الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – Guelma -

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

Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique

Option : Sciences et Technologies de l'Information et de Communication.

Thème :

Un système de détection d'intrusion pour la cybersécurité

Encadré Par :

Dr. Ferrag Med Amine

Présenté par :

Hamouda Djallel

Octobre 2020

Remerciements

Je tiens à remercier tout d'abord ALLAH le tout puissant de m'avoir donné le courage, la patience et la santé pour réaliser ce modeste travail.

Mes remerciements les plus chaleureux et les plus profonds s'adressent à mon directeur de recherche Monsieur MOHAMED AMINE FERRAG qui m'a guidé par ses précieux conseils, ses encouragements et ses orientations. Ainsi que pour l'aide et le temps qu'il n'a jamais manqué de m'apporter tout au long de l'élaboration de ce travail.

Je remercie également Monsieur RICARDO MORLA de l'Université de Porto- Portugal qui m'a aidé beaucoup pendant mon stage.

Mes remerciements les plus vifs s'adressent aussi aux membres de jury pour l'intérêt qu'ils ont porté à ma recherche tout en acceptant d'évaluer ce travail.

Je tiens aussi à exprimer mes sincères remerciements à tous les enseignants du département de l'informatique et surtout le chef du département Monsieur KOUAHLA ZINE EDDINE.

Enfin, je remercie tous ceux qui ont contribué de près ou de loin à la réalisation de ce mémoire

Dédicace

J'ai le bonheur de dédier ce modeste travail :

À ma source de joie, ceux qui ont toujours veillé sur mon bonheur, qui ont sacrifié pour me voir réussi et qui m'ont comblé tant d'amour et de tendresse, mes chers parents « YAZID et MACHTER RAZIKA ». Ils ont été toujours présents à mes côtés par leurs sacrifices et leurs prières. Que Dieu leur procure une longue vie avec une bonne santé.

À vous, mon adorable frère et sœur, ceux qui ont partagé avec moi tous les moments d'émotion lors de la réalisation de ce travail, merci pour votre support.

À vous aussi Monsieur MOHAMMED EL HADI RAHMANI merci pour votre soutien infini et votre aide incessante.

À mes chers amis avec qui j'ai partagé les meilleurs et les moments les plus agréables tout au long de mon parcours universitaire : Amine, Chouaib, Tamer, Zineddine et tous mes collègues de ma promo du département de l'informatique.

À tous ceux qui, par un mot, m'ont donné la force de continuer…

À tous ceux qui m'aiment et que j'aime…

Djallel

Résumé

Les systèmes de détection d'intrusion (IDS) ont fait l'objet de nombreuses recherches et jouent un rôle important dans la cybersécurité. L'objectif de cette étude est de modéliser un tel système pour aider les administrateurs du système à détecter et à identifier toute violation de la sécurité dans leur organisation afin de les prévenir avant de causer des dommages. Pour cela, nous avons étudié les performances des méthodes d'apprentissage machine (ML) appliquées à la détection des intrusions pour la cybersécurité. Ensuite, nous avons appliqué trois techniques de détection basées sur des approches d'apprentissage profond, un réseau neuronal profond (DNN), un réseau neuronal convolutif (CNN) et un réseau neuronal récurrent (RNN) pour détecter les intrusions dans les connexions réseau. Nous avons évalué les méthodes proposées avec l'ensemble de données CICDDoS2019 de référence pour la détection des attaques DDoS sur les réseaux. Nous avons également présenté une étude comparative avec quatre algorithmes d'apprentissage automatique de référence, en utilisant différentes mesures appliquées pour l'évaluation des performances d'apprentissage machine (Précision, Rappel, score F1), et deux autres indicateurs de performance importants pour la détection des intrusions (taux de détection, taux de fausses alarmes). Les résultats expérimentaux ont montré que les performances des approches de deep learning (DL) proposées sont supérieures à celles des algorithmes ML traditionnels en tant que modèles de détection avec une grande précision, un taux de détection idéale et un taux de fausses alarmes négligeable.

Mots clés : Cybersécurité, Système de détection des intrusions (IDS), L'apprentissage profond, L'apprentissage automatique (Machine learning), CICDDoS2019.

Abstract

Intrusion detection system (IDS) has been the subject of much research and play an important role in cybersecurity. The aim of this study is to model such a system to help system administrators to detect and identify any security violation in their organizations in order to prevent them before causing any damages. For that, we have studied the performance of machine learning (ML) methods applied in intrusion detection for cyber security. Then, we applied three detection techniques based on deep learning approaches, a deep neural network (DNN), a convolutional neural network (CNN) and a recurrent neural network (RNN) for detecting intrusions in network connection. We have evaluated the proposed methods with the benchmark CICDDoS2019 dataset for network DDoS attack detection. We also presented a comparative study with four machine learning algorithms in literature, using different measures applied for the evaluation of machine learning performances (Precision, Recall, F1-score), and two other important performance indicators for intrusion detection (detection rate, false alarm rate). The Experimental results showed that the performance of the proposed Deep learning (DL) approaches is superior than traditional ML algorithms as detection models with high precision, an ideal detection rate and a negligible false alarm rate.

Key words : Cybersecurity, Intrusion detection system (IDS), Deep learing (DL), Machine Learning (ML), CICDDoS2019.

Sommaire

Table des figures

Liste des tableaux

Introduction générale

Avec l'évolution des réseaux et en particulier les réseaux internet, les techniques de l'information et de communication nous offrent actuellement des facilités incontournables en matière de l'apprentissage à distance, l'achat et le paiement en ligne, la communication via les messageries instantanées et les vidéos conférences et bien d'autres technologies émergent comme les distributeurs automatiques, les véhicules autonomes . . .etc.Cependant, des nouvelles vulnérabilités de sécurité ont été développées avec l'omniprésence de ces outils informatiques.

Les réseaux de nos jours ainsi que les systèmes d'information connectés ont confronté des véritables menaces intentionnelles ou accidentelles. Il semble qu'il ne se passe pas un jour sans qu'une nouvelle histoire concernant les défis de la cybersécurité ne soit publiée, qu'il s'agisse du piratage de la vie privée via les réseaux sociaux, les fraudes par carte de crédit, l'espionnage économique, l'infection de système informatique est critiqué par les attaques de déni de service et beaucoup d'autres cybers menaces posent des problèmes et des défis majeurs dans les années prochaines. Dernièrement en 23 mars 2020, les hôpitaux de Paris ont été des victimes d'une cyber attaque par déni de service en pleine crise du Covid-19 (coronavirus). Ces cybers menaces de la cybersécurité sont considérés aujourd'hui ; une des principales préoccupations et le domaine le plus chaud des dépenses informatiques.

La cybersécurité est l'ensemble de techniques et de pratiques qui consistent à protéger tous les aspects concernant les technologies de l'information, y compris l'accès aux données, le stockage, le traitement et la protection des données, la transmission et la liaison des données). Elle est aussi connue sous le nom de la sécurité de l'information électronique.
Le système de détection d'intrusion (IDS) est une invention qui répond à ces exigences. Il est adopté pour le but d'empêcher toutes les menaces imminentes de violation des politiques de sécurité, des réseaux et des systèmes informatiques. le noyau de ce système est un module de reconnaissance des intrusions efficaces, robustes et évolutifs. Il est aussi l'élément clé de tout produit de cybersécurité. Les modules de reconnaissance des intrusions décident si un objet est une menace ou une utilisation légitime en fonction des

données qu'ils ont recueilli sur lui.
Avec l'évolution des cyberattaques en complexité, en volume et en fréquence, l'utilisation des méthodes classiques de détection n'est plus pratiqué et de nouvelles technologies de protection avancées étaient nécessaires. De ce fait,les entreprises de cybersécurité sont tournées vers l'utilisation de l'apprentissage machine(ML), un domaine de l'intelligence artificiel (AI) qui avait été utilisé avec succès dans la reconnaissance d'image, la recherche et la prise de décision, afin de renforcer l'efficacité de leurs produits face à divers problèmes de détection des cyberattaques comme la détection des intrusion, la détection des logiciels malveillants, et surtout aux problèmes de sécurité liés aux infrastructures critiques comme la sécurité du système électrique, les systèmes de contrôle industriel, la détection des intrusions dans les systèmes SCADA . . .etc.
L'apprentissage profond (Deep Learning) qui fait partie de l'apprentissage machine est un domaine très prometteur pour la cybersécurité avec la disponibilité des grandes quantités de données des cyberspaces. Les méthodologies traditionnelles d'apprentissage machine reposent sur l'ingénierie et la sélection des caractéristiques de domaine étudié, ce qui nécessite une bonne expertise afin d'accomplir ces taches.

Dans cette étude, nous étudions plusieurs approches de Deep Learning qui ont été déjà appliqués aux domaines de la détection des intrusions avec des nouveaux défis de la cybersécurité. Notre mémoire est organisé comme suit :

Le premier chapitre de notre travail est consacré à la présentation d'un système de détection d'intrusion, leurs principes de fonctionnement, ainsi que des différents concepts relatifs aux détections d'intrusion.

Le deuxième chapitre comprend l'apprentissage profond (deep learning, DL) pour la détection d'intrusion, quelques méthodes de DL et ses principes de fonctionnement.

Le troisième chapitre présente une synthèse des travaux reliés à la détection d'intrusion basé sur l'‘apprentissage profond.

Le quatrième chapitre présente les méthodes d'apprentissage profonds élaborée, les mesures d'évaluation à prendre en compte dans l'application de ces approches pour la détection d'intrusions, ainsi que tous les résultats obtenus.

La problématique

Avec l'explosion des données des différentes infrastructures informatiques (réseaux, systèmes, internet des objets ...etc). Les systèmes de détection d'intrusion confrontent des défis majeurs face à plusieurs formes des cyberattaques ainsi de plusieurs formes d'utilisation légitimes de ces infrastructures. Comment assurer l'efficacité de ces systèmes en termes de taux de détection et d'identification de toutes types d'activités malveillantes avec un taux de fausses alarmes plus petit. Comment s'assurer que les méthodologies de détection soient robustes et évolutifs.

Le but de travail

L'objectif de cette étude est d'implémenter plusieurs méthodes de détection d'intrusions en se basant sur les approches de Deep Learning et évaluer les performances des systèmes élaborés. Nous utilisons une nouvelle dataset CIC-DDoS-2019, qui représente un trafic réseau réel contenant plusieurs types des attaques DDoS malveillantes les plus répandus, pour aider les administrateurs des réseaux et des systèmes à détecter et identifier toute violation de la sécurité réseaux.

CHAPITRE 1 :
Un Système de Détection d'Intrusion. (IDS)

Un Système de Détection d'intrusion

1.1 Introduction

Avec le développement rapide de la technologie des réseaux et en particulier les réseaux sans fil, la sécurité de ces réseaux ainsi que ses terminaux connectés contre diverses menaces intentionnelles ou accidentelles, est devenue un problème crucial.
Toutes les informations concernées par les technologies Internet, les informations stockées dans des bases de données et qui sont transmises sur le réseau doivent être protégées. Les intrusions sont des véritables menaces qui peuvent être des activités non autorisées ou des utilisations malveillantes des ressources d'information qui offensent les politiques de sécurité.

Les systèmes et les techniques traditionnelles de prévention des intrusions comme les pare-feu, le cryptage et le contrôle d'accès sont dans la plupart du temps inefficaces face à l'évolution des nouvelles menaces sophistiquées.
Comment surmonter les défis de la cybersécurité, identifier les intrusions et protéger nos données est un problème clé qui ne doit jamais être contourné.
Un nouveau concept de détection d'intrusion a été proposé par James EAnderson en 1980, dans le but d'identifier toute activité non autorisée dans un réseau [18].

Dans ce chapitre, nous avons présenté une introduction aux systèmes de détections d'intrusions (IDSs), où nous avons défini le modèle de base de ces systèmes. Puis nous avons détaillé la taxonomie des IDSs et présenter une analyse des différentes techniques possibles de la détection, ainsi les différentes mesures d'évaluation des systèmes IDSs.

1.2 Définition d'un système de détection d'intrusion

Dans les systèmes d'information, On peut définir les intrusions par toutes les activités qui violent la politique de la sécurité du système.[38]. l'intrus ou l'attaquant essayant de trouver un moyen d'obtenir un accès non autorisé aux informations, de causer des

dommages ou de se livrer à d'autres activités malveillantes.

La détection d'intrusion est le processus de monitoring des événements survenant dans un réseau ou sur un système informatique et les analysant pour des signes des menaces imminentes de violation des politiques de sécurité des systèmes informatiques ou des pratiques de sécurité standard [42]. Un système de détection d'intrusion est un ensemble de composantes matérielles et logicielles conçues pour automatiser le processus de détection d'intrusion.

Ce type de logiciel de cybersécurité est lié à la sécurité du réseau de la manière d'un pare-feu (firewall). Il est considéré comme une deuxième ligne de défense pour identifier les différentes activités malveillantes qui ne peuvent pas être identifiées par un pare-feu traditionnel. Il surveille de manière dynamique les événements qui se produisent dans un système et décide si ces événements sont des signes d'attaque où constituent une utilisation légitime du système.
Ces systèmes sont devenues une module de sécurité indispensable pour augmenter le niveau de sécurité et détecter les menaces avant qu'elles ne causent des dommages étendues grâce à ses performances et ses puissances dans la reconnaissance des intrusions.

1.3 Le modèle de base d'un système de détection d'intrusion

Le système de détection d'intrusion se compose de plusieurs outils ou chaque outil à sa propre tache dont l'objectif général est la détection d'intrusion au premier temps, et ensuite informer l'opérateur ou le personnel informatique de la possibilité d'une intrusion dans le réseau. Un modèle générale pour la structure d'un système de détection d'intrusion a été proposé par IDWG (*Intrusion Detection Working Group*) de l'IETF englobe et standardise la structure d'un système de détection d'intrusion. La figure suivante montre en détail les différents composants de ce système.

Les IDSs peuvent ne pas avoir tous ces composants totalment séparés comme indiqué dans la figure 4.1. Certains IDS combineront ces composants en un seul module ; certains autres auront plusieurs instances de ces modules [52].

- **L'administrateur** :c'est le responsable de l'établissement de la politique de sécurité de l'organisation qui déploie et configure les différents composants d'IDS. il prend en charge la déclaration prédéfinie des activités qui sont autorisées à se dérouler sur le réseau ou sur des hôtes particuliers pour répondre aux besoins d'un système d'information.

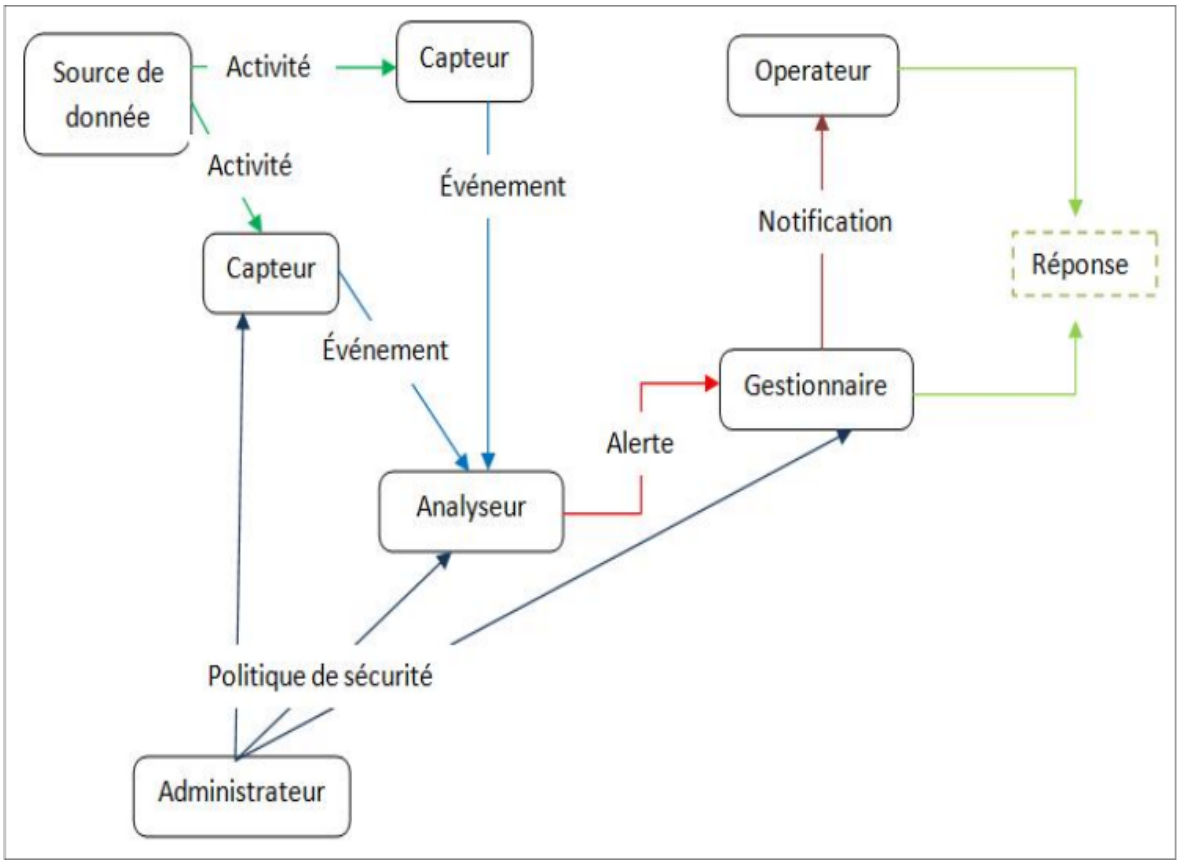


FIGURE 1.1 – Un modèle fonctionnel du Système de détection d'intrusion proposé par l'IDWG (Wood and Erlinger, 2012 [52])

- **La source de données :** Il existe différents types de données provenant de plusieurs sources (réseau, système, application et alertes). le Système IDS n'a pas de restriction sur les sources de données utilisées, il utilise pour cela des capteurs appropriés pour analyser les informations survenant de ces sources pour détecter les activités non autorisées ou non désirées.
- **Le capteur et l'analyseur :** sont des composants clés du système, au début, le capteur va accéder aux données brutes et collecter toutes les informations des activités survenant et les transférer à l'analyseur comme des évènements (séquence des activités). Ensuite, ce dernier va analyser ces événements pour signaler les activités non autorisées ou indésirables ou les événements qui pourraient avoir un intérêt pour l'administrateur de sécurité. Dans la plupart des IDSs existants, le capteur et l'analyseur font partie d'un même composant.
- **Le gestionnaire :** c'est aussi un composant clé et le moyen qu'à partir de lui, l'opérateur gère les différents composants du système. Les fonctions du gestionnaire comprennent généralement (mais ne sont pas limitées à) la configuration du capteur, la configuration de l'analyseur, la gestion de la notification d'événements, la consolidation des données et la gestion des rapports.

- **La réponse :** c'est les mesures prises comme réponse à un événement. Elle peut être effectuée automatiquement par une entité dans l'architecture de l'IDS comme elle peut être initiée par un humain. L'envoi d'une notification à l'opérateur est une réponse très commune. Autres réponses incluent (mais ne sont pas limités à) la journalisation de l'activité, l'enregistrement des données brutes (à partir de la source de données) qui ont caractérisé l'événement, l'arrêt du réseau ou de l'utilisateur ou la session de l'application, la modification des contrôles d'accès réseau ou système.

1.4 Taxonomie des IDSs

Il existe différents types de technologie des systèmes de détection d'intrusions, caractérisés par différentes approches de l'architecture du système, l'environnement de déploiement, les techniques surveillances et les stratégies de détection.
Il y a plusieurs taxonomies des IDSs proposé dans la littérature Un nouvelle perspective de ces taxonomies des IDSs a été introduite par Liao et al. (2013) [33] et adoptée selon différents critères basés sur des termes largement acceptés. Elle est présentée par 4 critères principales de classification comme elles sont illustrés dans la figure 1.2 :

1.4.1 Les sources des données à analyser

1. **Le type de données :** Les informations ou les données d'entrées à analyser par le système sont des caractéristiques essentielles des IDSs avant d'entamer le processus de détection,les données proviennent des trafic réseau filière ou sans fils, les Systèmes IDS se basent sur ces données sont appelés Network-based IDS (NIDS). Elles peuvent aussi être des données des machines hôtes comme les logs générées par le système d'exploitation ou par les applications, les Systèmes IDS basés sur ces données sont appelés Host-based IDS (HIDS). Un autre type d'IDS apparaît récemment c'est le Cloud-based IDS basé sur les données des cloud computing [54].
2. **La collection des données :** Deux architectures de systèmes différentes :
   - Centraliser : la source de données et le système de détection sont tous ensemble
   - Distribuer : les données sont collecté à l'aide de plusieurs capteurs situent en différents emplacements
3. **L'outil de collection :** Leur rôle est d'accéder aux données brutes, et les filtrer pour ne renvoyer que les informations intéressantes à un analyseur d'IDS. Cet outil peut être un capteur extérieur du système, ou un agent logiciel fonctionne sur le système.

