانات و تقييم العديد من نماذج اكتشاف التطفل على مجموعة

بيانات $Bot-IoT$. أظهرت النتائج التجريبية فعالية نهج التعلم العميق الخاص بنا لاكتشاف التسلل في شبكات إنترنت الأشياء.

الكلمات المفتاحية: أمان شبكات الإنترنت الأشياء، كشف الاختراق، التعلم العميق، معالجة البيانات، $Bot-IoT$.

REMERCIEMENT

Nous tenons à exprimer notre profonde gratitude à **Allah** le Tout-Puissant, qui nous a accordé la force et la volonté nécessaires pour mener à bien ce modeste travail.En premier lieu, nous souhaitons remercier chaleureusement **Dr. Noureddine ADJIR** pour sa précieuse guidance, son soutien et ses précieux conseils tout au long de ce travail.
Nous adressons également nos sincères remerciements à tous nos enseignants du département d'informatique, qui ont partagé leurs connaissances et leur expertise avec nous tout au long de notre parcours académique.

Nous exprimons notre gratitude envers les membres du jury d'avoir accepté de juger ce travail et d'apporter leurs précieuses contributions et suggestions. Nous tenons à remercier toutes les personnes qui nous ont apporté leur aide, que ce soit directement ou indirectement, lors de la préparation de ce mémoire. Nous sommes sincèrement reconnaissants envers tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail. Leur soutien et leur participation ont été d'une valeur inestimable.

AMARNI & HAMRI

DEDICACE

Je dédie ce travail à ma très chère famille. Je ressens une profonde gratitude envers **mes chers parents**, dont le soutien indéfectible et les encouragements ont été d'une importance capitale dans la réalisation de ce travail.
Je dédie également ce travail à mes frères **Mohamed** et **Saïd**, ainsi qu'à ma sœur **Chaimaa**, pour leur soutien inconditionnel et leur amour précieux. Leur présence et leur encouragement constants m'ont inspiré et motivé tout au long de ce parcours.
Enfin, je tiens à exprimer ma gratitude envers tous mes amis et les membres de la famille **AMARNI**, qui ont été une source de soutien. Leurs encouragements, leurs conseils et leur présence bienveillante ont contribué à ma réussite.

AMARNI Fatima Z.

Je dédie ce modeste travail avec grand amour et fierté : A mes très **chers parents** ,source de vie ,d'amour et d'affectation.
Pour leur patience , leur soutien ,et leurs encouragement.A tous les membres de Ma famille.
A tous mes amis et camarades et tout qui m'aiment. . ..

HAMRI Sara

ACRONYMES

IOT : Internet Of Things
IDO : Internet Des Objets
AI : Artificial Intelligence
ANN : Artificial Neural Network
DNN : Deep Neural Network
CNN : Convolution Neural Network
RNN : Recurrent Neural Network
LSTM : Long Short-Term Memory
GRU : Gated Recurrent Unit
GAN : Generative Adversarial Network
SGD : Stochastic Gradient Descent
IDS : Intrusion Detection System
DoS : Denial of Service
DDoS : Distributed Denial of Service
SSL : Secure Sockets Layer
TLS : Transport Layer Security

TABLE DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

FIGURE **Page**

CHAPITRE 1

INTRODUCTION GÉNÉRALE

Les Réseaux Internet Des Objets (IDO) jouent un rôle de plus en plus important dans notre société interconnectée. Ces réseaux prennent en charge la communication et l'échange de données entre des milliards d'appareils, e.g. des appareils électroménagers à l'infrastructure industrielle. Cependant, avec cette connectivité croissante, la sécurité des réseaux IDO est devenue une préoccupation majeure. Les attaques informatiques contre ces réseaux pourraient compromettre la confidentialité, l'intégrité et la disponibilité des données, avec des conséquences potentiellement désastreuses.
Dans ce contexte, la détection d'intrusion joue un rôle essentiel dans la prévention et la lutte contre les cyberattaques IDO. L'identification précoce des comportements malveillants et la réponse rapide aux incidents de sécurité sont essentielles pour maintenir un haut niveau de sécurité. Cependant, la détection des intrusions dans les réseaux IDO présente des défis uniques.

1.1 Problématique

La problématique de la détection d'intrusions dans les réseaux IDO réside dans la complexité des données et la nécessité d'une détection en temps réel. Les méthodes traditionnelles de détection d'intrusions ne sont pas adaptées à l'environnement dynamique et aux contraintes des réseaux IDO. Par conséquent, il était important d'explorer de nouvelles approches basées sur le Deep Learning pour améliorer la précision et l'efficacité de la détection.

1.2 Objectif de notre travail

L'objectif de notre travail est donc de développer et d'évaluer des modèles de détection d'intrusions basés sur le Deep Learning pour les réseaux IDO. Nous nous concentrons sur l'utilisation de réseaux de neurones récurrents en raison de leur capacité à capturer les dépendances séquentielles dans les données de trafic. Nous évaluons la performance de nos modèles en utilisant l'ensemble de données Bot-IoT, en nous concentrant sur des métriques telles que l'AUC, l'exactitude et la précision.

1.3 Organisation du mémoire

Ce mémoire est organisé en 3 chapitres :

— **Chapitre 01 Apprentissage profond :** Dans ce premier chapitre, nous introduisons le concept de l'apprentissage profond. Nous explorons les principes fondamentaux de cette branche de l'intelligence artificielle, en mettant l'accent sur les réseaux de neurones profonds, les architectures et les algorithmes d'apprentissage.

— **Chapitre 02 Réseaux internet des objets :**
Le deuxième chapitre se concentre sur la sécurité des réseaux IDO, en mettant l'accent sur l'utilisation des systèmes de détection d'intrusion (IDS) et des techniques de détection d'anomalies pour renforcer la sécurité de ces réseaux.

— **Chapitre 03 Implémentation et discussion des résultats :**
Dans ce troisième chapitre, nous décrivons en détail les étapes d'implémentation de notre système IDS basé sur l'apprentissage profond, et nous discutons les résultats obtenus lors de nos expérimentations.

Et on termine notre travail avec une conclusion générale et travaux futurs.

CHAPITRE 2

APPRENTISSAGE PROFOND

Le monde de la technologie évolue à pas de géant, et le domaine de l'intelligence artificielle ("Artificial intelligence" (AI)) est en train de révolutionner la façon dont nous vivons et travaillons. Parmi les nombreux sous-domaines de l'AI, le Machine Learning (apprentissage automatique) et le Deep Learning (apprentissage profond) sont deux des plus excitants et des plus prometteurs. Le Machine Learning a déjà permis de grands progrès dans la reconnaissance vocale, la détection de fraudes [38], et bien plus encore. Le Deep Learning, quant à lui, est en train de révolutionner la reconnaissance d'images [28], la traduction de langues, et autres tâches qui étaient auparavant impossibles pour les ordinateurs. Dans ce chapitre, nous allons explorer en profondeur ces deux domaines incroyablement passionnants de l'AI, en examinant leur fonctionnement.

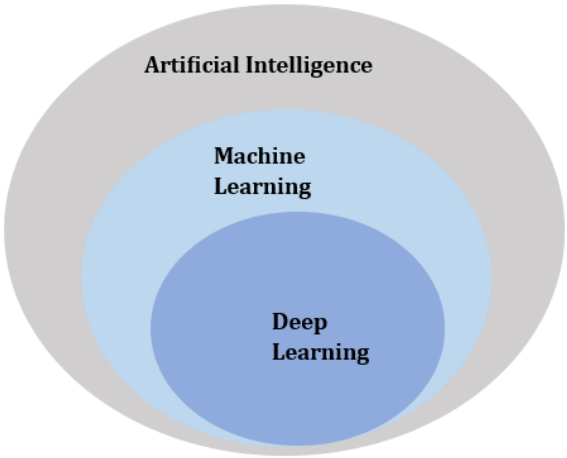

FIGURE 2.1: Artificial intelligence, Machine learning et Deep learning.

2.1 Apprentissage automatique

L'apprentissage automatique (Machine Learning) : est le processus par lequel les ordinateurs apprennent à partir de données et améliorent leur capacité à effectuer une tâche au fil du temps. Les algorithmes utilisés dans l'apprentissage automatique permettent aux logiciels de reconnaître des modèles, de faire des prédictions et de prendre des mesures en fonction des données. L'objectif de l'apprentissage automatique est de permettre aux ordinateurs d'apprendre et de s'adapter sans être explicitement programmés pour le faire [4].

2.1.1 Type d'apprentissage automatique

L'apprentissage automatique est souvent divisé en quatre types principaux : l'apprentissage supervisé, l'apprentissage non supervisé, l'apprentissage par renforcement et l'apprentissage semi-supervisé.

2.1.1.1 Apprentissage supervisé

Ce type est utilisé pour les tâches de classification et de régression, les données d'entrée sont étiquetées avec des résultats connus. L'algorithme apprend à partir de ces données étiquetées en tentant de prédire la sortie correcte pour de nouvelles données.

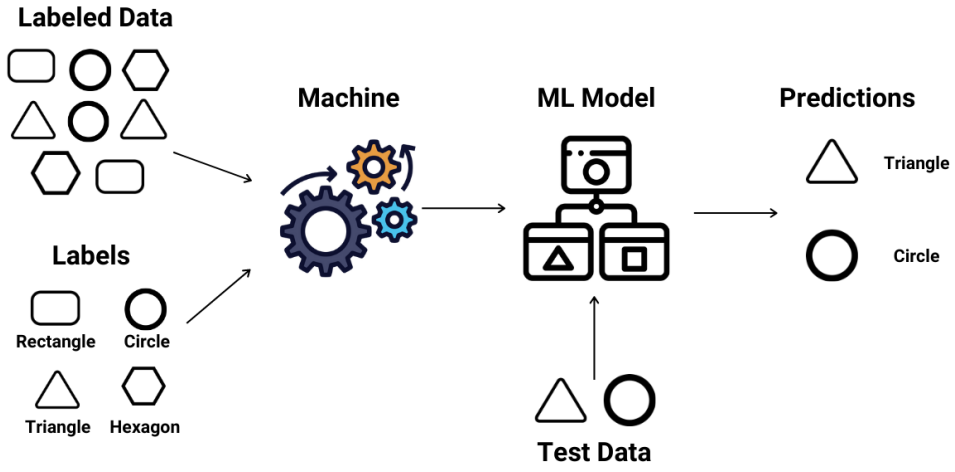

FIGURE 2.2: Apprentissage supervisé [29].

2.1.1.2 Apprentissage non supervisé

Ce type est utilisé dans des tâches telles que la segmentation d'image et la recherche de groupes de données similaires. Dans ce type d'apprentissage, les données d'entrée ne sont pas étiquetées. L'algorithme doit trouver des structures dans les données en regroupant les données similaires.

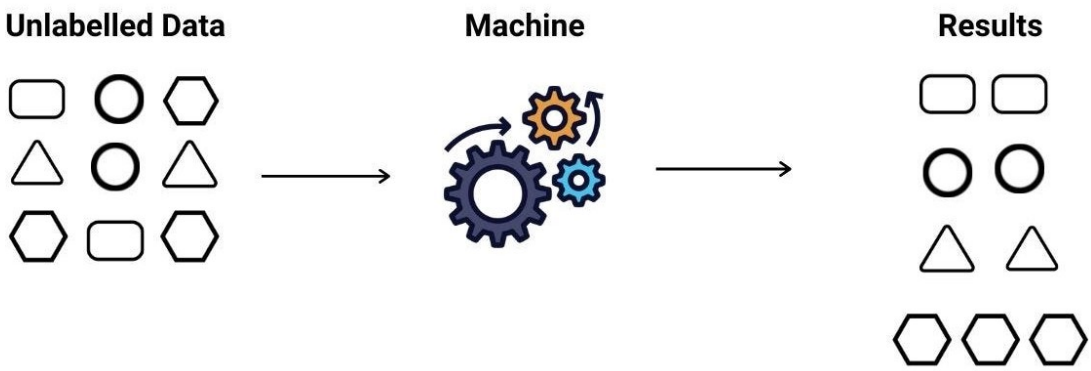

FIGURE 2.3: Apprentissage non supervisé [29].

2.1.1.3 Apprentissage semi supervisé

L'apprentissage semi-supervisé se réfère à l'apprentissage des étiquettes à partir d'un ensemble de données partiellement étiqueté. Cette approche présente plusieurs avantages. Tout d'abord, elle permet d'éviter d'étiqueter tous les exemples d'apprentissage, ce qui est particulièrement utile lorsque la collecte de données est facile mais que l'étiquetage nécessite un effort considérable. Prenons l'exemple de la classification d'images : il est souvent simple d'obtenir une grande quantité d'images, mais étiqueter chacune d'entre elles peut demander beaucoup de travail. De plus, les étiquettes fournies par des humains peuvent introduire des biais, qui peuvent ensuite être reproduits par un modèle entièrement supervisé. L'apprentissage semi-supervisé permet parfois de contourner cet obstacle [6].

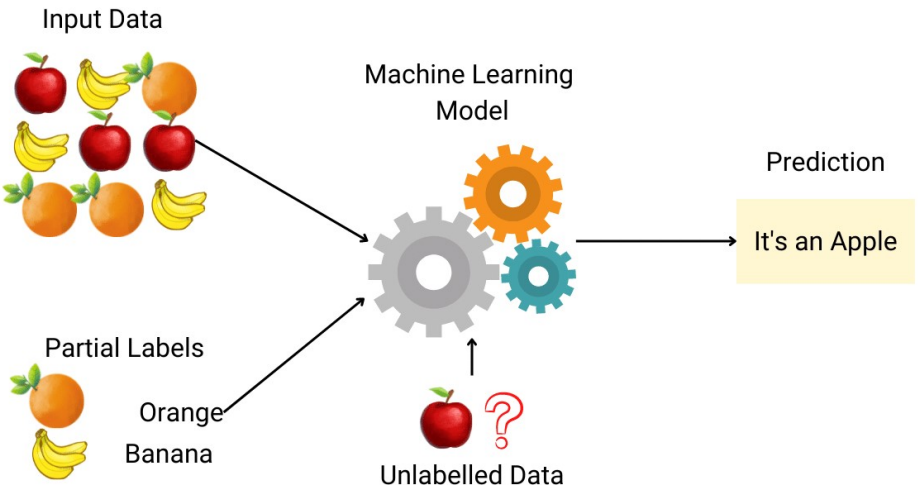

FIGURE 2.4: Apprentissage semi supervisé [29].

2.1.1.4 Apprentissage par renforcement

Dans le contexte de l'apprentissage par renforcement, un système d'apprentissage peut interagir avec son environnement et effectuer des actions. En fonction de ces actions, il reçoit une récompense, positive si l'action était judicieuse et négative sinon. La récompense peut parfois être accordée après une série prolongée d'actions. Ainsi, l'apprentissage consiste à définir une politique, c'est-à-dire une stratégie permettant d'obtenir systématiquement la meilleure récompense possible. Les domaines d'application principaux de l'apprentissage par renforcement se trouvent dans les jeux (e.g. les échecs) et la robotique [6].

2.1.2 Algorithmes d'apprentissage automatique

On va présenter quelques algorithmes d'apprentissage automatique :

Apprentissage automatique	
Apprentissage supervisé	Apprentissage non supervisé
Arbre de décision	K-means
Machines à vecteurs de support	Clustering
K-plus proches voisins	Réduction de dimensionnalité
Régression linéaire	Analyse en composantes (PCA)
Forêt aléatoire	Analyse de facteurs
Naïve Bayes	

TABLE 2.1: Algorithmes d'apprentissage automatique

2.2 Réseaux de neurones artificiels

Le réseau de neurones artificiels s'inspire du "réseau de neurones biologiques" (Figure2.5), en reproduisant son interconnexion de nœuds, analogues à des neurones. Chaque réseau de neurones est constitué de trois éléments clés : le caractère du nœud, la topologie du réseau et les règles d'apprentissage. Le caractère du nœud détermine la façon dont les signaux sont traités, notamment le nombre d'entrées et de sorties associées, les poids attribués à chaque entrée et sortie, ainsi que la fonction d'activation. La topologie du réseau détermine, quant à elle, l'organisation et les connexions entre les nœuds. Enfin, les règles d'apprentissage régissent l'initialisation et l'ajustement des poids du réseau [41].

2.2.1 Fonctions d'activation

Les réseaux de neurones artificiels utilisent des fonctions d'activation pour convertir les signaux d'entrée en signaux de sortie qui sont ensuite transmis à la couche suivante.
Pour obtenir la sortie de chaque couche, on calcule la somme pondérée des entrées, à laquelle on applique ensuite une fonction d'activation avant de la transmettre à la couche suivante [33].
On va présenter quelques fonctions :

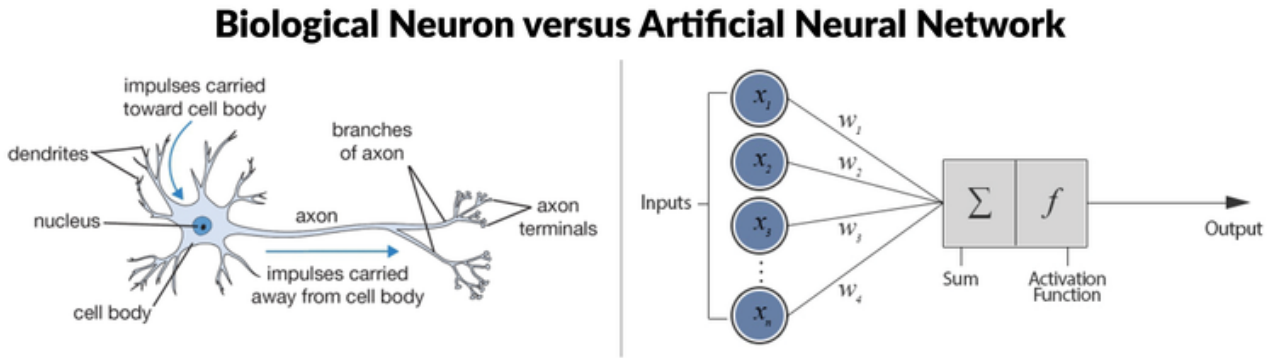

FIGURE 2.5: Neurone biologique (à gauche) et artificiel (à droite) [8].