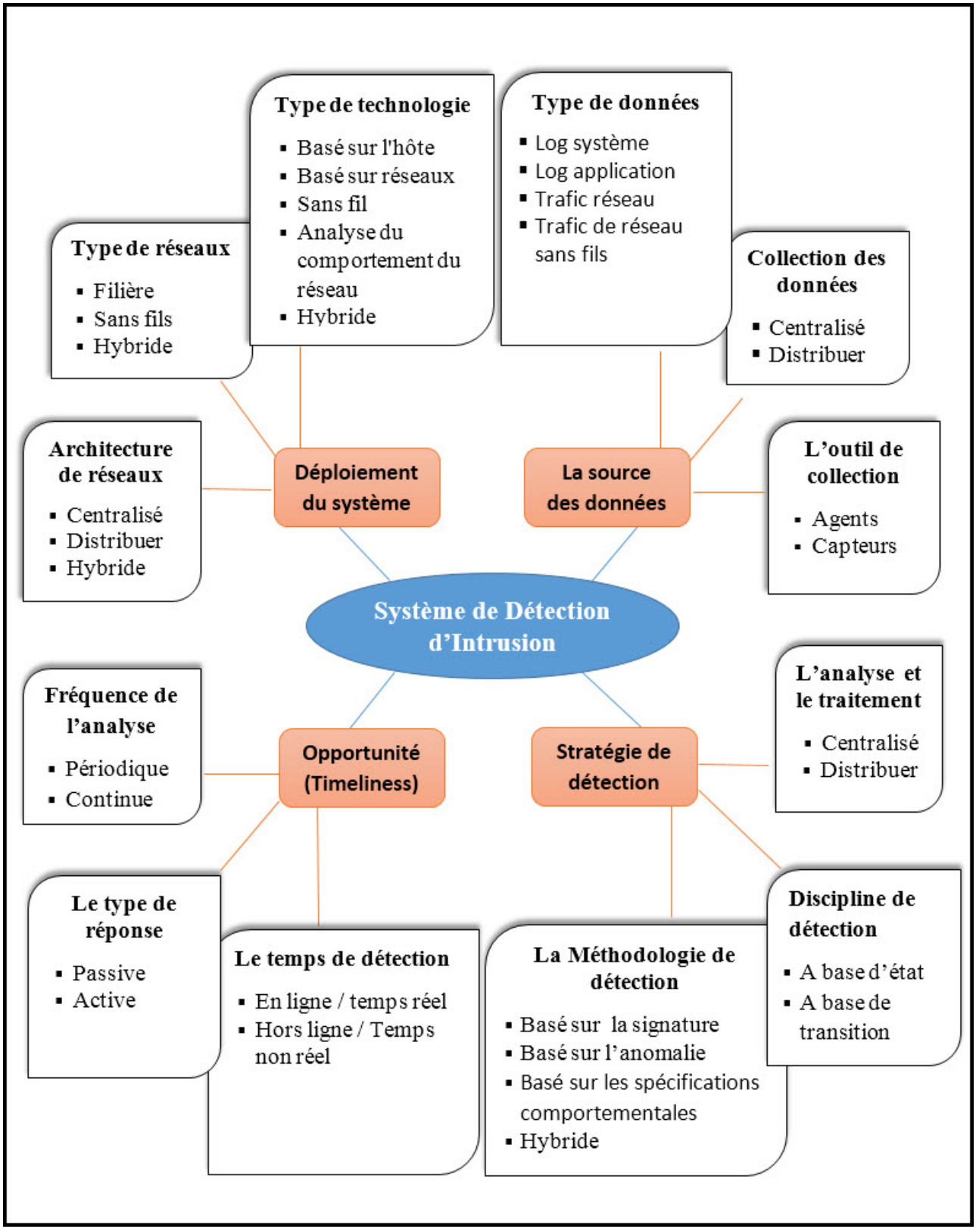


FIGURE 1.2 – Taxonomie des systèmes de détection d'intrusion proposé par Liao et al. (2013) [33])

1.4.2 La stratégie de détection

1. **L'analyse et le traitement :** c'est l'architecture de traitement utilisée "centralisé" ou "distribués".
2. **Discipline de détection :** Les outils de détection peut baser sur :
   - L'etat : reconnaître l'état sécurisé ou l'état non-sécurisé afin de détecter les intrusions.
   - Les transition : reconnaître certaines transitions qui conduisent à un état non-sécurisé.
3. **La Méthodologie de détection :**
   - La détection d'anomalies (Anomaly based détection) : Définir un modèle d'un comportement normal (profil) de l'entité à surveiller (trafic réseau, service, application . . .) et toute déviation significative entre le comportement observé et le modèle est potentiellement suspect et considéré comme une anomalie
   - La reconnaissance de signature (Signature based detection) : Basée sur les techniques d'appariement de motifs pour détecter l'intrusion Lorsqu'une signature d'intrusion correspond à une signature précédente qui existe déjà dans la base de données de signatures, un signal d'alarme sera déclenché.
   - Détection basé sur les spécifications (Specification-based detection) : si le système connaît auparavant les spécifications du protocole elle va signaler toutes les utilisations incorrectes de ce protocole comme activités malveillantes ;

     La première méthode a un taux de faux positifs élevé (fausse alarme), ainsi que la deuxième méthode incapable de détecter des nouvelles intrusions qu'ils n'existent pas dans la base de données, la dernière méthode est incapable de détecter les attaques qui se ressemblent à des utilisations bénins de protocole. Pour cela, un système IDS hybride combine plusieurs méthodologies pour fournir une détection plus étendue et précise.

1.4.3 L'Opportunité (*Timeliness*)

1. **Le temps de détection :** C'est le temps entre l'événement analysé et la détection, elle peut être une détection en temps réel (On-line Detection), ou bien une détection en temps non-réel (Off-line Detection).
2. **Le type de réponse :**
   - Passive : une réponse passive à l'intrusion s'agit d'une alarme ou bien d'un message qui notifie l'administrateur (l'opérateur) système lorsqu'une attaque est détectée.

- Active : une réponse active prend de plus une réponse passive des mécanismes et des mesures permettant une réponse automatique pour stopper l'attaque en cours, telle que le blocage de son adresse IP.

3. **La fréquence d'analyse :**
   - Périodique : qui sauvegarde une quantité de données dans une période de temps et ensuite commencer les traitements de détection sur elle.
   - Continue : la détection s'effectue de manière continue sur toutes les activités et les événements qui arrivent pour augmenter le niveau de sécurité dans des contextes sensibles.

1.4.4 Déploiement du système

De nombreux types de technologies d'IDSs selon l'endroit et l'environnement où ils sont déployés pour inspecter les activités suspectes ainsi que les types d'événements qu'ils peuvent reconnaître.

1. **Architecture de réseaux :** Liés au nombre de systèmes IDSs et à la corrélation des données entre eux, il existe 3 Architectures :
   - Systèmes centralisés : Un seul système qui prend la charge de collecter les données et identifier les intrusions
   - Systèmes Distribué : Les données sont collectées par plusieurs IDSs permettant la corrélation des données entre eux pour identifier les attaques distribuées
   - Systèmes hybrides
2. **Type de réseaux :** Dépend de l'interconnexion de l'IDS avec les systèmes de surveillance. Il peut être une connexion filière, sans fils, hybride
3. **Type de technologie :** L'adoption de plusieurs types de technologies IDS peut atteindre l'objectif d'une détection plus complète et plus précise.
   - IDS basé sur l'hôte : surveille et collecte les caractéristiques des hôtes contenant des informations sensibles, des serveurs exécutant des services publics et des activités suspectes.
   - IDS basé sur réseaux : capture le trafic réseau sur des segments de réseau spécifique via des capteurs, puis analyse les activités des applications et des protocoles pour reconnaître les incidents suspects.
   - IDS Sans fil : similaire à l'IDS basé sur réseaux, mais il capture le trafic réseau sans fil, comme les réseaux Ad-hoc, les réseaux de capteurs sans fil et les réseaux maillés sans fil.
   - Système basé sur l'analyse du comportement du réseau : cette classe des systèmes se diffère à la classe précédente (IDS basé sur des réseaux), ce système

inspecte le comportement du trafic réseau pour reconnaître les attaques avec des flux de trafic inattendus, comme DDos Attaques, malware et des services AP inattendus.

- Système Hybride : adoptée Les technologies précédentes pour atteindre l'objectif d'une détection plus complète et plus précise.

1.5 Les techniques de la détection d'intrusion

Le noyau d'un système de détection d'intrusion est un module de reconnaissance de toutes activité malveillantes. La première génération des systèmes se base sur les connaissances des experts de la sécurité pour identifier les attaques, après plusieurs approche de détection ont été développer pour construire un système de détection très précis et plus performant.

1.5.1 Les approches de détection

Les approches de détection des intrusions se divisent en deux groupes principaux : la détection des anomalies et la détection des signatures. Cependant, ces deux classes n'ont pas une différence considérable dans leurs caractéristiques pour voir l'ensemble des propriétés des approches de détection. une classification de cinq sous-classes a été proposée par Liao et al. (2013) [33]) avec une perspective approfondie de leur caractère Statistique : basée sur des statistiques, des modèles, des règles, des états et sur l'heuristique.

Les approches basées sur les statistiques consistent principalement en des caractéristiques de données statistiques telles que le seuil prédéfini, la moyenne, l'écart type et les probabilités d'identifier les intrusions. La détection basée sur les modèles se concentre sur les techniques de classification de modèle avec des attaques connues. Les techniques basées sur des règles sont principalement appliquées une base des règle comme "si-Alors" ou si-alors-Sinon pour construire le modèle et les profils des intrusions connues. Les méthodes basées sur l'état exploitent la machine à état fini dérivé des comportements de réseau pour identifier les attaques telles que l'analyse de protocole et le modèle de processus Markove.

La dernière est celle de l'approche heuristique, appliquant les techniques d'intelligence artificielle qui s'inspire des concepts biologiques tels que le système immunitaire, les algorithmes génétiques et l'intelligence en essaim. La plupart des travaux récents consistent à combiner ces différentes approches de détection en une approche sophistiquée pour donner plus de précision et une meilleure efficacité [33].

1.5.2 Les mesures d'évaluation de l'IDS

Les mesures qui nous permettent d'évaluer l'efficacité globale des systèmes de détection d'intrusion selon (Debar et al. [12]) sont :

La précision : le système IDS est précis lorsqu'il détecte les attaques sans faire des fausses alarmes. La non-précision survient lorsqu'il déclare comme anormale ou instructive une action légitime dans l'environnement.

La performance de traitement : est mesurée par la vitesse dans laquelle les événements sont traités. Lorsque le système IDS est plus performant alors la détection en temps réel sera possible.

La complétude : c'est la capacité d'un IDS de détecter toutes les attaques.

La tolérance aux pannes : la plupart des systèmes de détection d'intrusion s'exécutent dans des systèmes d'exploitation ou des matériels qui sont connus pour être vulnérables aux attaques. Donc un IDS devrait être résistant à ces attaques en particulier les attaques de déni de service.

La rapidité : L'IDS doit être plus rapide dans l'analyse et l'exécution pour minimiser le temps de réagir, et aussi pour empêcher l'attaquant d'altérer la source de vérification ou interrompre le fonctionnement du système [12].

Généralement, un taux de détection élevé est nécessaire pour un système IDS afin de prévenir les attaques avant de causer tout type de violations de sécurité pour les systèmes IDSs basés sur la machine learning une précision de détection élevée avec un faible taux de fausses alarmes est essentiel pour l'efficacité des Systèmes [46]. Les principaux aspects à considérer lors de la mesure de détection et la précision de classification des attaques sont :

- True Positive (TP) : nombre d'intrusions correctement détectées
- True Negative (TN) : nombre de non-intrusions correctement détectées
- False Positive (FP) : nombre de non-intrusions mal détectées
- False negative (FN) : nombre d'intrusions mal détectées

Il existe plusieurs types d'erreurs venant d'un détecteur, influençant plus ou moins sa puissance. Les vrais positifs sont les cas où une alarme se déclenche quand il y a une violation des politiques de sécurité. Les vrais négatifs sont les cas où aucune alarme ne se déclenche et rien d'anormal ne se produit. Les faux positifs sont les cas où une alarme se déclenche alors qu'il ne se produit rien d'anormal. Les faux négatifs sont les cas où une alarme ne se déclenche pas alors qu'il se produit une chose anormale. À la première vue, on pourrait supposer qu'un faux positif est moins dangereux qu'un faux négatif [32].

CHAPITRE 2 :
L'apprentissage automatique pour la cybersécurité

L'apprentissage automatique pour la cybersécurité

2.1 Introduction

L'apprentissage automatique est un domaine de l'intelligence artificiel (AI) qui donne aux ordinateurs la possibilité d'apprendre sans être explicitement programmé. Les chercheurs de ce domaine tentent d'imiter au maximum le fonctionnement du cerveau humain et les modes de traitement de l'information et de communication observée dans le système nerveux biologique, pour donner aux machines la capacité d'apprendre depuis les données, les interprètes et prendre des décisions éclairées.
Les méthodes d'apprentissage automatique ont été appliquées avec succès dans des plusieurs produits des TIC(la reconnaissance d'image, traduction automatique, diagnostic médical, ...etc), ainsi que d'autres différents secteurs technologiques de ces dernières années (voiture autonome, robots intelligents, ...etc).
Cependant, les performances de ce dernier reposent implicitement sur la qualité des données d'apprentissage, elles exigent une étape critique appelée l'ingénierie des caractéristiques (**Feature Ingeneering**), Elle se définie comme une méthode dictée par les experts du domaine pour sélectionner les caractéristiques ou les propriétés importantes des données de chaque problème. Pour cela, et avec la disponibilité du big-data, un nouveau procès de l'apprentissage automatique appelé le Deep learning (l'apprentissage profond) a été utilisé pour apprendre la représentation et abstraite implicitement les caractéristiques [9, 28].

Dans ce chapitre, nous avons commencé par l'importance du deep learning pour la détection d'intrusion. Ensuite, nous avons présenté une définition et une classification des déverses méthodes de DL. Nous avons détaillé aussi trois approches de DL qui feront le sujet de notre travail. A la fin de chapitre, nous avons cité les différents ensembles de données utilisées pour l'évaluation des systèmes IDSs basé sur l'apprentissage automatique.

2.2 L'apprentissage profond pour la cybersécurité

Avec la disponibilité de grandes quantités de données de la cyber infrastructure, des réseaux, des systèmes d'exploitation ou des systèmes d'informations et pour relever les défis de la cybersécurité, des méthodes et des techniques comme l'apprentissage automatique (machine learning), data mining, les statistiques et d'autres capacités interdisciplinaires ont été exploité [15].
L'apprentissage profond qui fait partie de l'apprentissage automatique pourrait être utilisé pour les Systèmes IDSs basés sur la signatures ou basés sur la détection des anomalies. Ces méthodes de classification et de prédiction peuvent être utilisées pour détecter des motifs et des comportements inhabituels des diverses cyberattaques qui permettent une cyber réponse en temps réel. Ils ont la capacité de détecter les attaques lorsqu'elle s'est produite et aussi de la capacité de prédire les futures attaques potentielles [35].
Les méthodes basées sur l'apprentissage approfondie peuvent aider à surmonter les défis liés au développement d'un IDS efficace [17, 41].
D'un autre coté, la collection des données et des trafic réseau ont conduit à un problème de big-data les experts en sécurité souhaitent toujours de meilleures performances IDS qui ont un taux de détection la plus élevée et un taux de fausses alarmes le plus bas. Par conséquent, les approches de deep learning qui s'adapte bien à une très grande quantité de données. Ces derniers sont été introduites pour la détection des anomalies de réseau dans le but de différencier les comportements normaux et des comportements anormaux afin de détecter des activités malveillantes ou suspectes d'être malveillantes [45].

2.3 Définition de l'apprentissage profond

L'apprentissage profonde (Deep learning ou DL) appartient à une classe de techniques d'apprentissage automatique (machine learning ou ML), il obtient un grand succès dans de nombreuses tâches de l'intelligence artificielle (IA) par rapport aux algorithmes de ML classiques. Les architectures des modèles profondes sont relativement récentes où de nombreuses étapes de traitement non linéaire de l'information sont exploitées, dans lesquelles les informations sont traitées en couches hiérarchiques, chacune recevant et interprétant les informations de la couche précédente pour l'apprentissage des représentations de données [13].

2.3.1 Fonctionnement

Généralement, l'architecture des réseaux profondes est organisée en couches de neurones pour n'importe quel type de ces réseaux ; une Couche d'entrée (Input Layer), une ou plusieurs Couches cachées (Hidden Layers) et une Couche de sortie (Output Layer).

Chaque paire de couches voisines est connectée. Les connexions entre eux appelées poids (Weights). Les "neurones" d'une même couche généralement appelés "nœuds" n'ont aucune association, la figure 2.1 illustré une architecture standard d'un modèle de réseau de neurones profond.

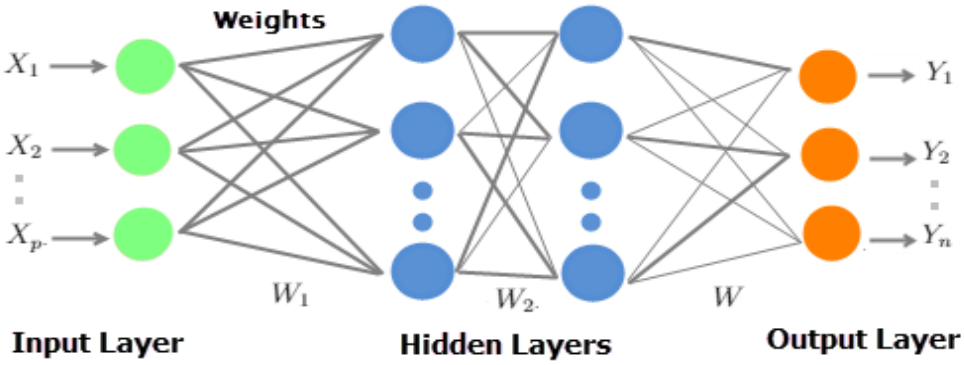


FIGURE 2.1 – L'architecture d'un modèle Deep Learning [34]

L'apprentissage profond se présente comme un système de calcul avancé, il est constitué d'une variété de techniques issues du domaine de l'apprentissage automatique qui utilisent un déluge de neurones (nœuds) non linéaires disposés en plusieurs couches de traitement qui extraient et convertissent des valeurs de variables d'entité à partir du vecteur d'entrée pour créer plusieurs niveaux d'abstraction afin de représenter les données [14].
L'apprentissage du DNN c'est l'optimisation des paramètres de poids et le paramètre de biais entre deux couches voisines, Il évalue la justesse du modèle et permet de mieux l'adapter aux données d'apprentissage ses besoins. Lorsque le modèle arrive à une précision maximale avec des paramètres optimaux, il sera généralisé pour les données réelles. La quantité et la qualité des données d'entraînement déterminent le degré d'apprentissage et donc la précision des modèles obtenus.

2.3.2 classification des méthodes DL

En pratique, toutes les approches d'apprentissage profond sont des réseaux de neurones (Neural networks), qui partagent certains propriétés de base communes. Ils sont tous constitués de neurones inter-connectés, ils sont organisés en couches. Ce qui les différencient, c'est l'architecture du réseau (où la manière dont les neurones sont organisés dans le réseau) et parfois la manière dont ils sont formés [31]. Ferrag et al. [16] ont présenté une étude et analysé 10 différents approches du Deep learing les plus utilisées pour la détection des intrusions dans la cybersécurité. Ces approches peuvent être classées en trois modèles, en fonction de la manière dont elles sont formées et destinées à être utiliser.

- Deep learning pour l'apprentissage supervisé : Il est utilisé lorsque les données d'étiquette cible sont disponibles, il s'agit des modèles profonds discriminatoires à sa savoir le Deep neural networks (DNNs), Recurrent neural networks (RNNs), Convolutional neural networks (CNNs).
- Deep learning pour l'apprentissage non-supervisé : Il est Utilisé lorsque les données d'entrée ne sont pas étiquetées ,il s'agit des modèles génératifs visent à regrouper les données selon certains critères de similarité à des fins de reconnaissance ou de synthèse de modèles, à savoir le deep belief networks (DBN), deep autoencoders (DA), Restricted Boltzmann machine (RBM) et deep Boltzmann machines (DBM).
- Deep learning hybrides : une combinaison hybride de ces modèles mentionnés ci-dessus. Les réseaux profonds non supervisés pourraient fournir une excellente initialisation sur la base pour laquelle la discrimination (l'apprentissage supervisé) pourrait être examinée.

2.4 Quelques méthodes d'apprentissage profond

Les réseaux neuronaux profonds sont un ensemble de neurones organisés en une séquence de couches inter-connectés. Ce qui les différencie, c'est l'architecture du réseau (la manière dont les neurones sont organisés dans le réseau et la manière dont ils se fonctionnent. Parmi de nombreuses implémentions de modèles d'apprentissage profond :

2.4.1 Deep Neural Network (DNN)

Les Deep Neural Networks (DNN) sont un ensemble de neurones organisés en une séquence de couches multiples appelée Multilayer Perceptrons (MLP). Ils se distinguent des réseaux neuronaux traditionnels (Artifical Neural Network) par leur profondeur et le nombre de couches, de nœuds (neurones) qui composent le réseau. Lorsqu'un ANN possède deux couches cachées ou plus, il est connu sous le nom de réseau neuronal profond. Ils tentent à modéliser des données contenant des architectures complexes en combinant différentes transformations non linéaires [3].

Le concept de base du perception a été introduit par Rosenblatt en 1958 [40]. La perception calcule une sortie unique à partir de multiples entrées à valeurs réelles (x_i) en formant une combinaison linéaire en fonction de ses poids (w) d'entrée, puis en plaçant la sortie via une fonction d'activation non linéaire. Mathématiquement, cela peut être écrit comme suit :

$$y = \delta(\sum_{n=1}^{n} W_i x_i + b) = \delta(W^T X + b) \tag{2.1}$$

Avec :

- W : est le vecteur des poids.
- X : est le vecteur des entrées.
- b : désigne le biais.
- δ : représente la fonction d'activation.

Un réseau typique de perception multi-couches (MLP) comprend un ensemble de nœuds sources formant la couche d'entrée, une ou plusieurs couches cachées de nœuds de calcul et une couche de sortie de nœuds. Le signal d'entrée se propage couche par couche sur le réseau. Le flux de signaux d'un tel réseau avec une couche cachée est illustré par la (figure 2.1).

Les réseaux DNNs sont généralement utilisés dans les problèmes d'apprentissage supervisé. La formation de modèle (l'apprentissage) signifie l'adaptation de tous les poids et les biais à leurs valeurs optimales.

2.4.2 Convolutional neural networks (CNNs)

Un réseau neuronal convolutionnel ou CNN est une extension des réseaux de feed forward traditionnels (FFN) dans le cadre de l'inspiration des facteurs biologiques [30]. Ceux-ci ont été initialement étudiés pour le traitement d'images dans lesquelles des motifs répétitifs peuvent être trouvés - par exemple, une image avec des bords répétitifs et d'autres motifs. Les CNNs surpassent tous les autres algorithmes ML classiques et fait un grand succès dans les tâches de traitement de vision par ordinateur (Computer Vision Tasks), ils ont des larges applications dans le traitement d'image et vidéo,le traitement du langage naturel (NLP), les systèmes de recommandation ...etc.
Les réseaux convolutifs sont particulièrement efficaces grâce à plusieurs types de couches spéciales : des couches de convolution, des couches groupement (Pooling) et de couches entièrement connectées [19], la figure 2.2 illustre un modèle d'un réseau conventionnel unidimensionnel (1D CNN).