— **Sigmoid** : Une couche d'activation saturée est une fonction qui prend un nombre réel en entrée et renvoie une sortie comprise entre [0,1] [17].

$$f(x) = \frac{1}{1+e^{-x}}$$

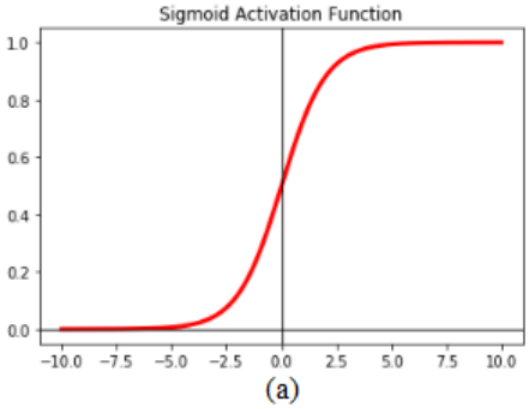

FIGURE 2.6: La fonction Sigmoid [17].

— **Tanh** : La fonction tangente hyperbolique est une couche d'activation saturée qui est couramment utilisée. Elle produit un nombre entre [-1, +1] [17].

$$f(x) = 2 \cdot \text{sigmoid}(2x) - 1 = \frac{1-e^{-2x}}{1+e^{-2x}}$$

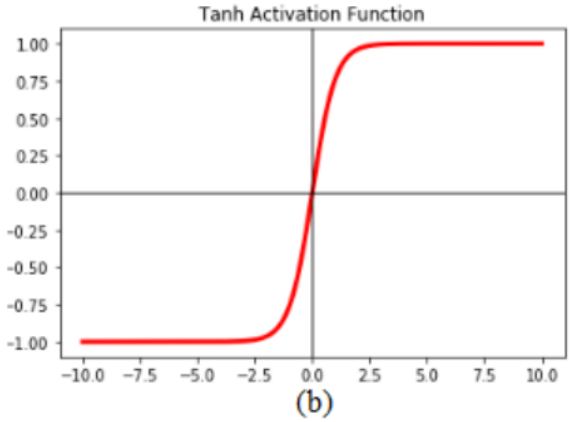

FIGURE 2.7: La fonction Tanh [17].

— **ReLU** : La couche d'activation linéaire rectifiée est une fonction d'activation non saturée qui est principalement utilisée pour supprimer toutes les valeurs négatives [17].

$$f(x) = \max(0,x) = \begin{cases} 0, & \text{si } x < 0 \\ x, & \text{si } x \geq 0 \end{cases}$$

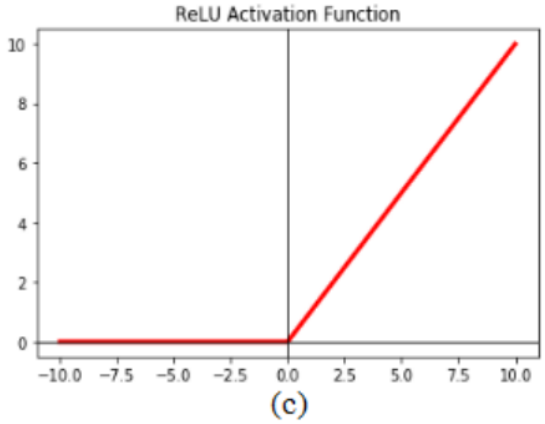

FIGURE 2.8: La fonction ReLU [17].

— **Softmax** : Softmax est une fonction qui se trouve généralement à la fin d'un réseau de neurone pour la classification multiclasse. Elle produit un vecteur de distribution de probabilité discret, assignant des probabilités à chaque classe possible.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

2.2.2 Descente de gradient

La descente de gradient est l'un des algorithmes les plus populaires pour effectuer l'optimisation et de loin le manière la plus courante d'optimiser les réseaux de neurones [31].
L'idée de base derrière la descente de gradient est de mettre à jour itérativement les poids du réseau de neurones en les déplaçant dans la direction du gradient négatif de la fonction d'erreur.

La mise à jour des poids se fait en utilisant la formule suivante :

$$\theta = \theta - \alpha \cdot \nabla J(\theta)$$

Où
θ : Le vecteur des paramètres du modèle (poids).
α : Learning rate qui va contrôler la taille du pas dans l'algorithme de descente de gradient.
$\nabla J(\theta)$: Le gradient de la fonction de coût J par rapport à θ .

2.3 Apprentissage profond

L'apprentissage profond est un domaine de l'apprentissage automatique qui vise à développer des algorithmes capables d'analyser et de comprendre des abstractions de données complexes, souvent au-delà des capacités des algorithmes d'apprentissage automatique traditionnels. Ces modèles avancés sont souvent influencés par divers domaines de connaissances, notamment les neurosciences, et imitent fréquemment la structure et le fonctionnement fondamentaux du système nerveux humain [7].

2.3.1 Réseaux de neurones profonds

Deep Neural Networks (DNN) sont un type de réseau de neurones artificiels (ANN) composé de plusieurs couches de nœuds interconnectés. Ces couches permettent aux DNN d'apprendre des représentations complexes de données en créant des représentations de caractéristiques de plus en plus abstraites et hiérarchiques à travers les couches.

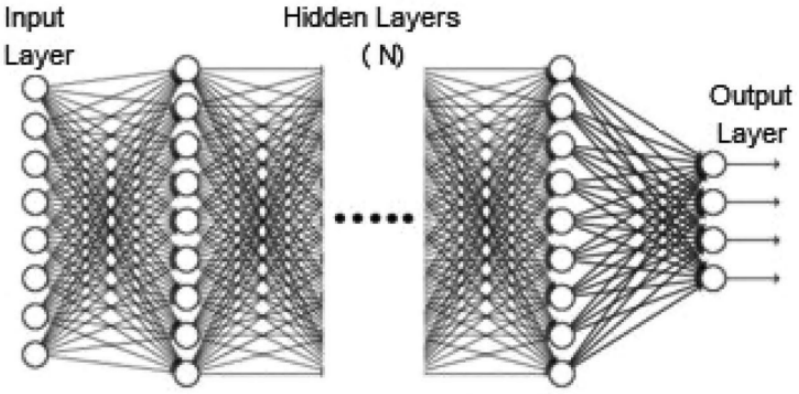

FIGURE 2.9: Architecture des réseaux de neurones profonds [32].

DNN contient plusieurs couches cachées (couche cachée = N et $N \geq 2$) en excluant la couche d'entrée et la couche de sortie [32].

2.3.2 Réseaux de neurones convolutifs

Convolution Neural Networks (CNN) est un type de réseau de neurones couramment utilisé principalement dans l'apprentissage en profondeur pour analyser l'imagerie visuelle.

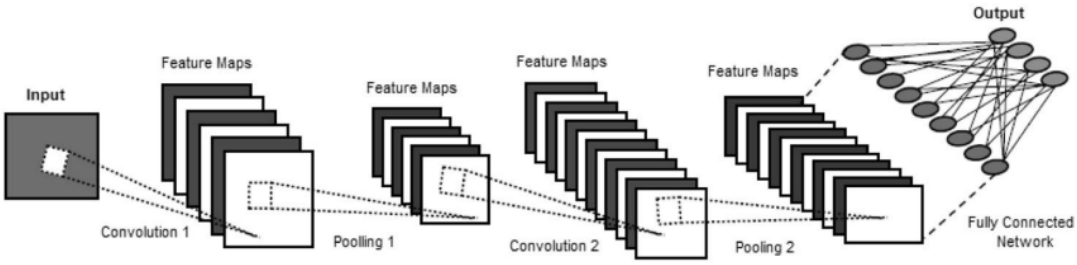

FIGURE 2.10: Architecture des réseaux de neurones convolutifs [32].

CNN est composé de trois types de couches principales : les couches de convolution, les couches de pooling et les couches entièrement connectées [15].

— **Les couches de convolution** sont responsables de l'extraction des caractéristiques à partir de l'image d'entrée. Elles appliquent des filtres sur l'image d'entrée pour produire une carte de caractéristiques. Le nombre de couches de convolution peut varier en fonction de la complexité de l'architecture du réseau.

— **Les couches de pooling** réduisent la taille de la carte de caractéristiques en agrégeant les caractéristiques les plus importantes. Le nombre de couches de pooling peut également varier en fonction de l'architecture du réseau.

— **Les couches entièrement connectées** sont responsables de la classification de l'image en utilisant les caractéristiques extraites par les couches de convolution et de pooling. Le nombre de couches entièrement connectées peut varier en fonction de la complexité du problème de classification.

2.3.3 Réseaux de neurones récurrents

Recurrent Neural Networks (RNN) sont un type de réseau de neurones qui est principalement utilisé pour détecter des modèles dans une séquence de données. Ces données peuvent être de l'écriture manuscrite, du texte ou des séries chronologiques numériques [20].
Le bloc de construction de base d'un RNN est l'unité récurrente, qui est une couche de réseau de neurones qui prend une entrée et un état caché de l'étape précédente, et produit une sortie et un nouvel état caché qui est transmis à l'étape suivante :

Où :

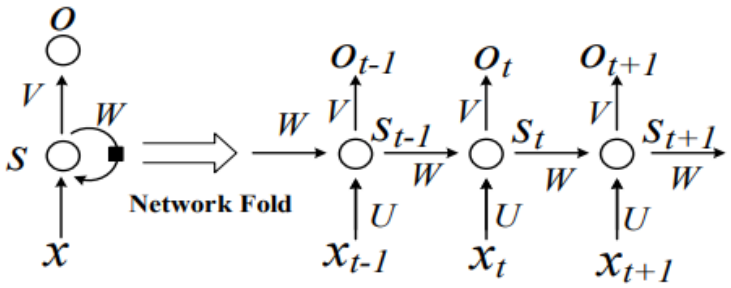

FIGURE 2.11: Architecture des réseaux de neurones récurrents [5].

x_t : est l'entrée à l'instant t.
S_t : est l'état caché à l'instant t.
L'état caché est calculé par :

$$S_t = f(Ux_t + WS_{t-1})$$

avec f une fonction d'activation et U et W des matrices de poids.

2.3.4 Mémoire à Long Terme et Court Terme

Long Short-Term Memory (LSTM) est un type du RNN qui permet de retenir des informations importantes sur longue période de temps (ou d'oublier des informations non importantes).
Pour ce faire, des cellules de mémoire sont introduites dans le modèle, lesquelles peuvent maintenir leur état sur une longue période. De plus, un ensemble de mécanismes de portes est utilisé pour contrôler le flux d'informations dans et hors des cellules, régulant ainsi le processus de rétention ou d'oubli.

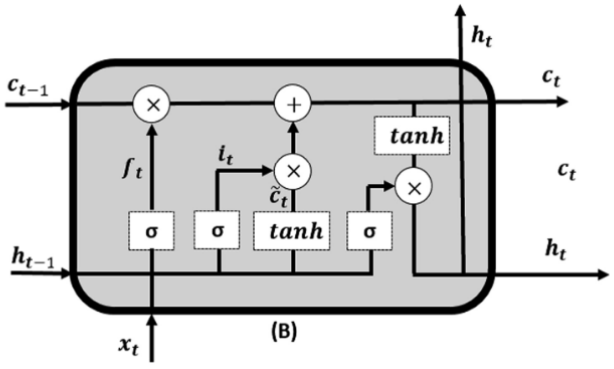

FIGURE 2.12: Architecture LSTM [5].

L'architecture générale de la cellule LSTM, qui se compose principalement d'une porte d'entrée, d'une porte de sortie et d'une porte d'oubli [5].

— La porte d'oubli permet de déterminer quelles informations sont importantes à conserver et quelles informations doivent être oubliées des unités de mémoire précédentes.

(2.1) $$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

— La porte d'entrée permet de contrôler les nouvelles informations qui sont intégrées aux unités de mémoire.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.2}$$

— La porte de sortie génère la nouvelle mémoire à long terme.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.3}$$

2.3.5 Unité récurrente à porte

Gated Recurrent Unit (GRU) est un type de RNN. Il est conçu pour résoudre certains des problèmes associés à LSTM, tels que sa grande complexité de calcul et la difficulté de former des réseaux plus profonds.

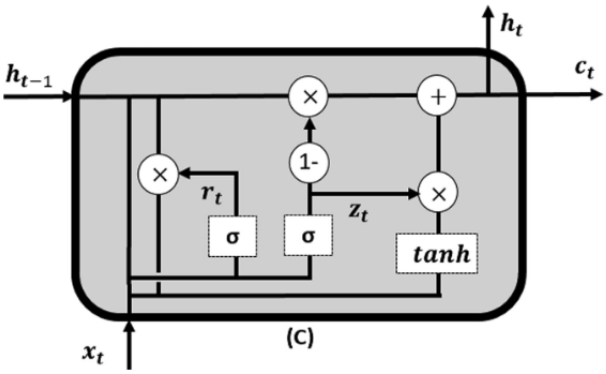

FIGURE 2.13: Architecture GRU [5].

En termes de fonctionnement, GRU et LSTM fonctionnent de manière similaire, mais la cellule GRU utilise un état caché qui fusionne la porte d'oubli et la porte d'entrée en une seule porte de mise à jour. De plus, GRU combine les états caché et cellulaire en un seul état. Les deux portes de GRU sont la porte de mise à jour et la porte de réinitialisation [5].

— La porte de réinitialisation détermine la quantité de l'état caché précédent qui doit être oubliée.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{2.4}$$

— La porte de mise à jour détermine la quantité de la nouvelle entrée qui doit être ajoutée à l'état caché actuel.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{2.5}$$

2.3.6 Auto encodeur

Un auto-encodeur est un type de réseau de neurones entraîné pour apprendre une représentation compressée (encodage) de données d'entrée, telles que des images ou du texte, de manière non supervisée, la représentation compressée est ensuite utilisée pour reconstruire les données d'entrée d'origine (décodage).

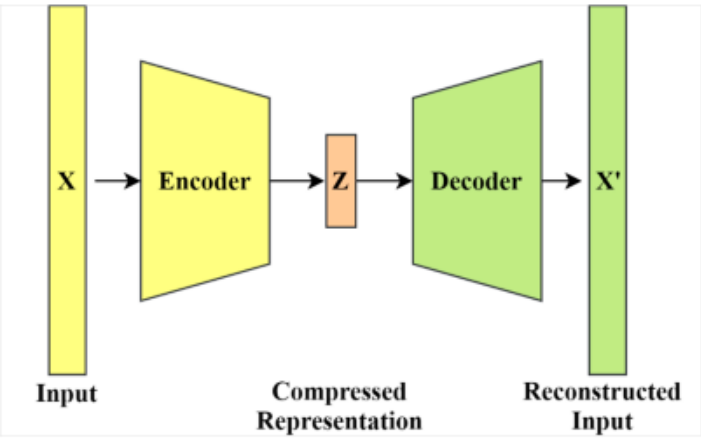

FIGURE 2.14: Architecture d'auto-encodeur [11].

Essentiellement, l'auto-encodeur apprend à compresser les données d'entrée dans un espace de dimension inférieure, puis à les reconstruire, en minimisant la différence entre les données d'entrée et de sortie.
Les auto-encodeurs peuvent être utilisés pour diverses applications telles que la réduction de dimensionnalité, la détection d'anomalies.

2.3.7 Réseaux antagonistes génératifs

Generative Adversarial Networks (GAN) sont conçus pour résoudre le problème de la modélisation générative. Le but d'un modèle génératif est d'étudier un ensemble d'exemples et d'apprendre la distribution de probabilité qui a permis de les générer. Bien que les modèles génératifs basés sur l'apprentissage en profondeur soient déjà populaires, les GAN sont considérés comme l'un des modèles les plus performants en termes de capacité à générer des images hautement réalistes et de grande qualité [14].

IL est constitué de deux composantes : le générateur et le discriminant. Le générateur prend un vecteur aléatoire en entrée et génère une image ou un échantillon de données, tandis que le discriminant prend en entrée l'image produite par le générateur ou une image réelle et tente de faire la distinction entre les deux.
Au cours de l'entraînement, le générateur est formé à produire des images trompeuses pour le discriminant, lui faisant croire qu'elles sont réelles. De son côté, le discriminant est entraîné à être capable de discerner les images réelles des images générées par le générateur.
Le processus de formation se poursuit jusqu'à ce que le générateur puisse générer des images convaincantes qui ne peuvent pas être différenciées des images réelles par le discriminant.

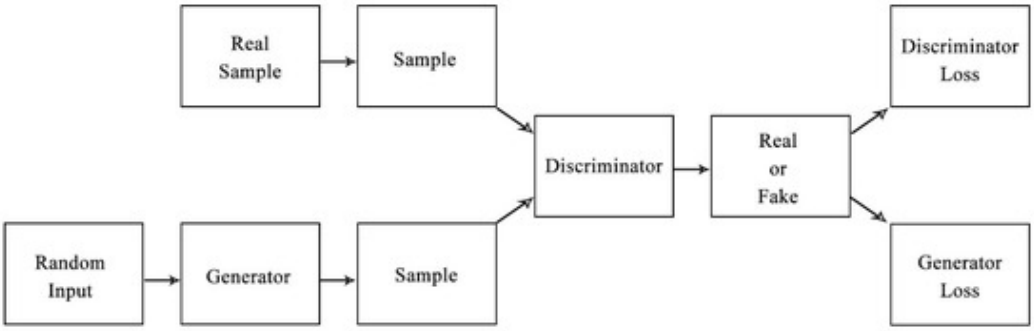

FIGURE 2.15: Architecture des réseaux antagonistes génératifs [21].

2.4 Techniques d'optimisation

L'optimisation d'un NN implique le réglage de ses paramètres pour minimiser une fonction de perte (Loss function), qui mesure la différence entre la sortie prévue et la sortie réelle. Une variété d'algorithmes d'optimisation sont utilisés dans les réseaux de neurones, notamment :

Algorithme	Description	Formule
Stochastic Gradient Descent (SGD)	Il est appliqué à un sous-ensemble sélectionné au hasard des données, ce qui peut aider à éviter de rester coincé dans les minima locaux.	$\theta = \theta - \alpha \cdot \nabla J(\theta; x(i); y(i))$
Momentum	Accélère SGD dans la direction appropriée et amortit les oscillations dans les directions perpendiculaires.	$v_t = \gamma v_{t-1} + \alpha \cdot \nabla J(\theta; x(i); y(i))$ $\theta = \theta - v_t$
Adagrad	Adapte le taux d'apprentissage aux paramètres, en réduisant le gradient pour les paramètres fréquemment mis à jour.	$\theta = \theta - \frac{\alpha}{\sqrt{G_t + \epsilon}} \cdot g_t$ G_t est la somme des carrés des gradients jusqu'au temps t
Adam	Combine les avantages des méthodes d'optimisation RMSprop et Momentum.	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$ $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$ $\hat{(m_t)} = \frac{m_t}{1-\beta_1^t}$; $\hat{(v_t)} = \frac{v_t}{1-\beta_2^t}$ $\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{(v_t)}} + \epsilon} \cdot \hat{(m_t)}$

TABLE 2.2: Algorithmes d'optimisation [31].