Convolution Layers :

L'objectif de la convolution est d'extraire les caractéristiques de haut niveau. Il est constitué d'un ensemble de filtres (ou noyaux) apprenants, chacun représente une certaine fonctionnalité indépendante avec le volume d'entrée. Ces filtres sont constitué d'une couche de poids de connexion, ils ont un petit champ de réception (la taille du noyau), mais lors de la passe en avant (feed forward), chaque filtre est convolé sur la largeur et la hauteur du volume d'entrée, calculant le produit des points entre les entrées et les valeurs du filtre produisant une nouvelle carte de caractéristiques qui représente mieux l'information. En conséquence, le réseau apprend les filtres qui s'activent lorsqu'il

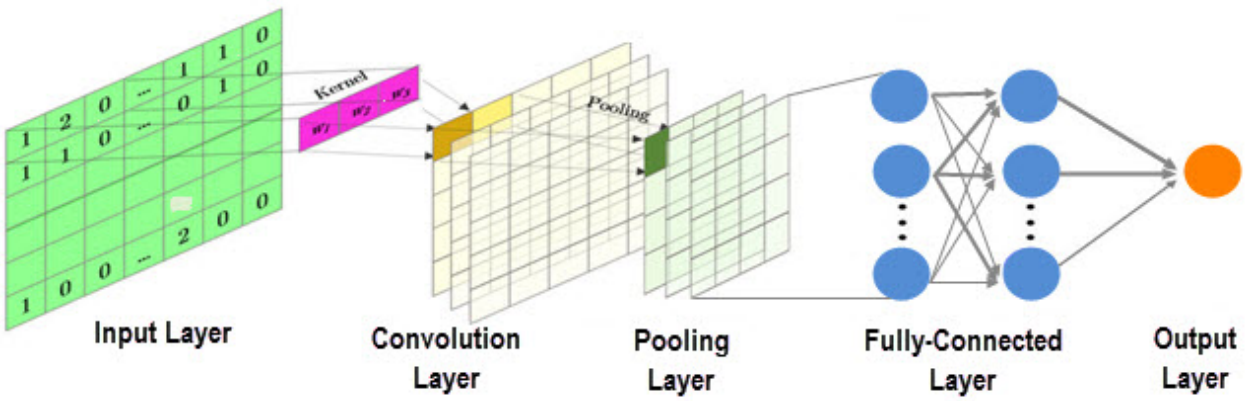


FIGURE 2.2 – L'architecture d'un modèle de réseau neuronal convolutif

détecte un type de caractéristique importante et spécifique à une certaine position spatiale dans l'entrée. La figure 2.3 présente une opération de convolution 1D avec une entré de dimension 1.

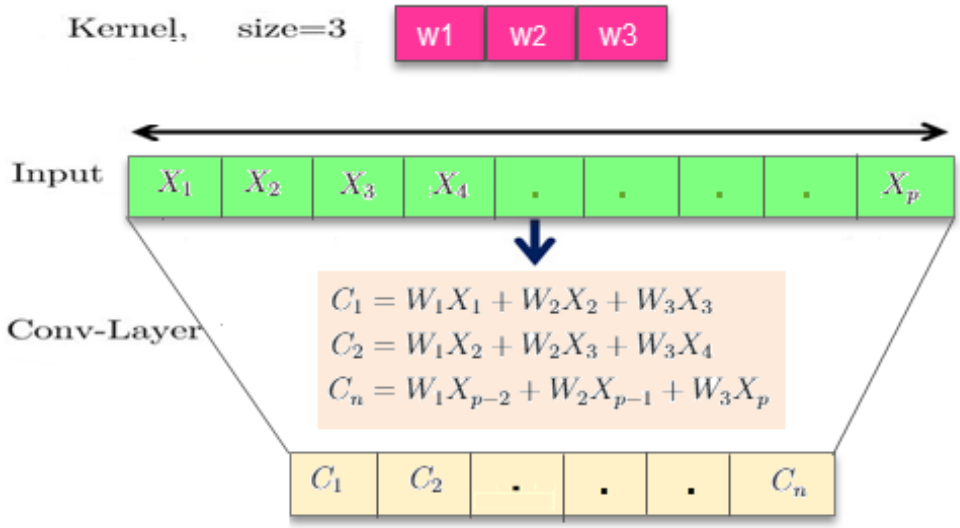


FIGURE 2.3 – Convolution

Une couche convolutionnelle partage le même noyau de convolution, ce qui réduit considérablement le nombre de paramètres nécessaires pour l'opération de convolution. Une fonction d'activation non linéaire sera appliqué immédiatement après chaque couche convolutionnelle. Les CNN profonds avec la fonction d'activation "Rectified Linear Units **ReLU**"

$$[f(x) = max(0, x)]$$

, renvoie x pour toutes les valeurs de $x > 0$ et renvoie 0 pour toutes les valeurs de $x <= 0$. s'entraînent plusieurs fois plus vite que leurs équivalents avec unités "Tanh Units" [29]

Pooling Layers :

Après la transformation ReLU, l'opération de mise en commun (Pooling) regroupe l'activation des neurones d'une couche en un seul neurone de la couche suivante. La couche de pooling fonctionne indépendamment sur chaque entité d'entrée, elle permet de réduire progressivement la taille des représentations afin de réduire le nombre de paramètres ou de poids, ce qui diminue le coût de calcul dans le réseau, tout en préservant les informations les plus critiques. Elle permet aussi de contrôler le sur-apprentissage.

Il peut utiliser deux méthodes de mise en commun différentes :

- La mise en commun maximale (Max-Pooling) : utilise la valeur maximale de chaque groupe de neurones de la couche précédente.
- La mise en commun moyenne (Average-Pooling) : utilise la valeur moyenne de chaque groupe de neurones de la couche précédente

Le Pooling est une forme de sous-échantillonnage non linéaire fonctionne de manière similaire à la convolution. le noyau de pooling se convolé sur le volume d'entrée et le diviser en un ensemble de région qui ne se chevauchent pas, et chaque sous-région produit une seule valeur en sortie qui est la valeur maximale pour Max-Pooling ou la valeur moyenne pour Average-Pooling la figure 2.4 décrit lopération de Max-Pooling avec un entré 1D et un noyau de taille 2.
la couche de Pooling n'a aucun paramètre pouvant être appris. De ce fait, ces couches ne

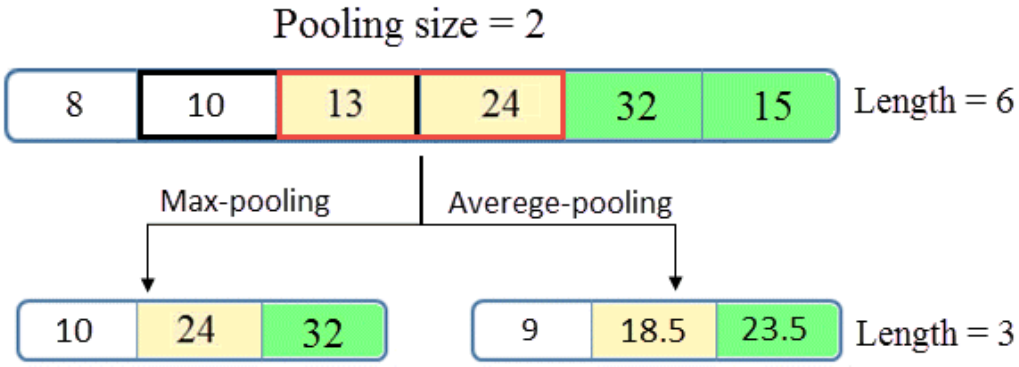


FIGURE 2.4 – Pooling

sont généralement pas incluses dans le nombre total de couches de réseaux de convolution.

Fully Connected Layers :

À la fin d'un réseau CNN, il y a une ou plusieurs couches entièrement connectées (chaque nœud de la première couche est connecté à chaque nœud de la couche suivante). Elles consistent à effectuer une classification basée sur les caractéristiques extraites des

convolutions. La couche finale contient une fonction d'activation Softmax, qui génère une valeur de probabilité de 0 à 1 pour chacune des étiquettes de classes que le modèle tente de prédire. Dans certaines architectures de réseaux CNNs récentes, les couches entièrement connectées peuvent se remplacer par plusieurs couches de mise en commun moyennes (average-pooling). Cela permet à ces réseaux de réduire considérablement le nombre total des paramètres et qui permet une meilleure prévention de sur-apprentissage [26].

2.4.3 Recurrent neural networks (RNNs)

les réseaux neuronal s'inspirent du fonctionnement des neurones biologiques du cerveau humain, ces neurones sont considère comme le centre de réflexion, et parfois ils doivent mémoriser certains évènements pour les utilisé ultérieurement avant de prendre la décision. Les réseaux neuronal traditionnel n'ont pas ce propriété, alors le fonctionnement d'un réseau de neurones récurrents (RNN) est motivé par le fait qu'un être humain raisonne en s'appuyant sur les connaissances qu'il a acquises et qui qu'il a mémorisé précédemment [11].
Les réseaux RNNs sont des réseaux de type Feed-Forward ayant un état interne (ou mémoire) qui prennent en compte tout ou partie des données vues précédemment (déjà fournies au réseau), en plus de la donnée vue actuellement pour adapter leur décision. L'idée clé de base de ces réseaux est le déploiement d'un calcul récurrent grâce aux boucles dans l'architecture du réseau. La sortie de réseau est une combinaison de son état interne (mémoire d'entrées) et le dernier l'entrée, au même temps, l'état interne change pour intégrer cette nouvelle donnée saisie. cela permet aux informations de persister en mémoire, comme le montre la figure 2.5.

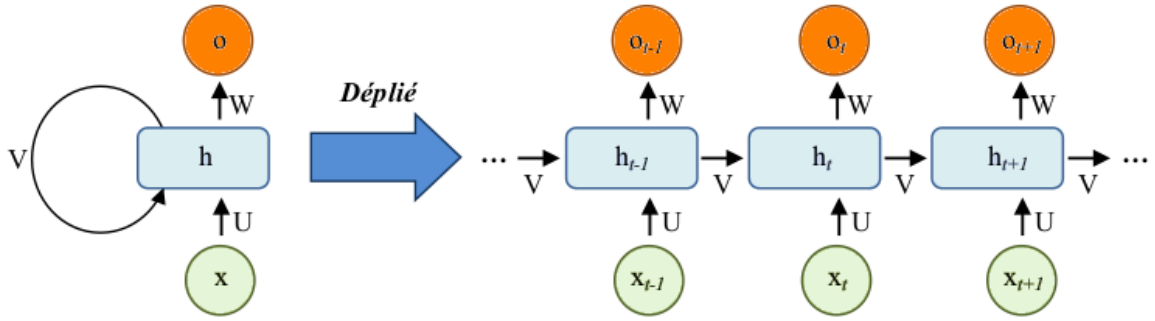


FIGURE 2.5 – L'architecture d'un modèle RNN

En raison de ces propriétés, les réseaux récurrents sont adaptés aux cas où la présence d'une forme n'est pas la seule information discriminante mais également un ordre d'apparition par exemple. ils sont de bons candidats pour les tâches qui traitent des données

séquentielles, telles que les données textuelles ou des données avec des caractéristiques temporelles. La description mathématique du processus de transfert de mémoire est comme suit :

$$h_t = \delta(Ux_t + Vh_{t-1} + b_h) \tag{2.2}$$

$$O_t = \delta(Wh_t + b_y) \tag{2.3}$$

ou :

- h_t : est l'état caché au temps t.
- x_t : est l'entrée au même temps t.
- U, V, W : sont les matrice de pondération, Input-to-Hidden, Hidden-to-Hidden et Hidden-to-Output respectivement (connue comme des matrice de transition).
- b_h : est la valeur du biais de l'état caché.
- b_y : est la valeur du biais de sortie.
- O_t : est la valeur de sortie au temps t.
- δ : est une fonction de non-linéarité appelées fonctions d'activation. (soit une fonction sigmoïde logistique ou tanh) qui est un outil standard de changement d'échelle pour condenser des valeurs très grandes ou très petites dans un espace logistique, ainsi que pour rendre les gradients exploitables pour la rétro-propagation.

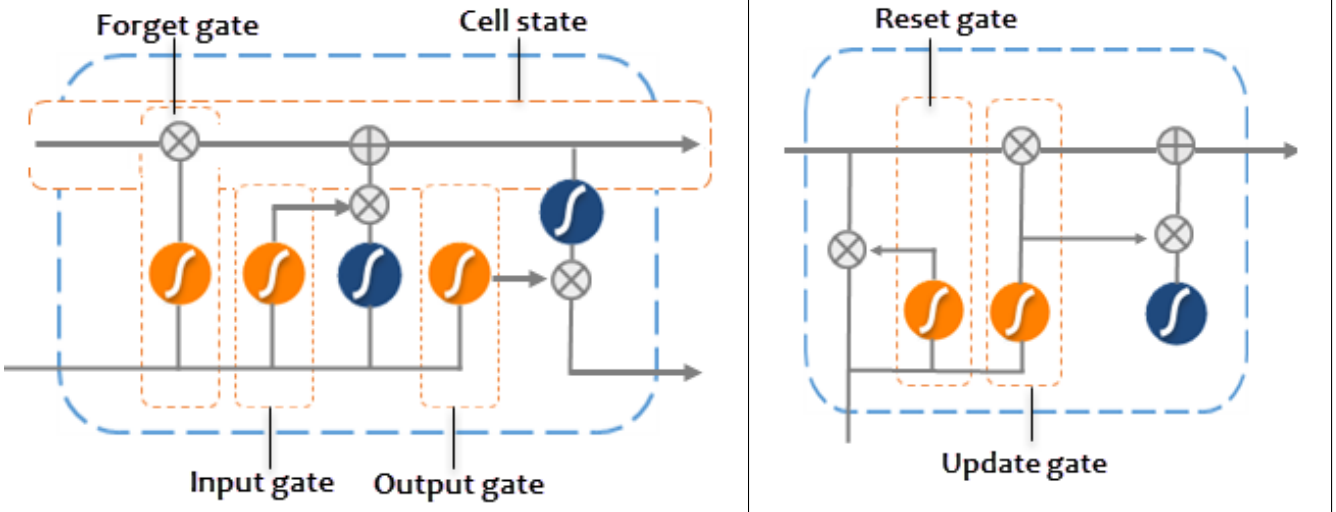
Un bloc de réseau neuronal, examine une entrée xt et génère une valeur o_t . Une boucle de rétroaction se produit à chaque pas de temps, chaque état caché h_t contient des traces non seulement de l'état masqué précédent, mais également de tous ceux qui ont précédé h_{t-1} aussi longtemps que la mémoire peut persister.

Unités de mémoire à court terme (LSTM)

Le réseau RNN a un long pas de temps car il prend en compte l'état sauvegardé précédent lors de la mise à jour du poids, les gradients lorsque l'entraînement devient de plus en plus petit et après quelques étapes, les erreurs n'ont pas pu être propagées à la fin du réseau. il n'y aura pas de différence significative dans le résultat, donc il ne peut pas faire de mise à jour des poids. Ce problème du RNN est appelé gradients de disparition (Vanishing Gradients). Pour surmonter ce problème, une architecture à mémoire longue et courte durée (LSTM) a été proposée au milieu des années 90 par les chercheurs allemands Sepp Hochreiter et Juergen Schmidhuber pour les réseaux neuronaux récurrents et aussi des étapes supplémentaires appelées Gated Recurrent Units (GRU). Ces étapes sont été utilisées pour améliorer les performances et la précision des RNNs.

L'idée clé de la méthode LSTM est l'état de la cellule. Elle a la capacité de supprimer ou d'ajouter des informations à l'état de la cellule. Cette technique est réglée par des

structures appelées portes (Gates). Ces derniers pourraient être une fonction sigmoïde où une valeur de 1 signifie que toutes les informations passent et une valeur de 0 signifie le contraire.
Les architectures LSTM et GRU se fonctionnent de la même manière. Cependant le GRU utilise moins de paramètres d'entraînement et donc moins de mémoires et s'entraînent plus rapidement que les LSTM. Alors que le LSTM est plus précis sur les ensembles de données utilisant une séquence plus longue.



(a) Cellule LSTM (b) Cellule GRU

FIGURE 2.6 – L'architecture d'un modèle LSTM_GRU

Les cellules LSTM sont les plus efficaces pour retenir les informations utiles lors de la rétro-propagation du gradient. Ce qui a leur permet de corriger les différences entre les prédictions sortantes et les catégories de référence en calculant le gradient de l'erreur pour chaque neurone, en allant de la dernière couche vers la première. La figure 2.6 illustre les quatre couches interactives (sigmoïde et tanh), les trois portes et les opérations Pointwise qui traitent le vecteur x à l'intérieur d'une cellule LSTM à un temps t.

2.5 Les Data-sets d'évaluation des IDSs basé sur Deep learning

Les Data-sets utilisés dans les travaux publiés pour l'application de l'apprentissage approfondi dans la cybersécurité jouent un rôle essentiel pour la validation de toutes approches DL proposés. Certains de ces ensembles de données ne sont pas facilement accessibles en raison de problèmes de confidentialité Les data-sets dédiés pour la détection des intrusions pour la cybersécurité peuvent être des [28] :

Data-set public	**Type**	**Étiqueté**	**Année**	**[Réf]**
KDD99	trafic du réseau	oui	1999	[49]
NSL-KDD	trafic du réseau	oui	2009	[37]
MAWI	trafic internet	oui	2011	[36]
ISCX dataset	trafic du réseau	oui	2012	[37]
CIC DoS dataset	trafic du réseau	oui	2017	[37]
Bot-IoT dataset	trafic IoT	oui	2018	[10]
CIC DDoS	trafic du réseau	oui	2019	[37]

Tableau 2.1 – Ensembles de données public relatives à la cybersécurité

- Benchmark Data-sets : Ces ensembles de données sont accessibles au public à des fins de recherche et pour l'évaluation des performances des algorithmes proposés.
- Data-sets Privé : Les données sont collectées à partir de sources publiques et en temps réel sont considérés comme des ensembles de données privés. Ces ensembles de données ne sont pas accessibles au public à des fins de recherche.
- En temps réel : Les données sont collectés dans un environnement en temps réel et sont considérés comme real-time data-sets.
- Collectés à partir de sources accessibles au public : Ces données sont collectés à partir de diverses sources accessibles au public. Dans la plupart des cas, ces Data-sets ne sont pas accessibles au public à des fins de recherche .

selon, [24, 16, 28], parmi les ensembles de données accessibles au public qui sont largement utilisés comme des benchmark il y a : DARPA, le KDD99, le NSL-KDD et l'ADFA-LD

2.6 Conclusion

le domaine du deep learning est très vaste.Il a une croissance rapide , des nouvelles algorithmes, architectures ou variantes apparaissent toutes les semaines. L'application des nouvelles méthodes de DL et l'évaluation des performances de diverses architectures DL existantes sont restées un objet et une orientation importante de la recherche pour les chercheurs en sécurité.

CHAPITRE 3 :
La détection d'intrusion basé sur le Deep learning

La détection d'intrusion basé sur le Deep learning

3.1 Introduction

Plusieurs approches de deep learning ont été étudié récemment pour la détection d'intrusion. Dans les systèmes de détection basé sur la signature, les intrusions sont détectées en comparant les comportements surveillés avec des motifs d'intrusion prédéfinis, tandis que les systèmes basés sur les anomalies se concentrent sur la connaissance du comportement normal afin d'identifier toute déviation et toutes activités suspects. Les méthodes de deep learning sont applicables pour les 2 types de détection grâce à ces capacités qui permettent d'extraire des niveaux plus élevés de relations non linéaires entre les données, afin d'identifier toute déviation d'une activité bénins. Cependant, ces méthodes exigent une quantité énorme des données dans le but de détecter et identifier les motifs de déférentes classes. Nous allons présenter ici quelques approches de DL qui ont été appliqué aux domaines de la détection des intrusions pour la cybersécurité.

Dans ce chapitre, nous avons étudié et analysé divers travaux de détection d'intrusion basés sur le deep learning. La synthèse comporte l'ensemble de données utilisé , l'architecture de réseaux et les mesures d'évaluation avec les résultats obtenus.

3.2 Travaux connexes pour la détection d'intrusion basé sur le DL

Des travaux antérieurs ont montré que les méthodes DL surpassent d'autres algorithmes d'apprentissage machine tels que la machine à vecteur de support (SVM) et les réseaux artificiels neuronales traditionnels (ANN) dans la détection des anomalies [23].

La détection d'intrusion basé sur l'apprentissage profond a été commencé en 2011, lorsque Salama et al. 2011 [41] ont présenté une approche hybride combinant le réseau de

croyances profondes (DBN) et SVM afin de classifier les intrusions du réseau en deux catégories : normale ou attaque. Le réseau DBN est composé de multiples couches de machines Boltzman restreinte (RBM) est utilisé comme une méthode de réduction de dimension pour obtenir de meilleures caractéristiques d'apprentissage suivies par un classeur SVM. Les performances de l'approche DBN-SVM proposées sont testées sur NSL-KDD data-set. Le nombre de caractéristiques a été réduit de 41 à 5 grâce au réseau DBN qui ont été ensuite transmises à un classeur SVM pour effectuer une classification binaire (normale/attaque). La méthode proposée atteint une exactitude supérieure à 90%. De plus, les résultats ont montré que le réseau DBN en tant qu'une méthode de réduction des caractéristiques est plus performant par rapport à d'autres méthodes d'analyse de données comme l'ACP, Gain Ratio et à Chi-Square.

Kang et al. [23] ont proposé un système de détection d'intrusion qui se base sur le réseau neuronal profond (DNN) dans le but de sécuriser les réseaux de véhicules. Le scénario d'attaque a été réalisé sur des paquets de données malveillants. Ces derniers sont injectés dans un bus de réseau de contrôleurs de véhicules. Le système proposé introduit le vecteur de caractéristiques dans les nœuds d'entrée afin de classer les paquets en deux classes (un paquet normal et un paquet d'attaque). Les auteurs utilisent le réseau de croyances profondes (DBN) pour former efficacement les paramètres du poids caché (Hidden weights) initialisant le réseau DNN. Ensuite, les couches cachées suivantes sont liées à ces sorties qui sont calculées sur la base de la fonction d'activation (ReLU). Le système proposé atteint un taux de fausse positive (TPR) inférieure à 1 ou 2 %, et un taux de détection (DR) de 99%.
Sandee et al. [20], un DNN pour NIDS composé d'un sparse auto-encoder utilisé pour l'apprentissage des caractéristiques non-supervisé, et d'une logistic regression pour une classification binaire sur l'ensemble de données NSL-KDD (normal/intrusion). Le système prend en compte 115 caractéristiques en entrée, le sparse auto-codeur a été utilisé pour la formation et l'apprentissage de nouvelles caractéristiques, donc ces caractéristiques ont été réduites à 50, puis à 10, puis elles sont attribuées au classificateur de régression logistique pour la classification. La performance du système a été mesurée en termes d'accuracy, de précision et de rappel. La précision globale du modèle était de 87,2%.

Tang et al. [47] appliquons le réseau neuronal profond (DNN) pour implémenter un IDS dans un contrôleur SDN (Software Defined Network) qui peut surveiller tous les flux des commutateurs OpenFlow. Ils ont entraîné le modèle sur l'ensemble de données NSL-KDD pour une classification binaire (normale / anomalie), où l'ensemble de données se composait de quatre catégories à savoir : attaques DoS, attaques R2L, attaques U2R et attaques probe. Ensuite, ils ont utilisé que 6 caractéristiques de base tirées des 41 caractéristiques (Features). Le modèle a été optimisé en trouvant l'hyper-paramètre ce qui a le conduit à des résultats de classification optimaux. Pour cela, ils ont donc utilisé

un taux d'apprentissage différent variant de 0,1 à 0,0001. Toutes les mesures d'évaluation étaient dans une tendance croissante quand elles diminuent le taux d'apprentissage (learning rate) de 0,1 à 0,001, et finalement ils ont obtenu les meilleurs résultats lors de l'utilisation de 0,001 comme taux d'apprentissage. Le modèle obtient des performances avec une précision de 75,75%. Ses expériences confirment que l'approche d'apprentissage approfondi DNN présente un fort potentiel d'utilisation pour la détection d'anomalies dans NIDS.
Un système de détection d'intrusion de réseau (NIDS) basé sur le deep learning a été proposé par Javaid et al. [22], Les auteurs ont utilisé l'apprentissage en autodidacte (Self-taught Learning STL). Cette méthode se représente en deux étapes pour la classification, la première étape appelée l'apprentissage de caractéristiques (non-supervisées). s'agit d'une bonne représentation des caractéristiques obtenue à partir d'une massive collection de données non étiquetées. Dans la deuxième étape, cette représentation apprise sera appliquée aux données étiquetées et utilisée pour la tâche de classification. L'approche STL a été testée sur le benchmark dataset NSL-KDD pour la détection des anomalies de réseau. Il s'agit d'une classification multi-classes des records du trafic normal et autres types d'attaques réseaux. Les meilleurs résultats atteint une précision de 85.44% et un rappel (recall) de 95.95% pour une classification binaire (trafic normal / trafic malveillant) lorsqu'ils ont appliqué des données des testes malgré que ces résultats a un taux de fausse positive considérable, cette approche du SLT donne de bons résultats comparent une simple soft-max regression (machine learning algorithme pour multi- classification) et d'autres travaux précédents.

Il y a eu quelques études utilisant les CNN dans le domaine de la détection d'intrusion ; le plus grand avantage de CNN est qu'il partage les mêmes noyaux convolutionnels, ce qui réduirait le nombre de paramètres et la quantité de calculs de formation : Une fois que le système peut identifier plus rapidement le type d'attaque des données de trafic, il est plus facile d'identifier les données de trafic [50, 51].

Vinayakumar et al. [50] ont analysé l'efficacité du réseau neuronal convolutionnel (CNN) pour la détection des intrusions en modélisant les événements de trafic du réseau en tant que time-series de paquets TCP/IP dans des périodes de temps prédéfinie. les auteurs ont suivi la méthode CNN appliquée au traitement du langage naturel avec la couche Convolution 1D [27]. Sur cette base, les auteurs modélisent les événements de trafic du réseau comme une série chronologique de données, où, au lieu de prendre des données d'image 2D comme entrée, la CNN traitera une série de données sous forme de 1D dans laquelle les données sont disposées dans un intervalle de temps. Les auteurs ont proposé différentes architectures du CNN contient une couche d'entrée, une couche cachée et une couche de sortie, ou la couche cachée contient une ou plusieurs couches CNN

suivies de FFN ou RNN/LSTM/GRU afin de déterminer l'architecture optimal.Toutes les expériences se déroulent sur 1000 époques. Les performances de chaque modèle sont évaluées sur l'ensemble de test du KDDCup 99 data-set, Les résultats montrent que le CNN-LSTM a obtenu de bons résultats par rapport aux autres structures du réseau CNN et atteint une précision de 99%. ainsi selon les auteurs, les algorithmes de CNN ont surpassé les résultats du défi KDDCup 99 et d'autres résultats publiés de détection d'intrusion.

Kim et al. [25] ont proposés IDSs basé sur RNN. les paquets TCP/IP du trafic réseau a été représenté sous forme de séquence. Les auteurs ont utilisé la méthode Hessian-Free Optimisation pour surmonter le problème explosion/vanishing gradiant du RNN, ce technique aide l'algorithme de gradient de trouver le minimum globale pendant l'entraînement et donc améliorer la précision. L'ensemble de données DARPA a été utilisé pour l'évaluation de modèle proposé composé de 41 caractéristiques et 22 différentes attaques. Le taux de détection était de 95,37% et le taux de fausses alertes de 2,1%. La précision de la classification de chaque classe d'attaque était bonne, et les auteurs concluent que l'utilisation de RNN avec optimisation sans hélium pour la détection des intrusions est une assez bonne approche.

GAO et al. [18] ont utilisé le Deep beleif network DBF pour l'appliquer dans le domaine de la reconnaissance des intrusions. L'architecture de ce modèle constitué d'une combinaison d'un réseaux d'apprentissage mutti-couches non supervisés, appelée "machine de Boltzmann restreinte" (Resticted Boltzmann machine), et d'un réseau d'apprentissage supervisé appelé réseau de rétro-propagation. Les résultats expérimentaux sur l'ensemble de données KDD CUP 99 démontrent que les performances du modèle DBN sont meilleures que celles du SVM et de l'ANN.

Rezvy et al. [39] ont choisi l'approche AutoEncoder qui fonctionne comme un constructeur de caractéristiques, le modèle a été d'abord préformé par l'AutoEncoder pour réconfigurer les caractéristiques, puis un classificateur DNN a été utilisé pour classer le trafic. L'auteur a testé le modèle avec le NSL-KDD'99 à cinq classes. L'AutoEncoder a reconstruit les 122 caractéristiques qui ont été ensuite intégrées dans un DNN à trois couches. Le DNN est un classificateur à cinq classes. La précision des cinq classes varie de 89,2% à 99,9%. La précision globale des attaques est de 99,3%.

Alom et al. [4] ont utilisé le réseau de croyances profondes (Deep beleif network) pour la détection des intrusions. Le système proposé est capable de détecter les attaques, et aussi identifier et classer la précision des activités du réseau en cinq groupes sur la base de sources de données limitées, incomplètes et non linéaires. Par rapport au système exis-

tant, la précision de détection du système a atteint 97,5%, et n'a nécessité que 50 itérations.

Wu et al. [53] ont proposé un modèle IDS en utilisant des réseaux neuronaux convolutifs (CNN). Le CNN est utilisé pour sélectionner automatiquement les caractéristiques du trafic à partir d'un ensemble de données brutes. Les auteurs ont utilisé l'ensemble de données NSL-KDD pour l'évaluation, et aussi pour résoudre le problème de déséquilibre des ensembles de données (the imbalanced Dataset problem), ils fixent le coefficient de pondération de la fonction de coût de chaque classe en fonction de ses nombres. Pour réduire le coût du calcul, ils convertissent le format vectoriel de trafic brut en format d'image. Le modèle proposé améliore la précision de la classe avec de petits nombres et réduit également le taux de fausses alarmes (FAR).

Torres et al. [48] analysent la viabilité des réseaux neuronaux récurrents (RNN) pour détecter le comportement du trafic du réseau en le modélisant comme une séquence d'état qui se change avec le temps. La méthode de détection LSTM a été proposée et évaluée sur 2 ensembles de données différents provenant du trafic du réseau, CTU13-42 et CTU13-47, les deux contiennent 2 classes : la connexion au botnet et la connexion normale. Un botnet peut être conçu comme un groupe d'ordinateurs compromis qui peuvent être contrôlés à distance pour exécuter des attaques coordonnées ou commettre des actes frauduleux. Les auteurs ont également analysé le nombre d'états par connexion pour le modèle comportemental et les expériences prouvent que le RNN est capable de classer le trafic avec un taux élevé de détection des attaques arrive à (ADR) 0.970% et un taux de fausses alarmes très faible (FAR) qui arrive à 0.018%. L'influence des techniques d'échantillonnage (sur-échantillonnage /sous-échantillonnage) afin de traiter les classes déséquilibrées a été analysé aussi sur les performances des LSTM. Les résultats montrent que les valeurs de l'ADR (Attack detection Rate) ne présentent pas une déférence significative sur les trois cas évalués. Lorsqu'ils ont utilisé ou pas de techniques d'échantillonnage, et qu'il n'y a pas de déférence considérable entre ces techniques. Cependant, les expérimentations ont montré que les modèles de détection RNN ont des problèmes pour traiter les comportements du trafic qui ne sont pas facilement différenciés ainsi que certains cas particuliers de trafic de réseau déséquilibré.
Le tableau 3.1 présente une sélection d'études portant sur la détection des intrusions basés sur des modèles DL, les caractéristiques utilisés, le secteur d'intérêt, les ensembles de données et les mesures de performance.

Méthode	Dataset	Mesures de performances	Année	Auteurs	Cité*
DBN-SVM	NSL-KDD	ACC = 90%	2011	salama et al, [41]	143
DNN-DBN	Vehicular network communication	DR = 99% - 99.46% FPR = 1-2% ROC = _	2016	kang et al, [23]	258
DBN	NSL-KDD	ACC = 97.5%	2015	Alom et al, [4]	121
CNN	NSL-KDD	ACC = 97.88% - 99.46% DR = 68.66% FPR = 27.9%	2018	Wu et al, [53]	54
STL	KDD'99	ACC = 79.10% - 88.39% PR = 85.44% RC = 95.95% F1 = 90.39%	2016	Javaid et al. [22]	450
RNN	DARPA	DR = 95.37% FPR = 2.1%	2015	Kim et al, [25]	26
RNN-LSTM	CTU-13	DR = 97.96% FPR = 2.27%	2016	Torres et al,[48]	57
CNN-RNN	KDD'99	ACC = 99.99% PR = 99.99% RC = 99.99% F1 = 99.99%	2017	vinayak et al, [50]	119
Deep auto-encoder	NSL-KDD	ACC = 99.99% PR = 84.6% RC = 92.8% specificity =80.7%	2018	Sandee et al, [20]	13
DNN	NSL-KDD	ACC = 75.75% PR = 83% RC = 76% F1 = 75% ROC_AUC = 86%	2016	Tang et al[47]	306
Deep auto-encoder	NSL-KDD	ACC = 99.3%	2018	rezvy et al [39]	8
DBN	KDD'99	ACC = 74.22% - 93.49% DR = 75.6% - 92.33% FAR = 3.15% - 0.76%	2014	GAO et al.[18]	152

Tableau 3.1 – Travaux antérieurs connexes pour la détection d'intrusion basé sur le deep learning

3.3 Conclusion

Des différentes approches et architectures deep learning ont été appliquées pour la détection des intrusions. Ces algorithmes d'apprentissage profond proposés ont des performances différentes selon les ensembles de données sélectionnés et les caractéristiques d'entrée. Cependant, l'utilisation des mêmes approches et des mêmes techniques d'apprentissage ne garantissent pas toujours les mêmes résultats pour une variété de classes différentes d'attaques possibles.

CHAPITRE 4 :
Conception et réalisation

Conception et réalisation

4.1 Introduction

Les systèmes de détection d'intrusion basées sur l'apprentissage profond ont fait l'objet de nombreux travaux de recherche. Nous avons discuté dans le chapitre précédant les différentes méthodes de deep learning qui ont été appliqué avec succès dans la tâche de détection d'intrusion en utilisant des différents ensembles de données dédiés pour la cybersécurité. Les performances des IDS basés sur l'apprentissage profond dépendent fortement de l'ensemble de données utilisées et aucun modèle de référence pour la détection d'intrusion n'a été trouvé. Dans ce travail, nous avons choisi un ensemble de données récentes fourni par l'institut canadien de cybersécurité (CIC) nommée CIC-DDoS-2019 pour la détection des attaques de déni de service distribué connues sous le nom DDOS (distributed deny of services). Nous avons proposé 3 modèles deep learning (DNN, CNN, RNN) en matière de cybersécurité permet à l'IDS de faire face à ces types d'attaques du réseau afin d'identifier les activités DDoS malveillantes dans un trafic de réseau réel. Nous avons commencé d'abord par un modèle DNN simple. Ensuite, nous avons implémenté le CNN et le RNN pour la classification de ce type d'attaque réseau afin de remédier le problème de détection des diverses DDoS attaques possibles avec un taux de détection très élevé et un taux de fasses alarmes négligeables. Les performances des approches de détection proposées ont été évalué prendre en compte les différentes mesures d'évaluation des algorithmes de l'apprentissage profond savoir , la précision, le rappelle, le score F1, le taux de détection et le taux de fausse alarmes.

4.2 Environnement d'exécution

Le Deep Learning est un domaine avec des exigences de la disponibilité des ressources matériel (surtout en GPU) capable de faire des calculs intenses. Au premier temps, On a commencé par la configuration d'un environnement local de développement Python utilisant la plate-forme Anaconda [43].

Anaconda : Est une distribution libre et open source des langages de programmation Python et R appliqué au développement d'applications dédiées à l'apprentissage automatique et à la science des données. localement, on a fait une exploration et une analyse générale sur l'ensemble des données. la préparation des données aussi a été faite localement par plusieurs étapes et à l'aide des techniques qu'il n'exige pas beaucoup de calculs.
Après pour aller vers le deep learning notre travail, on avait besoin d'aller vers le Cloud, ce dernier fournit des ressources de calculs et de mémoires importantes qui dépassent notre ordinateurs. Selon le besoin le Cloud peut fourni l'accès à un processeur graphique GPU totalement gratuit. Parmi les outils Cloud les plus utilisés dans le domaine de la machine learning c'est Google Colab [7].

Google Colaboratory : est un service Cloud basé sur Jupyter Notebooks, permet de développer des applications en Deep Learning en Python, il offre un processeur GPU gratuit, 12 Go de RAM et plus de 100 Go de stockage. Pour l'accès dans ce service il nous suffit simplement d'avoir qu'un compte Google.
Pour le langage de développement on a choisi Python, un langage de programmation interprété, multi- paradigme et multi-plateformes, il est aussi un langage plus commun et plus populaire pour l'apprentissage automatique et l'intelligence artificielle grâce à sa flexibilité et aussi parce qu'il a un nombre important de bibliothèques logicielles open source disponible. Permet ses bibliothèques utilisées dans notre projet : Pandas, Numpy, Scikit-Learn . . .etc. Pandas et Numpy sont utilisés pour la manipulation des données (le chargement, la réorganisation et le traitement des données), Scikit-Learn nous permet d'expérimenter différentes techniques et algorithmes d'apprentissage automatique et d'analyse de données prédéfinis rapidement et facile a utilisé.
les frameworks TensorFlow et Keras ont été choisir pour l'implémentions des méthodes deep learning proposé. TensorFlow est une bibliothèque de deep Learning open source développée par Google utilisée pour effectuer des opérations numériques complexes et plusieurs autres tâches pour modéliser les architectures de Deep Learning. il peut déployer facilement des calculs sur plusieurs plates-formes comme les CPU, les GPU.
Keras est une API de haut niveau qui vise à créer et à entraîner des modèles de Deep Learning basé sur python. Elle a été développée dans le but de permettre des expérimentations rapides.Parmi ces avantages

- Était capable d'aller de l'idée au résultat avec le plus faible délai possible. Et ça c'est la clé d'une recherche efficace.
- Supporte à la fois les réseaux convolutifs (CNN) et les réseaux récurrents (RNN) ainsi que la combinaison des deux items Pas de fichiers séparés de configuration des modèles, tout est déclaré dans le code.

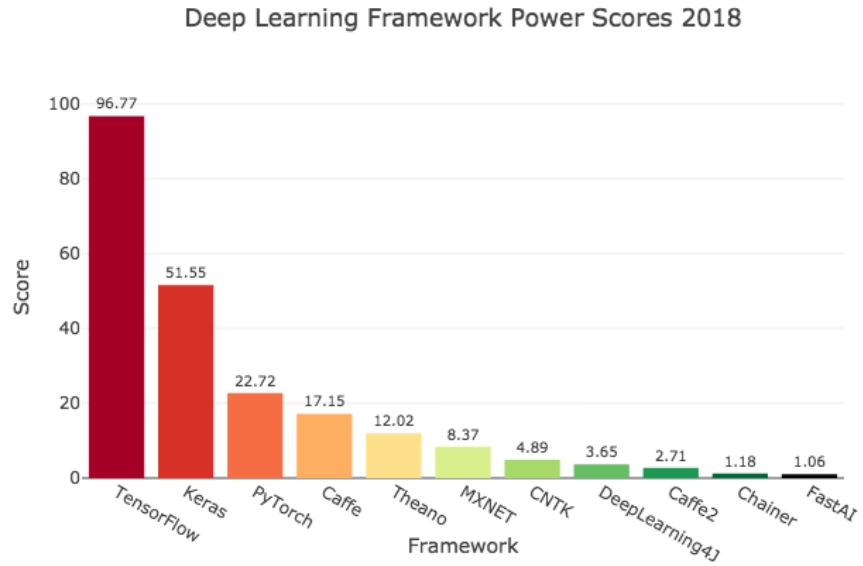


FIGURE 4.1 – Les 10 frameworks Deep learning les plus populaire [21])

- fonctionne sur CPU et GPU.

Bien que Keras fournisse toutes les fonctionnalités générales nécessaires à la création de modèles d'apprentissage en profondeur, il ne fournit pas autant que TensorFlow qui offre plus des opérations plus avancées pour obtenir un bon contrôle afin de développer un type particulier de modèles, il nous permet aussi de comprendre mieux ce qui se passe à l'intérieur d'un réseau DL.Afin de bénéficier les avantages des deux nous avons utilisé TensorFlow comme back-end avec keras. L'environnement d'expérimentation s'est google Colab avec un processeur GPU et une RAM de 12 Go.

4.3 Dataset

L'ensemble de données choisi pour cette étude c'est CIC-DDoS-2019 data-set, fourni par l'Institut canadien de cybersécurité (CIC) [37], elle constitue des données des flux réseau réel avec plusieurs types d'attaques DDOS les plus récentes et les plus répandus. Ces données sont des formats plus condensés qui contiennent principalement des métas-informations sur des connexions réseau, ou chaque donnée (chaque flux de réseau) regroupe tous les paquets qui partagent certaines propriétés dans une fenêtre temporelle et qui n'incluent pas de charge utile (Payload). L'ensemble de données comporte deux versions de données, des données PCAP bruts et des données CSVs Les auteurs sont utilisées l'analyseur du trafic CICFlowMeter-V3 pour extraire plus 80 caractéristiques depuis les fichiers PCAPs et les résultats ont été sauvegardé dans des fichiers CSVs structurés et étiquetés par l'Université de New Brunswick [44].

Cet ensemble de données comprend également 12 différents attaques DDOS le plus à jour, L'ensemble complet des données contient 50063112 instances, dont 50006249 sont des attaques DDoS et que 56863 sont des instances d'un trafic réseau bénin (légitime / normale), le nombre d'instances pour chaque type d'attaque DDOS est indiqué dans le tableau 4.1. Ce data-set contient également 86 caractéristiques (Features), ou 6 d'eux sont étiquetés et caractérisés le flux lui-même, en fonction de **Source IP, Source Port, Destination IP, Destination Port, Protocole** et **Timestamp** (temps d'attaques), et plus de 80 caractéristiques sur le flux trafic du réseau. Les données des data-set ont 2 désavantages, le premier grand désavantage c'est qu'elles sont déséquilibrées. les instances du trafic réseau bénin représentent seulement 1.13% de l'ensemble de données. Et aussi les pourcentages entre les classes d'attaques sont variés. Le deuxième désavantage c'est que l'étiquetage des données (la classe label) CSVs se diffère entre les CSVs des données d'apprentissage qui sont générées dans le premier jour d'expérimentation. Et les CSVs des données de test généré dans le deuxième jour. En plus, ces données de test ne comportent que 5 types de classe comme illustrées dans le tableau 4.6. Ce qui est imposé de modifier l'étiquetage de l'un des 2 ensembles avant de l'utiliser et pour faire l'équivalence entre les classes des données d'entraînements et les classes des données de test.

Les attaques qui ne se produisent que le jour d'entraînement sont les suivantes : DrDoS_S, DrDoS_NTP, DrDoS_SNMP, DrDoS_SSDP, DrDoS_UDP, TFTP, UDP-lag, WebDDoS.