2.5 Techniques de régularisation

Les techniques de régularisation des neurones artificiels sont des méthodes utilisées pour prévenir le surapprentissage (ou overfitting en anglais) des réseaux de neurones artificiels.
Le surapprentissage se produit lorsqu'un réseau de neurones apprend trop bien à partir des données d'entraînement et perd sa capacité à généraliser pour des données nouvelles, ce qui réduit les performances du modèle sur des données inconnues. Voici quelques techniques courantes de régularisation :

2.5.1 Régularisation L1/L2

L1 et L2 sont des techniques utilisées pour empêcher le surajustement d'un modèle aux données d'apprentissage en ajoutant un terme de pénalité à la fonction de perte L(W).

— **L2** ajoute un terme qui encourage le modèle à avoir des poids petits mais non nuls.

$$L(W) = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{2.6}$$

$$L2 = \frac{\lambda}{2m}|W|_2^2 \tag{2.7}$$

Donc

$$\tilde{L}(W) = L(W) + \frac{\lambda}{2m}|W|_2^2 \tag{2.8}$$

où λ est le paramètre de régularisation.

— **L1** pénalise les coefficients du modèle qui sont trop grands.

$$L1 = \frac{\lambda}{m}|W|_1 \tag{2.9}$$

Donc

$$\tilde{L}(W) = L(W) + \frac{\lambda}{m}|W|_1 \tag{2.10}$$

2.5.2 Dropout

Dropout empêche le surajustement et fournit un moyen de combiner efficacement de manière exponentielle de nombreuses architectures de réseaux neuronaux différentes. Le terme dropout fait référence aux unités d'abandon (cachées et visibles) dans un réseau de neurones.
Par abandonner une unité, nous entendons la retirer temporairement du réseau, ainsi que toutes ses connexions entrantes et sortantes. Le choix des unités à abandonner est aléatoire [34], la figure 2.16 représente cette technique.

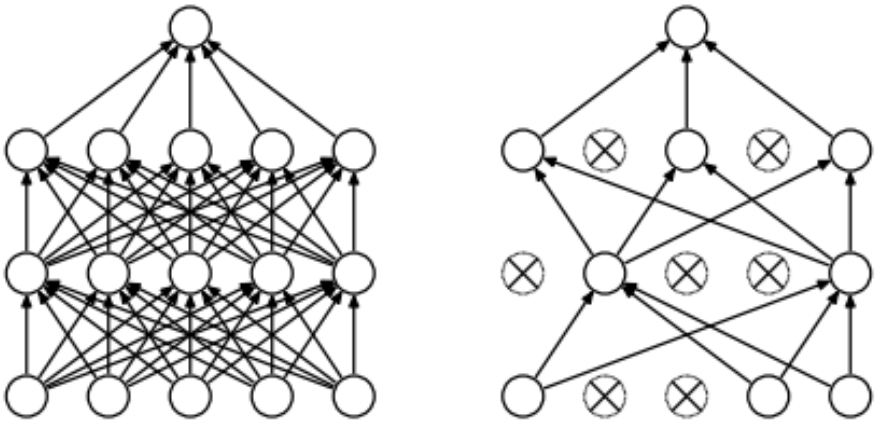

FIGURE 2.16: Technique dropout [34].

2.6 Conclusion

En conclusion, nous avons abordé dans ce chapitre les fondamentaux du ML, du DL et des réseaux de neurones artificiels. Nous avons vu les différentes architectures de ces réseaux, les fonctions d'activation les plus couramment utilisées et les techniques d'optimisation et régularisation.

CHAPITRE 3

RÉSEAUX INTERNET DES OBJETS

La sécurité des réseaux internet des objets est devenue un enjeu crucial du fait de la prolifération des appareils et de leurs nouveaux enjeux de sécurité. Des mesures de sécurité traditionnelles et nouvelles ont été suggérées, mais elles ont toutes des limites. Par conséquent nouvelles stratégies sont nécessaires pour sécuriser efficacement ces réseaux. Deep Learning s'est révélé très prometteur pour relever les défis de sécurité difficiles. L'objectif de ce chapitre est de discuter les différentes approches basées sur Deep Learning qui ont été suggérées pour améliorer la sécurité des réseaux IDO. Nous parlons des dangers liés à l'insécurité des réseaux, donnons un aperçu des précautions de sécurité et parlons des méthodes basées sur Deep Learning et des études de cas de leur utilisation.

3.1 Internet des objets

L'Internet Des Objets "IDO" (en anglais :Internet Of Things "IOT") est une infrastructure de réseau mondial dynamique qui permet une auto-configuration et une communication interopérable. En termes simples, cela fait référence à la capacité de connecter tout ce qui nous entoure à Internet, en commençant par (machines, appareils, téléphones portables et voitures) et en progressant vers (villes et routes) avec un comportement intelligent et en tenant compte de l'existence de l'autonomie et de la vie privée [3].

3.1.1 Architecture d'IDO

L'architecture de l'Internet Des Objets (IDO) est un cadre qui décrit les composants et l'infrastructure nécessaires à la création, au déploiement et à la gestion des systèmes IOT. L'architecture est composée de trois couches (Figure 3.4 représentant l'architecture de l'IOT) :

— **La couche de perception** : Elle est la couche la plus basse de l'architecture classique de l'IOT. Cette couche est composée de dispositifs, de capteurs, d'actionneurs et de contrôleurs. La tâche fondamentale de cette couche est de recueillir des informations précieuses à

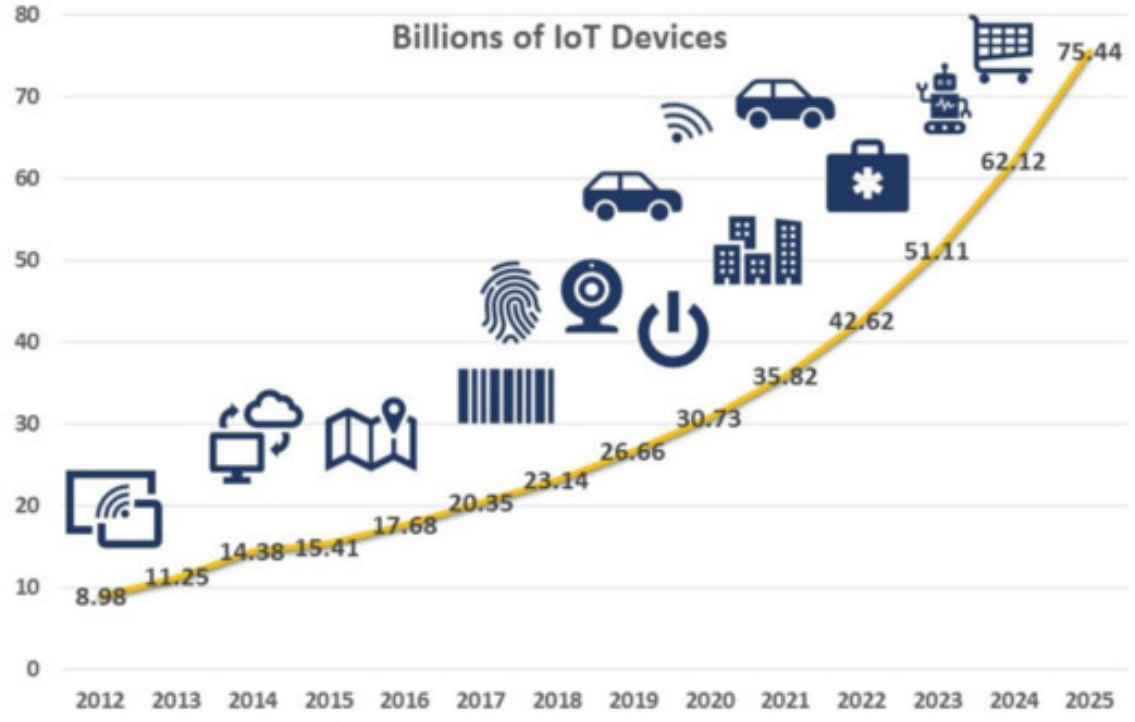

FIGURE 3.1: La croissance des dispositifs IoT au fil des années [37].

partir des systèmes de capteurs IOT [18].

— **La couche réseau** : Elle garantit la transmission réussie des données [18]. Grâce à cette couche, les données collectées par les capteurs et les dispositifs de la couche de perception peuvent être acheminées vers les applications de la couche d'application pour être traitées et visualisées. Elle garantit également que les données parviennent à leur destination finale de manière efficace, en évitant les pertes ou les retards.

— **La couche d'application** : La couche supérieure, elle traite les données pour leur visualisation. Cette couche est composée de différentes applications qui utilisent essentiellement les données fournies par les couches sous-jacentes [18].

3.2 Sécurité des Réseaux IDO

La sécurité des réseaux IDO est un processus de protection des réseaux IOT, des appareils et des données qu'ils hébergent contre les menaces et les intrusions. Cela implique de mettre en place des politiques et des technologies pour empêcher le vol de données, l'accès non autorisé et d'autres failles de sécurité susceptibles de compromettre la confidentialité, la fonctionnalité et la sécurité des systèmes IDO.

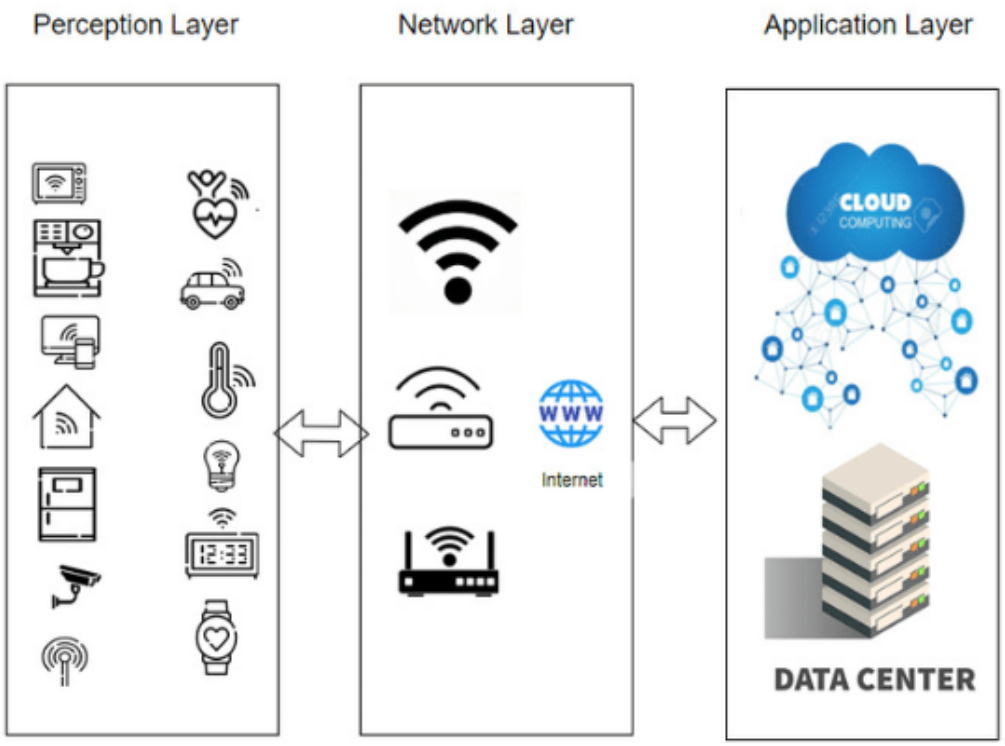

FIGURE 3.2: Architecture d'Internet Des Objets [18].

3.2.1 Les menaces de la sécurité dans les réseaux IDO

Les réseaux IOT peuvent être vulnérables à une variété d'attaques. Ces attaques prendre différentes formes, notamment :

3.2.1.1 Attaque par déni de service (DoS)

Une des attaques les plus courantes est l'attaque de Déni de Service (DoS). Elle se produit chaque fois qu'un attaquant inonde un serveur avec différents types de paquets, provoquant ainsi son arrêt en raison de la charge [30].

3.2.1.2 Attaque "Man in the Middle" (MitM)

MitM survient lorsque quelqu'un s'interpose dans la communication entre deux appareils sur un réseau IOT, et altère éventuellement les données qui y transitent. Cela permet à l'attaquant de capturer des informations confidentielles telles que des identifiants de connexion, qu'il pourra ensuite exploiter à des fins malveillantes.

3.2.1.3 Attaque de botnet

Botnet se produit lorsque des attaquants prennent le contrôle de volumes importants d'appareils IOT et les manipulent à diverses fins. L'objectif d'un Botnet est de prendre le contrôle à

distance et de diffuser des logiciels malveillants. Les criminels utilisent des botnets pour voler des informations privées, e.g. abuser des données bancaires en ligne et extraire de la crypto-monnaie [30].

3.2.1.4 Attaque par déni de service distribué (DDoS)

DDoS est une forme d'attaque qui ressemble à une attaque de déni de service (DoS), mais qui est effectuée à partir de plusieurs appareils simultanément. Cette technique rend la défense contre l'attaque beaucoup plus difficile, car elle implique l'utilisation de plusieurs sources d'attaques pour submerger une cible spécifique, telle qu'un site Web ou un serveur, avec un trafic de données considérable.

3.2.1.5 Usurpation d'identité

Usurpation d'identité (en anglais "Spoofing") est une technique utilisée par les attaquants pour falsifier ou manipuler l'information d'identification dans le but de se faire passer pour une autre entité légitime. Dans le contexte des réseaux IOT, l'attaque de spoofing peut être utilisée pour tromper les appareils IOT ou les systèmes de gestion en leur faisant croire qu'ils communiquent avec une source de confiance légitime, alors qu'en réalité, ils interagissent avec un attaquant.
Il existe plusieurs types d'attaques de spoofing, telles que les attaques de DNS, IP et MAC spoofing [30].

— **IP Spoofing** consiste à falsifier l'adresse IP source des paquets, donnant l'impression qu'ils proviennent d'un expéditeur différent. Cela leur permet d'usurper l'identité de quelqu'un en particulier ou même de simplement cacher leur propre identité [30].

— **MAC Spoofing** : l'attaquant modifier l'adresse MAC d'un appareil pour usurper l'identité d'un autre appareil connecté au réseaus.

— **DNS Spoofing** consiste à rediriger un utilisateur vers un faux site web qui imite l'original, dans le but de voler ses informations d'identification pour le site web. Cela peut être utilisé pour n'importe quel site web, des réseaux sociaux aux sites bancaires [30].

3.2.2 Vulnérabilités communes dans les réseaux IDO

3.2.2.1 Mots de passe

L'absence ou l'utilisation de mots de passe faibles et par défaut sont une menace importante pour la sécurité des dispositifs IOT, car ils peuvent être facilement devinés par des attaquants qui cherchent à accéder aux appareils ou aux réseaux auxquels ils sont connectés.
Pendant une période de 45 jours en 2021, le mot de passe le plus utilisé dans le monde était "admin",qui a été rencontré près de 21 millions de fois.Ainsi que les autres mots de passe comme "root","nc11","user","enable" ont été utilisé plus de trois millions de fois. Cette répétition de mots de passe peu sécurisés expose les utilisateurs à des risques d'attaques.

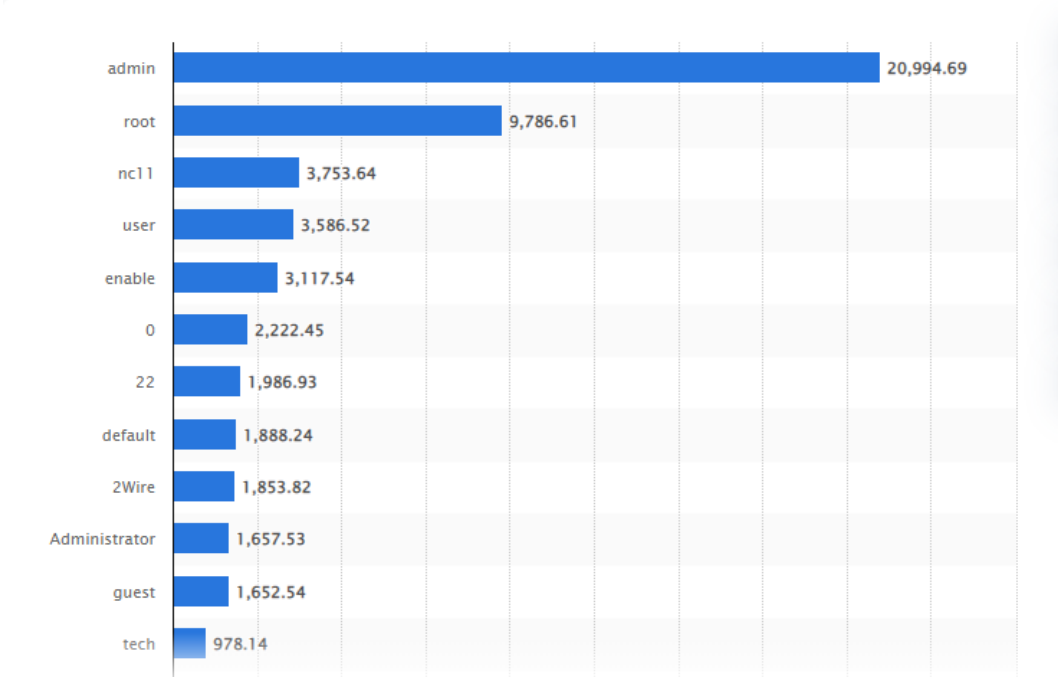

FIGURE 3.3: Mots de passe les plus utilisés dans les appareils d'IDO sur une période de 45 jours dans le monde en 2021 [35].

3.2.2.2 Absence des mises à jour

L'absence des mises à jour est une autre vulnérabilité courante dans les réseaux IOT. Les appareils IOT peuvent être vulnérables à des failles de sécurité qui peuvent être exploitées par des attaquants pour accéder à des données sensibles, perturber le fonctionnement des appareils ou même les prendre en charge. Les fabricants peuvent publier des correctifs de sécurité pour ces vulnérabilités sous forme de mises à jour de firmware ou de logiciels, mais si ces mises à jour ne sont pas installées, les appareils resteront vulnérables.

3.3 Mesures de sécurité existantes pour les réseaux IDO

3.3.1 Les mesures de sécurité traditionnelles

Les mesures de sécurité sont essentielles pour protéger les réseaux IOT contre les menaces. Parmi ces mesures, on peut citer l'authentification, la confidentialité et le contrôle d'accès.

3.3.1.1 Authentification

Les appareils IOT doivent pouvoir s'authentifier auprès du réseau et des autres appareils du réseau, pour garantir que seuls les appareils autorisés peuvent accéder au réseau. Ce processus implique la présentation d'informations d'identification, telles qu'un mot de passe, pour prouver l'identité.

3.3.1.2 Confidentialité

La confidentialité vise à protéger les données transmises et stockées contre l'accès non autorisé. En effet, les données transmises par les dispositifs IOT peuvent inclure des informations sensibles. On peut l'assurée par des techniques de chiffrement, telles que SSL/TLS ou le protocole IPsec, qui rendent les données illisibles pour les personnes non autorisées.

3.3.1.3 Contrôle d'accès

Le contrôle d'accès permet de restreindre l'accès aux ressources du réseau en fonction des autorisations accordées. Il utilise des mécanismes de contrôle d'accès tels que les listes de contrôle d'accès (ACL) ou les règles de pare-feu utilisées pour limiter l'accès à certaines parties du réseau en fonction de l'identité ou des autorisations d'un utilisateur.

3.4 Système de détection d'intrusion (IDS)

IDS est un élément essentiel de la sécurité informatique pour protéger les systèmes et les réseaux contre les attaques malveillantes et les intrusions. C'est un processus de surveillance des événements et de leur analyse pour identifier les signes d'intrusion ou d'accès non autorisé à un système informatique ou à un réseau [23].
Cette analyse peut se faire en temps réel ou en différé, et peut impliquer l'utilisation de techniques telles que l'analyse de journal, la surveillance du trafic réseau, la surveillance des fichiers et la surveillance du comportement des utilisateurs pour détecter les activités suspectes.

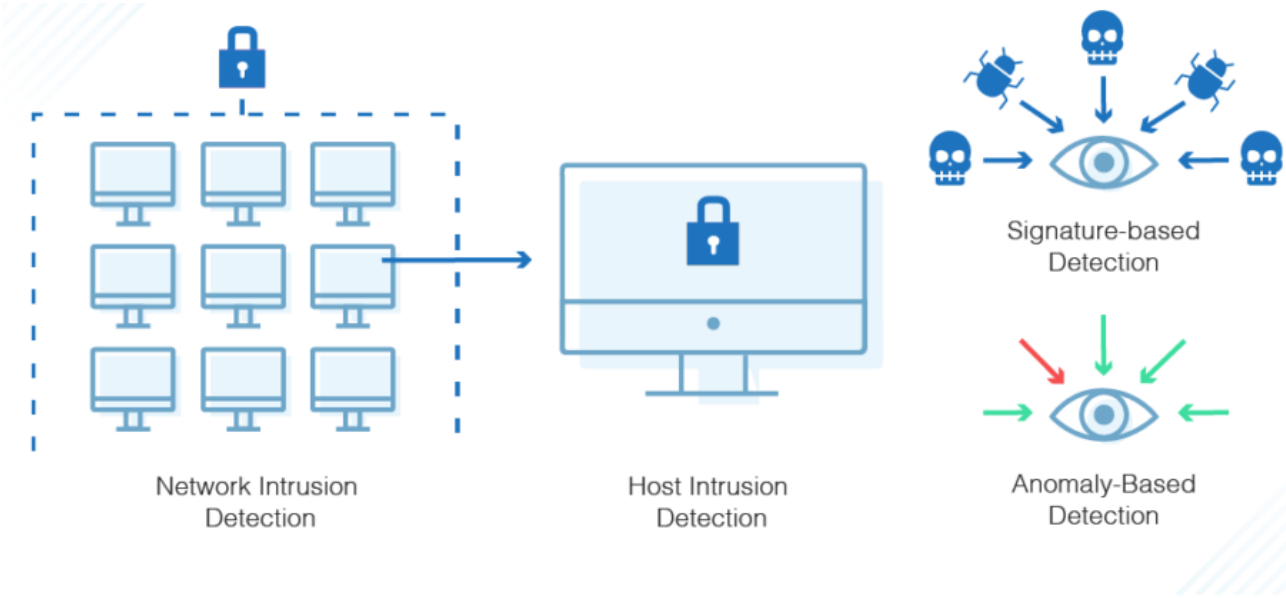

FIGURE 3.4: Système de détection d'intrusion [10].

3.4.1 Les méthodologies d'IDS

Les méthodologies de détection d'intrusion sont des approches utilisées pour identifier les activités suspectes dans un système informatique ou un réseau. On va présenter les deux catégories les plus utilisées :

3.4.1.1 IDS basé sur les signatures

Une signature fait référence à un modèle ou à une chaîne spécifique qui correspond à une attaque ou à une menace connue. La détection basée sur la signature (SD) est un processus de comparaison des modèles avec les événements capturés pour reconnaître les intrusions possibles. SD utilise les connaissances accumulées par des attaques spécifiques et des vulnérabilités du système pour identifier et faire correspondre des modèles d'attaque spécifiques, permettant une détection plus rapide et plus précise des attaques connues [23].

3.4.1.2 IDS basé sur les anomalies

IDS basé sur les anomalies est utilisé pour détecter les intrusions et superviser les activités d'utilisation abusive. Il différencie ces activités comme normales ou anormales grâce à l'utilisation d'un seuil. Dans le contexte de l'IOT, ces types d'IDS peuvent observer le comportement normal du réseau pour établir un seuil. Pour identifier les intrusions, le comportement du réseau est comparé au seuil, et tout écart par rapport à cette valeur est classé comme une anomalie [16].

3.4.2 Les types d'IDS

Il existe plusieurs types d'IDS utilisés pour surveiller les réseaux et détecter les intrusions, parmi lesquels les HIDS (Host-based Intrusion Detection System), les NIDS (Network-based Intrusion Detection System) et les hybrides sont les plus populaires.

3.4.2.1 Système de détection d'intrusion basé sur l'hôte (HIDS)

Un HIDS surveille et collecte des informations sur les hôtes individuels, tels que les serveurs qui exécutent des services publics ou les hôtes contenant des informations sensibles [23].
Il peut détecter des activités suspectes sur un hôte, comme des tentatives de connexion non autorisées, des modifications de fichiers système ou l'exécution de programmes malveillants.

3.4.2.2 Système de détection d'intrusion basé sur le réseau (NIDS)

Un NIDS capture le trafic réseau sur des segments de réseau spécifiques à l'aide de capteurs, puis analyse les activités des applications et des protocoles pour détecter les incidents suspects [23]. Il surveille le trafic réseau en temps réel pour détecter les anomalies, les comportements inhabituels et les activités potentiellement malveillantes.

3.4.2.3 Hybride

Les IDS hybrides sont conçus pour fournir une protection plus robuste contre les menaces de sécurité en utilisant une combinaison de plusieurs techniques de détection.

3.4.3 Problèmes liés à la détection des intrusions dans les réseaux IDO

La détection des intrusions dans les réseaux IDO peut être confrontée à plusieurs problèmes. Certains des problèmes les plus couramment rencontrés sont les suivants :

— **Manque de visibilité** : En raison de la complexité des réseaux IDO, il peut être difficile d'obtenir une visibilité complète sur l'ensemble du réseau et de détecter les activités malveillantes [24].

— **Architecture réseau complexe** : Les réseaux IDO sont souvent composés de nombreux appareils connectés de différentes natures, ce qui rend la détection des intrusions plus complexe en raison de la diversité des protocoles et des comportements de communication [24].

— **Journalisation limitée** : De nombreux appareils IDO ont des capacités de journalisation limitées, ce qui rend difficile la collecte d'informations détaillées sur les activités et les événements du réseau nécessaires pour détecter les intrusions [24].

— **Complexité et distribution des réseaux IDO** : ces réseaux peuvent être étendus et distribués géographiquement, ce qui complique la détection des intrusions et le suivi des incidents de sécurité à travers l'ensemble du réseau [24].

3.5 Détection d'anomalies

La détection d'anomalies est une tâche essentielle dans l'analyse de données, visant à identifier des données qui se démarquent de manière significative ou inhabituelle au sein d'un ensemble de données donné [1].
Dans un réseau, la détection d'anomalie se concentre sur l'identification des comportements anormaux ou suspects dans le trafic réseau, ce qui peut indiquer une activité malveillante ou une panne système.

3.5.1 Les techniques de détection d'anomalies

Les techniques les plus couramment utilisées pour la détection des anomalies du réseau appartiennent aux catégories des statistiques, du regroupement (Clustering) et de l'apprentissage automatique (Machine Learning) [13].

3.5.1.1 Techniques statistiques

Technique qui se base sur l'analyse statistique des données d'un réseau pour identifier les anomalies. Elle utilise des modèles de distribution de probabilité pour identifier les valeurs qui s'écartent significativement de la normale et les considère comme des anomalies potentielles.

3.5.1.2 Technique d'apprentissage automatique

Les méthodes d'apprentissage automatique utilisent des algorithmes pour apprendre à partir de données sans avoir besoin de programmation explicite pour chaque tâche. Elles se divisent en

deux catégories principales : supervisées et non supervisées (Clustering).

1. **Clustering** : Le clustering est une technique d'apprentissage non supervisé qui vise à regrouper des données similaires sans nécessiter des données pré-étiquetées. Il existe plusieurs types de techniques de clustering, mais dans le contexte de la détection d'anomalies de réseau, on utilise généralement le clustering régulier et le co-clustering [1].

   — Le clustering régulier, tel que l'algorithme k-means, regroupe les données en considérant uniquement les lignes de l'ensemble de données, c'est-à-dire les instances de données.

   — Le co-clustering considère simultanément les lignes et les colonnes de l'ensemble de données, c'est-à-dire les instances de données et les caractéristiques ou attributs de ces données.

2. **Classification** :
   Classifier les instances de données en deux catégories : normales ou anormales.
   Il existe plusieurs algorithmes de classification supervisée qui peuvent être utilisés pour la détection d'anomalies dans un réseau, tels que les arbres de décision, les réseaux de neurones, les SVM (Support Vector Machines) et les algorithmes basés sur les k-plus proches voisins [27].

3.6 Rôle du deep learning dans la sécurité des réseaux IDO

En raison de la quantité et de la complexité des données générées par les réseaux IDO, deep learning est devenu un élément essentiel pour assurer la sécurité de ces réseaux. Les réseaux de neurones profonds peuvent être utilisés pour détecter les anomalies et les comportements malveillants en apprenant à partir des données massives. De plus, ces modèles peuvent être entraînés en temps réel pour s'adapter rapidement aux nouvelles menaces et fournir une protection en temps réel contre les attaques.

3.7 Approches basées sur deep learning pour la sécurité des réseaux IDO

3.7.1 Détection des intrusions

L'amélioration de la détection des intrusions grâce au Deep Learning est un sujet de recherche très actif dans le domaine de la sécurité des réseaux. Le Deep Learning peut être utilisé pour améliorer la capacité de l'IDS à détecter les menaces et les attaques en temps réel en apprenant à partir des données massives et complexes des réseaux IoT. Les modèles de deep learning peuvent être entraînés pour reconnaître les schémas de trafic normaux et pour détecter les anomalies et les comportements malveillants en temps réel.

On va présenter dans un tableau (Tableau 3.1) quelques recherches avec différents modèles Deep Learning utilisé pour IDS avec les résultats obtenus :

Référence	Modèle	Dataset	Résultat
Xu et al. [39]	GRU	NSL-KDD	Recall : 99.31%, False Positive Rate : 0.84%
		KDD CUP 99	Recall : 99.42%, False Positive Rate : 0.05%
Yousefi-Azar et al.[40]	AE	NSL-KDD	Accuracy : 83.34%
Liang et al. [22]	DNN	NSL-KDD	Accuracy : 97%
Nsunza et al. [26]	CNN	NSL-KDD	Accuracy : 82.83%
Diro et al. [9]	LSTM	ISCX	Accuracy : 99.91%
		AWID	Accuracy : 98.22%
Malik et al.[25]	LSTM-CNN	CICIDS2017	Accuracy : 98.60%
Ferrag et al.[12]	RNN	Bot-IOT	Accuracy : 99.91%
		CICIDS2017	Accuracy : 99.81%
		Power System	Accuracy : 99.82%

TABLE 3.1: Méthodes Deep Learning utilisées pour détection des intrusions

3.8 Conclusion

Ce chapitre présente l'utilisation du Deep Learning dans la sécurité des réseaux IDO. Il commence par introduire les défis de la sécurité et la nécessité d'utiliser des techniques avancées telles que le Deep Learning pour y faire face. Ensuite, il explique en détail comment le Deep Learning peut être appliqué pour la détection d'intrusion dans les réseaux IDO. Le chapitre se termine par une discussion sur les travaux de recherche récents sur l'utilisation du Deep Learning pour la sécurité des réseaux IDO.

CHAPITRE 4

IMPLÉMENTATION ET DISCUSSION DES RÉSULTATS

Dans le chapitre précédent, nous avons exploré les différents aspects des systèmes de détection d'intrusion (IDS) et les problèmes qui leur sont associés dans les réseaux IDO. Nous avons également discuté de l'utilisation du deep learning comme une approche prometteuse pour améliorer les performances des IDS. Dans ce chapitre, nous allons nous concentré sur l'implémentation de notre système IDS basé sur le deep learning et sur la discussion des résultats obtenus. Notre objectif principal est de développer un modèle efficace de détection d'intrusion qui peut identifier les activités malveillantes dans les réseaux IDO en utilisant des modèles RNN.
Pour atteindre cet objectif, nous avons choisi d'utiliser le BoT-IoT Dataset. Dans ce qui suit, nous présenterons en détail la méthodologie que nous avons suivie pour implémenter notre système IDS basé sur le Deep Learning.

4.1 Dataset

Pour notre étude sur la détection d'intrusion dans les réseaux IOT, nous avons choisi le BoT-IoT dataset en raison de sa pertinence et de sa représentativité des attaques courantes dans ce contexte.

4.1.1 Présentation du dataset

BoT-IoT [1] a été créé en simulant un environnement réseau réaliste au sein du laboratoire Cyber Range de l'UNSW Canberra. Cet environnement est caractérisé par une combinaison de trafic normal et de trafic de botnet.

1. https://research.unsw.edu.au/projects/bot-iot-dataset

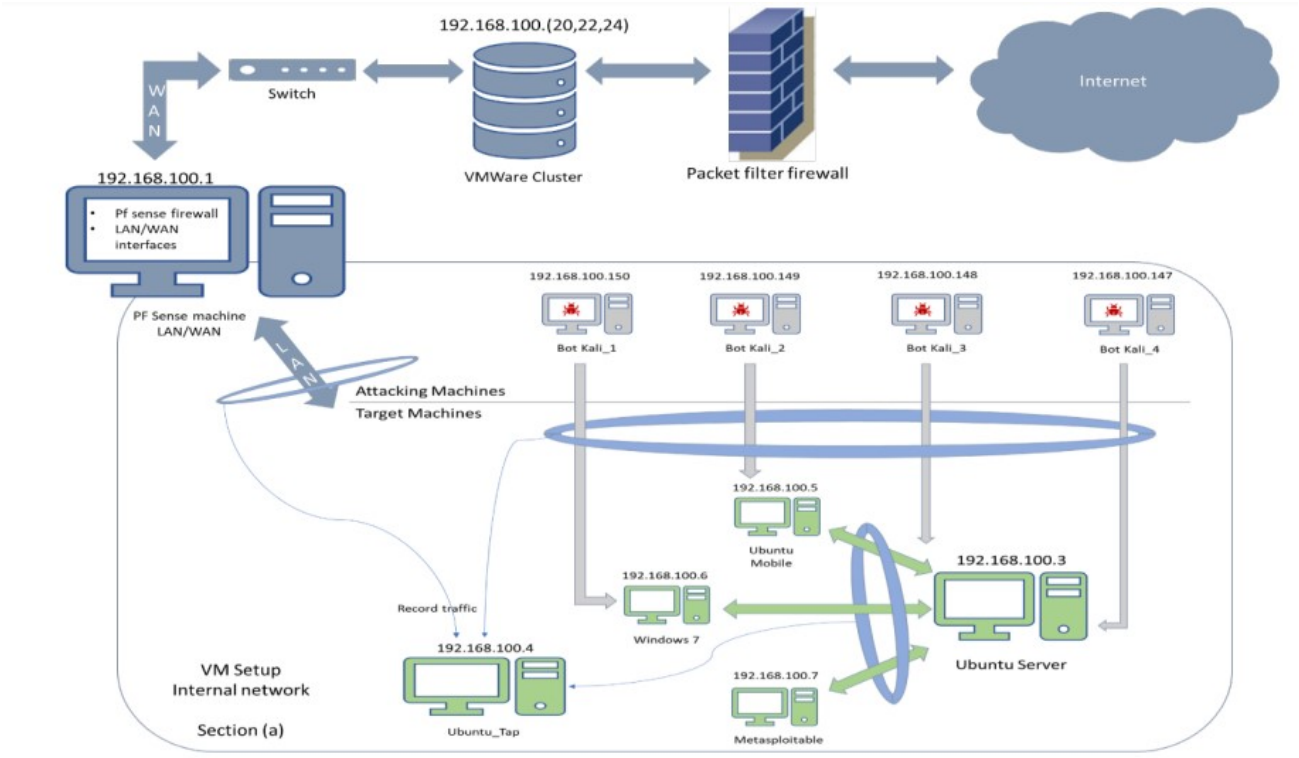

FIGURE 4.1: Environnement de test d'ensemble de données BoT-IoT [19].