Class Label	Nombre des Instances
Benign (legitimate traffic)	56863
DDoS_DNS	5071011
DDoS_LDAP	2179930
DDoS_MSSQL	4522492
DDoS_NetBIOS	4093279
DDoS_NTP	1202642
DDoS_SNMP	5159870
DDoS_SSDP	2610611
DDoS_SYN	1582289
DDoS_TFTP	20082580
DDoS_UDP	3134645
DDoS_UDP-Lag	366461
DDoS_WebDDoS	439

Tableau 4.1 – CICDDoS2019 : Le nombre d'enregistrements pour chaque catégorie d'attaques DDOS dans l'ensemble des données

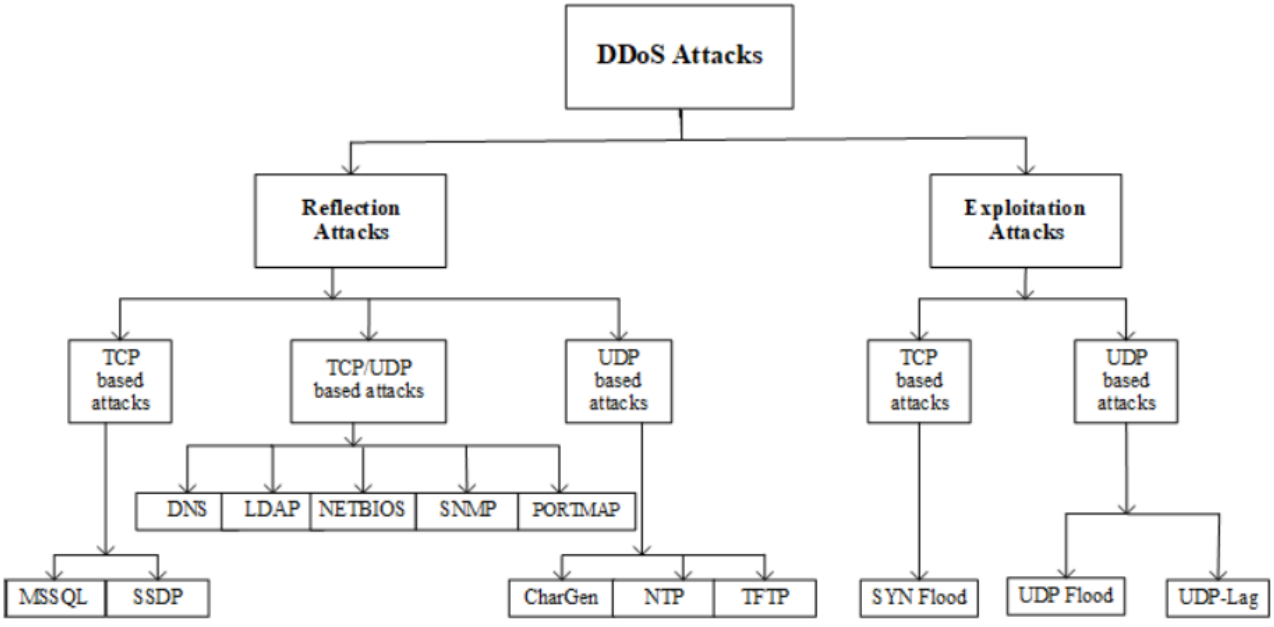
4.4 Taxonomie des attaques DDoSs

Le déni de service distribué (DDos) est une tentative intrusive s'agit d'une série d'attaques qui envoient une énorme quantité de paquets à une ressource cible pour le but de désactiver temporairement ou définitivement les services fournis par cette ressource. Une taxonomie des ces attaques a été proposer par les auteurs composées de 2 grandes catégories qui sont : les attaques par réflexion et les attaques par l'exploitation [44].

Les attaques DDoS basées sur la réflexion : Ils s'agit d'une série d'attaques dans lesquelles l'identité de l'attaquant est dissimulée en utilisant des botnets pour envoyer le trafic d'attaque (par exemple des requêtes HTTP) à la victime. Ces demandes sont envoyées aux serveurs réflecteurs par l'attaquant avec l'adresse IP source définie à l'adresse IP cible dans le but d'inonder la victime par les paquets de réponses. Ces attaques sont effectuées à l'aide des protocoles de la couche transport à savoir transmission contrôle protocole (TCP), User datagram protocole (UDP) ou une combinaison de ces protocoles [44].

Les attaques DDos basées sur l'exploitation : Il s'agit d'une série d'attaques dans lesquelles l'attaquant tente d'exploiter directement le service à distance. Ces attaques peuvent être menées à l'aide des protocoles de la couche transport TCP ou UDP.

Une classification de ces DDoSs attaques s'est basée sur ces catégories. Ainsi que les protocoles réseaux qui sont utilisés sont illustrés dans la figure **??** aussi une briefe description de ces attaques a été montré dans le tableau ci-dessous 4.2.



les attaques par réflexion et les attaques par l'exploitation [44].

Les attaques	Description
Syn Flood	Une attaque par déni de service qui exploite la faiblesse de la séquence de connexion TCP. Elle se déroule en trois temps : le SYN, le SYN-ACK et l'ACK l'attaquant envoie une succession de requêtes SYN à un serveur cible sans une réponse ACK pour tenter de consommer ses ressources [1].
LDAP	L'injection LDAP est une attaque utilisée pour exploiter les applications web qui construisent des déclarations LDAP basées sur les entrées de l'utilisateur [5].
SSDP	Dans cette variante attaque par réflexion, l'attaquant exploite de façon abusive le protocole SSDP (Simple Service Discovery Protocol) et envoyer une requête avec une adresse d'origine mystifiée à une victime cible pour surcharger ses ressources informatiques avec les réponses de ce protocole [6].
UDP	Une attaque DDos dans laquelle un grand nombre de paquets est envoyé à une victime en utilisant la connexion UDP sans avoir besoin d'une connexion établie. Cela est dans le but de surcharger sa capacité de traitement et de réponses qui entraîne une indisponibilité limitée pour les utilisateurs. [44].
DNS	Une attaque par réflexion, exploitée les vulnérabilités du DNS (Domain Name Service Protocol)
MSSQL	Microsoft Structured Query Language est une attaque qui permet d'exécuter des instructions SQL malveillantes
NetBIOS	Un exploit de sécurité dans le Network Basic Input/Output System (NetBIOS) permet à un attaquant de voir des informations dans la mémoire d'un ordinateur sur un réseau
SNMP	dans cette variante d'attaque de surcharge Une attaque par le protocole SNMP génère une grande quantité de trafic qui est dirigée vers des victimes provenant de plusieurs réseaux
UDP-lag	une attaque utilisée pour perturber la connexion entre le client et le serveur [44].
TFTP	Cette attaque exploite la vulnérabilité de débordement de tampon dans un serveur TFTP (Trivial File Transfer Protocol)
WebDDoS	Le WebDDoS est une attaque qui vise à faire tomber le site web cible ou à le ralentir en inondant le réseau, le serveur ou l'application de faux trafic
NTP	Le NTP est une attaque par amplification dans laquelle l'attaquant exploite les serveurs NTP accessibles au public pour surcharger la cible de trafic UDP
PortMap	une attaque sur le port 111 TCP ou UDP qui est un service utilisé pour diriger les clients vers le numéro de port approprié afin qu'ils puissent communiquer avec le service d'appel de procédure à distance (RPC)

Tableau 4.2 – Attaques DDoS basées sur la réflexion et sur l'exploitation

4.5 La préparation des données

Les performances des méthodes de deep learning dépendent fortement sur la quantité et la qualité des données d'apprentissage, plus de donnés avec qualité plus de précision et de bon résultats. Dans notre cas, Nous avons une masse de données très suffisante. Cependant, à cause de la dés-équilibrage des classes de données . Il est nécessaire de faire réduire ces données.
La préparation des données s'agit de 2 opérations essentielles avant de les traitées. qui sont : la réduction des données et la résolution de l'étiquetage.

4.5.1 La réduction des données

Le volume de l'ensemble de données nous oblige à faire réduire un nombre important des échantillons depuis les données d'apprentissage et de test.
Cette masse de données constitue de 11 fichiers CSVs de tailles plus de 22 Go pour l'apprentissage. Ainsi de 7 autres fichiers CSVs pour le test de taille plus de 8 Go. La réduction de données d'apprentissage et de test se fait de manière séparée suivant la même procédure. La lecture et la préparation des données n'étaient pas des taches faciles avec cette énorme quantité de données même lorsque on a travaillé sur colabe qui offre 12 Go de RAM, pour cela on a :

- Commencé à analyser et traiter chaque fichier CSVs séparément, afin de charger les données pour certains fichiers de grande taille (plus 8 Go), on a utilisé le fractionnement horizontal des données en 2 ou plusieurs fichiers CSVs.
- Choisit aléatoirement un nombre prédéfini des instances pour chaque classe majoritaire dans l'ensemble des instances de cette classe.
- Garder tous les instances des classes minoritaire, qui sont collectées depuis plusieurs fichiers.
- Combiner les données d'apprentissage ainsi les données de test en 2 seules fichiers séparés appelés "Train.csv" et "Test.csv".
- Générer 3 différentes sous-ensembles de données depuis les données d'apprentissage et de test pour 3 différentes expérimentations. Les tableau 4.3, 4.4 et 4.5 montre la répartition de ces ensembles de données.

4.5.2 La résolution de l'étiquetage

Comme il est illustré dans le tableau 4.6, l'étiquetage des données est mal fait, à cause de cela, nous avons choisi de ré-étiqueter les données de test manuellement utilisant l'éditeur "Notepad++". Les données de l'ensemble de Tests du CICDDoS_2019 constitués

	Les Classes concernés	Nb d'instances pour l'apprentissage	Nbr d'instances pour le Test
Dataset_1 (7-Classes)	BENIGN	56101	17146
	DrDoS_NetBIOS	619700	136729
	DrDoS_MSSQL	619446	157076
	DrDoS_LDAP	619251	150701
	DrDoS_UDP	618696	150706
	UDP-lag	183662	1873
	Syn	790662	150416

Tableau 4.3 – Sous ensembles_1 constitue de 6 différentes DDoS attaques (Dataset_1)

	Les Classes concernés	Nb d'instances pour l'apprentissage	Nb d'instances pour le Test
Dataset_2 (2_Classes)	BENIGN	56101	17146
	Attack	997054	314716

Tableau 4.4 – Sous ensembles_2 constitue de 2 classes pour la détection des attaques DDoS (Dataset_2)

de 7 fichiers CSVs comportent que 8 classes dont 7 classes sont des attaques DDos et le 8e classe désigne le trafic normale (La classe BENIGN). la modification des étiquettes des classes de données de test se fait comme la suivante :

- la classe LDAP a été modifié en DrDoS_LDAP classe.
- la classe MSSQL a été modifié en DrDoS_MSSQL classe.
- la classe NetBIOS a été modifier en DrDoS_NetBIOS classe.
- la classe UDPLag a été modifier en UDP-lag classe
- la classe UDP a été modifier en DrDoS_UDP classe.
- la classe BENIGN et Syn restent inchangé.
- La classe Portmap qui n'existe pas dans les données d'apprentissage a été éliminé la première fois pour la classification multi-classes, mais elle est considérée pour la classification binaire (2-classes) pour la détection des attaques DDoS.

Dans une autre expérience, elle a été modifiée en Other classe qui désigne une autre DDos attaque classe, dans l'autre coté des donnés d'apprentissage les attaques qui ne se produisent que le jour d'entraînement comme : DrDoS_S, DrDoS_NTP, DrDoS_SNMP, DrDoS_SSDP, DrDoS_UDP, TFTP, UDP-lag et WebDDoS qui n'existent pas dans les données de test ont été modifié aussi en Other classe.

	Les Classes concernés	Nb d'instances pour l'apprentissage	Nb d'instances pour le Test
Dataset_3 (13-classes)	BENIGN	56101	Utilisé l'échantillonnage stratifié (Stratified sampling) avec 25% des données d'apprentissage afin d'éviter l'erreur d'échantillonnage des données déséquilibrées pourcentage et aussi assurer que les données d'entraînement et de test ont le même pourcentage de division pour chaque classe.
	DrDoS_LDAP	99943	
	DrDoS_SSDP	98576	
	DrDoS_UDP	97932	
	DrDoS_DNS	96567	
	DrDoS_MSSQL	95700	
	DrDoS_NetBIOS	93560	
	DrDoS_SNMP	91578	
	Syn	99983	
	DrDoS_NTP	76457	
	UDP-lag	74203	
	TFTP	72116	
	WebDDoS	439	

Tableau 4.5 – Sous ensembles_3 constitue de 12 différents DDoS attaques (Dataset_3)

Les étiquettes des données d'apprentissage	Les Étiquettes des Données de Test
BENIGN	BENIGN
DrDoS_DNS	LDAP
DrDoS_LDAP	MSSQL
DrDoS_MSSQL	NetBIOS
DrDoS_NetBIOS	Portmap
DrDoS_SNMP	UDPLag
DrDoS_UDP	UDP
DrDoS_SSDP	
Syn	Syn
DrDoS_NTP	
TFTP	
UDP-lag	

Tableau 4.6 – Les étiquettes des différentes attaques dans la journée d'entraînement ainsi dans la journée de Test

4.5.3 Les pré-traitements des données

Afin de construire un modèle très précis, il est important d'effectuer des analyses exploratoires sur l'ensemble de données et ses caractéristiques. Le pré-traitement de l'ensemble de données est effectué avant d'être appliqué au réseau neuronal profond. les

étapes de pré-traitement sont comme les suivantes :

- Tout d'abord, l'ensemble de données a été filtré afin de supprimer toutes les lignes redondantes qui représente les instances de classe. Ensuite, une analyse a été effectué pour détecter toute valeur '**NAN**' (Not A Number) ou '**INF**' (Infinite Value), ces valeurs peuvent être considérées comme des valeurs manquantes. Les algorithmes des deep learning ou de machine learning en général, traitent très mal ces valeurs qui affectent directement et négativement les performance des modèles finales.
  Il est apparaît que les données sélectionnées comme l'ensemble de données concernés par cette étude ont plusieurs valeurs de 'NAN' pour la colonne **Flow Bytes**, afin de garder cette caractéristique et comme on a des données suffisantes, les lignes avec des valeurs NAN ou INF ont été supprimé.
- Les statistiques descriptives qui résument la dispersion et la distribution de l'ensemble de données ont montré qu'il y a des colonnes vides (ses valeurs sont toujours 0), ces caractéristiques ne contiennent aucune information discriminatoire permettant de différencier les classes d'attaque, par contre ils peuvent donner des mauvaises résultats, ces colonnes sont : **' Bwd PSH Flags',' Fwd URG Flags',' Bwd URG Flags','FIN Flag Count',' PSH Flag Count',' ECE Flag Count', 'Fwd Avg Bytes/Bulk',' Fwd Avg Packets/Bulk',' Fwd Avg Bulk Rate',' Bwd Avg Bytes/Bulk', ' Bwd Avg Packets/Bulk','Bwd Avg Bulk Rate'** qui ont été supprimé.
- il y a un certain nombre de caractéristiques de type catégoriel dans l'ensemble de données qui doivent être encodées. la colonne **Flow Packets/s** a été convertie en une colonne numérique. Cependant, les autres colonnes catégorielles comme les colonnes Adresse IP et Timestamp ont été supprimées. On a considéré que ces caractéristiques sont liées aux informations de connexion et ne représentent pas les propriétés des attaques DDOS, car ce dernier peut être produit à tout moment par n'importe quelle machine contre n'importe quelle machine victime, pour cette raison les caractéristiques **'Flow ID', 'Source IP', ' Destination IP', ' Timestamp' et 'Inbound'**. sont aussi supprimées pour qu'il ne reste que les caractéristiques du trafic réseau (Trafic features). L'ensemble des caractéristiques qu'ils nous reste pour cette étude sont indiquées dans le tableau 4.7
- Pour la colonne **Label** qui représente la classe de chaque instance, il a été encodé avec une technique populaire appelée "One-Hot-Encoding". Le codage va convertit les lignes contenant des catégories en leur propre colonne avec une valeur 1 signifie vrai (cette instance est celle de cette classe) ou 0 signifie faux (cette instance n'est pas de cette classe).
- Lorsque on obtient des données de qualité, ou chaque instance des classes comporte une information qui bien décrite sa classe . La prochaine étape avant de passer

vers l'apprentissage c'est la normalisation. Les données d'entrées doivent être normalisées, cette étape a un effet dans la construction de modèle en réduisant le taux d'apprentissage, la formation de modèle sa converge rapidement. Elle peut aussi avoir un effet de régularisation en réduisant l'erreur de généralisation.

- les données d'entraînement sont divisée en 2, des données pour l'apprentissage et pour la validation de modèle. Ensuite, le modèle sera tester uniquement sur l'ensemble de test du CICDDos2019 Data-et
- Comme on a mentionné auparavant,ces ensembles des données sont très déséquilibres, on aura de mauvaises performances sur les classes minoritaires. Dans notre cas, la classe minoritaire **BENIGN** est la plus importante pour différencier le trafic normale d'une trafic malicieux contenant des attaques DDoS. Un taux élevé de vrai négatifs va réduit le taux de fausse alarme du système. Pour traiter ce problème, on a essayé l'un des algorithmes de sur-échantillonnage (oversampling) : **Synthetic Minority Oversampling Technique (SMOTE)** (figure 4.2) sur les données d'apprentissage afin d'augmenter la taille des classes minoritaires dans les 3 expérimentations.
- SMOTE est une algorithme de sur-échantillonage crée des observations synthétiques basées sur les observations des classes minoritaires existantes. Dépend de la quantité de sur-échantillonnage nécessaire, SMOTE calcule les k plus proches voisins pour créer des exemples synthétiques [2]. .

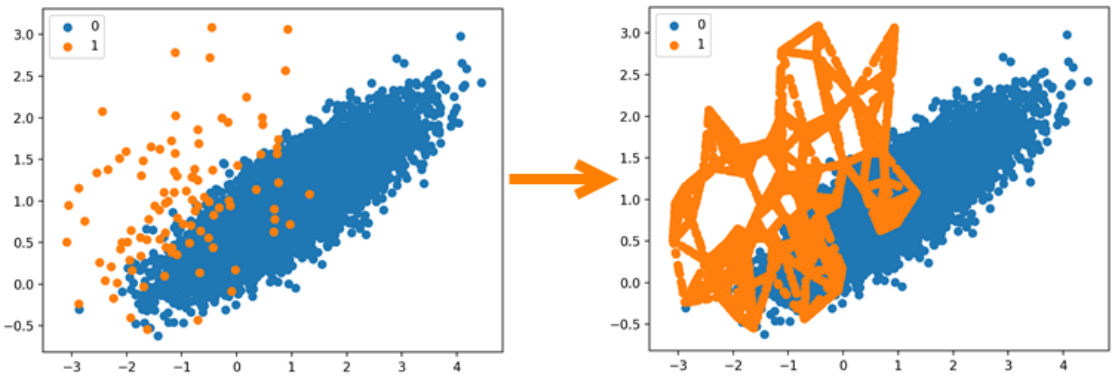


FIGURE 4.2 – Le sur-échantillonage via SMOTE (Oversampling Technique) [8]

- La bibliothèque Scikit learn offre aussi un algorithme pour gérer un ensemble de données déséquilibrées en définissant le paramètre class_weight.

Features	Type	Features	Type
Protocol	int64	Bwd IAT Max	float64
Flow Duration	int64	Fwd PSH Flags	int64
Total Fwd Packets	int64	Bwd PSH Flags	int64
Total Backward Packets	int64	Fwd URG Flags	int64
Total Len of Fwd Packets	float64	Bwd URG Flags	int64
Total Len of Bwd Packets	float64	Fwd Header Len	int64
Fwd Packet Length Max	float64	Bwd Header Len	int64
Fwd Packet Length Min	float64	Fwd Packets/s	float64
Fwd Packet Length Mean	float64	Bwd Packets/s	float64
Fwd Packet Length Std	float64	Min Packet Length	float64
Bwd Packet Length Max	float64	Max Packet Length	float64
Bwd Packet Length Min	float64	Packet Len Mean	float64
Bwd Packet Length Mean	float64	Packet Len Std	float64
Bwd Packet Length Std	float64	Packet Len Variance	float64
Bwd IAT Min	float64	Fwd IAT Max	float64
Bwd IAT Total	float64	Fwd IAT Min	float64
Bwd IAT Mean	float64	Fwd Avg Packets/Bulk	int64
Bwd IAT Std	float64	Fwd Header Length.1	int64
Fwd Avg Bytes/Bulk	int64	FIN Flag Count	int64
Flow Packets/s	object	SYN Flag Count	int64
Flow IAT Mean	float64	RST Flag Count	int64
Flow IAT Std	float64	PSH Flag Count	int64
Flow IAT Max	float64	ACK Flag Count	int64
Flow IAT Min	float64	URG Flag Count	int64
Fwd IAT Total	float64	CWE Flag Count	int64
Fwd IAT Mean	float64	ECE Flag Count	int64
Fwd IAT Std	float64	Down/Up Ratio	float64
Average Packet Size	float64	Avg Fwd Segment Size	float64
Avg Bwd Segment Size	float64	Active Std	float64
Fwd Avg Bulk Rate	int64	Bwd Avg Bytes/Bulk	int64
Bwd Avg Bytes/Bulk	int64	Bwd Avg Bulk Rate	int64
Subflow Fwd Packets	int64	Subflow Fwd Bytes	int64
Subflow Bwd Packets	int64	Subflow Bwd Bytes	int64
Init_Win_bytes_forward	int64	Init_Win_bytes_backward	int64
act_data_pkt_fwd	int64	min_seg_size_forward	int64
Active Mean	float64	Idle Max	float64
Active Max	float64	Active Min	float64
Idle Mean	float64	Idle Std	float64
Idle Min	float64		

Tableau 4.7 – L'ensemble des caractéristiques utilisées pour la détection des intrusions basé réseau (NIDS)

4.6 Un système de détection d'intrusion pour la détection des attaques DDoS dans les Réseaux

En utilisant les techniques de classification du Deep Learning ; nous avons implémenté trois types de modèle discriminatoire basé sur l'apprentissage profond qui sont : le réseau de neurone profond (DNN), le réseau de neurone convolutif (CNN) et le réseau de neurone récurrent (RNN). Ces modèles ont été construit et évalué sur 3 sous ensembles de données de CICDDos2019 Data-set pour 3 expérimentations différentes :

1. - Une multi-classification (7-classes) avec la classe BENIGN qui désigne le trafic réseau bénin (légitime), et six catégories d'attaques différentes, sur l'ensemble de données illustrées dans le tableau 4.3. Nous avons utilisé les données d'apprentissage pour l'entraînement et la validation de modèle, ensuite le modèle a été évalué sur les données de test du CICDDos2019 Data-set.
2. - Une classification binaire (2 classes)sur l'ensemble de données illustrées dans le tableau 4.4. Nous avons groupé à la fois les classes des données d'apprentissage et de Test en 2 catégories : la classe BENIGN et la classe Attack désigne le trafic réseau malicieux qui contient tous les attaques DDoS des deux ensembles de données (Apprentissage/Test). Nous avons utilisé les données d'apprentissage pour l'entraînement et la validation de modèle, ensuite le modèle a été évalué sur les données de test du CICDDos2019 Data-set. Cette expérimentation est pour objectif d'évaluer l'efficacité et le taux de détection des anomalies des attaques DDoS.
3. - Une multi-classification (13-classes) avec 12 différentes DDoS attaques classes, sur l'ensemble de données illustrées dans le tableau 4.5. Nous avons utilisé les données d'apprentissage du CICDDos2019 Data-set pour l'entraînement et l'évaluation de modèle, cette expérimentation est pour l'objectif de tester l'efficacité de détection de système face à plusieurs types attaques DDos qui ayant des comportements différents et pour évaluer aussi la capacité d'identifier le type d'attaque. Dans ce cas, les performances de système repose sur le taux de détection et la précision totale de classification.