Le dataset est fourni avec des fichiers pcap, argus et CSV, et été divisé en catégories et sous-catégories pour faciliter l'étiquetage. Il est volumineux, mais une version réduite de 5% a été extraite pour faciliter son utilisation [19].

L'approche choisie implique des techniques telles que le coefficient de corrélation et l'entropie pour identifier les caractéristiques hautement corrélées et réduire la dimensionnalité de l'ensemble de données.

Ces deux dernières techniques ont permis d'identifier les caractéristiques les plus significatives. En utilisant ces techniques, (Nickolaos Koroniotis et Nour Moustafa) [19] visaient à extraire les principales caractéristiques afin d'obtenir des performances élevées. Ils ont conservé uniquement 10 caractéristiques qui seront utilisées dans le modèle final [19], avec la classe "category" catégorie pour permettre la classification des données.

	Description
seq	Numéro de séquence Argus.
stddev	Écart type des enregistrements agrégés.
N_IN_Conn_P_SrcIP	Nombre de connexions entrantes par adresse IP de source.
min	Durée minimale des enregistrements agrégés.
state number	Représentation numérique de l'état d'une caractéristique.
mean	Durée moyenne des enregistrements agrégés.
N_IN_Conn_P_DstIP	Nombre de connexions entrantes par adresse IP de destination.
drate	Paquets destination à source par seconde.
srate	Paquets source à destination par seconde.
max	Durée maximale des enregistrements agrégés.
category	Catégories des attaques.

TABLE 4.1: Description des 10 principales caractéristiques avec la classe "category"

4.1.2 Catégories des attaques

Le jeu de données BoT-IoT propose une classification en quatre catégories d'attaques, à savoir : DDoS, DoS, theft et reconnaissance.

— **Attaque par déni de service (DoS)** : Cette attaque vise à rendre un dispositif ou un service IOT spécifique indisponible en le submergeant de trafic ou en exploitant une vulnérabilité spécifique.

— **Attaque par déni de service distribué (DDoS)** : Cette attaque a pour but de submerger un réseau IOT avec un trafic malveillant provenant de multiples sources.

— **Theft (Vol de données)** : Cette attaque consiste à accéder et extraire des données sensibles à partir des dispositifs IOT ou du réseau lui-même.

— **Reconnaissance** : Dans cette attaque, les attaquants tentent de collecter des informations sur les dispositifs IOT, comme des adresses IP, des ports ouverts, etc. Cette phase de reconnaissance permet aux attaquants de mieux comprendre le réseau IOT cible et de planifier des attaques ultérieures de manière plus précise.

Nombre de catégories (classe "category") et le nombre de chaque attaque dans catégories est donné par le tableau 4.2 :

Categories	**Nombre**
DDoS	1,926,624
DoS	1,650,260
Reconnaissance	91,082
Normal	477
Theft	79

TABLE 4.2: Nombre de catégories

4.2 Prétraitement des données

Nous avons effectué une exploration des données en chargeant les deux fichiers CSV 'train.csv' et 'test.csv'. Une représentation succincte de ces deux fichiers est la suivante :

— **Données d'entraînement** : Le fichier "train.csv" contient un total de 2 934 817 lignes et 19 colonnes.

	pkSeqID	proto	saddr	sport	daddr	dport	seq	stddev	N_IN_Conn_P_SrcIP	min
0	3142762	udp	192.168.100.150	6551	192.168.100.3	80	251984	1.900363	100	0.00000
1	2432264	tcp	192.168.100.150	5532	192.168.100.3	80	256724	0.078003	38	3.85693
2	1976315	tcp	192.168.100.147	27165	192.168.100.3	80	62921	0.268666	100	2.97410
3	1240757	udp	192.168.100.150	48719	192.168.100.3	80	99168	1.823185	63	0.00000
4	3257991	udp	192.168.100.147	22461	192.168.100.3	80	105063	0.822418	100	2.97999
...	...	...	...	...	...	...	...	...	...	...
2934812	1132803	udp	192.168.100.149	56044	192.168.100.5	80	253370	0.016992	100	4.08250
2934813	3384621	udp	192.168.100.150	21546	192.168.100.3	80	231693	1.922317	100	0.00000
2934814	775893	udp	192.168.100.149	30897	192.168.100.5	80	158616	2.112228	100	0.00000
2934815	443484	tcp	192.168.100.147	36904	192.168.100.7	80	179855	0.000000	100	0.00000
2934816	96906	tcp	192.168.100.150	14302	192.168.100.3	80	95429	0.053820	100	0.06476

2934817 rows × 19 columns

FIGURE 4.2: Données d'entraînement.

— **Données de test** : Le fichier "test.csv" contient un total de 733705 lignes et 19 colonnes.

	pkSeqID	proto	saddr	sport	daddr	dport	seq	stddev	N_IN_Conn_P_SrcIP	min
0	792371	udp	192.168.100.150	48516	192.168.100.3	80	175094	0.226784	100	4.100436
1	2056418	tcp	192.168.100.148	22267	192.168.100.3	80	143024	0.451998	100	3.439257
2	2795650	udp	192.168.100.149	28629	192.168.100.3	80	167033	1.931553	73	0.000000
3	2118009	tcp	192.168.100.148	42142	192.168.100.3	80	204615	0.428798	56	3.271411
4	303688	tcp	192.168.100.149	1645	192.168.100.5	80	40058	2.058381	100	0.000000
...	...	...	...	...	...	...	...	...	...	...
733700	1571905	udp	192.168.100.148	17412	192.168.100.6	80	168162	1.743940	39	0.000000
733701	2787099	udp	192.168.100.147	932	192.168.100.3	80	158482	0.694618	65	3.002272
733702	2255382	tcp	192.168.100.149	47980	192.168.100.3	80	79841	1.744651	53	0.000000
733703	588946	tcp	192.168.100.147	25096	192.168.100.7	80	63165	0.000000	100	0.000000
733704	2577420	tcp	192.168.100.147	34718	192.168.100.3	80	139733	0.091420	65	0.000000

733705 rows × 19 columns

FIGURE 4.3: Données de test.

Avant de commencer l'entraînement du modèle, nous avons effectué plusieurs étapes de prétraitement sur les données.

Tout d'abord,

— Nous avons éliminé certaines colonnes et conservé uniquement les 10 meilleures colonnes sélectionnées précédemment.

— Ensuite, nous avons vérifié s'il y a des valeurs dupliquées. On a trouvé 13 valeurs en double dans les données d'entraînement. On les a supprimés pour ne pas influencer de manière négative les résultats.

```
In [23]: print("les valeurs dupliquées dans les données d'entraînement sont:",traindata.duplicated().sum())

les valeurs dupliquées dans les données d'entraînement sont: 13
```

FIGURE 4.4: Nombre de valeurs dupliquées.

```
In [22]: traindata.drop_duplicates(inplace=True)
```

FIGURE 4.5: Suppression des valeurs dupliquées.

Après les deux étapes précédentes de prétraitement, l'ensemble de données d'entraînement compte maintenant **2934804 lignes, 11 colonnes**.

De même, l'ensemble de données de test compte **733705 lignes, 11 colonnes**.

Après avoir analysé la distribution des attaques dans le dataset que nous avons gardé (ensembles données d'entraînement et de test), nous avons constaté les répartitions présentées dans le tableau 4.3 et illustrées graphiquement par la figure 4.6.

Categories	**Train (%)**	**Test (%)**
DDoS	52.518	52.516
DoS	44.982	44.992
Reconnaissance	2.484	2.476
Normal	0.013	0.015
Theft	0.002	0.002

TABLE 4.3: Distribution des attaques dans l'ensemble d'entraînement et de test

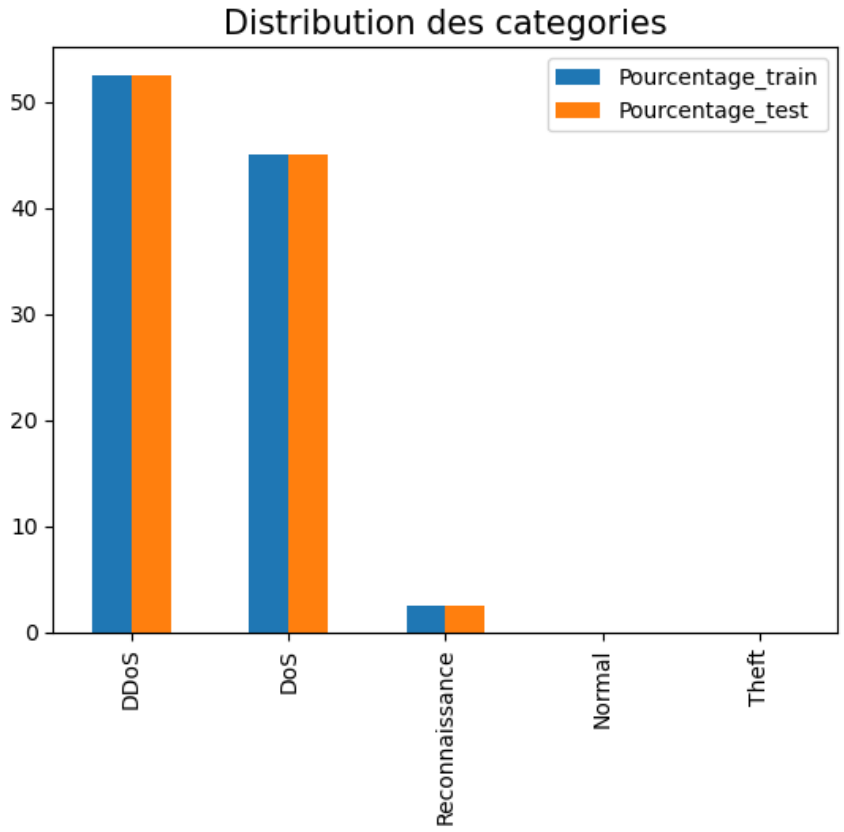

FIGURE 4.6: Représentation graphique de la distribution des attaques dans l'ensemble d'entraînement et de test.

Comme vous pouvez constater que le nombre des attaques DDoS et DoS sont les plus prédominantes dans notre jeu de données, représentant respectivement **52,518%** et **44,982%** des données d'entraînement, ainsi que **52,516%** et **44,992%** des données de test.
Cela indique que notre modèle de détection d'intrusion sera principalement axé sur la détection de ces types d'attaques, compte tenu de leur prévalence plus élevée dans les ensembles de données.

4.2.1 Normalisation

Pour accélérer les calculs, diminuer le nombre d'étapes d'entraînement et éviter les valeurs dominantes, il est nécessaire de normaliser nos données. À cet effet, nous avons utilisé la méthode de normalisation min-max, qui redimensionne les valeurs de chaque attribut dans un intervalle spécifique entre [0,1]. Cela garantit que toutes les caractéristiques sont à la même échelle.

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{4.1}$$

Dans les données d'entraînement et de test, les 10 colonnes considérées pour la normalisation sont : ['seq', 'stddev', 'N_IN_Conn_P_SrcIP', 'min', 'state_number', 'mean', 'N_IN_Conn_P_DstIP', 'drate', 'srate', 'max'].

4.2.2 Encodage

Pour classifier des données en utilisant les réseaux de neurones, on doit encoder nos classes dans des nombres naturels : à partir de 0 et jusqu'au nombre de classes moins un. Nous avons utilisé la méthode du Label Encoding pour encoder la colonne "category". Cette technique attribue à chaque catégorie un entier unique de manière séquentielle.

Par exemple :
Les catégories "DDoS", "DoS", "Reconnaissance", "Normal" et "Theft" ont été respectivement codées avec les entiers 0, 1, 3, 2 et 4.

Pour valider les entraînements, l'ensemble des données d'entraînement est subdivisé en deux répartitions : une répartition de 80% pour l'entraînement et 20% pour la validation.

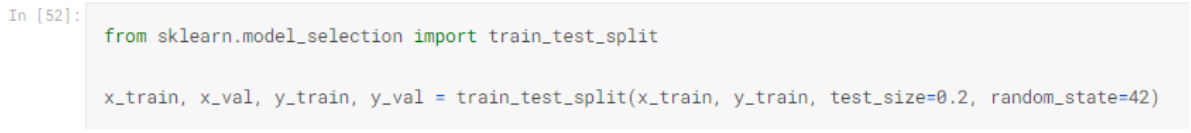

FIGURE 4.7: Division de l'ensemble d'entraînement en un ensemble d'entraînement et un ensemble de validation.

4.3 Modèles

Dans cette partie, nous allons présenter trois modèles qui nous avons élaborés pour la détection d'intrusion dans les réseaux IDO, qui sont basés sur : LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit) et LSTM-GRU (une combinaison de LSTM et GRU).

4.3.1 Architecture des modèles

4.3.1.1 Premier modèle à base de LSTM

Nous avons choisi les LSTM (Long Short-Term Memory) pour la détection d'intrusion dans les réseaux IDO, car ils ont spécialement conçu pour capturer les dépendances à long terme dans les séquences de données.
Le modèle comprend trois couches LSTM, LSTM_1, LSTM_2 avec 128, 64, 32 unités respectivement, qui permettent de capturer les motifs séquentiels et les relations complexes entre les données, en utilisant la fonction 'ReLU'.

Les couches LSTM sont intercalées avec des couches de dropout, qui aident à régulariser le modèle et à prévenir le surapprentissage. Enfin, le modèle se termine par une couche dense qui effectue la classification des attaques avec la fonction 'Softmax' pour une classification multi-classe.

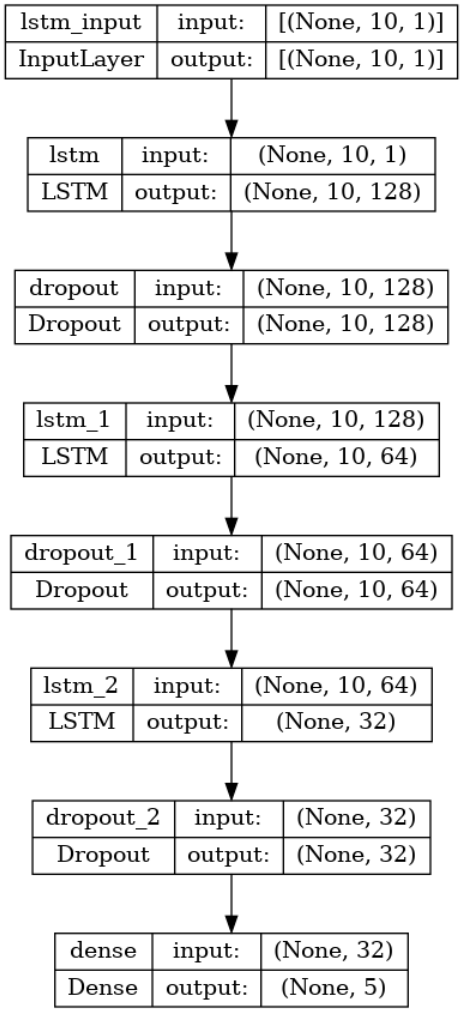

FIGURE 4.8: Architecture du modèle LSTM.

4.3.1.2 Deuxième modèle à base de GRU

GRU (Gated Recurrent Unit) est un autre modèle que nous avons utilisé pour la détection d'intrusion dans les réseaux IDO.
Il présente une architecture similaire au modèle LSTM, avec trois couches GRU, GRU_1, GRU_2 contenant respectivement 128, 64 et 32 unités.

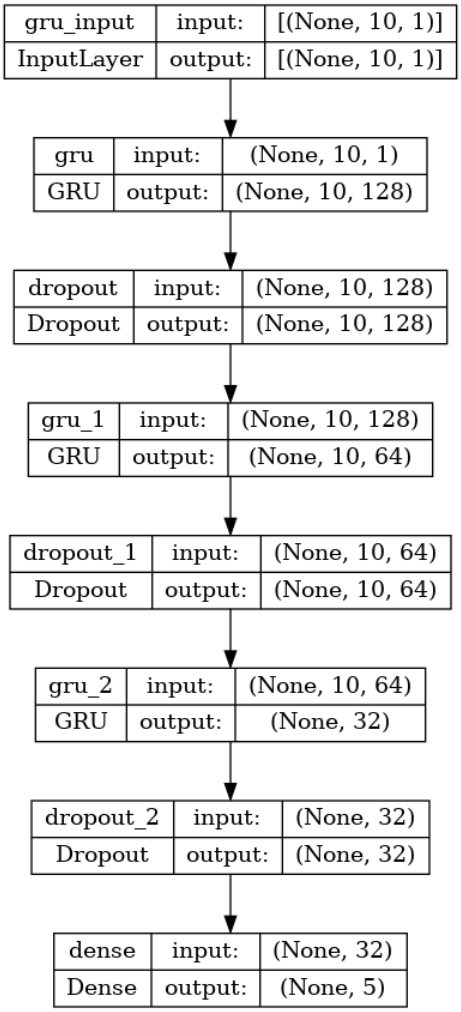

FIGURE 4.9: Architecture du modèle GRU.

4.3.1.3 Troisième modèle à base de LSTM et GRU

LSTM-GRU est une combinaison des modèles LSTM et GRU. Cette architecture hybride contient une couche LSTM avec 128 unités et 2 couches GRU avec 64, 32 unités respectivement.

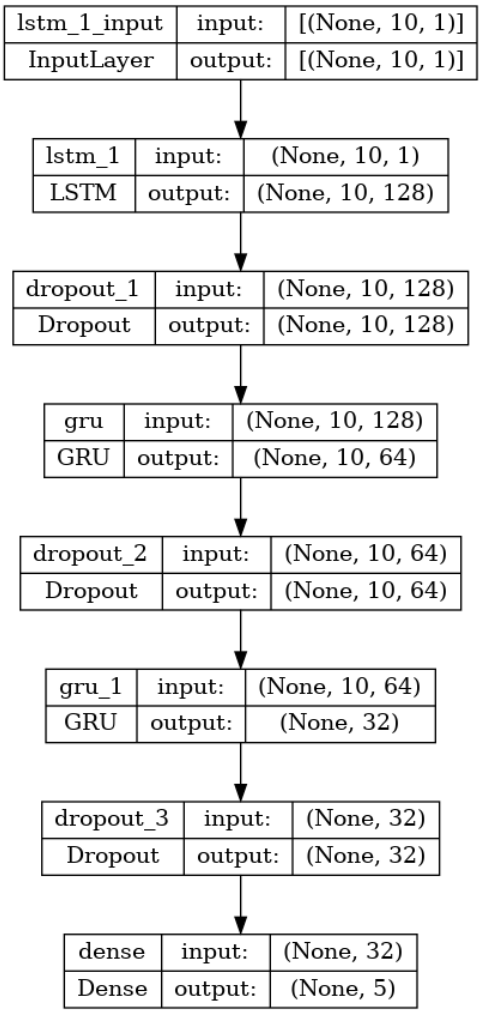

FIGURE 4.10: Architecture du modèle LSTM-GRU.