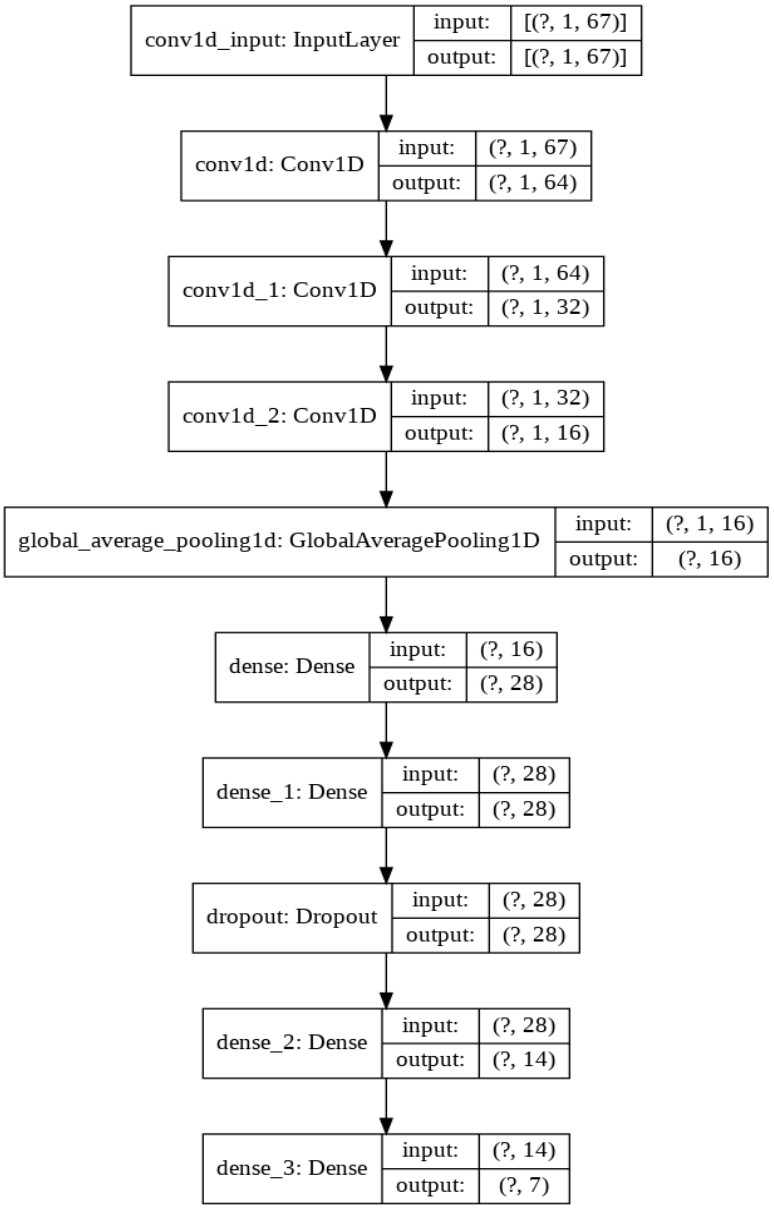
Cette étude est basée sur les caractéristiques de trafic extraites par CICFlowMeter-V3. Nous avons évalué le taux de détection et le taux de fausse alarme ainsi que d'autre métrique de classification pour toute approche proposée pour les trois expérimentations.

4.6.1 L'Architecture des modèles

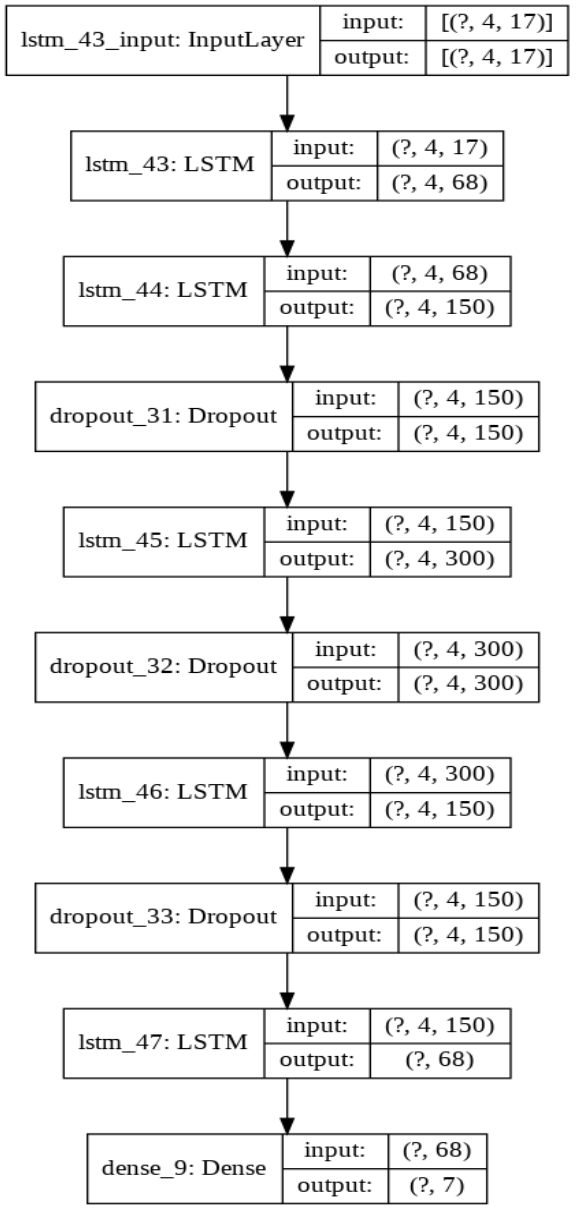
Nous avons implémenté trois types d'approches de réseaux profonds (DNN, CNN et RNN), l'architecture de chacune de ces approches a été modifiée et améliorée en essayant

plusieurs combinaisons de plusieurs paramètres pour chaque expérimentation.
Néanmoins, ces architectures des modèles proposées partagent certains propriétés et paramètres communs :

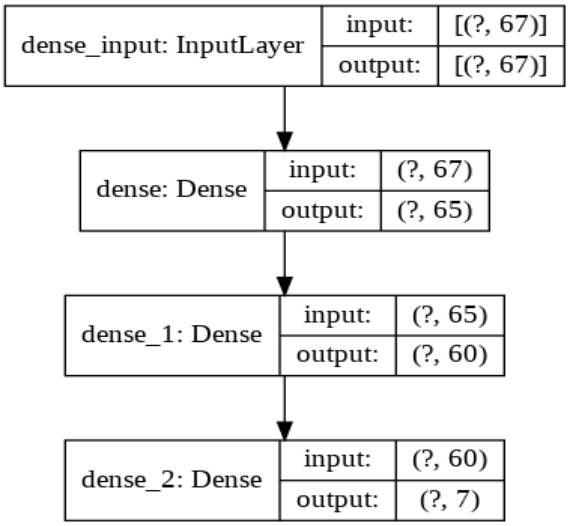
- Les couches d'entrées ont les mêmes dimensions (nombre de neurones) que le nombre de caractéristiques (Features) dans le vecteur d'entrée pour chaque modèle.
- La fonction d'activation utilisée était ReLU, différentes autres fonctions comme tanh et sigmoid ont été expérimentées, mais le ReLU a toujours les meilleurs résultats.
- Les couches de sortie ont les mêmes dimensions que le nombre de classes, pour la classification multi-classes la fonction d'activation ” **Softmax** ” a été choisit. Elle donne une probabilité (dont la somme vaut 1) en sortie de chaque neurone, le neurone de sortie avec la probabilité la plus grande permettant alors de décider que sa classe associée est la classe prédite.
- La technique dropout a été utilisée aussi, lorsqu'on tombe au problème de sur-apprentissage (Overfitting). Cette technique s'agit de considérer aléatoirement qu'un pourcentage de neurones d'une couche dans le but d'obtenir un modèle généralisable.
- La fonction de perte (Loss function) sélectionnée était “ **categorical-cross-entropy** ” pour la classification multi-classes et “ **binary-cross-entropy** ” pour la classification binaire (normale/Attack).
  L'optimiseur “ **Adam** “ a été utilisé avec un taux d'apprentissage (Learning Rate) de 0.001, au lieu de l'algorithme d'optimisation stochastique du gradient descendant (SGD) qui nous a données des mauvais résultats.
  La fonction de perte va mesurer l'écart entre les prédictions de modèle et les résultats attendus. Ensuite, l'algorithme d'optimisation "Adam" va dicter comment mettre à jour les poids d'un réseaux de neurones pour diminuer le perte, qui au modèle de converge rapidement et obtient des meilleurs prédiction avec le minimum d'erreur.



(a) le modéle Cnn utilisé pour 7-Class classification



(b) le modéle Rnn utilisé pour 7-Class classification



(c) le modéle Dnn utilisé pour 7-Class classification

FIGURE 4.3 – L'architecture des modèles proposée pour la classification 7-classes (expérimentation 1)

4.6.2 Un modèle de détection d'intrusion basée sur Deep Neural network (DNN)

L'architecture proposée du modèle DNN comporte deux couches cachées, avec une couche d'entrée et une couche de sortie (figure 4.3c).
Grâce à ces couches cachées qui contiennent un nombre important de paramètres constituant le modèle, le réseau DNN peut effectuer une extraction automatique des caractéristiques complexes correspondant à partir de données brutes. Cela est dans le but de déterminer les propriétés statistiques sous-jacentes des paquets normaux et des paquets de différentes attaques.
Plus de couches cachées mènent à un modèle plus complexe, les résultats peuvent être meilleurs,, mais ils peuvent entraîner un sur-apprentissage. Après avoir réglé les fonctions d'activation, le nombre des échantillons par lot et la fonction d'optimisation. le modèle se converge plus rapidement que d'autres modèles du CNN et RNN. Ce modèle a été formé sur plus de 10 époques pour la classification de 7-classes, jusqu'à 30 époques pour la classification de 13-classes.

4.6.3 Un modèle de détection d'intrusion basé sur Convolution Neural Network CNN

Les CNNs 1D ont été initialement étudiées pour le traitement du langage naturel utilisant des couches de convolution 1D [27]. Dans notre étude, les événements du trafic réseau sont représentés comme des données d'une série chronologique sous forme 1D. Les caractéristiques de flux sont capturées dans des périodes de temps équivalents, mais ils ont des comportements différents des millions de connexions bénignes et des attaques DDoS malveillants. De ce fait, nous avons essayé d'extraire des caractéristiques discriminatoires spatiales en appliquant le CNN 1d.
Au départ, nous avons commencé par un réseau CNN de taille moyenne en utilisant des différents nombres de couches convolutifs, différents nombres des filtres (16, 32 et 64) avec des longueurs 5, 3 et 2, afin de trouver les bons paramètres et la meilleure structure du réseau.
La figure 4.4a montre l'évaluation de l'exactitude des modèles CNNs utilisant plusieurs couches convolutifs pour les 3 expérimentations. Le nombre des couches de convolution nécessaires dépend généralement de la complexité des données. Plus nous avons utilisé des couches convolutifs, plus nous avons obtenu une meilleure précision, bien qu'après environ deux ou trois couches, le gain de précision devienne plutôt stable ainsi que l'apprentissage prend beaucoup de temps.

L'architecture du CNN utilisée dans cette étude est illustrée dans la figure 4.3a. elle est composée de 3 couches de convolution 1D, une couche Average-pooling 1D et 4 couches entièrement connectées. Le CNN commence avec une convolution 1D, le modèle nécessite une entrée tridimensionnelle. [batch_shape, steps, features], l'input-shape dans notre modèle c'est (None, 1, 67), ce modèle va accepter n'importe quel batch_shape avec des séquences de longueur 1 et une vecteur d'entrée de 67 caractéristiques.

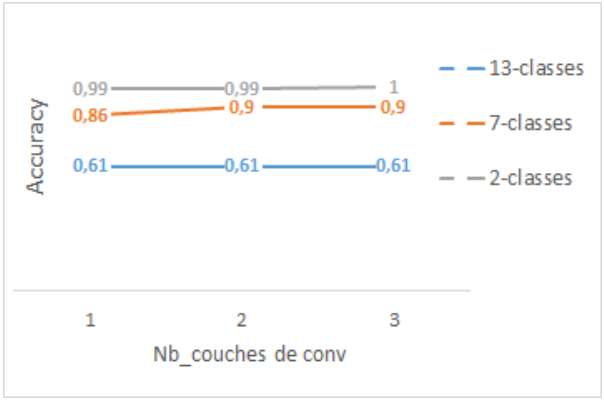
Les 3 couches de convolution ont des nombres de filtres variés de différentes tailles, ils ont le **Padding same** pour conserver la même taille des cartes de caractéristiques d'entrées. Étant donné un vecteur d'entrée 1D d'événements chronologiques du trafic réseau, l'opération de convolution 1D utilisant plusieurs filtres nous donne une carte de caractéristiques applées (**Features map**), ensuite, la fonction d'activation ReLU sera appliquée à chaque valeur du features maps.
Ces caractéristiques seront encore passé par 2 autres couches convolutifs où des nouvelles features maps seront construites. Après, nous appliquons l'opération Average-pooling sur chaque carte de caractéristiques afin d'obtenir des nouvelles caractéristiques plus significatives dans lesquelles la nouvelle caractéristique c'est la valeur moyenne des autres valeurs de la carte. Ces nouvelles caractéristiques seront finalement transmises à 4 couches entièrement connectées, une couche de **Dropout** est utilisée afin d'éviter le sur-apprentissage. La dernière couche contient la fonction **Soft-max** qui donne la distribution de probabilité sur chaque classe.
La version efficace de l'algorithme déscente de gradient stochastique "Adam " était utilisée pour optimiser le réseau, et la fonction de perte d'entropie croisée catégorique était utilisée étant donné que nous apprenons un problème de classification multi-classes.

4.6.4 Un modèle de détection d'intrusion basé sur Recurrent Neural network (RNN_LSTM)

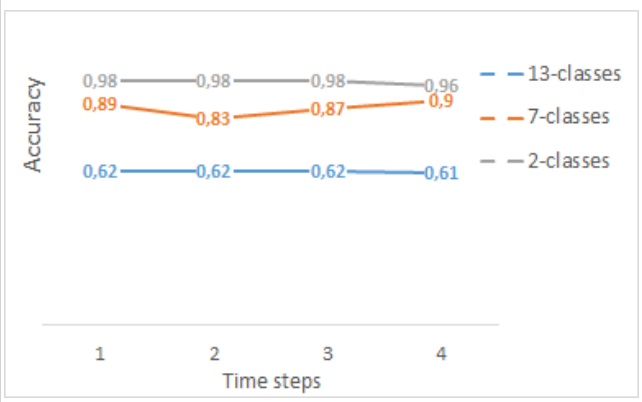
En partant de l'idée que l'enregistrement actuel d'une connexion au trafic réseau peut être classé sur la base des enregistrements passés de cette connexion à ce réseau. Nous avons essayé d'extraire des caractéristiques temporelles discriminatoires.
De ce fait, nous avons implémenté un réseau RNN_LSTM qui classe les événements de trafic du réseau comme des données de séries chronologiques. Le modèle proposé est composé d'une couche d'entrée avec une entrée tridimensionnelle [batch_size, time_step, features], notre modèle a une entrée de (None, 4, 17), où 4 représente le nombres de séquences (time step) utilisé, et 17 représente les caractéristiques (features) d'une seule séquence. Au total il ya 4*17 caractéristique du flux utilisé, le modèle va mémoriser 17 caractéristiques à la fois au temps t. La figure 4.4b montre l'évaluation de l'exactitude des modèles RNNs avec différents nombres de pas de temps (time steps) pour les 3 expérimentations. D'après les expérimentations, la division des caractéristiques d'entrées

en plusieurs pas de temps pour l'apprentissage augmente un peu la précision des modèles, mais elles n'ont pas un effet considérable. Les cellules LSTM peuvent mémoriser à la fois toutes les caractéristiques d'entrées et les utiliser pour différencier différents enregistrements. De plus, l'utilisation de plusieurs pas de temps nécessite un considérable temps de formation.

Quatre couches LSTMs cachées ont été utilisées, avec 4 couches dropout de différents pourcentages de régularisation entre elles et une couche de sortie (figure 4.3b).
L'ensemble de données peut être défini comme une liste de $[x_t, y_t]$. Étant donné un vecteur d'entrée d'évènements du trafic réseau au temps t $x_t = (x_1, x_2, ..., x_{n-1}, y)$,(ou $x_n \in R$ représente les caractéristiques et $y \in R$ représente sa classe). les couches LSTM ajoutent un module de mémoire plus long pour construire des modèles plus profonds afin de prédire les $\overline{y_t}$. Pour cela, nous avons divisé les caractéristiques d'entrées en plusieurs et pas de temps pour que le modèle apprend depuis les propriétés temporelles des données. les 3 premières couches LSTM ont une return_State = True qu'il s'agit de renvoyer le dernier état mémorisé en plus de la sortie. Les fonctions d'activation habituellement recommandées pour le réseau RNN_LSTM sont le sigmoïde, et ReLU. La dernière couche contient la fonction **Soft-max** pour une classification multi-classes.



(a) L'exactitude du Cnn avec multiples couches de convolution



(b) L'exactitude du Rnn avec multiples pas de temps (time step)

FIGURE 4.4 – L'évaluation de l'exactitude des modèles CNN et RNN avec différents nombres de couches de convolution et différents nombres de pas de temps pour les 3 expérimentations

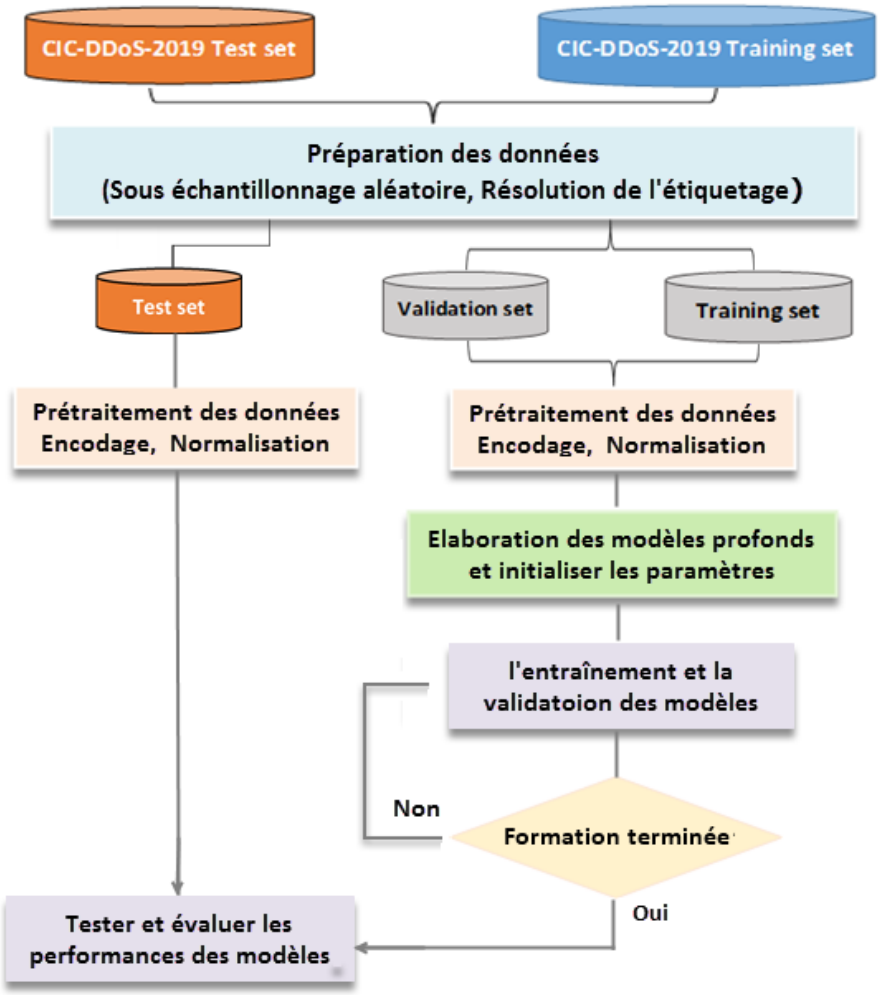


FIGURE 4.5 – Schéma conceptuel de notre méthode d'implémentation des méthodes DL proposés

4.7 Résultat et Discussion

Nous avons implémenté 3 modèles de Deep learning, le DNN, CNN et RNN. Ces modèles ont été formés et testés sur 3 sous ensembles de données différentes du CICDDoS2019 Dataset.

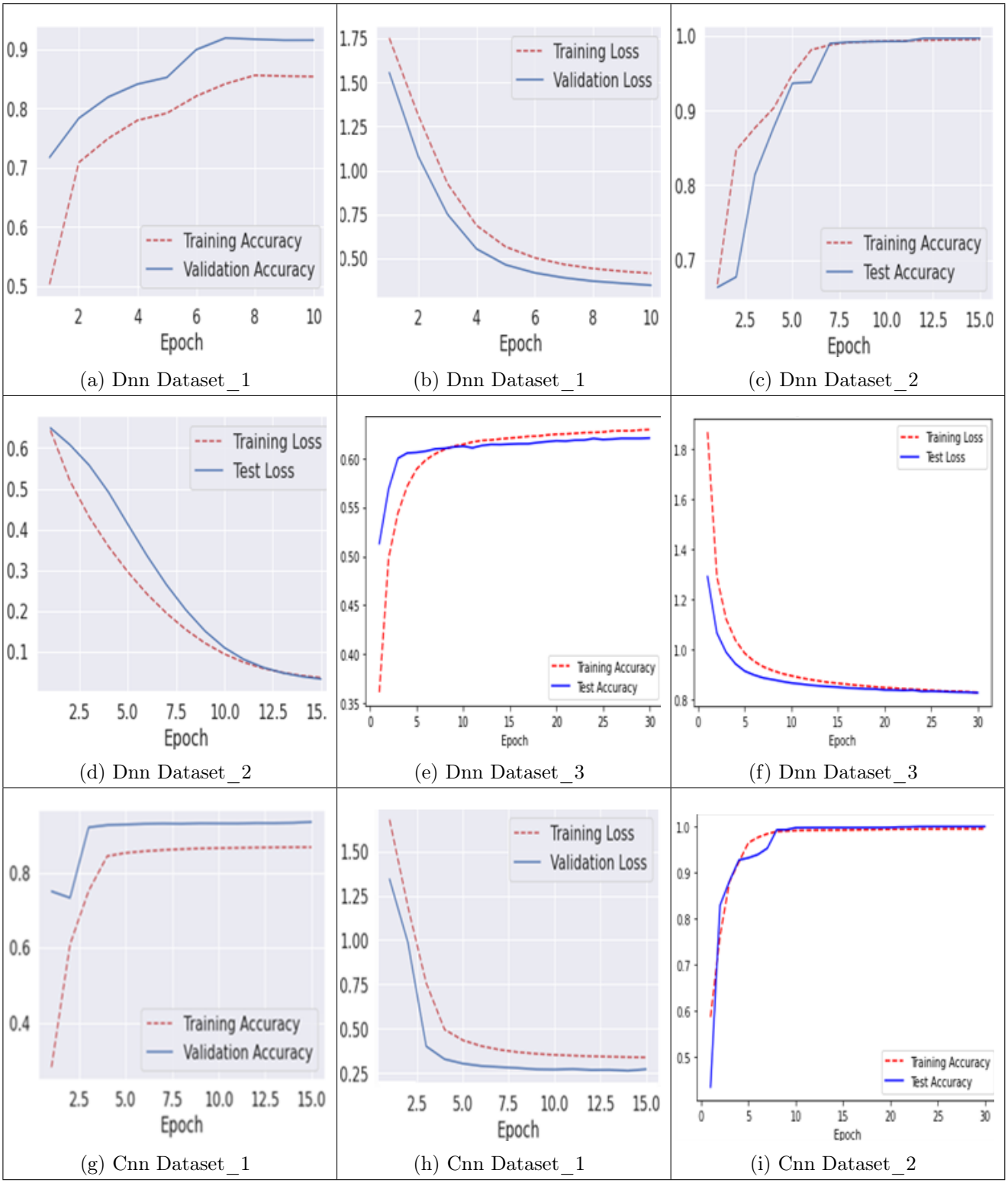
Plusieurs tests ont été faits afin d'obtenir les bonnes Hyper-paramètres pour chaque modèle. Ces paramètres ne peuvent pas ajuster durant la phase de l'apprentissage, pourtant qu'ils ont un grand impact sur les performances des modèles durant l'apprentissage. Ils comprennent les variables qui déterminent la structure du réseau (Nbr de neurones, Nbr de couches, fonction d'activation, ...), le lot d'échantillons (Batch Size) et le nombre d'itérations ...etc.
lorsqu' on arrive à un bon modèle avec le minimum de taux d'erreur et le maximum

d'exactitude, nous avons ensuite testé ce modèle sur le sous-ensemble de test. Les résultats sont présentés dans les figures ci-dessus 4.5.

Dans la première expérimentation (7-classes classification) qui a été faite sur le sous-ensemble de données du CICDDoS2019 indiquées aux tableau 4.3. Les données d'entraînement ont été divisé sur 2 : 80% pour l'apprentissage et 20% pour l'évaluation. L'apprentissage ne prend pas beaucoup de temps, les modèles ont été formés des 15 à 20 époques. Ils ont obtenu une exactitude très bonne 86.8% pour le Cnn, 82.5% pour le Dnn et 93.2 % pour le Rnn. Nous notons ici que les 3 modèles convergent vers une valeur de perte minimale, ou ils ont presque la même valeur de perte d'apprentissage et d'évaluation. Ce qui indique que ces modèles seront généralisés bien au-delà de l'ensemble d'apprentissage. Ensuite, nous avons testé ces modèles sur l'ensemble de tests.

La deuxième expérimentation (classification binaire) qui a été faite sur le sous-ensemble de données du CICDDoS2019 Data-set indiqué aux tableau 4.4. les modèles ont étaient évaluer directement sur l'ensemble de test. Le modèles Cnn a été formé sur 30 itérations, tandis que les les modèles DNN et RNN ont été formé sur 15 itérations. Pour tous les modèles, nous notons que l'exactitude d'apprentissage et de validation augmente constamment du début à la fin, elle atteint une valeur maximale tend vers 1. Nous notons aussi que la valeur de perte se diminue fortement durant l'entraînement et l'évaluation et atteint une valeur minimale tend vers 0. Cela signifie que ces modèles apprennent mieux et effectuent des meilleures prédictions après chaque époque d'optimisation.

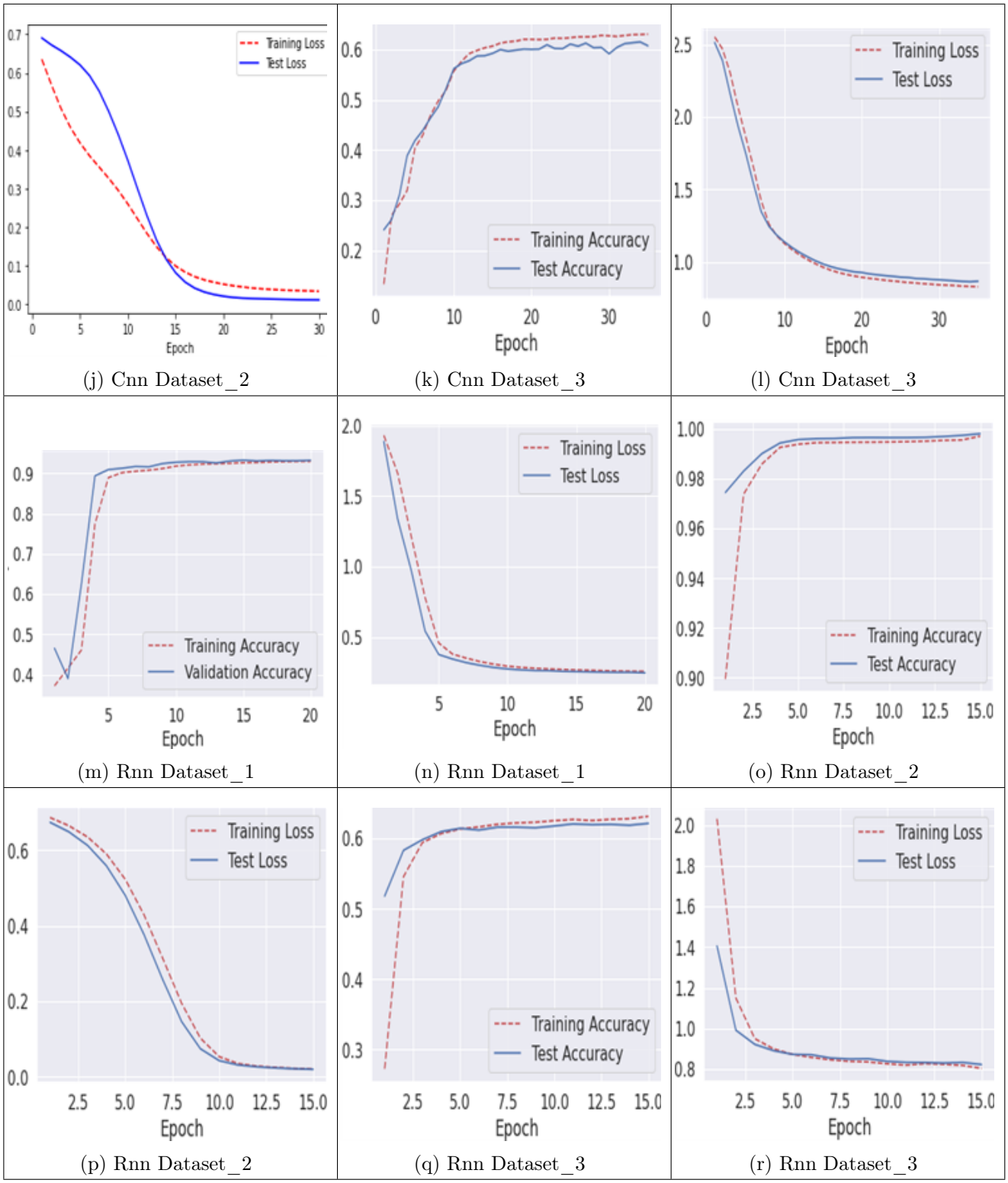
La troisième expérimentation (13-classes classification) qui a été faite sur le sous-ensemble de données d'apprentissage du CICDDoS2019 Data-set indiqué au tableau 4.5. les données d'entraînement ont été divisées sur 2 : 75% pour l'apprentissage et 25% pour le test, les modèles ont été évaluées et testées sur le sous-ensemble des données d'apprentissage. Les modèles Cnn et Dnn ont été formés sur 30 époques, tandis que le Rnn a été formé sur 15 itérations. Nous notons que la valeur de perte d'apprentissage et la valeur de perte de test ainsi que la valeur d'exactitude d'apprentissage et la valeur de l'exactitude de test se convergent toujours à la même valeur pour tous les modèles, ce qui signifie qu'on est bien traité les problèmes de sur-apprentissage et de sous-apprentissage. La meilleure exactitude obtenue était de Cnn 63%. les valeurs de perte des 3 modèles ont été proches ils se diminuent de 2.5 jusqu'à 0.8.



(a) Dnn Dataset_1 (b) Dnn Dataset_1 (c) Dnn Dataset_2

(d) Dnn Dataset_2 (e) Dnn Dataset_3 (f) Dnn Dataset_3

(g) Cnn Dataset_1 (h) Cnn Dataset_1 (i) Cnn Dataset_2



(j) Cnn Dataset_2 (k) Cnn Dataset_3 (l) Cnn Dataset_3

(m) Rnn Dataset_1 (n) Rnn Dataset_1 (o) Rnn Dataset_2

(p) Rnn Dataset_2 (q) Rnn Dataset_3 (r) Rnn Dataset_3

FIGURE 4.5 – L'exactitude et le perte des modèles proposés par rapport aux époques d'apprentissage et de validation pour les 3 expérimentations

4.7.1 Les mesures d'évaluation des modèles

- Précision (Pr) : le pourcentage des attaques DDoS identifiées comme des attaques TP parmi tous les exemples pré dictés comme attaque, il est donné par :

$$Pr = \frac{TP_{Attack}}{TP_{Attack} + FP_{BENIGN}}$$

- Recall (Rc) : le pourcentage des attaques DDoS identifiées comme des attaques TP parmet tous les attaques dans l'ensemble de données :

$$Rc = \frac{TP_{Attack}}{TP_{Attack} + FN_{Attack}}$$

- F1-score ($F1$) : la moyenne harmonique pondérée de précision et de rappel (Recall), il est donné par :

$$F1 = \frac{2 * (Pr * Rc)}{(Pr + Rc)}$$

- True Positive Rate (TPR) : le nombre d'attaques DDoS identifiées comme des attaques divisées par le nombre d'attaques DDoS dans l'ensemble de données, il est donné par :

$$TPR = \frac{\sum TP_{Attack}}{\sum DDoSAttaques}$$

- False Positive Rate (FPR) : le nombre de cas bénins incorrectement classés comme des attaques DDoS divisés par le nombre total de cas bénins dans l'ensemble de données, il est donné par :

$$FPR = \frac{\sum FP_{BENIGN}}{\sum TraficBenin}$$

- Matrice de confusion : Est une disposition de tableau spécifique permettant de visualiser les performances d'un algorithme ML pour un problème de classification, elle est connue sous le nom de matrice d'erreur.

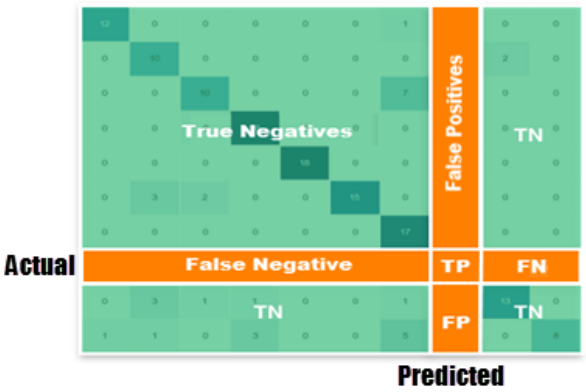


FIGURE 4.6 – Illustration d'une matrice de confusion

4.7.2 Les modèles de base utilisés pour la comparaison

Les auteurs [44] ont testé 4 algorithmes de machine learning communs à partir des données d'entraînement (Training Data) et de test (Test Data) des CICDDoS_2019 Dataset afin de servir comme référence de base pour une étude comparative. Ils ont commencé par la sélection des caractéristiques les plus importantes à l'aide de la bibliothèque SKLearn, après ils ont choisi 4 algorithmes ML les plus communs : ID3 (Decision Tree), Random forest, Naive Bayes, et Logistic Regression. Les mesures d'évaluation ainsi que les résultats obtenus sont illustrées dans le tableau ci-dessous 4.8 :

Méthode	precision	recall	f1-score
DNN	0,85	0,83	0,82
CNN	0,91	0,90	0,89
RNN	0,90	0,89	0,88
ID3	0,78	0,65	0,69
Random Forest	0,77	0,56	0,62
Naive Beyes	0,41	0,11	0,05
Logistic Regresion	0,25	0,02	0,04

Tableau 4.8 – Comparaison des résultats entre les méthodes DL proposés et les algorithmes ML de référence

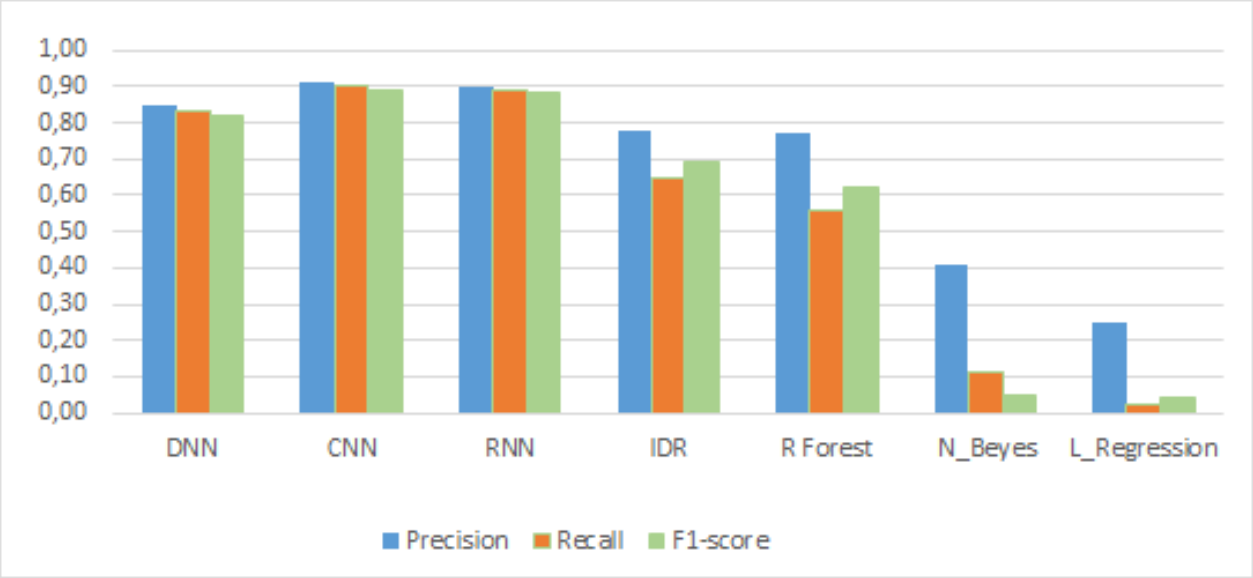


FIGURE 4.7 – Comparaison entre les méthodes DL proposées et les méthodes ML de références

Les précisions de la détection des attaques DDoS des 3 modèles DNN, CNN et RNN pour les 3 expérimentations ont été illustrées dans le tableau 4.11.

La première expérimentation

Le figure 4.8 qui illustre les résultats de tests de notre classifieurs DL (DNN, CNN, RNN) et les classiffieurs ML de référence. Il est clair que notre méthodes DL surpassent largement toutes les les autres méthodes ML, avec un taux de précision et de rappelle (Recall) élevé.
Parmi notre méthodes le CNN a les meilleurs résultats avec une précision 91% , grâce à sa capacité de reconnaissance des motifs discriminatoires pour chaque classe, le tableau 4.9 montre le rapport de classification des attaques utilisant la méthode CNN.
Comme les performances des systèmes IDSs reposent sur sa capacité de différencier les

	precision	recall	f1-score	support
BENIGN	0.99	0.97	0.98	16954
DrDoS_LDAP	0.97	0.99	0.98	147342
DrDoS_MSSQL	0.95	0.61	0.75	151505
DrDoS_NetBIOS	0.94	1.00	0.97	131395
DrDoS_UDP	0.71	0.93	0.81	147674
Syn	1.00	0.97	0.99	140427
UDP-lag	0.00	0.00	0.00	1873
accuracy			0.90	737170
macro avg	0.80	0.78	0.78	737170
weighted avg	0.91	0.90	0.89	737170

Tableau 4.9 – Le rapport de classification 7_classes avec CNN

comportements normaux des comportements malicieux, le CNN a été bien identifié. La classe minoritaire BENIGN qui représente le trafic normal avec une précision converge vers le 1 et un rappel de 97%. Cette précision signifie le nombre des fausses alarmes est minimale, le rappel signifié que le modèle est efficace dans l'identification des attaques réseaux avec un taux de fausse négative (FN) réduits.
Le modèle CNN a été bien identifié aussi certains types d'attaques comme DrDoS_LDAP, DrDoS_NetBIOS, DrDoS_UDP et Syn. Cepandant, la classe DrDoS_MSSQL a été mal classé avec 61% de rappel et le pire c'est la classe UDP-lag qui n'a pas du tout été reconnue comme une classe d'attaque, cette dernière est une classe minoritaire et même avec différents pourcentages de sur-échantillonnage sa précision n'a pas été amélioré.

La deuxième expérimentation

Dans cette expérimentation, nous avons testé l'efficacité de notre méthodes dans la détection des attaques DDoS quelles que soient ses types, les résultats de détection ont

montré dans le tableau 4.10 On peut constater clairement que notre méthode DL proposée a été très performante. La classification binaire avec la classe BENIGN qui représente les trafiques normaux et la classe Attack qui représente les trafics malicieux a été effectué avec un taux de vrai positive (TPR) nettement élevé qui converge vers 1 et un taux de fausse négative très réduit converge vers 0. Cela signifie que les modèles proposés ont bien identifié le comportement normal du trafic ce qui permet de réduire le nombre des fausses alarmes aussi qu'ils ont bien identifié le comportement malicieux du trafic ce qui permet une sécurité très élevée contre ce type des attaques réseau. Le tableau 4.10 et les figures 4.9 présentent une étude comparative entre les 3 modèles proposés.

	TP	FP	TN	FN	TPR%	FPR%
DNN	301984	140	16814	781	99.74	0.82
CNN	302602	153	16801	163	99.94	0.90
RNN	302571	716	16238	194	99.94	4.22

Tableau 4.10 – Le rapport de classification binaire

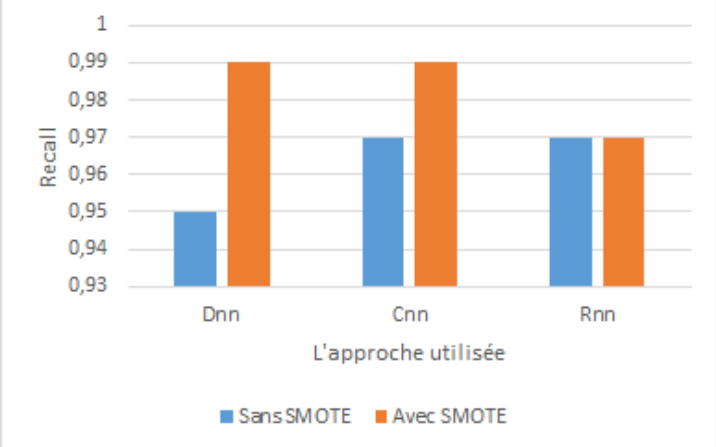
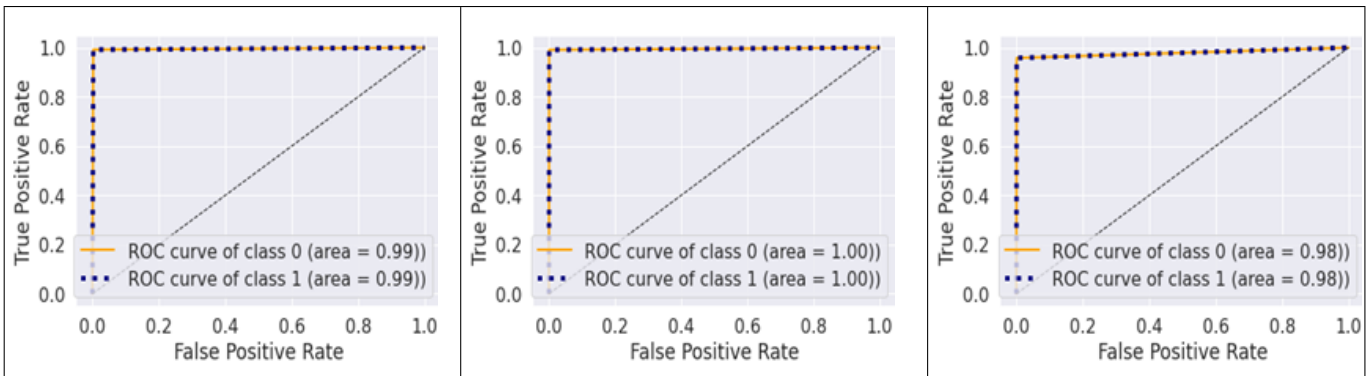


FIGURE 4.8 – Le rappel des 3 modèles avec et sans utiliser le sur-échantillonnage via SMOTE

La figure 4.8 montre l'impact du sur-échantillonnage avec SMOTE sur les performances des 3 modèles pour la prédiction de la classe minoritaire BENIGN.
Les résultats des approches proposées sans utiliser Smote ont été très bons, la classe BENIGN été bien prédite utilisant des petits pourcentages de sur- échantillonnage, les résultats de rappel ont été encore amélioré un peu, cependant, L'utilisation des grands pourcentages de sur- échantillonnage nos donnes des mauvais résultats.
La courbe Receiver Operating Characteristic (ROC) est une mesure de performance souvent utilisé pour les classifieurs binaires et surtout dans le cas où les classes sont déséquilibrées. Elle trace le taux de vrai positif en fonction du taux de faux positif à



(a) La courbe ROC du Dnn (b) La courbe ROC du Cnn (c) La courbe ROC du Rnn

FIGURE 4.9 – Les résultats du ROC courbes pour les 2-classes (BENIGN /Attack)

différents seuils de classification, afin de montrer le compromis entre la sensibilité (TPR) et la spécificité (FPR). Mathématiquement, il est calculé par la surface sous la courbe appelée Area Under Curve (AUC), Dans le cas idéal, nous aimerions avoir un Auc = 1, de sorte que la courbe ROC passe par le coin supérieur gauche du carré au point (FPR = 0, TPR = 1).