4.4 Optimisation des hyperparamètres

Pour choisir les meilleurs hyperparamètres de chaque modèle, nous avons effectué des tests en utilisant différents optimiseurs, des tailles de lots (Batch sizes) variées et des taux d'apprentissage (Learning rates) différents. Ces tests nous ont permis de trouver les combinaisons des hyperparamètres qui ont donné les meilleurs résultats pour chaque modèle.

Tableau 4.4 représente les différentes optimiseurs, batch sizes et learning rates utilisées.

	Tailles de lots	**Optimiseurs**	**Taux d'apprentissage**
LSTM	64 - 128 - 256 - 512	Adam - SGD - RMSprop	0.001 - 0.002 - 0.003
GRU	64 - 128 - 256 - 512	Adam - SGD - RMSprop	0.001 - 0.002 - 0.003
LSTM-GRU	64 - 128 - 256 - 512	Adam - SGD - RMSprop	0.001 - 0.002 - 0.003

TABLE 4.4: Configuration des modèles LSTM, GRU et LSTM-GRU

4.5 Mesures d'évaluation

4.5.1 Matrice de confusion

C'est un outil permettant de visualiser la performance d'un modèle de classification. Elle représente les prédictions du modèle par rapport aux observations réelles en les classant en quatre catégories : vrais positifs (VP), faux positifs (FP), vrais négatifs (VN) et faux négatifs (FN).

— TP (True Positive) représentent le nombre d'échantillons positifs qui ont été correctement classés comme positifs par le modèle.

— TN (True Negative) représentent le nombre d'échantillons négatifs qui ont été correctement classés comme négatifs par le modèle.

— FP (False Positive) représentent le nombre d'échantillons négatifs qui ont été incorrectement classés comme positifs par le modèle.

— FN (False Negative) représentent le nombre d'échantillons positifs qui ont été incorrectement classés comme négatifs par le modèle.

		Valeur Réelle	
		Positive	Négative
Valeur Prédite	Positive	TP	FP
	Négative	FN	TN

TABLE 4.5: Matrice de confusion

4.5.2 Rapport de classification

Il fournit des mesures détaillées pour chaque classe prédite par le modèle. Il comprend :

— Précision (precision) : représente la proportion des vrais positifs parmi les prédictions positives. Elle évalue la capacité du modèle à limiter les faux positifs.

$$\text{Précision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.2}$$

— Rappel (recall) : mesure la proportion des vrais positifs parmi les observations réellement positives.

$$\text{Rappel} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.3}$$

— F-mesure (f1-score) : une mesure qui combine à la fois la précision et le rappel d'un modèle.

$$\text{F1-score} = 2 \cdot \frac{\text{precision} \cdot \text{Recall}}{\text{precision} + \text{Recall}} \tag{4.4}$$

4.5.3 Courbe ROC

— **(Receiver Operating Characteristic)** : C'est une représentation graphique de la performance d'un modèle de classification. Elle montre la relation entre le taux de vrais positifs et le taux de faux positifs pour différents seuils de classification.

— **AUC-ROC (Area Under the ROC Curve)** : est une mesure quantitative de la performance d'un modèle de classification basée sur la courbe ROC. Elle représente l'aire sous la courbe ROC et varie de 0 à 1. Plus la valeur de l'AUC-ROC se rapproche de 1, plus le modèle est performant.

4.6 Outils de développement

4.6.1 Environnement d'exécution

Nous avons utilisé **Kaggle** comme environnement d'exécution, c'est une plateforme en ligne qui offre un environnement intégré pour développer et exécuter des projets.
On a utilisé l'accélérateur GPU : **GPU P100** pour réduire le temps d'exécution.

4.6.2 Bibliothèque utilisée

— **SKlearn** : une bibliothèque populaire d'apprentissage automatique en Python. Elle offre une large gamme d'outils et d'algorithmes pour la préparation des données, la construction de modèle.

— **Matplotlib** : une bibliothèque de visualisation de données en Python. Elle permet de créer une grande variété de graphiques, tels que des histogrammes, des diagrammes en boîte, etc. Matplotlib est souvent utilisée pour représenter graphiquement des données, pour une communication des résultats.

— **Pandas** : bibliothèque Python utilisée pour la manipulation et l'analyse des données. Elle est couramment utilisée pour charger, prétraiter et manipuler des données structurées, notamment des tableaux de données et des fichiers CSV.

— **Numpy** : bibliothèque fondamentale en Python pour le calcul scientifique et numérique. Elle est utilisée pour effectuer des calculs numériques rapides.

— **Seaborn** : bibliothèque de visualisation de données basée sur Matplotlib. Elle permet de créer facilement des graphiques statistiques avancés et une meilleure intégration avec les données pandas.

— **Keras** : est une bibliothèque d'apprentissage en profondeur. Elle offre la construction de modèles d'apprentissage en profondeur.

4.7 Résultats

Après avoir réalisé plusieurs tests en explorant différentes combinaisons des hyperparamètres présenté précédemment, nous avons évalué les performances de chaque modèle en utilisant diverses mesures d'évaluation, notamment l'exactitude (accuracy), la perte (loss), la matrice de confusion, le rapport de classification et ROC-AUC.
Ces évaluations nous ont permis de sélectionner les configurations d'hyperparamètres qui ont donné les meilleurs résultats pour chaque modèle.

4.7.1 Premier modèle LSTM

Les meilleurs hyperparamètres identifiés pour le modèle LSTM sont présentés dans le tableau 4.7.1.

Learning rate	**Batch size**	**Optimiseur**	**Résultat**	
			Accuracy	**Loss**
0.001	512	Adam	0.97	0.05

TABLE 4.6: Performance du modèle LSTM

Ces valeurs ont été sélectionnées après des tests approfondis pour optimiser les performances du modèle. L'apprentissage a pris beaucoup de temps, le modèle a été entraîné pendant **20 époques**. Notre modèle LSTM a atteint une exactitude de **97,3%**.

4.7.1.1 Courbes accuracy et loss

La courbe d'exactitude de ce modèle est présentée dans la figure 4.11 :

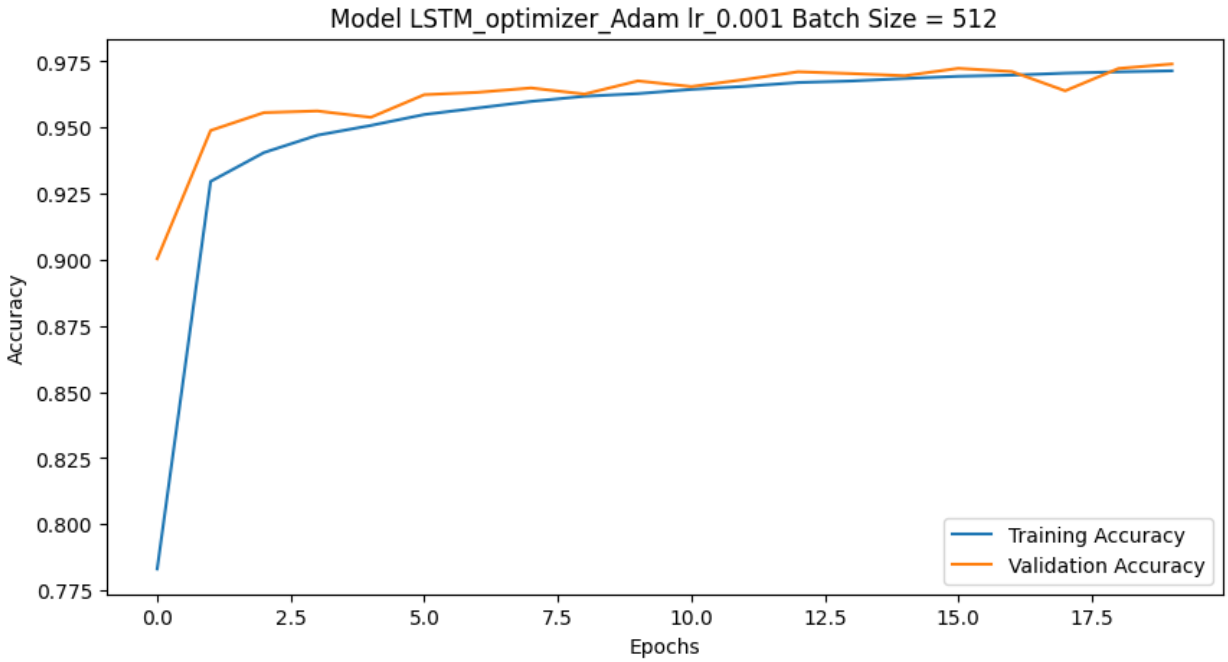

FIGURE 4.11: Courbe d'exactitude du modèle LSTM.

La courbe d'erreur de ce modèle est présentée dans la figure 4.12 :

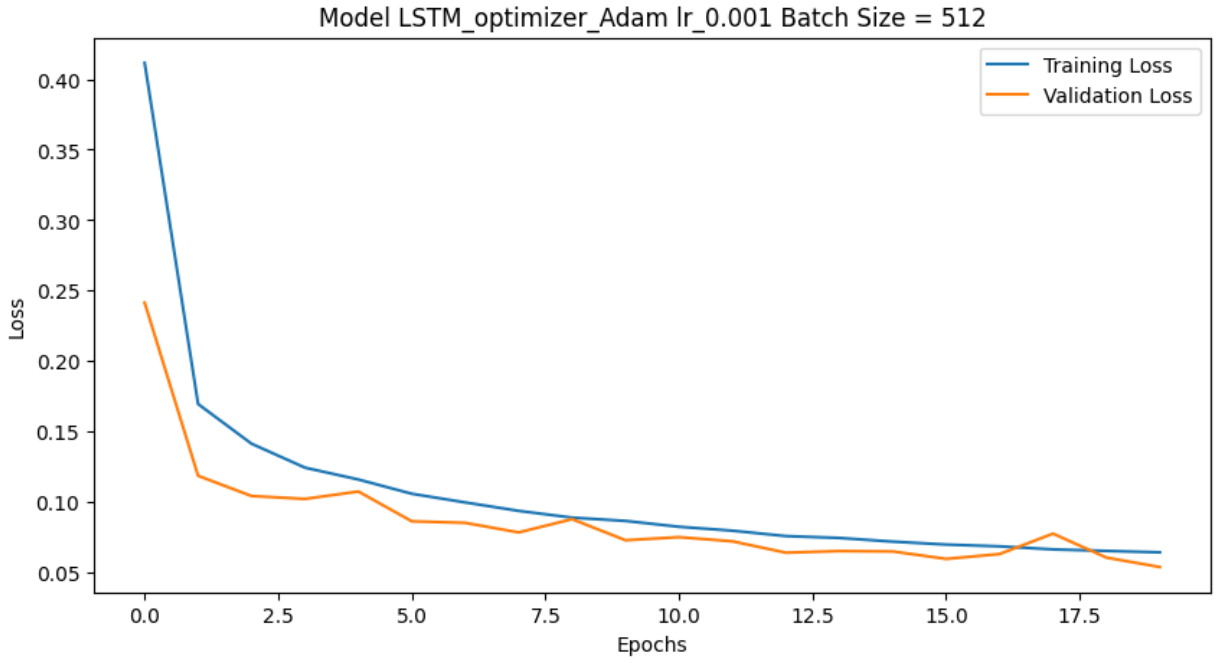

FIGURE 4.12: Courbe d'erreur du modèle LSTM.

4.7.1.2 Rapport de classification

Le rapport de classification du modèle LSTM

Classification Report

	precision	recall	f1-score	support
0	0.99	0.96	0.98	385309
1	0.96	0.99	0.97	330112
2	0.91	0.54	0.68	107
3	0.95	0.99	0.97	18163
4	1.00	0.93	0.96	14
accuracy			0.97	733705
macro avg	0.96	0.88	0.91	733705
weighted avg	0.97	0.97	0.97	733705

FIGURE 4.13: Rapport de classification du modèle LSTM.

4.7.1.3 Matrice de confusion

La matrice de confusion du modèle LSTM

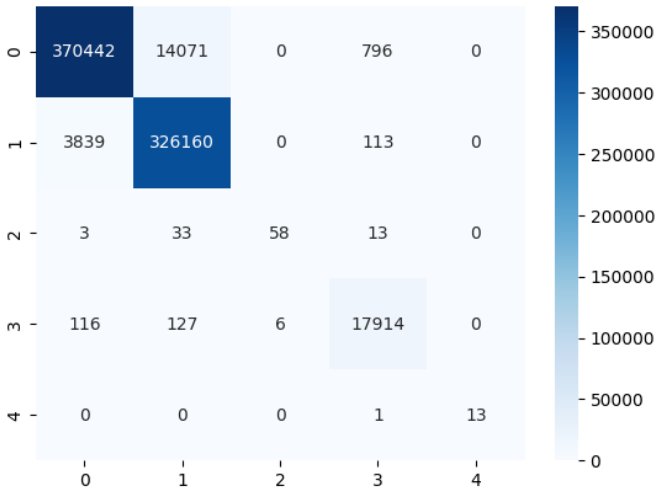

FIGURE 4.14: Matrice de confusion du modèle LSTM.

1. La classe **DDOS**, avec **370442** prédictions correctes à partir d'un total de **385309**. Cela indique que le modèle est capable de détecter efficacement les attaques DDOS.

2. Pour la classe **DOS**, notre modèle a prédit correctement **326,160** échantillons à partir de **330,112** échantillons.

3. Pour la classe **Normal**, notre modèle a prédit correctement **58** échantillons à partir de **107** échantillons. Le résultat indique que notre modèle a eu du mal à classer correctement cette classe en raison d'un nombre d'échantillons extrêmement réduits.

4. Pour la classe **Reconnaissance**, notre modèle a prédit correctement **17,914** échantillons à partir de **18,163** échantillons. Cela indiquent une bonne capacité de notre modèle à détecter les tentatives de reconnaissance dans le réseau.

5. Pour la classe **Theft**, notre modèle a prédit correctement **13** échantillons à partir de **14** échantillons. Bien que le nombre d'échantillons soit très faible, il est encourageant de constater que notre modèle est capable de détecter ce type d'activité suspecte.

4.7.1.4 ROC-AUC

Courbe ROC-AUC pour le modèle LSTM :

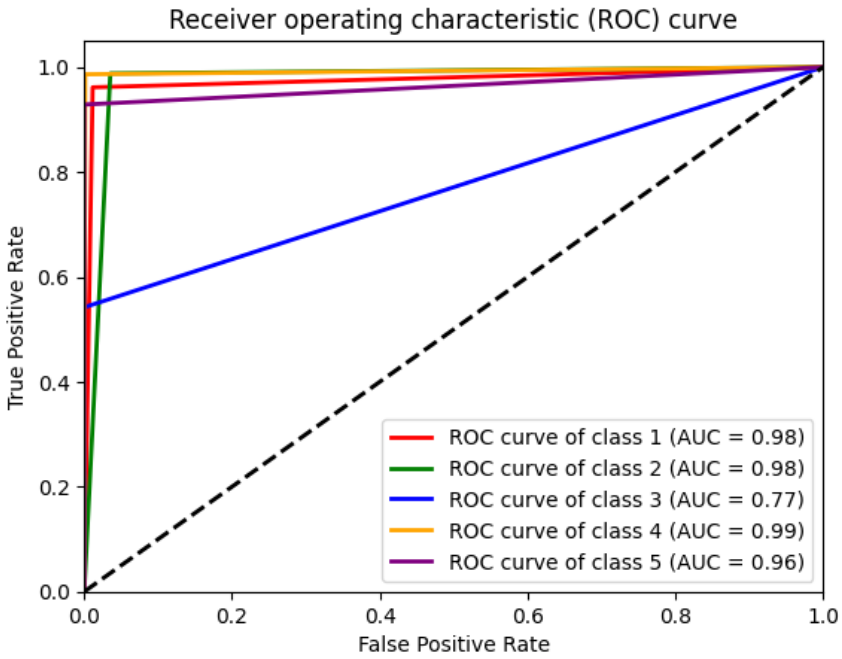

FIGURE 4.15: ROC-AUC du modèle LSTM.

Les résultats indiquent que le modèle LSTM a une très bonne performance de classification pour les classes 1 : DDoS, 2 : DoS, 4 : Reconnaissance et 5 : Theft, avec des ROC-AUC de 0,98, 0,98, 0,99 et 0,96 respectivement.

Cependant, **la classe 3** : Normal présente un ROC-AUC de 0,77, ce qui est relativement plus bas par rapport aux autres classes. Cela peut indiquer que le modèle rencontre des difficultés pour classifier cette classe.

4.7.2 Deuxième modèle GRU

Les meilleurs hyperparamètres identifiés pour le modèle GRU sont présentés dans le tableau 4.7.2.

Le modèle GRU avec ces hyperparamètres a atteint une exactitude : 97.7% .