Les courbes ROCs nous permettent de bien évaluer la précision de notre classifieur avec des classes très déséquilibrées ou la classe BENIGN ne représente qu'une 5.3% de l'ensemble de données. Nous pouvons constater que notre méthodes DL sont très performantes dans la discrimination entre la classe BENIGN et la classe Attack. Le CNN surpasse encore fois le DNN et RNN avec un Auc = 1 pour les 2 classes.

La troisième expérimentation

Dans cette expérimentation, nous avons testé notre méthodes pour la classification des 12 classes d'attaques DDoS avec la classe BENIGN. Le tableau 4.11 donne les résultats obtenus des 3 modèles. Nous avons essayé d'augmenter le nombre de classes en utilisant toutes les classes de l'ensemble d'apprentissage afin de d'évaluer l'efficacité de notre modèles face à une variété des attaques DDoS et sa capacité de différencier le trafic bénin parmi eux. La meilleure précision que nous ayons obtenue est 63%, cela se réfère au nombre de classes concernées dans cette étude, ainsi le déséquilibrage les données d'entraînement et des tests. La matrice de confusion normalisée ci-dessous montre que la classe BENIGN a été bien prédite avec d'autres classes d'attaques qui ont été prédites convenablement. Cependant, certains types d'attaques comme : DrDoS_DNS, DrDoS_NetBios, DrDoS_SSDP, TFTP et UDP-Lag ont été mal classé, cela signifie que ces attaques ayant des comportements similaires et partagent certaines propriétés entre eux ce qui rendent la tache de reconnaissance de ces derniers plus difficiles.

Les bonnes caractéristiques discriminantes de la classe BEGNIN ont fait l'objet afin

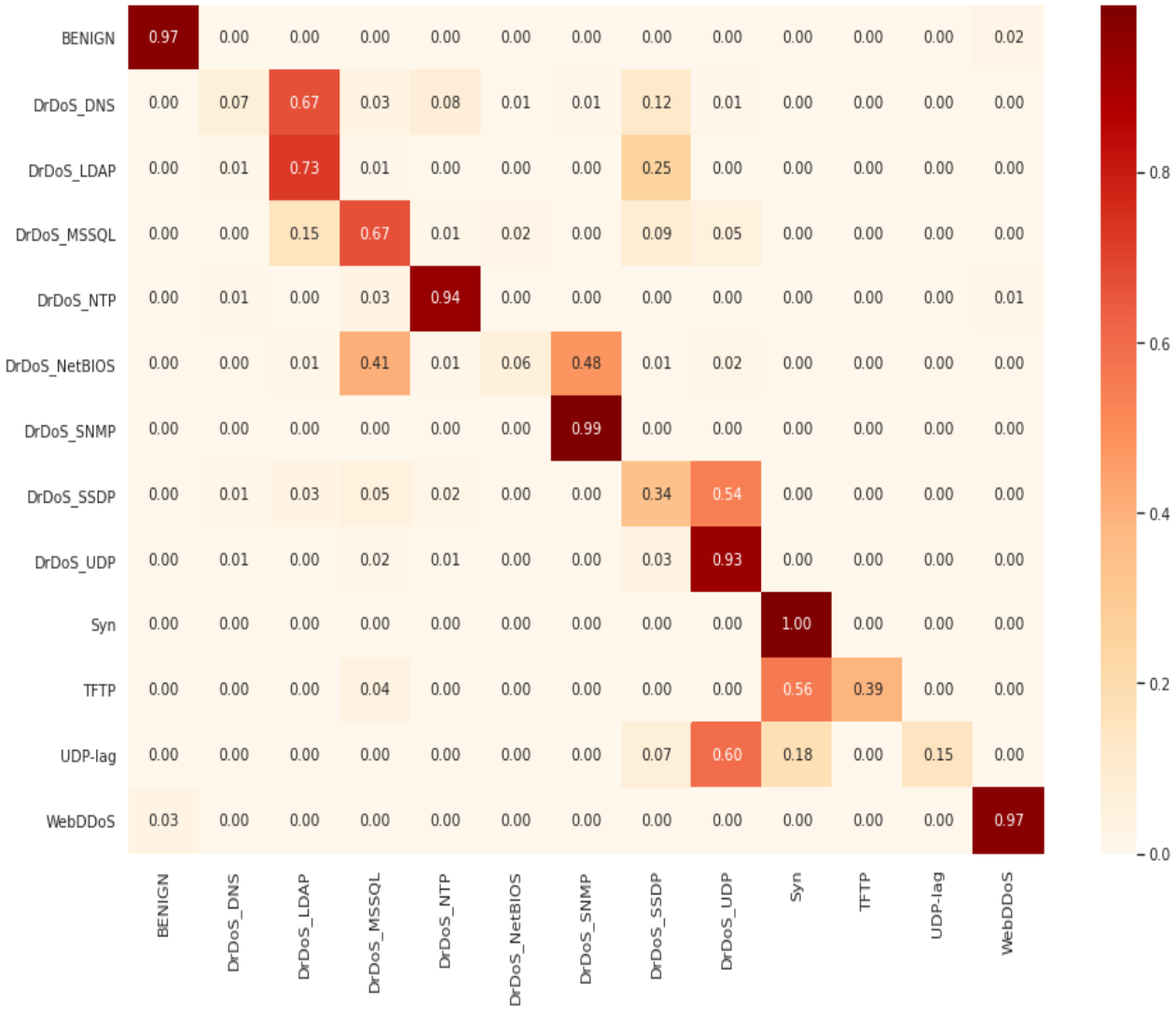


FIGURE 4.10 – Matrice de confusion du CNN (13-classes)

d'obtenir un taux de vrai négative (TNR) très élevé pour tous les modèles proposés. Le taux de vrai négative (TNR) est définie comme suit :

$$TNR = \frac{TN_{BEGNIN}}{TN_{BEGNIN} + FP_{BEGNIN}} \tag{4.1}$$

Les modèles qui ont un TNR convergent vers 1, ce qui nous a donnés des résultats très satisfaits en matière de taux de détection des attaques (TPR ; sans considérer le type d'attaque) qui a été extrêmement élevé 99.98%, ainsi que le taux de fausses alarmes (FPR) a été trop faible, 1.01% pour le DNN , 1.39% pour CNN et 1.54% pour le RNN.

4.8 Conclusion

Nous avons implémenté 3 modèles discriminatoires de deep learning tout en utilisant l'ensemble de données CICDDoS2019 Data-set pour la détection des attaques D.DoSs.

	DL-approche	Précision	Recall	F1-measure	TPR%	FPR%
1-expérimentation Data-set_1 (7-Classes)	DNN	0.85	0.83	0.82	99.87	3.43
	CNN	0.91	0.90	0.89	99.96	2.47
	RNN	0.90	0.89	0.88	99.90	3.58
2-expérimentation Data-set_2 (2-Classes)	DNN	0.98	0.99	0.99	99.74	0.82
	CNN	0.99	1.00	1.00	99.94	0.90
	RNN	0.99	0.98	0.99	99.94	04.22
3- expérimentation Data-set_3 (13-Classes)	DNN	0.71	0.62	0.56	99.98	01.01
	CNN	0.67	0.61	0.55	99.97	01.39
	RNN	0.71	0.62	0.56	99.99	01.54

Tableau 4.11 – Les résultats des méthodes proposées

Le développement avait de nombreux problèmes qui nous a fallu beaucoup de temps pour les résoudre. La masse des données du data-set, le déséquilibre et l'étiquetage des données et ainsi les limites des outils matériels disponibles (processeur, mémoire). Afin de surmonter ces problèmes, nous avons reproduit 3 sous-ensembles de données utilisant le sous-échantillonnage aléatoire (Random under sampling) pour 3 différentes classifications. la première classification était une étude comparative avec 4 méthodes d'apprentissage automatiques (ML) de références. le 2ème et la 3 ème classification étaient pour évaluer les modèles proposés en matière de taux de détection et le taux de fausse alarme avec une classification binaire (normale/ attaque) et une classification multi-classes comprennent des différents types d'attaques.

Des différentes techniques de sur-échantillonnage avec des différents pourcentages ont été essayé afin de surmonter le problème de dés-équilibrage de données, ensuite, de nombreuses expériences ont été faites pour trouver la meilleure architecture avec les bons hyper-paramètres des modèles pour chaque type de classification. Les résultats des modèles de DL proposés étaient meilleurs que celles qui sont obtenues avec les modèles classiques de ML en utilisant les mêmes ensembles de données d'apprentissage et de tests. Les modèles proposés ont obtenu aussi des résultats très satisfaits avec une précision de détection très élevée de différents types d'attaques D.DoSs. Et même que ces attaques n'étaient pas connues auparavant lors de la phase d'apprentissage. Ces modèles peuvent être considérés comme le noyau d'un IDS basé sur la détection des anomalies du trafic réseau.

Conclusion générale

La cybersécurité est un ensemble de pratiques qui consistent à sécuriser les éléments vulnérables grâce aux technologies de l'information et de communication (TIC). Les systèmes de détection d'intrusions ont fait partie de ces pratiques de surveillance dans le but de couvrir l'insuffisance des différents modules de sécurité comme les logiciels antivirus ou les firewalls. Ces logiciels sont dans la plupart du temps inefficaces face à l'évolution des nouvelles menaces plus sophistiquées. Cette étude a été menée afin de démontrer l'efficacité du deep learning pour le domaine de la cybersécurité. Notre objectif est d'implémenter des méthodes de détection d'intrusions basées sur l'apprentissage profond et évaluer ses performances.

Nous avons commencé par le choix de l'ensemble de données où nous avons choisi de travailler avec une Data-set très récente nommé CIC-DDoS-2019 pour la détection des attaques de déni de service distribué (D. Dos). Ces cybers-attaques réseau sont les plus fréquentes et les plus répandus, et comme elles peuvent être lancées à distance et répercutées par des utilisateurs légitimes sur les réseaux . Il est difficile de les détecter et de les prévenir. Notre objectif est d'explorer la détection de ces attaques, et en particulier celles qui sont apparues ces dernières années.
Ensuite, nous avons choisi d'implémenter trois modèles discriminatoires de deep learning (apprentissage supervisé) : un réseau neuronal profond (DNN), un réseau de neurones convolutif (CNN) et un réseau neuronal récurrent (RNN) pour la classification. Le choix de ces méthodes a été fait lorsqu'elles conviennent avec un ensemble de données étiquetées .Ainsi, lorsqu'elles ont fait de bon résultat dans des travaux antérieurs connexes.

Les résultats obtenus sont très satisfaisants, où seules les caractéristiques du trafic réel d'un réseau public sont prises en compte sans aucune information relative aux terminaux connectés, ce qui nous donne à penser que le taux de détection serait encore plus élevé si nous appliquons ces méthodes sur un réseau spécifique.
Il ne suffit que sauvegarder les poids du modèle et mettre en place un capteur de réseau et un analyseur où les flux peuvent être lus en temps réel et envoyés dans le modèle pour la prédiction. Le temps de réponses pour une seule prédiction dépend de la complexité de

modèles (nombre de paramètres du modèle) qui doit suffisamment minimal pour l'utiliser comme un système de détection d'alarme en temps réel.

Perspective

Dans nos travaux futures, nous allons Travailler sur les fichiers pcap du Data-set en utilisant les différentes générateurs de flux du trafic réseau, puis effectuer une analyse exploratoire sur les caractéristiques générés pour extraire des informations supplémentaires sur les profils de diverses attaques DDoS et l'utiliser avec d'autres méthodes d'apprentissage approfondi non-supervisé telles que l'apprentissage autodidacte, l'encodage automatique ...etc.

Bibliographie

[1] Dos attaqurs. https://www.ionos.fr/digitalguide/serveur/know-how/dos-et-ddos-apercu-des-types-dattaques/. [En ligne ; Consulté le 06/08/2020].

[2] imbalanced learning. https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over_sampling.SMOTE.html. [En ligne ; Consulté le 06/06/2020].

[3] Hassan Hadi Al-Maksousy, Michele C Weigle, and Cong Wang. Nids : Neural network based intrusion detection system. In *2018 IEEE International Symposium on Technologies for Homeland Security (HST)*, pages 1–6. IEEE, 2018.

[4] Md Zahangir Alom, VenkataRamesh Bontupalli, and Tarek M Taha. Intrusion detection using deep belief networks. In *2015 National Aerospace and Electronics Conference (NAECON)*, pages 339–344. IEEE, 2015.

[5] Jose Maria Alonso, Rodolfo Bordon, Marta Beltran, and Antonio Guzmán. Ldap injection techniques. In *2008 11th IEEE Singapore International Conference on Communication Systems*, pages 980–986. IEEE, 2008.

[6] DDoS Attacks. Sécurité publique canada. https://www.securitepublique.gc.ca/cnt/rsrcs/cybr-ctr/2014/al14-033-fr.aspx. [En ligne ; Consulté le 06/08/2020].

[7] Ekaba Bisong. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 59–64. Springer, 2019.

[8] Jason Brownlee. Imbalanced classification. https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/. [En ligne ; Consulté le 13/05/2020].

[9] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[10] CIC. Bot iot dataset. https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/bot_iot.php. [En ligne ; Consulté le 20/02/2020].

[11] Clément Dalloux, Natalia Grabar, and Vincent Claveau. Détection de la négation : corpus français et apprentissage supervisé. *Revue des Sciences et Technologies de l'Information-Série TSI : Technique et Science Informatiques*, pages 1–21, 2019.

[12] Hervé Debar, Marc Dacier, and Andreas Wespi. A revised taxonomy for intrusion-detection systems. In *Annales des télécommunications*, volume 55, pages 361–378. Springer, 2000.

[13] Li Deng. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, 3, 2014.

[14] Li Deng, Dong Yu, et al. Deep learning : methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4) :197–387, 2014.

[15] Sumeet Dua and Xian Du. *Data mining and machine learning in cybersecurity*. Auerbach Publications, 2016.

[16] Mohamed Amine Ferrag, Leandros Maglaras, Sotiris Moschoyiannis, and Helge Janicke. Deep learning for cyber security intrusion detection : Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, 50 :102419, 2020.

[17] Ugo Fiore, Francesco Palmieri, Aniello Castiglione, and Alfredo De Santis. Network anomaly detection with the restricted boltzmann machine. *Neurocomputing*, 122 :13–23, 2013.

[18] Ni Gao, Ling Gao, Quanli Gao, and Hai Wang. An intrusion detection model based on deep belief networks. In *2014 Second International Conference on Advanced Cloud and Big Data*, pages 247–252. IEEE, 2014.

[19] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 77 :354–377, 2018.

[20] Sandeep Gurung, Mirnal Kanti Ghose, and Aroj Subedi. Deep learning approach on network intrusion detection system using nsl-kdd dataset. *International Journal of Computer Network and Information Security (IJCNIS)*, 11(3) :8–14, 2019.

[21] Jeff Hale. Deep learning framework. `https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a`. [En ligne ; Consulté le 16/05/2020].

[22] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pages 21–26, 2016.

[23] Min-Joo Kang and Je-Won Kang. Intrusion detection system using deep neural network for in-vehicle network security. *PloS one*, 11(6), 2016.

[24] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems : techniques, datasets and challenges. *Cybersecurity*, 2(1) :20, 2019.

[25] Jihyun Kim and Howon Kim. Applying recurrent neural network to intrusion detection with hessian free optimization. In *International Workshop on Information Security Applications*, pages 357–369. Springer, 2015.

[26] Kwangjo Kim, Muhamad Erza Aminanto, and Harry Chandra Tanuwidjaja. *Network Intrusion Detection Using Deep Learning : A Feature Learning Approach*. Springer, 2018.

[27] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv :1408.5882*, 2014.

[28] Soman KP, Mamoun Alazab, et al. A comprehensive tutorial and survey of applications of deep learning for cyber security. 2020.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[30] Yann LeCun et al. Generalization and network design strategies. *Connectionism in perspective*, 19 :143–155, 1989.

[31] Collins Achepsah Leke and Tshilidzi Marwala. *Deep learning and missing data in engineering systems*. Springer, 2019.

[32] Liran Lerman, Olivier Markowitch, and Gianluca Bontempi. *Les systèmes de détection d'intrusion basés sur du machine learning*. PhD thesis, Thèse de doctorat, Université libre de Bruxelles, 2008.

[33] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system : A comprehensive review. *Journal of Network and Computer Applications*, 36(1) :16–24, 2013.

[34] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234 :11–26, 2017.

[35] Mehedy Masud, Latifur Khan, and Bhavani Thuraisingham. *Data mining tools for malware detection*. Auerbach Publications, 2016.

[36] MAWILab. Mawi dataset. `http://www.fukuda-lab.org/mawilab/data.html`, note=.

[37] Univ of new Brunswick. Cic datasets. `https://www.unb.ca/cic/datasets/.html`, note=.

[38] Manoranjan Pradhan, Chinmaya Kumar Nayak, and Sateesh Kumar Pradhan. Intrusion detection system (ids) and their types. In *Securing the Internet of Things : Concepts, Methodologies, Tools, and Applications*, pages 481–497. IGI Global, 2020.

[39] Shahadate Rezvy, Miltos Petridis, Aboubaker Lasebae, and Tahmina Zebin. Intrusion detection and classification with autoencoded deep neural network. In *International Conference on Security for Information Technology and Communications*, pages 142–156. Springer, 2018.

[40] Frank Rosenblatt. The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6) :386, 1958.

[41] Mostafa A Salama, Heba F Eid, Rabie A Ramadan, Ashraf Darwish, and Aboul Ella Hassanien. Hybrid intelligent intrusion detection scheme. In *Soft computing in industrial applications*, pages 293–303. Springer, 2011.

[42] Karen Scarfone and Peter Mell. Guide to intrusion detection and prevention systems (idps). Technical report, National Institute of Standards and Technology, 2012.

[43] Data scientists. Anaconda for data scientists. `https://www.anaconda.com/`, note=.

[44] Iman Sharafaldin, Arash Habibi Lashkari, Saqib Hakak, and Ali A Ghorbani. Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In *2019 International Carnahan Conference on Security Technology (ICCST)*, pages 1–8. IEEE, 2019.

[45] Robin Sommer and Vern Paxson. Outside the closed world : On using machine learning for network intrusion detection. In *2010 IEEE symposium on security and privacy*, pages 305–316. IEEE, 2010.

[46] Nasrin Sultana, Naveen Chilamkurti, Wei Peng, and Rabei Alhadad. Survey on sdn based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications*, 12(2) :493–501, 2019.

[47] Tuan A Tang, Lotfi Mhamdi, Des McLernon, Syed Ali Raza Zaidi, and Mounir Ghogho. Deep learning approach for network intrusion detection in software defined networking. In *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pages 258–263. IEEE, 2016.

[48] Pablo Torres, Carlos Catania, Sebastian Garcia, and Carlos Garcia Garino. An analysis of recurrent neural networks for botnet detection behavior. In *2016 IEEE biennial congress of Argentina (ARGENCON)*, pages 1–6. IEEE, 2016.

[49] Irvine Univ of California. kddcup99 dataset. `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`. [En ligne ; Consulté le 08/05/2020].

[50] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. Applying convolutional neural network for network intrusion detection. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1222–1228. IEEE, 2017.

[51] Wei Wang, Yiqiang Sheng, Jinlin Wang, Xuewen Zeng, Xiaozhou Ye, Yongzhong Huang, and Ming Zhu. Hast-ids : Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. *IEEE Access*, 6 :1792–1806, 2017.

[52] Wood, Mark and Erlinger, Michael. Intrusion detection message exchange requirements. `https://tools.ietf.org/html/rfc4766`, 2002. [En ligne ; Consulté le 15/02/2020].

[53] Kehe Wu, Zuge Chen, and Wei Li. A novel intrusion detection model for a massive network using convolutional neural networks. *IEEE Access*, 6 :50850–50859, 2018.

[54] Chiba Zouhair, Noreddine Abghour, Khalid Moussaid, Amina El Omri, and Mohamed Rida. A review of intrusion detection systems in cloud computing. In *Security and Privacy in Smart Sensor Networks*, pages 253–283. IGI Global, 2018.