Learning rate	**Batch size**	**Optimiseur**	**Résultat**	
			Accuracy	**Loss**
0.002	512	Adam	0.97	0.04

TABLE 4.7: Performance du modèle GRU

4.7.2.1 Courbes accuracy et loss

Les courbes d'exactitude et d'erreur pour notre deuxième modèle sont :

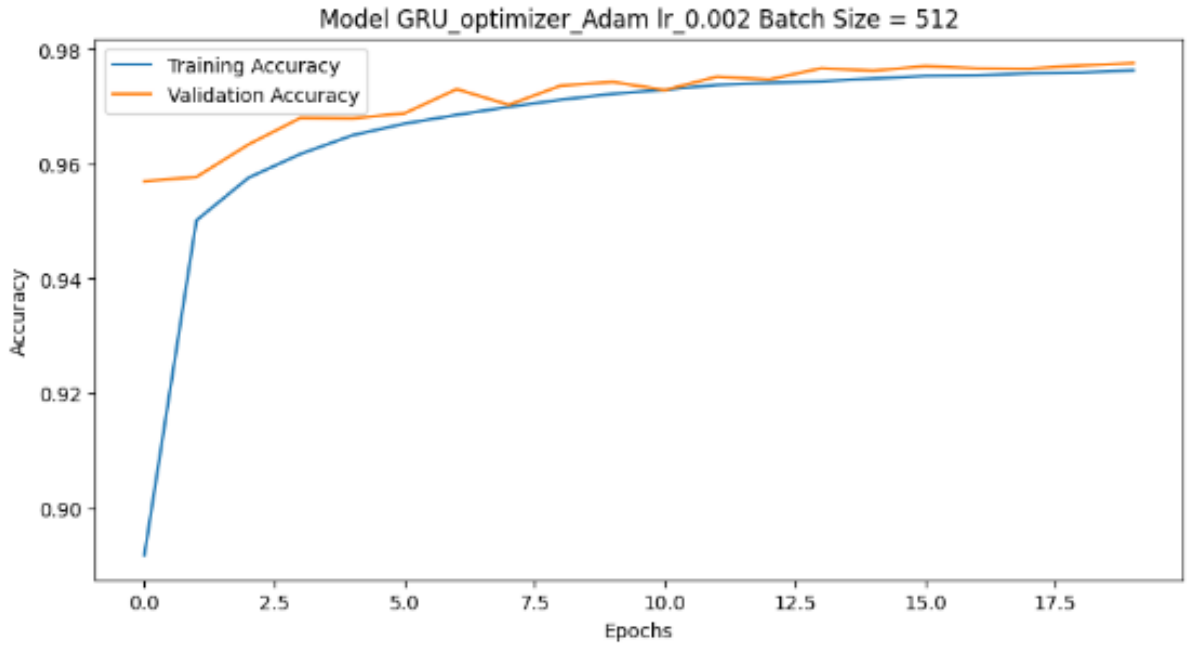

FIGURE 4.16: Courbe d'exactitude du modèle GRU.

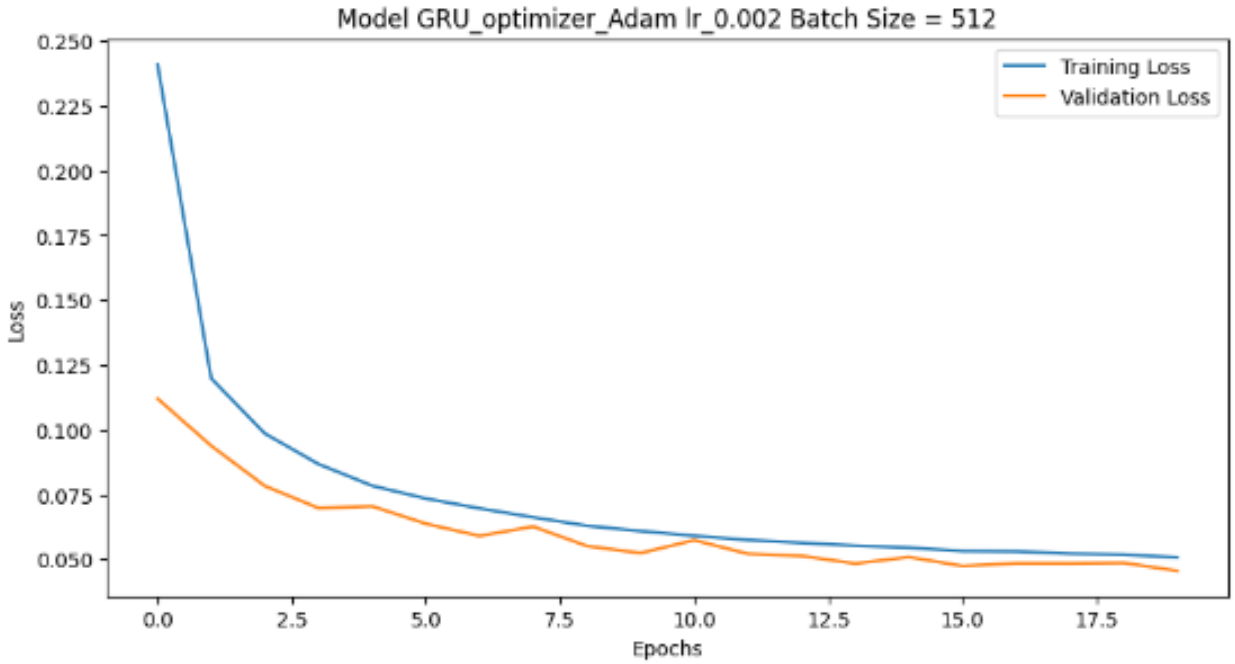

FIGURE 4.17: Courbe d'erreur du modèle GRU.

4.7.2.2 Rapport de classification

Le rapport de classification du modèle GRU est présenté dans la figure 4.18 :

Classification Report

	precision	recall	f1-score	support
0	0.99	0.97	0.98	385309
1	0.96	0.99	0.98	330112
2	0.71	0.75	0.73	107
3	0.97	0.99	0.98	18163
4	1.00	0.57	0.73	14
accuracy			0.98	733705
macro avg	0.93	0.85	0.88	733705
weighted avg	0.98	0.98	0.98	733705

FIGURE 4.18: Rapport de classification du modèle GRU.

4.7.2.3 Matrice de confusion

La matrice de confusion du modèle GRU :

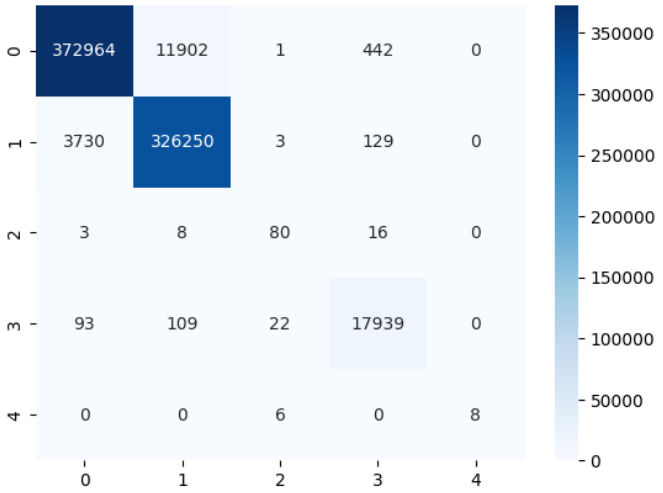

FIGURE 4.19: Matrice de confusion du modèle GRU.

Pour la classe normale, le modèle GRU a prédit correctement 80 échantillons à partir de 107 échantillons, soit une amélioration par rapport au modèle LSTM.
Cependant, la performance de classification pour les autres classes est assez similaire entre les deux modèles, avec des taux de prédiction correcte élevés pour les classes 0, 1, 2 et 3, et une performance relativement faible pour la classe 4.

4.7.2.4 ROC-AUC

La courbe ROC-AUC pour le modèle GRU :

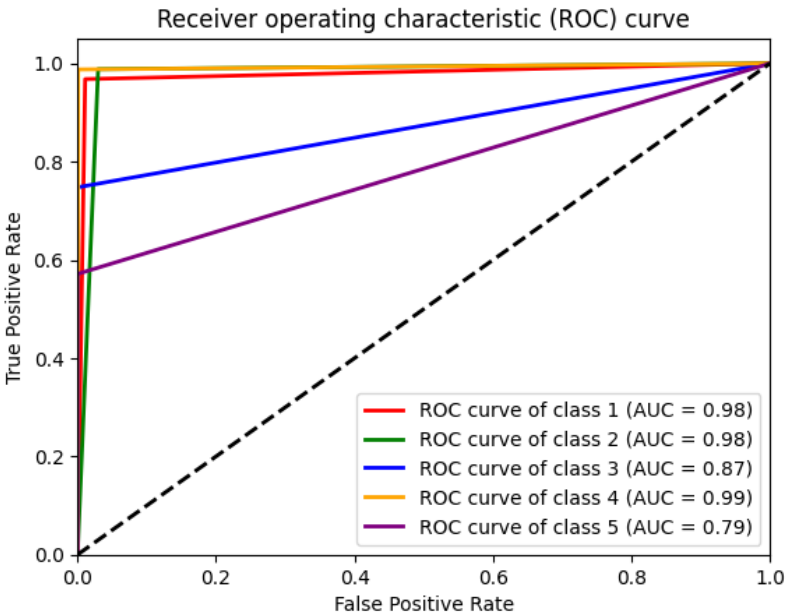

FIGURE 4.20: ROC-AUC du modèle GRU.

4.7.3 Troisième modèle LSTM-GRU

Le modèle LSTM-GRU qui combine les 2 modèles LSTM et GRU avec ces hyperparamètres a atteint une exactitude : 97.4%.

Learning rate	Batch size	Optimiseur	Résultat	
			Accuracy	Loss
0.002	512	Adam	0.97	0.05

TABLE 4.8: Performance du modèle LSTM-GRU

4.7.3.1 Courbes accuracy et loss

Les deux courbes de modèle LSTM-GRU sont présentées dans les figures 4.21 et 4.22 respectivement :

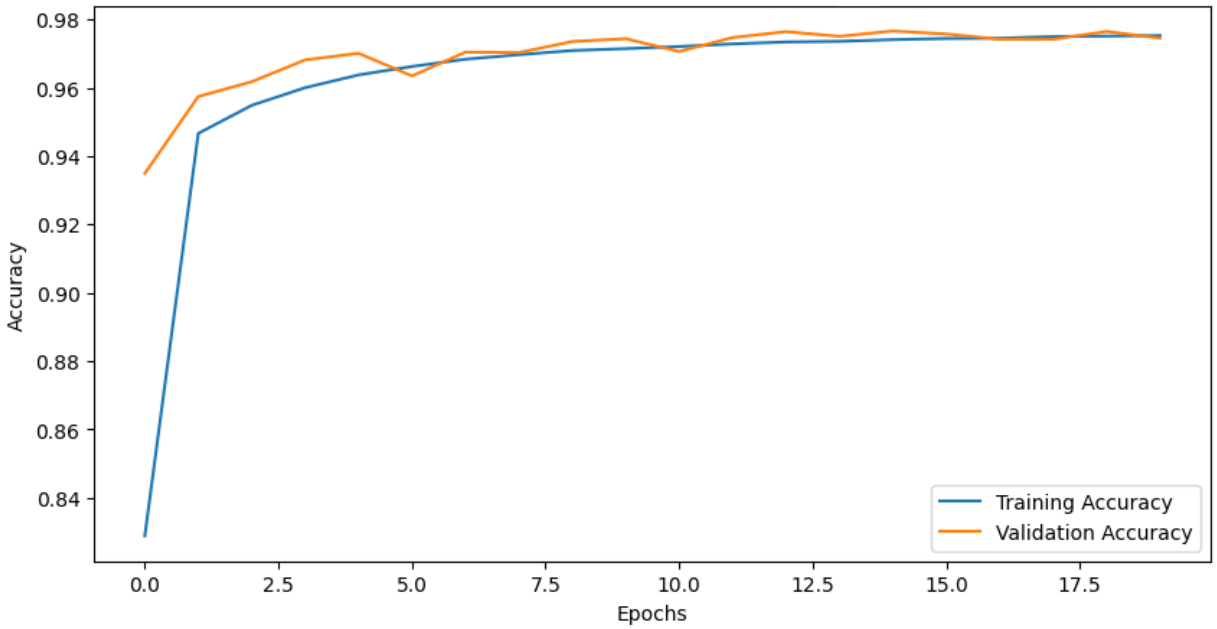

FIGURE 4.21: Courbe d'exactitude du modèle LSTM-GRU.

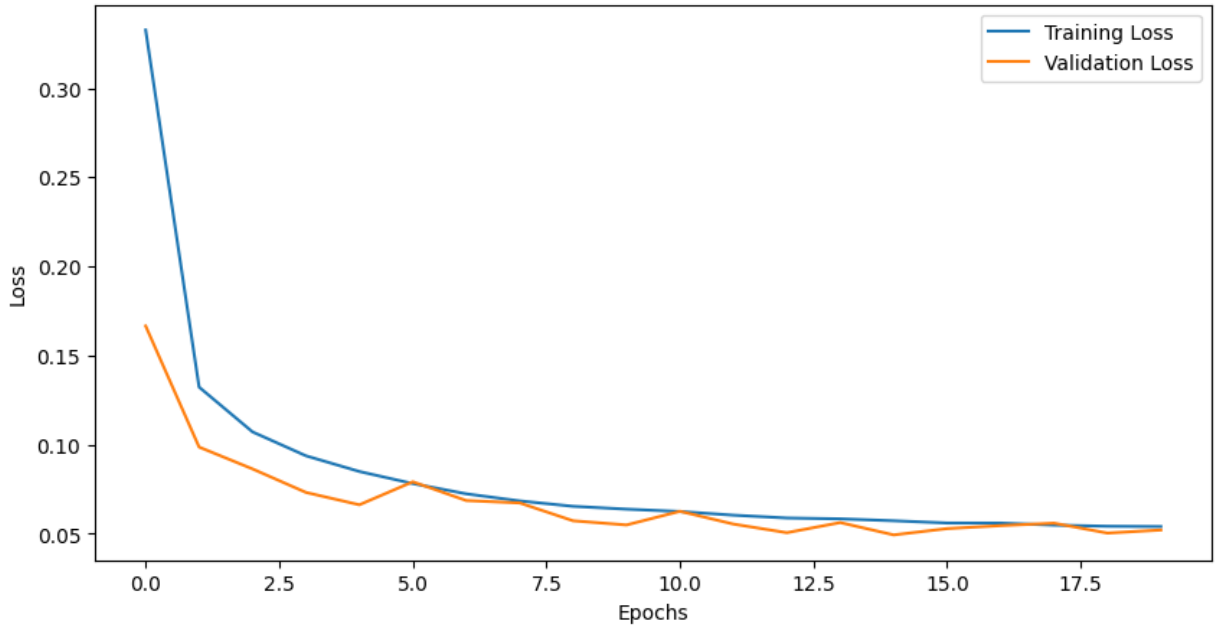

FIGURE 4.22: Courbe d'erreur du modèle LSTM-GRU.

4.7.3.2 Rapport de classification

Le rapport de classification de ce modèle (LSTM-GRU) est :

Classification Report

	precision	recall	f1-score	support
0	0.99	0.96	0.98	385309
1	0.96	0.99	0.97	330112
2	0.67	0.83	0.74	107
3	0.96	0.98	0.97	18163
4	1.00	0.50	0.67	14
accuracy			0.97	733705
macro avg	0.92	0.85	0.87	733705
weighted avg	0.98	0.97	0.97	733705

FIGURE 4.23: Rapport de classification du modèle LSTM-GRU.

4.7.3.3 Matrice de confusion

Le modèle LSTM-GRU, a une amélioration dans la prédiction de la classe 2 par rapport aux deux autres modèles. Ce modèle a réussi à prédire correctement 89 échantillons sur 107 pour

cette classe, ce qui indique une amélioration significative par rapport aux autres modèles.

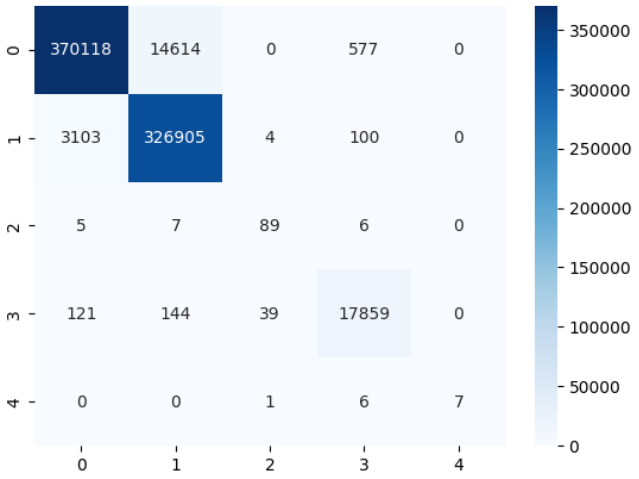

FIGURE 4.24: Matrice de confusion du modèle LSTM-GRU.

En ce qui concerne la classe 4, les modèles GRU et LSTM-GRU ont des performances similaires, avec une légère différence. Le modèle GRU a prédit correctement 8 échantillons sur 14, tandis que le modèle LSTM-GRU a prédit correctement 7 échantillons sur 14.

4.7.3.4 ROC-AUC

La courbe ROC-AUC pour le modèle LSTM-GRU est :

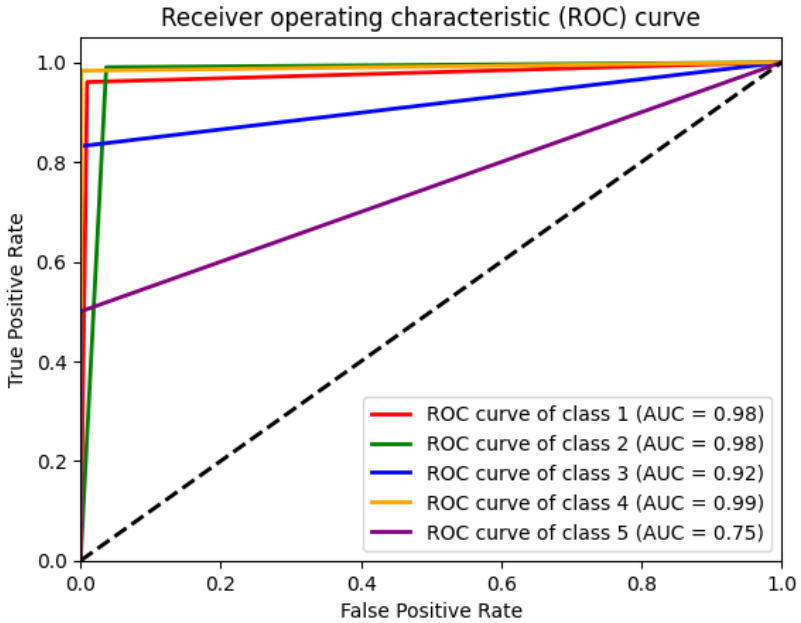

FIGURE 4.25: ROC-AUC du modèle LSTM-GRU.

Les résultats de la courbe ROC-AUC pour le modèle LSTM-GRU montrent une performance élevée pour les classes 1 : DDoS, 2 : DoS et 4 : Reconnaissance, avec des ROC-AUC de 0,98, 0,98 et 0,99 respectivement. La classe 3 : Normal présente également une amélioration significative par rapport aux modèles précédents, avec une ROC-AUC de 0,92.
Cependant, la classe 5 : Theft montre une performance légèrement inférieure, avec une ROC-AUC de 0,75.
Il semble que ce modèle a eu plus de difficulté à classifier la classe 5 par rapport aux autres classes.

4.8 Comparaison de nos modèles avec d'autres modèles dans la littérature

Nous avons comparé nos résultats avec d'autres résultats qui sont obtenus à partir d'autres modèles qui sont largement utilisés dans la détection d'intrusion, plus particulierement : **CNN** (Convolutional Neural Network) [36], **MLP** (Multilayer Perceptron) [36], **SVM** [18] et **Random Forest** [2].

Les résultats de cette comparaison sont présentés dans le tableau 4.8, où nous avons évalué les modèles en termes d'AUC (Area Under the Curve) et d'exactitude (Accuracy) pour chaque classe d'attaque.

Modèle	**AUC 1 (DDoS)**	**AUC 2 (DoS)**	**AUC 3 (Normal)**	**AUC 4 (Reconnaissance)**	**AUC 5 (Theft)**	**Accuracy**
CNN [36]	0.98	0.98	0.99	0.99	1	91.27%
MLP [36]	0.56	0.51	0.97	0.99	0.99	97.01%
SVM [18]	/	/	/	/	/	89.52%
Random Forest [2]	/	/	/	/	/	92.67%
Nos modèles						
LSTM	0.98	0.98	0.77	0.99	0.96	97.3%
GRU	0.98	0.98	0.87	0.99	0.79	97.7%
LSTM-GRU	0.98	0.98	0.92	0.99	0.75	97.4%

TABLE 4.9: Performance des modèles

Pour les classes "DDoS" et "DoS", le modèle CNN et nos modèles ont des scores AUC similaires de 0.98, ce qui dénote une bonne capacité de classification pour ces types d'attaque.

Cependant, dans la classe "Normal", le modèle CNN se démarque avec un score AUC élevé de 0.99, tandis que nos modèles LSTM, GRU et LSTM-GRU obtiennent des scores AUC inférieurs de 0.77, 0.87 et 0.92 respectivement. Il semble que le modèle CNN soit plus performant dans la détection des données normales.

En ce qui concerne la classe "Reconnaissance", tous les modèles, y compris le modèle CNN, affichent un score AUC élevé de 0.99, ce qui suggère une bonne capacité de reconnaissance des activités malveillantes.

Pour la classe "Theft", le modèle CNN et le modèle MLP obtiennent des scores AUC presque parfaits de 1 et 0.99 respectivement, tandis que nos modèles LSTM, GRU et LSTM-GRU affichent des scores AUC légèrement inférieurs de 0.96, 0.79 et 0.75 respectivement.

En termes d'exactitude, nos modèles LSTM, GRU et LSTM-GRU surpassent le modèle CNN avec des valeurs allant de 97.3% à 97.7%. Le modèle MLP obtient une exactitude élevée de 97.01%, tandis que les modèles SVM et Random Forest affichent des exactitudes respectives de 89.52% et 92.67%.

En conclusion, nos modèles LSTM, GRU et LSTM-GRU présentent des performances similaires ou meilleures que le modèle CNN dans la plupart des classes, à l'exception des classes "Normal" et "Theft" où le modèle CNN se démarque. Nos modèles ont également de meilleures exactitudes par rapport aux autres modèles.

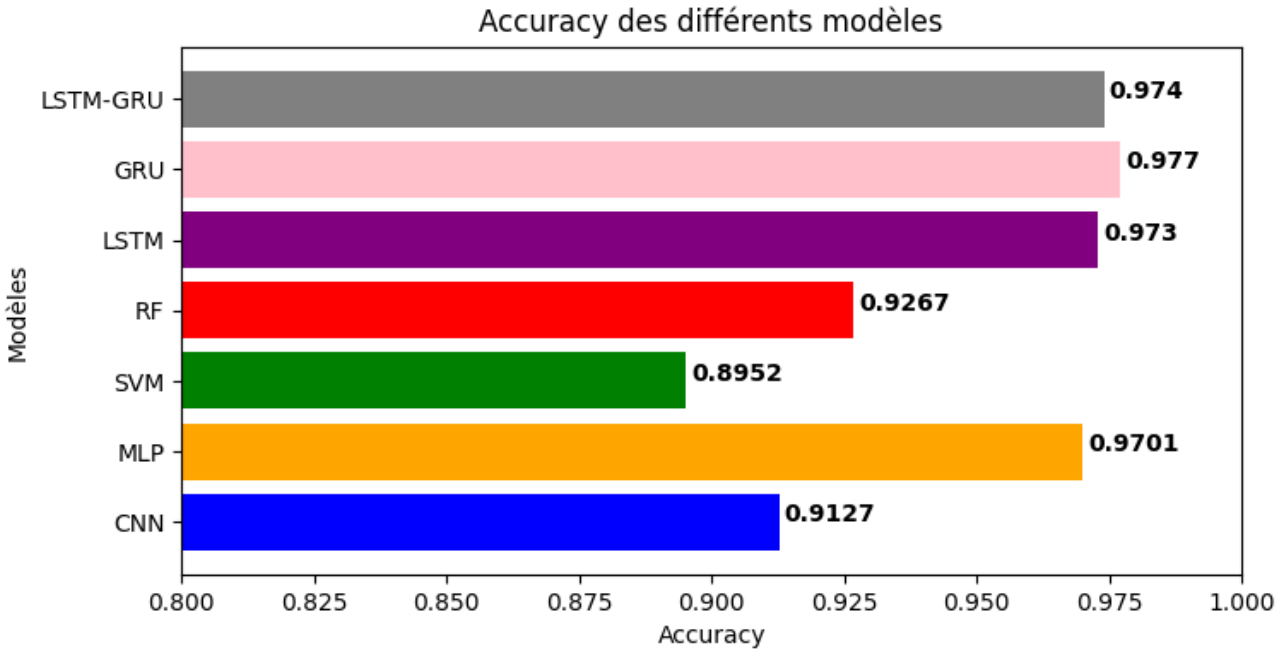

FIGURE 4.26: Comparaison nos modèles avec d'autres modèles en terme d'exactitude.

4.9 Meilleur modèle

En analysant les résultats des modèles LSTM, GRU et LSTM-GRU en termes des différentes mesures d'évaluation, il semble que le modèle GRU soit le meilleur choix final parmi les trois.

Le modèle GRU présente des performances élevées et équilibrées dans la plupart des classes, avec des scores élevés de ROC-AUC, d'exactitude et de f1-score pour les classes DDoS, DoS, Reconnaissance et Theft. Bien que son score de f1-score pour la classe Normal soit légèrement inférieur, il reste relativement bon par rapport aux autres modèles.

Le modèle GRU offre donc un compromis solide entre les différentes mesures d'évaluation et semble être le plus performant globalement.

4.10 Conclusion

En conclusion, ce chapitre a présenté les différentes étapes de l'implémentation de notre projet de détection d'intrusions dans les réseaux IDO à l'aide des réseaux de neurones récurrents (LSTM, GRU et LSTM-GRU). Nous avons présenté les détails de la préparation des données, le processus de modélisation et les mesures d'évaluation utilisées pour évaluer la performance de nos modèles. Les résultats ont montré que nos modèles ont atteint des performances remarquables avec des scores d'AUC. De plus, une comparaison avec d'autres modèles publiés dans la littérature a montré que nos modèles étaient compétitifs.

CHAPITRE 5

CONCLUSION GENERALE

5.1 Conclusion

En conclusion, notre projet a porté sur la sécurité des réseaux IDO en utilisant des techniques de Deep Learning pour la détection d'intrusions. La détection d'intrusions joue un rôle crucial dans la protection des réseaux IDO, car ces derniers sont vulnérables à diverses attaques en raison de leur nature interconnectée et de la diversité des dispositifs qui y sont connectés.
La détection d'intrusions permet d'identifier et de prévenir les activités malveillantes dans les réseaux IDO, telles que les attaques de déni de service (DDoS), les tentatives de reconnaissance, les attaques de type botnet et bien d'autres. En détectant ces intrusions, les systèmes de sécurité peuvent prendre des mesures pour atténuer les attaques et protéger les dispositifs et les données sensibles. Dans notre projet, nous avons utilisé des modèles de réseaux de neurones récurrents (LSTM, GRU et LSTM-GRU) pour la détection des intrusions.
Ces modèles ont démontré leur efficacité dans la détection d'intrusions. Cependant, nous avons également identifié une faible capacité de nos modèles à détecter l'attaque de type "Theft".

5.2 Travaux futurs

Dans nos travaux futurs, nous envisageons d'explorer plusieurs perspectives prometteuses pour renforcer la sécurité des réseaux IDO en utilisant le deep learning. Ces perspectives comprennent :

Implémentation des modèles IDS en environnement réel : Nous prévoyons de déployer nos modèles de détection des intrusions dans des environnements réels de réseaux IDO.

Détection d'anomalies et systèmes IPS : nous souhaitons approfondir notre travail en développant des techniques de détection d'anomalies spécifiques aux réseaux IDO. Cela comprendra également l'intégration de systèmes de prévention des intrusions (IPS).

Adaptation aux nouvelles formes d'attaques : nous nous engageons à améliorer continuellement nos algorithmes de détection afin de rester à la pointe des nouvelles formes d'attaques

émergentes. Cela implique une recherche active sur les techniques de détection des intrusions les plus récentes pour répondre aux menaces spécifiques au réseau IDO.

Intégration des techniques de renforcement : nous explorons également l'utilisation de techniques de renforcement combinées avec le deep learning pour améliorer la détection des intrusions dans les réseaux IDO.

BIBLIOGRAPHIE

[1] M. AHMED, A. N. MAHMOOD, AND J. HU, *A survey of network anomaly detection techniques*, Journal of Network and Computer Applications, 60 (2016), pp. 19–31.

[2] Z. AL-OTHMAN, M. ALKASASSBEH, AND S. A.-H. BADDAR, *A state-of-the-art review on iot botnet attack detection*, arXiv preprint arXiv :2010.13852, (2020).

[3] Z. H. ALI, H. A. ALI, AND M. M. BADAWY, *Internet of things (iot) : definitions, challenges and recent research directions*, International Journal of Computer Applications, 128 (2015), pp. 37–47.

[4] E. ALPAYDIN, *Introduction to machine learning*, MIT press, 2020.

[5] K. ARUNKUMAR, D. V. KALAGA, C. M. S. KUMAR, M. KAWAJI, AND T. M. BRENZA, *Comparative analysis of gated recurrent units (gru), long short-term memory (lstm) cells, autoregressive integrated moving average (arima), seasonal autoregressive integrated moving average (sarima) for forecasting covid-19 trends*, Alexandria engineering journal, 61 (2022), pp. 7585–7603.

[6] C.-A. AZENCOTT, *Introduction au machine learning*, (2018).

[7] T. BEYSOLOW II, *Introduction to deep learning using R : A step-by-step guide to learning and implementing deep learning models using R*, Apress, 2017.

[8] DATACAMP, *Deep learning in python*.
`https://www.datacamp.com/tutorial/deep-learning-python`, s.d.

[9] A. DIRO AND N. CHILAMKURTI, *Leveraging lstm networks for attack detection in fog-to-things communications*, IEEE Communications Magazine, 56 (2018), pp. 124–130.

[10] DNSSTUFF, *Ids vs ips : What's the difference ?*
`https://www.dnsstuff.com/ids-vs-ips`, 2019.

[11] M. ELBATTAH, C. LOUGHNANE, J.-L. GUÉRIN, R. CARETTE, F. CILIA, AND G. DEQUEN, *Variational autoencoder for image-based augmentation of eye-tracking data*, Journal of Imaging, 7 (2021), p. 83.

[12] M. A. FERRAG AND L. MAGLARAS, *Deepcoin : A novel deep learning and blockchain-based energy exchange framework for smart grids*, IEEE Transactions on Engineering Management, 67 (2019), pp. 1285–1297.

[13] V. GARCIA-FONT, C. GARRIGUES, AND H. RIFÀ-POUS, *A comparative study of anomaly detection techniques for smart city wireless sensor networks*, sensors, 16 (2016), p. 868.

[14] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, AND Y. BENGIO, *Generative adversarial networks*, Communications of the ACM, 63 (2020), pp. 139–144.

[15] H. GU, Y. WANG, S. HONG, AND G. GUI, *Blind channel identification aided generalized automatic modulation recognition based on deep learning*, IEEE Access, 7 (2019), pp. 110722–110729.

[16] S. HAJIHEIDARI, K. WAKIL, M. BADRI, AND N. J. NAVIMIPOUR, *Intrusion detection systems in the internet of things : A comprehensive investigation*, Computer Networks, 160 (2019), pp. 165–191.

[17] I. KANDEL AND M. CASTELLI, *Transfer learning with convolutional neural networks for diabetic retinopathy image classification. a review*, Applied Sciences, 10 (2020), p. 2021.

[18] A. KHRAISAT, I. GONDAL, P. VAMPLEW, J. KAMRUZZAMAN, AND A. ALAZAB, *A novel ensemble of hybrid intrusion detection system for detecting internet of things attacks*, Electronics, 8 (2019), p. 1210.

[19] N. KORONIOTIS, N. MOUSTAFA, E. SITNIKOVA, AND B. TURNBULL, *Towards the development of realistic botnet dataset in the internet of things for network forensic analytics : Bot-iot dataset*, Future Generation Computer Systems, 100 (2019), pp. 779–796.

[20] Q. V. LE ET AL., *A tutorial on deep learning part 2 : Autoencoders, convolutional neural networks and recurrent neural networks*, Google Brain, 20 (2015), pp. 1–20.

[21] D.-C. LI, S.-C. CHEN, Y.-S. LIN, AND K.-C. HUANG, *A generative adversarial network structure for learning with small numerical data sets*, Applied Sciences, 11 (2021), p. 10823.

[22] C. LIANG, B. SHANMUGAM, S. AZAM, M. JONKMAN, F. DE BOER, AND G. NARAYANSAMY, *Intrusion detection system for internet of things based on a machine learning approach*, in 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), IEEE, 2019, pp. 1–6.

[23] H.-J. LIAO, C.-H. R. LIN, Y.-C. LIN, AND K.-Y. TUNG, *Intrusion detection system : A comprehensive review*, Journal of Network and Computer Applications, 36 (2013), pp. 16–24.

[24] K. MAHANTA AND H. B. MARINGANTI, *Security in the internet of things (iot) : Developing intrusion detection systems for iot devices and networks and addressing the unique security challenges posed by this connected environment.*

[25] J. MALIK, A. AKHUNZADA, I. BIBI, M. IMRAN, A. MUSADDIQ, AND S. W. KIM, *Hybrid deep learning : An efficient reconnaissance and surveillance detection mechanism in sdn*, IEEE Access, 8 (2020), pp. 134695–134706.

[26] W. W. NSUNZA, A.-Q. R. TETTEH, AND X. HEI, *Accelerating a secure programmable edge network system for smart classroom*, in 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), IEEE, 2018, pp. 1384–1389.

[27] S. OMAR, A. NGADI, AND H. H. JEBUR, *Machine learning techniques for anomaly detection : an overview*, International Journal of Computer Applications, 79 (2013).

[28] M. PAK AND S. KIM, *A review of deep learning in image recognition*, in 2017 4th international conference on computer applications and information processing technology (CAIPT), IEEE, 2017, pp. 1–3.

[29] R. RAJ, *Supervised, unsupervised, and semi-supervised learning*. https://www.enjoyalgorithms.com/blogs/supervised-unsupervised-and-semisupervised-learning. Consulté le 20 mai 2023.

[30] S. RIZVI, R. ORR, A. COX, P. ASHOKKUMAR, AND M. R. RIZVI, *Identifying the attack surface for iot network*, Internet of Things, 9 (2020), p. 100162.

[31] S. RUDER, *An overview of gradient descent optimization algorithms*, arXiv preprint arXiv :1609.04747, (2016).

[32] I. H. SARKER, *Deep learning : a comprehensive overview on techniques, taxonomy, applications and research directions*, SN Computer Science, 2 (2021), p. 420.

[33] S. SHARMA, S. SHARMA, AND A. ATHAIYA, *Activation functions in neural networks*, Towards Data Sci, 6 (2017), pp. 310–316.

[34] N. SRIVASTAVA, G. HINTON, A. KRIZHEVSKY, I. SUTSKEVER, AND R. SALAKHUTDINOV, *Dropout : a simple way to prevent neural networks from overfitting*, The journal of machine learning research, 15 (2014), pp. 1929–1958.

[35] STATISTA, *Frequently seen passwords in iot devices*. https://www.statista.com/statistics/1298495/frequently-seen-passwords-in-iot-devices/, 2021.

[36] B. SUSILO AND R. F. SARI, *Intrusion detection in iot networks using deep learning algorithm*, Information, 11 (2020), p. 279.

[37] W. T. TOOR, M. ALVI, AND M. AGIWAL, *Combined access barring scheme for iot devices using bayesian estimation*, Electronics, 9 (2020), p. 2191.

[38] D. VARMEDJA, M. KARANOVIC, S. SLADOJEVIC, M. ARSENOVIC, AND A. ANDERLA, *Credit card fraud detection-machine learning methods*, in 2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH), IEEE, 2019, pp. 1–5.

[39] C. XU, J. SHEN, X. DU, AND F. ZHANG, *An intrusion detection system using a deep neural network with gated recurrent units*, IEEE Access, 6 (2018), pp. 48697–48707.

[40] M. YOUSEFI-AZAR, V. VARADHARAJAN, L. HAMEY, AND U. TUPAKULA, *Autoencoder-based feature learning for cyber security applications*, in 2017 International joint conference on neural networks (IJCNN), IEEE, 2017, pp. 3854–3861.

[41] J. ZOU, Y. HAN, AND S.-S. SO, *Overview of artificial neural networks*, Artificial neural networks : methods and applications, (2009), pp. 14–22.